FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Development of an Integrated Monitoring Platform for the Motio System: Advancing Efficiency and User Experience

Paula Mika Ogata





Master in Bioengineering - Biomedical Engineering

Supervisor: Hélder Filipe Pinto de Oliveira (Prof. Dr.)

Second Supervisor: Sofia Assis

July 24, 2024

© Paula Mika Ogata, 2024

Development of an Integrated Monitoring Platform for the Motio System: Advancing Efficiency and User Experience

Paula Mika Ogata

Master in Bioengineering - Biomedical Engineering

Approved by . . . :

President: José Machado Da Silva
Referee: Inês Domingues
Referee: Hélder Filipe Pinto de Oliveira

July 24, 2024

Resumo

No cenário competitivo atual, as empresas com *insights* aprofundados sobre as suas operações e desempenho têm frequentemente uma vantagem significativa sobre aquelas que não investem em ferramentas de Business Intelligence (BI) ou análise de dados. Esses *insights*, derivados de dados internos, permitem que elas façam previsões precisas, melhorem a tomada de decisões, reduzam custos e aumentem a eficiência operacional. Além disso, empresas envolvidas no desenvolvimento de aplicações se beneficiam muito com *insights* sobre seus sistemas de *issue tracking*. Ao obtê-los, essas empresas podem melhor gerenciar os *issues* e otimizar seus processos de resolução, levando a um melhor desempenho geral e à satisfação do cliente.

O presente trabalho, realizado em colaboração com a Adapttech, descreve o desenvolvimento de uma plataforma de monitorização para o seu produto, Motio StepWatch™ System, que ajuda a classificar os níveis funcionais de utilizadores de próteses, monitorizando-os no seu dia a dia. A aplicação, denominada *Motio System Monitoring Platform*, fornece *insights* sobre o uso da aplicação do Motio StepWatch™ System e o gerenciamento de *feedback tickets* (*issues*, *improvements*, *complaints* e CAPA). Além disso, também permite descarregar relatórios de *outcome tests* padrões, como o PLUS-M™, o AMPPRO® e o Timed Up and Go.

A aplicação web, Motio System Monitoring Platform, foi desenvolvida em três etapas. Primeiramente, foram realizados alguns estudos iniciais, que incluíram levantamento de requisitos e *conceptual designing*. A segunda etapa consistiu em quatro iterações de prototipagem, tendo como *input* os requisitos, feedback dos stakeholders e teste de usabilidade. Em seguida, a implementação da Coleta de Dados (conexão com a base de dados e obtenção de dados), Preparação e Análise de Dados (processamento de dados) e Visualização de Dados (criação da interface do usuário baseada no protótipo e com ligação à base de dados usando *Dash*) resultaram no produto final.

Palavras-chave: Platforma de Monitorização, Visualização de Dados, Sistemas de Issue Tracking, Desenvolvimento de Aplicações Web, Design de Interface

Abstract

In today's competitive landscape, businesses with in-depth insights into their operations and performance often have a significant advantage over those that do not invest in Business Intelligence (BI) tools or data analytics. These insights, derived from internal data, enable organizations to make accurate predictions, improve decision-making, reduce costs, and enhance operational efficiency. Moreover, companies involved in application development benefit greatly from understanding their issue tracking systems. By gaining insights into these systems, they can better manage issues and optimize their resolution processes, leading to improved overall performance and customer satisfaction.

The present work, done in collaboration with Adapttech®, describes the development of a monitoring platform for their product, Motio StepWatch™ System. This product helps classify prosthetic users' functional levels by monitoring them in their everyday lives. The developed application, named Motio System Monitoring Platform, provides insights into the usage of the Motio StepWatch™ System's application and the management of feedback (issues, improvements, complaints, or CAPA) tickets. Furthermore, it also allows the export of standard outcome test reports, such as PLUS-M™, AMPPRO®, and Timed Up and Go.

The Motio System Monitoring Platform, a web application, was developed in three steps. First, some initial studies were conducted, which included requirements gathering and conceptual designing. The second step consisted of four iterations of prototyping, having as input the requirements, stakeholders' feedback, and usability evaluation. Following that, the implementation of Data Collection (database connection and data retrieval), Data Preparation and Analysis (data processing), and Data Visualization (creation of the user interface using Dash based on the prototype and with connection with the data) resulted in the final product. .

Keywords: Monitoring Platform, Data Visualization, Issue Tracking Systems, Web App Development, User Interface Design

Acknowledgments

Escrevo esses agradecimentos na véspera do dia de São João, feriado que se tornou o meu preferido desde que me mudei para Portugal. Mas não foi só o meu feriado preferido que mudou desde então, esses últimos cinco anos contribuíram muito para a pessoa que sou hoje. Foram tantos acontecimentos e aprendizados, de lágrimas sem fim até risadas genuínas e contagiantes. Por isso que, para mim, escrever essa dissertação não representa só minha graduação em Bioengenharia, mas também o fim desse período, da Paula que saiu do Brasil porque o sonho dela era estudar fora.

Dessa forma, gostaria de dedicar essa dissertação a todos que estiverem presentes (mesmo à distância) nessa fase e agradecê-los por todo o apoio e companhia que me fizeram.

Primeiramente, ao meu orientador na FEUP, Prof. Hélder Oliveira, pelas reuniões e comentários que foram essenciais para o desenvolvimento desta Dissertação. À Sofia, por todo o tempo dedicado a mim nesses últimos seis meses, este projeto não teria sido tão divertido desenvolver se não fosse tua ajuda.

Aos meus pais, por sempre confiarem em mim e porque nada disso seria possível sem vocês. À minha irmã, que me recebeu no Porto com os braços abertos e me faz companhia todos os dias. À Molly, minha cadela que veio comigo para Portugal, esteve lá todos os dias por mim e que presenciou minhas graduações no Ensino Fundamental I, Ensino Fundamental II, Ensino Médio, Licenciatura e, agora, no Mestrado. À minha avó, que desde que eu mudei para cá dizia que queria me visitar e esse ano veio para ver minha Queima das Fitas como Finalista. À minha família, por sempre enviarem mensagens de suporte e congratulações.

E, como sempre dizem que amigos são a família que nós escolhemos, aproveito para agradecê-los também. Ao João, minha dupla "Compre 1 e leve 2" do curso, por todos os trabalhos que fizemos juntos e companhia pelos grupos que participamos nestes últimos anos. Bei Nici, mein größtes Geschenk vom Erasmus in Köln, für all die Unterstützung und Gespräche. Às meninas do Clube do Livro (Mah, Kath, Guti, Laís e Paula), meus presentes do Porto que eu não trocaria por nada, até porque o que seria de mim sem nossas idas a restaurantes novos e rolês de senhora? À Érica, minha companheira de tese, por todas as idas à Adapttech e almoços juntas. Ao Fe, que apesar de longe, estar sempre disposto a ter conversas que só nós dois entendemos. Ao grupo Bitchuss, grupo de Bio que me acolheu e que estou sempre animada para as viagens. Ao grupo Biomigos, que apesar da maioria ter voltado a terras tupiniquins, foram muito importantes nos meus primeiros anos aqui. Aos meus amigos do Brasil, que apesar da distância, nunca perdemos contato e estão lá para o que der e vier.

Por fim, gostaria de agradecer a todos que tiraram um tempinho de suas vidas agitadas para participarem do meu teste que fiz para a Dissertação, em especial Andreia, Anna Gabi, Érica, Fe, Flora, Guti, Inês, João, Kath, Laís, Ligia, Ling, Mah, Mafalda, Manel, Maria, Matheus, Naomi, Nici, Rafa e Serena.

Paula Ogata

"Viver é um rasgar-se e remendar-se."

João Guimarães Rosa

Contents

1 Introduction **1**
- 1.1 Motivation . . . 3
- 1.2 Goals . . . 5
- 1.3 Document Structure . . . 5

2 Background and Literature Review **7**
- 2.1 Motio StepWatch™ System . . . 7
  - 2.1.1 Outcome Tests . . . 8
  - 2.1.2 How Motio App Works . . . 11
  - 2.1.3 Feedback (Issues, Improvements, Complaints, and CAPA) System . . . 11
- 2.2 Business Intelligence . . . 16
  - 2.2.1 Why Business Intelligence? . . . 17
  - 2.2.2 Structure of a BI Application . . . 17
  - 2.2.3 BI Implementation Success . . . 19
  - 2.2.4 Current Solutions . . . 19
- 2.3 Dashboards and Data Visualization . . . 21
- 2.4 Issue Tracking Systems . . . 25
  - 2.4.1 Current Solutions . . . 25
- 2.5 Summary . . . 28

3 Methodologies for Application Development **29**
- 3.1 Design Principles and Practices . . . 29
  - 3.1.1 User-Centered Design (UCD) . . . 29
  - 3.1.2 Iterative Design . . . 30
  - 3.1.3 Usability . . . 31
- 3.2 Initial Studies . . . 32
  - 3.2.1 Requirements Gathering . . . 32
  - 3.2.2 Conceptual Design . . . 34
- 3.3 Prototyping and Designing . . . 35
  - 3.3.1 Low-Fidelity (Lo-Fi) Prototyping . . . 35
  - 3.3.2 High-Fidelity (Hi-Fi) Prototyping . . . 36
  - 3.3.3 Style Guide for Designing . . . 38
- 3.4 Evaluation . . . 40
  - 3.4.1 Usability Testing . . . 40
  - 3.4.2 Feedback Gathering from Stakeholders . . . 40
- 3.5 Implementation . . . 40
- 3.6 Summary . . . 42

4 Initial Studies **43**
4.1 Stakeholder Requirements . . . 43
4.1.1 Stakeholders Identification . . . 43
4.1.2 Requirements Gathering . . . 44
4.1.3 Requirements Analysis . . . 44
4.2 Conceptual Design . . . 49
4.2.1 Conceptual Model . . . 49
4.2.2 System Model . . . 49
4.3 Summary . . . 57

5 UI Prototyping and Validation **61**
5.1 Low-Fidelity (Lo-Fi) Prototype . . . 61
5.1.1 Feedback Gathering - Lo-Fi Prototype . . . 64
5.2 High-Fidelity (Hi-Fi) Prototypes . . . 68
5.2.1 Hi-Fi Prototype - Prototype Version 1 . . . 68
5.2.2 Hi-Fi Prototype - Prototype Version 2 . . . 76
5.2.3 Hi-Fi Prototype - Prototype Version 3 . . . 79
5.3 Prototype Validation - Usability Testing . . . 84
5.3.1 Methodology . . . 84
5.3.2 Results - Usability Testing . . . 85
5.4 Summary . . . 87

6 Implementation **91**
6.1 Implementation Architecture . . . 91
6.2 Backend Implementation - Data Collection . . . 92
6.2.1 SQL Interface . . . 92
6.2.2 MS Interface . . . 93
6.3 Backend Implementation - Data Preparation and Analysis . . . 94
6.4 Backend and Frontend Implementation - Data Visualization . . . 97
6.4.1 Available Technologies - Data Visualization . . . 97
6.4.2 Requirements Implementation . . . 99
6.5 Deployment and Environment . . . 106
6.6 Results . . . 108
6.7 Summary . . . 108

7 Conclusion and Future Work **125**
7.1 Acquired Knowledge . . . 126
7.2 Challenges Encountered . . . 126
7.3 Future Work . . . 127

References **129**

A Usability Testing Details **135**
A.1 Test Instructions . . . 135
A.2 Satisfaction Questionnaire . . . 136
A.3 Quant-UX Testing Consent . . . 138
A.4 Effectiveness and Efficiency Measures - All Results . . . 139
A.5 Satisfaction Questionnaire - All Answers . . . 139
A.6 Other Outcomes from the Usability Testing . . . 142

B Motio App Performance Queries **147**
B.1 Patient Query 147
B.2 Appointment Query 147
B.3 Outcome Tests Query 147
B.4 Appointments Related to the Outcome Tests Query 148
B.5 PLUS-M™ Query 148
B.6 AMPPRO Query 150
B.7 TUG Query 151

List of Figures

1.1 Ancient toe prostheses . . . 1
(a) Cairo toe . . . 1
(b) Greville Chester toe . . . 1
1.2 Anglesey leg . . . 2
1.3 Motio StepWatch™ System . . . 4
(a) Motio StepWatch™ Device . . . 4
(b) Motio App . . . 4
1.4 Project pipeline . . . 5

2.1 Motio App - Patients page . . . 12
2.2 Motio App - Create new patient page . . . 12
2.3 Motio App - Create new appointment . . . 13
2.4 Motio App - Create new outcome test on the appointment page . . . 13
2.5 Report Feedback Page on Report Motio Feedback App . . . 15
2.6 Evolution of Decision Support Systems . . . 16
2.7 *Business Intelligence* architecture . . . 18
2.8 Gartner Magic Quadrant for Analytics and Business Intelligence technologies in the year 2023 . . . 20
2.9 Simplified life cycles of issues . . . 25

3.1 Typical application development pipeline . . . 29
3.2 Iterative user interface design and evaluation process . . . 31
3.3 Example of wireframe . . . 36
3.4 Scheme of a Hi-Fi prototype . . . 37
3.5 "One Vase or Two Faces?" - Figure-ground segregation example . . . 39
3.6 Standard System Usability Scale . . . 41

4.1 Use Case diagram . . . 50
4.2 UML Package diagram - System components . . . 51
4.3 UML Activity diagram - Motio App data visualization . . . 52
4.4 UML Activity diagram - Motio App and Feedback data export . . . 53
4.5 UML Sequence diagram - Motio App data visualization and export . . . 53
4.6 UML Activity diagram - Feedback data visualization . . . 54
4.7 UML Sequence diagram - Feedback data visualization and export . . . 55
4.8 UML Activity diagram - Outcome test reports export . . . 55
4.9 UML Sequence diagram - Outcome test reports export . . . 56
4.10 UML Component diagram - SQL Interface . . . 57
4.11 Class diagram - Motio App (SQL database) . . . 58
4.12 UML Component diagram - Microsoft Interface . . . 59

4.13 UML Class diagram - Feedback (Microsoft Planner and Sharepoint List) 59

5.1 Wireframe - Motio App Performance page 62
5.2 Wireframe - Filter menu 63
5.3 Wireframe - Post-Market Surveillance page 65
5.4 Suggested data visualization format for every requirement 66
5.5 Wireframe - Clinical Insights page 67
5.6 Functional prototype - Version 1 - Home page 69
5.7 Functional prototype - Version 1 - Motio App Performance page 70
5.8 Functional prototype - Version 1 - Filter menu 71
5.9 Functional prototype - Version 1 - Post-Market Surveillance page 72
5.10 Functional prototype - Version 1 - Test Reports page 73
5.11 Functional prototype - Version 2 - Motio App Performance page 77
5.12 Functional prototype - Version 2 - Post-Market Surveillance page 78
5.13 Functional prototype - Version 2 - Outcome Test Reports page 79
5.14 Functional prototype - Version 3 - Motio App Performance initial page 80
5.15 Functional prototype - Version 3 - Post-Market Surveillance initial page 80
5.16 Functional prototype - Version 3 - Motio App Performance page 81
5.17 Functional prototype - Version 3 - Post-Market Surveillance page 82
5.18 Functional prototype - Version 3 - Outcome Test Reports page 83
5.19 Distribution of answers per question in the Satisfaction Questionnaire 86
(a) Question 2 86
(b) Question 3 86
(c) Question 4 86
(d) Question 5 86
(e) Question 6 86
(f) Question 7 86
(g) Question 8 86
(h) Question 9 86
(i) Question 10 86

6.1 Motio System Monitoring Platform implementation architecture 91
6.2 UML Class Diagram - SQL and MS Interfaces 92
6.3 Group 1 graph data 95
(a) Appointments graph data - Example of the Quarterly view (June 2024) . . 95
(b) Outcome Tests graph data - Example of the Quarterly view (June 2024) . . 95
6.4 Group 4 graph data 96
(a) New issues graph data - Example of the Quarterly view (June 2024) 96
(b) Open X New graph data - Example of the Quarterly view (June 2024) . . . 96
(c) Average Closing Times graph data - Example of the Quarterly view (June 2024) 96
6.5 Popularity of data visualization tools 97
6.6 Callback diagram of the Motio App Performance page 100
6.7 Callback diagram of the Post-Market Surveillance page 103
6.8 Callback diagram of the Outcome Test Reports page 105
6.9 Label display when the user hovers the cursor over the chart 106
6.10 Scheme of Motio System Monitoring Platform deployment 107
6.11 Implemented version - Home page 109
6.12 Implemented version - Motio App Performance page - Initial page 112

6.13 Implemented version - Motio App Performance page - Monthly View (Viewed in June 2024) . . . 113
6.14 Implemented version - Motio App Performance page - Quarterly View (Viewed in June 2024) . . . 114
6.15 Implemented version - Motio App Performance page - Yearly View (Viewed in June 2024) . . . 115
6.16 Implemented version - Motio App Performance page - Custom View (05/Mar/2024-19/Apr/2024 as selected dates) . . . 116
6.17 Implemented version - Motio App Performance page - Location filter (Location 2 selected) . . . 117
6.18 Implemented version - Post-Market Surveillance page - Initial page . . . 118
6.19 Implemented version - Post-Market Surveillance page - Monthly view (Viewed in June 2024) . . . 119
6.20 Implemented version - Post-Market Surveillance page - Quarterly view (Viewed in June 2024) . . . 120
6.21 Implemented version - Post-Market Surveillance page - Yearly view (Viewed in June 2024) . . . 121
6.22 Implemented version - Post-Market Surveillance page - All filters selected . . . 122
6.23 Implemented version - Outcome Test Reports . . . 123
6.24 Implemented version - Test Reports options . . . 123
6.25 Implemented version - Table export button . . . 123
6.26 Implemented version - Export graphics as png option . . . 124
6.27 Group 4 graph data . . . 124
(a) MAP and PMS pages - Missing View error message . . . 124
(b) MAP and PMS pages - Missing field error message . . . 124
(c) Outcome Test Reports page - Missing field error message . . . 124

A.1 Satisfaction Questionnaire - Part 1 . . . 136
A.2 Satisfaction Questionnaire - Part 2 . . . 137
A.3 Quant-UX's initial message about privacy . . . 138
A.4 Boxplot concerning interactions and duration for each task . . . 142
A.5 Data distribution of tests - Interactions X Duration . . . 142
A.6 Heatmap - Motio App Performance - Initial page . . . 143
A.7 Heatmap - Motio App Performance - Selecting filters . . . 143
A.8 Heatmap - Motio App Performance - Exporting data . . . 144
A.9 Heatmap - Post-Market Surveillance - Initial page . . . 144
A.10 Heatmap - Outcome Test Reports - Downloading data . . . 145

List of Tables

2.1 Correlation between AMPPRO® scores and K levels . . . 8
2.2 Overview of current Business Intelligence solutions listed in Gartner Magic Quadrant . . . 22
2.2 Overview of current Business Intelligence solutions listed in Gartner Magic Quadrant . . . 23
2.3 Brief overview of four current Issue Tracking Systems . . . 26

3.1 Knowledge obtained from the task analysis data . . . 34
3.2 Benefits and challenges of gathering stakeholder feedback . . . 41

4.1 Functional requirements related to Motio App Performance . . . 45
4.2 Functional requirements related to Post-Market Surveillance of Motio App . . . 46
4.3 Functional requirements related to Clinical Insights . . . 47
4.4 Other functional requirements . . . 48

5.1 Stakeholder Feedback - Warnings and advice . . . 74
5.2 Stakeholder Feedback - Ideas . . . 75
5.3 Stakeholder Feedback - Positive Feedback . . . 75
5.4 Summary of usability testing measures regarding effectiveness and efficiency . . . 85
5.5 Traceability Matrix after Prototyping - Motio App Performance requirements . . . 88
5.6 Traceability Matrix after Prototyping - Post-Market Surveillance requirements . . . 89
5.7 Traceability Matrix after Prototyping - Clinical Insights requirements . . . 90
5.8 Traceability Matrix after Prototyping - Other requirements . . . 90

6.1 Group 1 - Steps of the Data Preparation and Analysis . . . 95
6.2 Group 4 - Steps of the Data Preparation and Analysis . . . 96
6.3 Used components and their IDs in the Motio App Performance page . . . 101
6.4 Used components and their IDs in the Post-Market Surveillance page . . . 104
6.5 Used components and their IDs in the Outcome Test Reports page . . . 105
6.6 Traceability Matrix after Implementation - Motio App Performance requirements 109
6.7 Traceability Matrix after Implementation - Post-Market Surveillance requirements 110
6.8 Traceability Matrix after Implementation - Clinical Insights requirements . . . 111
6.9 Traceability Matrix after Implementation - Other requirements . . . 111

A.1 Usability testing measures per each test and each task . . . 140
A.2 Satisfaction Questionnaire answers per test . . . 141

Abbreviations

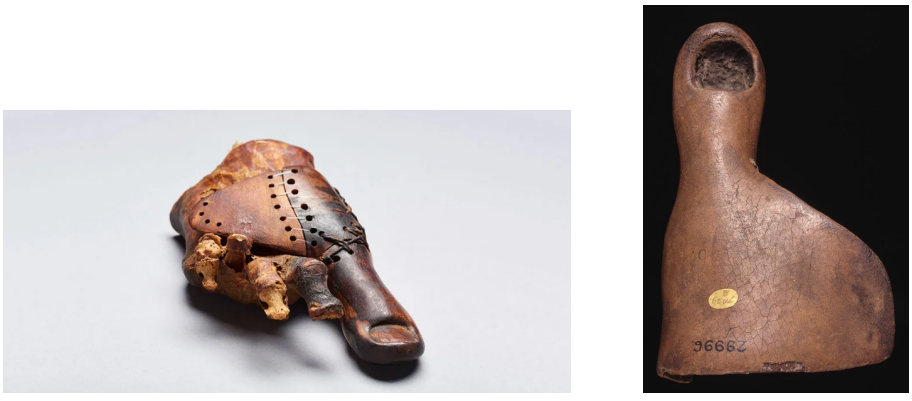
ABI	Analytics and Business Intelligence
API	Application Programming Interface
AMP	Amputee Mobility Predictor
BI	Business Intelligence
BI&A	Business Intelligence and Analytics
CAPA	Corrective and Preventive Action
CMS	Centers for Medicare/Medicaid Services
DM	Data Mining
ITS	Issue Tracking System
KPI	Key Performance Indicator
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
MAP	Motio App Performance
MFCL	Medicare Functional Classification Level
MS	Microsoft
NLQ	Natural Language Query
OLAP	Online Analytical Processing
O&P	Orthotics and Prosthetics
PLUS-M	Prosthetic Limb Users Survey of Mobility
PMS	Post-Market Surveillance
SEO	Search Engine Optimization
SDK	Software Development Kits
SQL	Structured Query Language
SUS	System Usability Scale
TUG	Timed Up and Go
UCD	User-Centered Design
UI	User Interface
UML	Unified Modeling Language
UX	User Experience
VPN	Virtual Private Network

Chapter 1

Introduction

By 2050, it is estimated that the quantity of limb-loss people living in the United States will almost double, according to Ziegler-Graham et al. [1]. The primary causes are a larger elderly demographic and the rising incidence of illnesses like diabetes that may cause inadequate blood flow in the legs. As a consequence, there will also be an increase in the need for prostheses, which fortunately have evolved significantly throughout the years.

The "Cairo toe" (Figure 1.1a), housed at the Egyptian Museum in Cairo, is considered the oldest existing intravital limb prosthesis as it is believed to have been used between 950 and 710 BC. It was found attached to the right foot of a female mummy and is comprised of three leather and wood components joined together by an intricate system of laces. Meanwhile, the British Museum in London, United Kingdom, has possession of another primitive replacement toe, the "Greville Chester toe" (Figure 1.1b), which dates back to before 600 BC. It is made of cartonnage, a type of paper maché consisting of layers of linen soaked in animal glue and covered in dyed plaster [2].



(a) Cairo toe (b) Greville Chester toe

Figure 1.1: Ancient toe prostheses[1][2]

[1] https://www.smithsonianmag.com/smart-news/study-reveals-secrets-ancient-cairo-toe-180963783/. Extracted on 2024-04-29.

[2] https://www.britishmuseum.org/collection/object/Y_EA29996. Extracted on 2024-04-29.

Roughly 300 years later - 300 BC - in the northern region of Naples, Italy, the most famous European prosthetic nowadays was crafted for an ancient Roman noble [3]. Known as the "Capua leg", it was made of a wooden core, covered in bronze and hollow near the top, presumably to accommodate padding for the owner. Thin rods and leather straps helped secure the limb in place[1].

Additionally, references to body part replacements can be found in the classical literature of several cultures. The possibly earliest reference is an archaic, sacred poem of India—Rig Vega, written between 3500 BC and 1800 BC [4]. Therefore, it is reasonable to say that prostheses have been part of humankind for a considerable time.

From the late fifteenth century through the eighteenth century, prostheses became more sophisticated, capable of rotation and bending through cables, gears, cranks, and springs [3]. A notorious name for prosthetic improvement was Ambroise Paré (1510-1590), considered the inventor of the modern upper-limb and lower-limb prostheses. Among his inventions, his above-knee prosthesis consisting of a kneeling peg and a prosthetic foot stands out since its features, a locking knee, a suspension strap, and a fixed equinus position, are still used today. His "Le Petit Lorrain", a mechanical hand used in combat and powered by springs and catches, was another remarkable creation of him [4].

Other notable figures in the prosthetic field were Pieter Verduyn (ca. 1625–1700) and James Potts. Verduyn, a Dutch surgeon, introduced the first non-locking, below-knee prosthesis. While James Potts, in 1800, devised the "Anglesey Leg" (Figure 1.2), a prosthetic leg with a steel knee joint, a wooden shank and socket, and an articulated foot controlled by catgut tendons from the knee to the ankle [4].



Figure 1.2: Anglesey leg[2]

[1] https://daily.jstor.org/a-brief-history-of-prosthetic-limbs/. Accessed on 2024-04-29.

[2] https://collection.sciencemuseumgroup.org.uk/objects/co476453/artificial-leg-anglesey-type-1890-1920-medicine-artificial-limbs-artificial-legs-prostheses-disability. Extracted on 2024-04-29.

In the following centuries, the need for prosthetics unfortunately increased due to large-scale conflicts, such as World War I and II, which resulted in fast advancements in the field. The "Hanger limb", for example, was a wooden leg prosthesis conceived during the American Civil War. Since James E. Hanger, an amputee himself, designed it, it had the user's comfort in mind, featuring ankles and cushioned heels[3] [3].

Today, replacement limbs can be made of state-of-the-art materials and tailored to specific needs and preferences. For example, in the case of a lower-limb prosthesis, several types of feet and knees are available, each with different functionality, depending on how active the patient is [5]. To assess the functional level and rehabilitation potential of individuals with lower limb loss, the United States' Medicare Functional Classification Level (MFCL), also known as K level, is used. It ranges from 0 (no ability or potential to ambulate or transfer; lower limb prosthesis will not enhance the quality of life or mobility) to 4 (ability or potential to exceed basic ambulation skills) [6].

However, since the classification procedure is done by prosthetists, whose opinions can diverge, it can be rather subjective. Besides, the difference between the person's capability to perform at a high functional level during clinical tests and their high-functioning conduct in the day-to-day is often disregarded [7]. These inaccuracies can make the optimal componentry selection more challenging and impact the patient's possibility of reimbursement, as US Centers for Medicare/Medicaid Services (CMS) only approve the reimbursement when the prosthetic components correspond to the patient's K level [7].

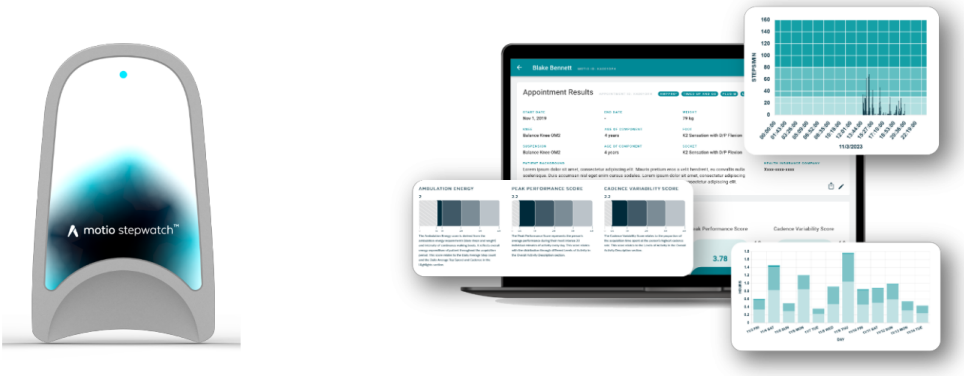
The Motio StepWatch™ System, developed by Adapttech®, addresses this issue. It collects real-world data about prosthetic users' daily activities by tracking them for a minimum period of seven days. This yields insights into the prosthetic user's daily needs and the potential for ambulation they actually have during daily life, generating additional evidence to support the evaluation of the patient's functional level.

The system comprises two main components: the Motio StepWatch™ device and the Motio App. The Motio StepWatch™ device (Figure 1.3a), worn by the patient, collects step data on cadence, distance, speed, and other metrics throughout their daily activities. Meanwhile, the web-based Motio App (Figure 1.3b) manages the entire system, congregates patients' ambulation data acquired from outside the clinical setting, and generates a customized report indicating their functional level. It also allows O&P professionals to conduct current standard outcome measures and clinical tests, such as PLUS-M™, AMPPRO®, and Timed Up and Go. Furthermore, the system can also monitor whether the patient's quality of life improved or remained stable [8].

1.1 Motivation

Throughout the last two decades, the use of the internet skyrocketed [9]. Today, it is estimated that people spend an average of 6.5 hours per day on the internet [10], using it for everything from

[3] https://www.battlefields.org/learn/articles/story-james-e-hanger-amputee-turned-entrepreneur/. Accessed on 2024-04-30.



(a) Motio StepWatch™ Device (b) Motio App

Figure 1.3: Motio StepWatch™ System

basic activities, such as browsing websites, to more sophisticated ones, like transferring money or purchasing goods. Consequently, robust and reliable applications became crucial to fulfill these needs.

Nevertheless, even the most well-designed and precisely engineered applications can experience issues during their lifecycle. Furthermore, the complexity of maintaining and monitoring them grows dramatically as they evolve and scale. From that, it is clear the demand for an application monitoring platform that offers real-time visibility into an application's performance. In this manner, businesses can proactively detect and address issues before they escalate into critical problems.

Alongside monitoring platforms, current corporations employ *Business Intelligence* (BI) solutions to obtain a more in-depth understanding of their operations. This term encompasses the techniques, technologies, applications, architectures, and practices that transform data into information or knowledge, enabling the discovery of hidden patterns and trends [11]. By providing insights into the performance of the organization, these BI tools support decision-making and enhance operational efficiency [12][13].

The presented work, developed at Adapttech®, takes into consideration both concepts. Adapttech is a biomedical innovation company whose goal is to improve the quality of life for lower-limb prosthetic users. They stand out for introducing novel smart technologies to the O&P market, such as remote patient monitoring and data-driven solutions to support clinicians' decisions[4].

Among their developments is the Motio StepWatch™ System, described previously. In order to gain more detailed insights into the use and performance of this system, there is a need for a monitoring platform capable of general system improvement, resource reallocation, and proactive issue detection.

The development of an efficient integrated monitoring platform for the Motio StepWatch™ System has the potential to improve its functionality and user experience while also meeting essential demands across several areas, such as process optimization, energy management, and

[4] https://www.adapttech.eu/. Accessed on 2024-05-11.

industrial automation. To achieve this, the platform must gather advanced data visualization, analytics, and stakeholder collaboration.

An analogous platform was previously developed by Adapttech for its now obsolete product, INSIGHT (Inner Socket Information Gathering Tool), a smart prosthetic-fitting solution [5]. One of the drawbacks of this platform was the limited focus on engineering aspects and its usage only for customer support.

1.2 Goals

The main goal of this project is to develop a complete monitoring platform for the Motio StepWatch™ System that combines useful insights to support the decision process with real-time visibility into the system's operational aspects. Therefore, the platform must provide information about the Motio App usage and the feedback system, as well as allow export of all data.

The initial steps entail an in-depth comprehension of the Motio product and the database, requirements gathering from internal stakeholders, model conceptualization, and the development of a functional monitoring dashboard prototype. Next, stakeholders will iteratively validate the prototype to ensure alignment with actual user preferences and demands. Then, the validated platform with all functionalities and features seamlessly integrated will be implemented. The project culminates with the deployment of the monitoring application, which will enhance user satisfaction and optimize system functionality. Figure 1.4 summarizes the complete project workflow.

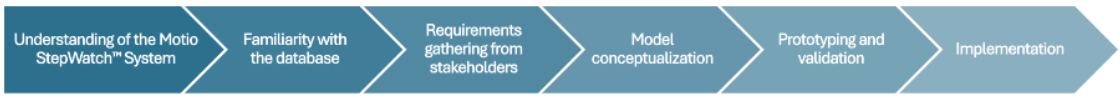


Figure 1.4: Project pipeline

The entire development process, from conceptualization to execution, will be documented. The documentation serves as a comprehensive guide for users and administrators and is essential for knowledge transfer, system transparency, and future maintenance.

1.3 Document Structure

Besides the introduction, this document is organized in the following manner.

- Chapter 2 introduces the Motio StepWatch System along with concepts of Business Intelligence and Issue Tracking Systems and their current solutions;
- Chapter 3 reviews a typical pipeline for application development, which served as the base of this project;
- Chapter 4 corresponds to the initial part of the pipeline, the Initial Studies, comprised of the requirements gathering and conceptual design;

- Chapter 5 refers to the second step of the pipeline, the UI Prototyping, and describes all Motio System Monitoring Platform prototype versions and their evaluations;
- Chapter 6 corresponds to the third step of the pipeline, the Implementation, in which the application takes shape from the database connection to the development of the user interface;
- Lastly, Chapter 7 concludes the project and reflects on the obtained knowledge, encountered difficulties, and potential future work.

Chapter 2

Background and Literature Review

This project aims to develop a monitoring app for the Motio StepWatch™ System that provides insights into the Motio App use and Adapttech's feedback system. Therefore, it combines knowledge of Business Intelligence and Issue Tracking Systems.

To provide some theory about these concepts, Section 2.1 delves deeply into the function of the Motio StepWatch™ System, describing the available outcome tests, the functionality of the Motio App, and the feedback system inside Adapttech. Then, the topic of Business Intelligence is discussed in Section 2.2, which discloses its benefits, structure, steps to success, and current solutions. Although dashboards and data visualization are part of Business Intelligence, they were given a separate section, Section 2.3, since they are one of the most essential aspects of this project. This section provides the motivation and benefits of its implementation, and strategies for achieving a successful design. Finally, Issue Tracking Systems are introduced in Section 2.4, along with a subsection dedicated to their current solutions.

2.1 Motio StepWatch™ System

As briefly described in the introduction of Chapter 1, the Motio StepWatch™ System is a tool for evaluating activity patterns and classifying activity levels in lower limb prosthetic users. It monitors patients during community ambulation to assess their K level and provides the Motio Functional Report to insurance companies that oversee prosthetic component supply. Therefore, it can address issues, such as the subjectivity of traditional tests and classifications based mostly on the clinician's assessment, the challenge of determining the optimal componentry for each prosthetic user's needs, and the manual generation of reimbursement data.

The Motio StepWatch™ System consists of two components. On the one hand, the Motio StepWatch™ device supports the clinician to accurately classify their patient's activity level by collecting data on cadence, distance, speed, and other metrics for a minimum period of seven days. On the other hand, the Motio App allows O&P professionals to assess patients' ambulation data acquired from their everyday lives outside of the clinical environment, in addition to performing the current standard outcome tests.

To use the system, the clinician must follow the following steps:

1. Place the Motio StepWatch™ device on the patient's prosthetic ankle
2. Evaluate the patient with one or more of the functional tests supported by the App
3. Calibrate and set up the device to record activity for a long-term acquisition
4. Send patient for a 7-day real-world evaluation
5. Seven days later, retrieve the device and patient data from the Motio App
6. Generate the Motio Functional Report and share it with the rehabilitation team

2.1.1 Outcome Tests

This subsection describes the outcome tests available in the Motio System, namely AMPPRO®, PLUS-M™, Timed Up and Go (TUG), and Long-Term Acquisition (LTA).

AMPPRO®

The Amputee Mobility Predictor (AMP) is an objective in-clinic examination in which the patient needs to perform several tasks, including static and dynamic sittings; standing activities; and transfer and gait skills of progressing difficulty. In the case of AMPPRO®, this questionnaire aims to measure the functional status of a prosthetic user [8].

Most of the questionnaire items offer three scoring options [14]:

- **0**: inability to complete the task;
- **1**: minimal level of achievement or the need for help in completing the task;
- **2**: full independence or mastery of the task.

At last, the sum of all items is computed to yield the AMPPRO score, which ranges from 0 to 47. This score can assist in determining the patient's functional level, or K level, as there appears to be some correlation between the two values (Table 2.1) [15].

Table 2.1: Correlation between AMPPRO® scores and K levels

K level	**K0**	**K1**	**K2**	**K3**	**K4**
AMPPRO® Scores range	0-14	15-26	27-36	35-42	43-47

PLUS-M™

Prosthetic Limb Users Survey of Mobility (PLUS-M™) is a self-report tool designed to assess mobility in adults with lower limb amputation. It measures a prosthetic user's ability to move independently, encompassing actions from simple domestic movement to outdoor activities. The described actions pertain to locomotion (i.e., movement in a continuous, repeating pattern) and/or postural (i.e., movement from one position to another) transitions. Nevertheless, the test doesn't measure unintentional movements, assisted movements, or mobility with seated or wheeled devices.

The PLUS-M™ test is available in two versions: PLUS-M™ 7-item short form and PLUS-M™ 12-item short form; the latter includes all items of the 7-item version. Since Adapttech's Motio StepWatch™ System only employs the PLUS-M™ 12-item short form, the following description will focus only on this version.

Each item of this questionnaire is scored from 1 (unable to do) to 5 (without any difficulty), depending on the patient's answer. The sum of all these responses results in the raw score, ranging from 12 to 60. Then, the raw score is converted into a T-score through a Look Up table developed by the University of Washington using compiled data from a development sample of 1091 unilateral lower limb prosthetic users [16].

The obtained T-score, which varies from 21.8 to 71.4, allows a comparison of the patient's mobility to those reported by the development sample or by specific subgroups, such as level of amputation, etiology of amputation, gender, and age. But generally speaking, a higher T-score translates into a higher functionality and greater mobility [8].

Timed Up and Go (TUG)

Timed Up and Go (TUG) assesses a person's mobility by testing both static and dynamic balance [8]. The procedure requires the patient to wear their regular prosthesis and to identify a 3-meter line on the floor. Then, the steps to perform the TUG test are:

1. Patient begins sitting back in a standard armchair;
2. On the word "Go", they must stand up from the chair and walk to the line on the floor at their normal pace;
3. At the end of the line, the patient must turn and walk back to the chair at their normal pace;
4. Patient ends the test sitting down again;
5. In the meantime, the physician must record the duration of the test and annotate some observations.

The observations annotated by the physician are related to the patient's postural stability, gait, stride length, and sway. They include the following checklist:

- Slow tentative pace;

- Loss of balance;
- Short strides;
- Little or no arm swing;
- Steadying self on walls;
- Shuffling;
- En bloc turning;
- Not using assistive device properly.

Lastly, the test duration correlates with the patient's fall risk; if an individual takes longer than 12 seconds to complete the task, it is suggestive of a fall risk [17].

Long-Term Acquisition (LTA)

Following the acquisition period, the collected step data from the Long-Term Acquisition test is divided into four distinctive groups under the "Daily Activity Highlights". Then, this data is separated into two distinct contexts: *daily average* and *best*. The *daily average* corresponds to a mean of the gathered data per day, providing an overview of the patient's typical activity level and, as a result, valuable information for the clinician to understand the patient's regular accomplishments. In the meantime, the *best* indicates the patient's highest activity level recorded on their "best day", allowing the clinician to acquire insights into the patient's peak performance and assess their potential [8].

Furthermore, the four distinctive groups from the "Daily Activity Highlights" are:

- **Steps**: provide valuable insights into the functional capabilities and mobility improvements over time of prosthetic users;
- **Top Speed**: aids in assessing the functional capacity and mobility of a lower limb prosthetic user, while providing information about their ability to move swiftly, efficiently, and safely in different contexts. It can also serve as a relevant measure of the progress of individuals undergoing prosthetic rehabilitation;
- **Top Cadence**: also known as maximum step rate, refers to the maximum number of steps taken in a minute. It monitors their progress over time and helps to ensure that the prosthetic device is appropriately designed and fitted;
- **Maximum Continuous Walking Distance**: corresponds to the longest distance or period of continuous walking. It provides important details regarding the patient's aerobic capacity and possibility of engagement in their community.

Aside from the "Daily Activity Highlights", the Long-Term Acquisition also displays the "Overall Activity Description", composed of two sections:

- **Active Days**: display how many days the Motio StepWatch™ System recorded step activity during the entire acquisition period;
- **Active Time**: show time spent at different intensity levels of walking activity, relevant to assess walking performance. The levels of intensity are low (1-30 steps per minute), moderate (31-80 steps per minute), and high (more than 80 steps per minute).

During the Long-Term Acquisition, the Motio StepWatch System can also conduct three blind evaluations: Two-Minute Continuous Walk (2MCW), Six-Minute Continuous Walk (6MCW), and 10-Meter Continuous Walk (10MCW), in which the patient needs to walk for 2 minutes, 6 minutes, and 10 meters, respectively. Since they eliminate the awareness of being tested, they acquire the most accurate and unbiased depiction of a patient's functional level. However, unlike the validated outcome measures, the setting is not controlled in these blind tests [8].

2.1.2 How Motio App Works

During the patient's first appointment, the clinician creates a *New Patient* in the Motio App (Figure 2.1) and enters the required information, including name, gender, date of birth, amputation level and side, weight, height, and health insurance company (Figure 2.2). After selecting the patient, they create a *New Appointment* with the required fields (Figure 2.3). An *appointment* is mandatory every time a clinician wishes to perform a long-term acquisition or any other available outcome measure.

On the *appointment* page, the clinician can select which *outcome test* to perform between AMPPRO®, PLUS-M™, TUG, and Long-Term Acquisition (Figure 2.4). Each test requires them to fill out forms and answer questions. Then, the results of each *outcome test* are recorded on the corresponding *appointment* page. In the end, a clinician evaluation, i.e., K level given by the clinician, is required to generate the Motio Functional Report, which includes the results of all performed *outcome tests*.

All these data are then stored in an SQL relational database inside Adapttech's server.

2.1.3 Feedback (Issues, Improvements, Complaints, and CAPA) System

As part of the Post-Market Surveillance process of Adapttech, all feedback received is logged by Adapttech's collaborators in the Report Motio Feedback App and can originate internally or from partners and clients. The feedback is, then, categorized into:

- **Issues**: broad term to describe any problem, task, request, or inquiry that needs to be tracked, managed, and resolved when developing [18];
- **Improvements**: refers to any feature or component that needs to be improved;
- **Complaints**: deficiencies related to the identity, quality, durability, reliability, usability, safety, or performance of the Motio StepWatch or the Motio App that affect their performance. Can be internal or external [19];

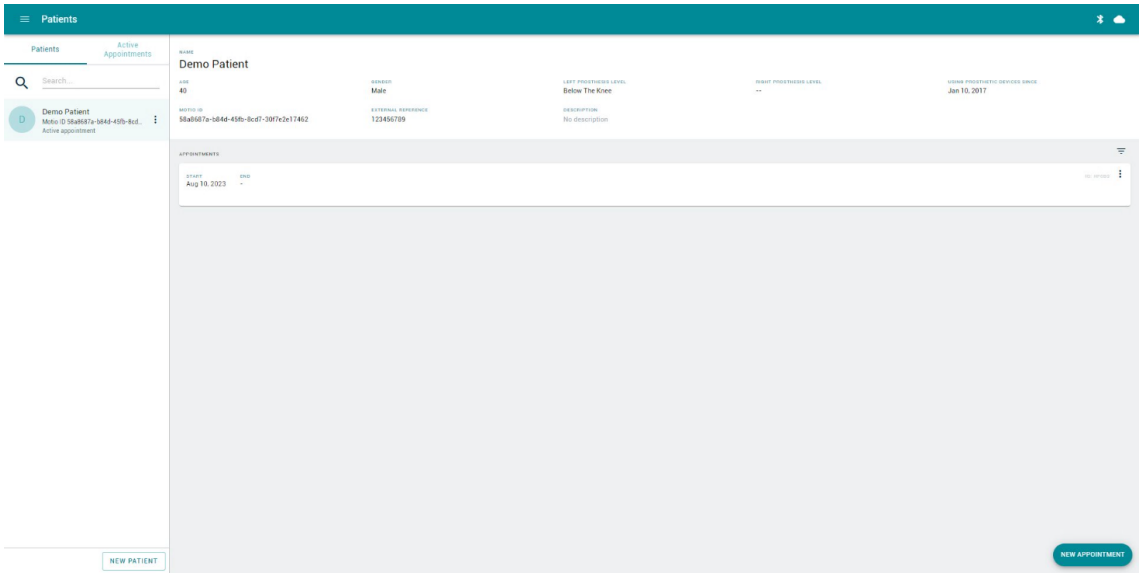


Figure 2.1: Motio App - Patients page

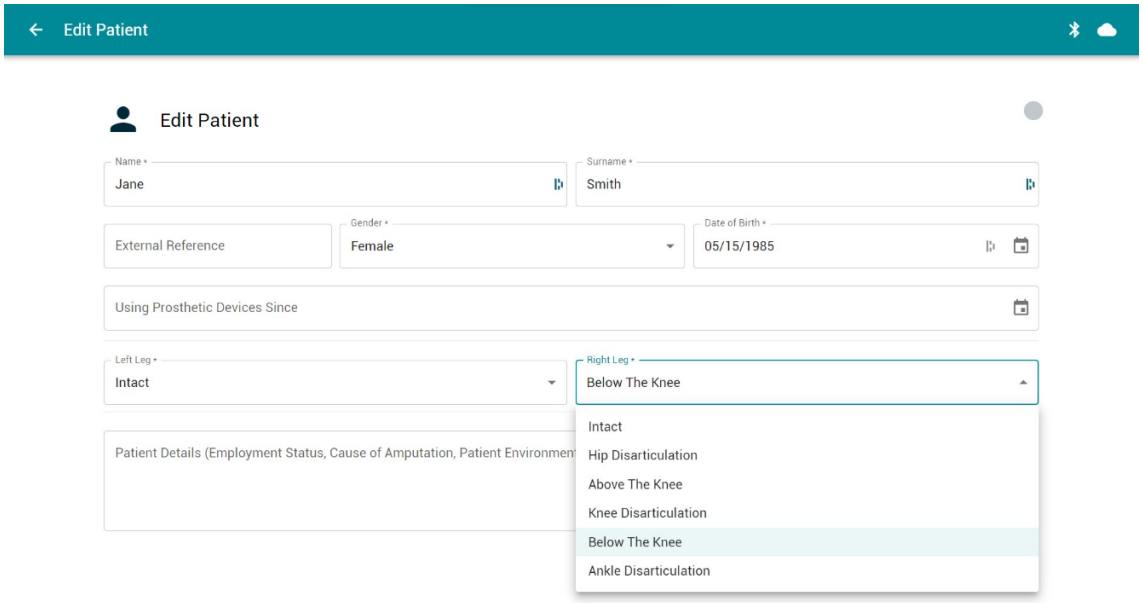


Figure 2.2: Motio App - Create new patient page

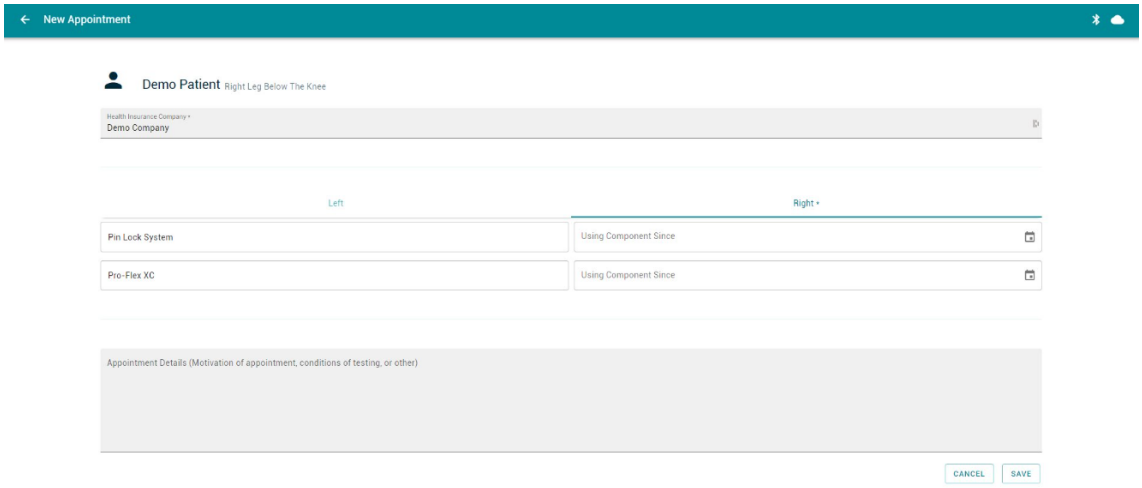


Figure 2.3: Motio App - Create new appointment

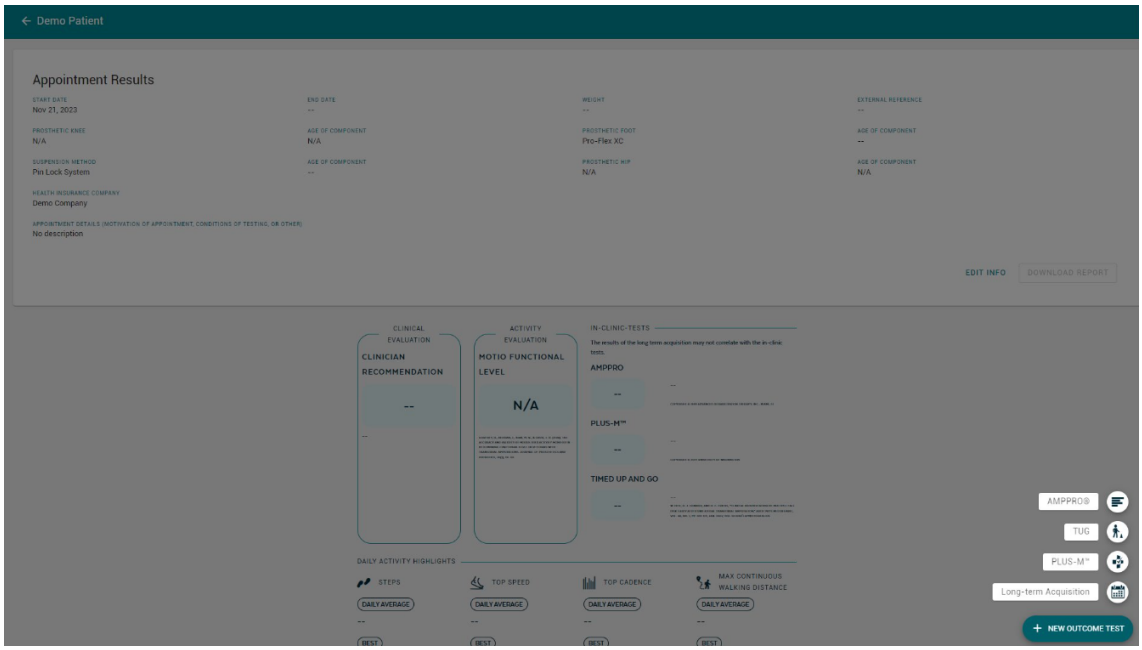


Figure 2.4: Motio App - Create new outcome test on the appointment page

- **CAPA (Corrective and Preventive Action)**: most severe feedback, created after three reported complaints addressing the same topic to prevent recurrence and occurrence. Seeks to eliminate the cause of a potential non-conformity or other undesirable situation. A non-conformity may include a problem or malfunction with the device [19].

2.1.3.1 How the Feedback System Works

When a user wants to report an issue, improvement, or complaint, or start a CAPA, they must access the Report Motio Feedback app, a Microsoft Power App designed for this purpose. On the Report Feedback page, the user needs to fill out the following information (Figure 2.5):

- **Issue**: name of the issue, improvement, complaint, or CAPA
- **Feedback Type**: Issue, Improvement, Complaint, or CAPA
- **Severity**: Low, Medium, or High
- **Safety related?**: On/Off
- **Product/system**: Motio StepWatch, Motio App, or Motio Backend
- **Main area** (Optional)
- **Description**
- **Date of finding** (Optional)
- **Source of feedback**: location from where the ticket came
- **Person who received the feedback**
- **Product Serial Number** (Optional)
- **Steps to reproduce the issue** (Optional)
- **Other information** (Optional)
- **Attachments** (Optional)

After the feedback was successfully submitted, the Microsoft Power App automatically creates a list item with the input information on Adapttech's Microsoft Sharepoint List. Subsequently, the list item is automatically converted using Microsoft Power Automate to an item on Adapttech's Microsoft Planner, in which both the list and planner items share an ID.

The reason for using two platforms for feedback management is that while the Microsoft Sharepoint List allows customizable fields, thus providing all of the information inserted by the user, the Microsoft Planner is more strict, being capable of displaying only the feedback type and product/system. However, the latter has some built-in task management information—not found in the Lists, namely:

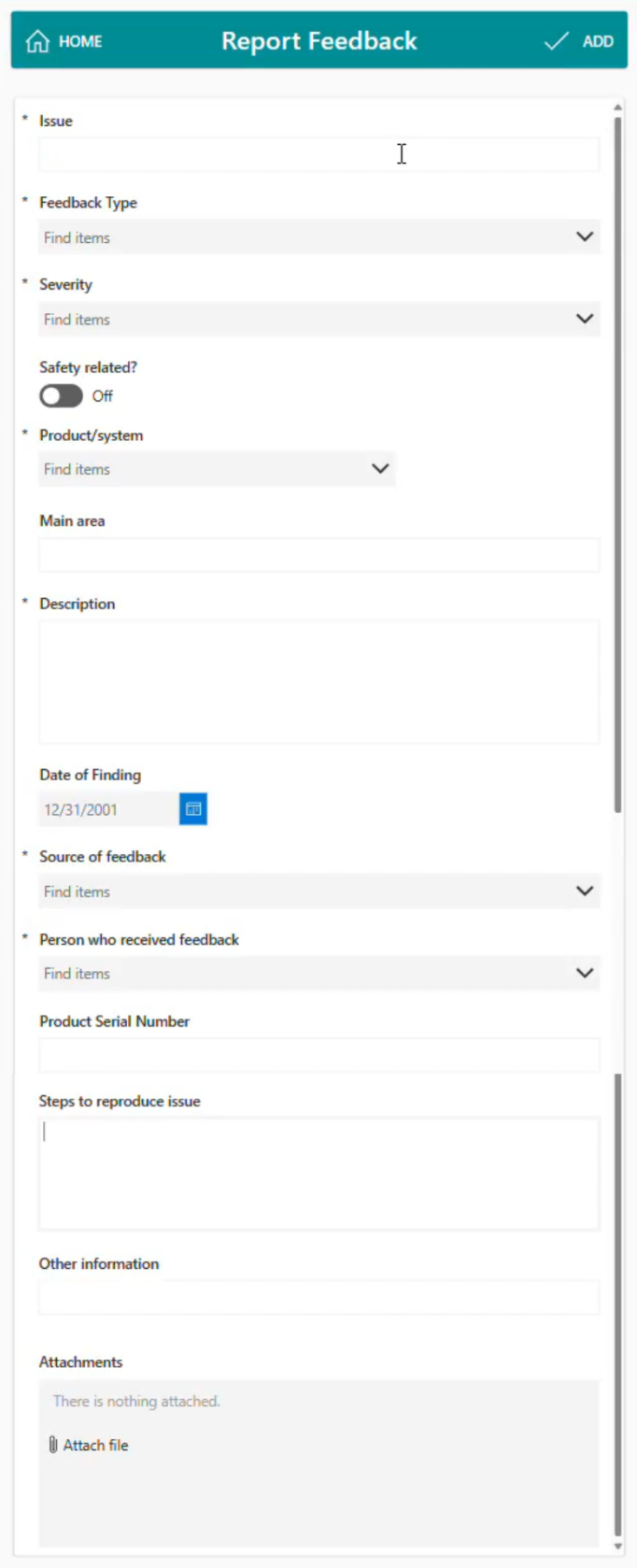


Figure 2.5: Report Feedback Page on Report Motio Feedback App

- **Progress**: Not started, In progress, or Completed
- **Priority**: Urgent, Important, Medium, or Low
- **Start date**
- **Due date**
- **Repeat**: Does not repeat, Daily, Weekdays, Weekly, Monthly, Yearly, or Custom
- **Checklist**

Additionally, Microsoft Planner divides the items into Product buckets for better visualization.

2.2 Business Intelligence

The term "*Business Intelligence*" first appeared in Richard Millar Devens' Cyclopædia of Commercial and Business Anecdotes in 1865. He employed it to explain how banker Sir Henry Furnese profited before his competitors by learning about and acting upon his surrounding information [20].

The first applications of decision support systems, known as Management Information Systems (MISs), arose around the end of the 1960s. The goal of MISs was to assist managers plan and optimize specific objectives and tasks, including production planning or investment optimization [21]. Over the years, a wide range of new applications emerged with increasingly different names and characteristics, as shown in Figure 2.6 [22] [23].

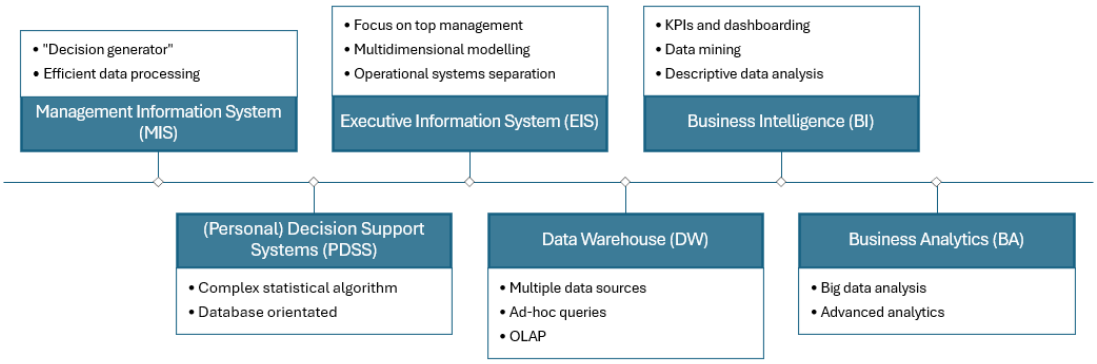


Figure 2.6: Evolution of Decision Support Systems [24]

It was just in 1989 that Howard Dresner from the Gartner Group adopted the term for decision support systems, executive information systems, and management information systems to describe "concepts and methods to improve business decision-making by using fact-based support systems" [21].

Nowadays, literature frequently defines *Business Intelligence* (BI) in two ways: a technological and a strategic perspective. The former refers to a broader "category" of tools that can

assist users in decision-making by collecting, storing, analyzing, and accessing data. In contrast, the strategic viewpoint not only defines it as a set of tools but also as any measurement method that involves the systematic collection and preparation of data, as well as their combination with knowledge developed in organizations to improve the efficiency of their processes [25].

2.2.1 Why Business Intelligence?

Business Intelligence remains a top priority for any company that needs to know what is happening, what is expected to happen, and what measures to take to obtain the best results [26]. Furthermore, its capabilities play a crucial role in helping a business handle the fast-growing volume, velocity, and variety of data produced internally and externally [13]. Especially in cases like this, of abundant data, the use of analysis tools in management practices is seen to have a positive impact on the organization's performance [27] [28] [29].

Concerning data, *Business Intelligence* allows [30] [31]:

- Integration of heterogeneous information and its correlation;
- Analysis of large amounts of data and the generation of comparative reports;
- More efficient access to higher quality data and information.

Other aspects that make BI stands out are the ability to perform simulations and predictions; produce faster and more detailed analyses; improve decision-making; and create dashboards to monitor indicators, such as KPIs (Key Performance Indicators) [30] [32] [31].

All these benefits not only have a positive financial impact since they allow income increase and/or cost reduction but also enhance the organization's ability to adapt, enabling a fast response in the event of market or internal changes [31] [30]. In addition, this strategy makes the company learn more about its customers, allowing it to tailor its offers of products and services that will maintain current customers while attracting new ones [33]. Although this notion is previously known, BI makes it even easier and faster to implement.

In summary, the importance of BI within an organization comes with the increased efficiency in accessing and analyzing data, which aids decision-making and, consequently, results in achieving objectives more easily and quickly.

2.2.2 Structure of a BI Application

A typical BI application is mainly composed of data collection, preparation, analysis, and visualization, as seen in Figure 2.7 .

Data Collection and Preparation

The first step of building a BI application is to collect the data from various sources and integrate them using Extract, Transform, and Load (ETL) tools to obtain a standard, homogeneous Data Warehouse (DW). This step must be performed with high regard since it serves as the data

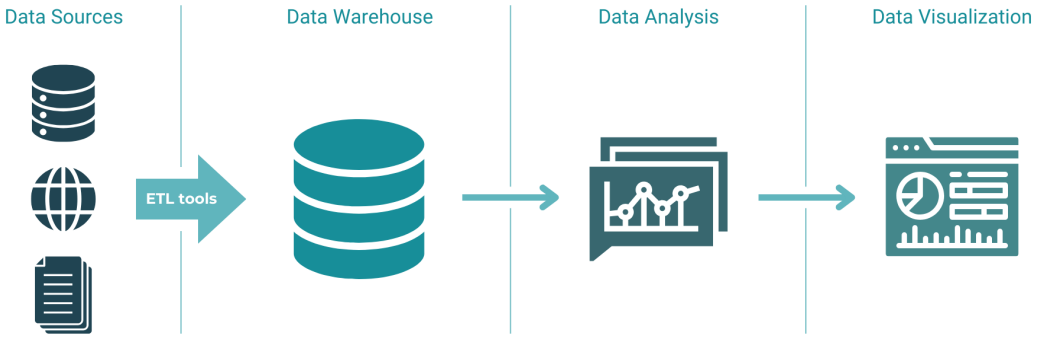


Figure 2.7: *Business Intelligence* architecture

source for the application and, consequently, is partially responsible for the implementation success [24] [34].

The ETL phases are characterized by:

- **Extract phase**: data reading and relevant data extraction;
- **Transform phase**: data formatting, if necessary, of the previously extracted data. It involves data correction and cleansing, such as removal of incorrect or duplicate samples, creation of missing data, and correction of data errors [35];
- **Load phase**: data transfer to its final destination, DW. It includes regular updates from the data sources to provide up-to-date data for following operations [36].

Data Warehouses receive and store data and are managed by one or more servers. Therefore, it can be defined as a database that serves as the construction foundation of decision support systems and BI architectures [37].

Data Analysis and Visualization

The data analysis and visualization, also known as the end user layer, provide the user with quantitative and statistical analyses; predictive and explanatory models; and fact-based management to guide actions and decisions using the loaded data in DW [38] [39].

Nowadays, there are several tools available for data analysis and visualization, namely Online Analytical Processing (OLAP), Data Mining (DM), ad-hoc queries, visualization tools, and dashboards [40] [41] [38]. These tools allow the user to determine the outlines of their analysis and analyze the findings [42].

Online Analytics Processing (OLAP) delivers a multidimensional viewpoint of data, including filtering, aggregation, and other operations. Data Mining (DM) is the process of discovering significant information, such as anomalies, trends, structures, changes, and relationships, in large volumes of data stored in the database, while also providing predictive models [43]. An ad hoc query corresponds to a one-time or specific query executed in real-time to address spontaneous information needs [44]. More about dashboards and data visualization are described in Section 2.3.

2.2.3 BI Implementation Success

The organization needs to succeed both technologically and organizationally to reap the benefits of a BI application [13].

Achieving technological success requires that technical platforms and standardized databases communicate with each other seamlessly, i.e., correctly defining user access to tools and integrating BI systems with the organization's system. To achieve this, factors like high-quality, up-to-date, and integrated data; highly adaptable tools; BI systems integration with other systems of the organization; and correct definition of access of users to the tools are essential [41] [30] [45].

Meanwhile, there must be a structure inside the business capable of supporting a BI approach to have organizational success [13]. In this case, a top-down approach is considered the most suitable one, as it ensures the support and commitment from high-ranking roles to encourage change within the company and provide the necessary financial and human resources [30] [31] [45] [46]. Alongside that, it is crucial to identify the business' needs and priorities and use them as the cornerstone of the implementation process [45].

2.2.4 Current Solutions

Regarding current solutions, Gartner created the Magic Quadrant, a product that analyses a specific business market—in this case, *Analytics and Business Intelligence* (ABI). It provides a broad overview of the leading market corporations in this niche based on their completeness of vision and ability to execute, along with a quick verification of their technological development level[1]. The Magic Quadrant of 2023 for Business Intelligence is pictured in Figure 2.8.

As seen, the graphic divides the companies into four types of technology providers: Leaders, Visionaries, Niche Players, and Challengers. Within the *Business Intelligence* scope, these types have the following meanings [47]:

- *Leaders*: have a solid understanding of product capabilities and prioritize customer success while offering understandable pricing models that support value proof, progressive purchases, and enterprise scale. Their products are considered easy to buy and use, requiring little technical knowledge. Despite the market's rapid evolution and constant innovation, Leaders maintain a robust roadmap to secure their market leadership, safeguarding buyers' investments;

- *Visionaries*: offer a distinctive, modern product with extensive functionality in their areas. Though innovative, they may lack broader functionality, score lower in customer experience, operations, and sales execution, and may face challenges in scaling and sustaining consistent execution;

- *Niche Players*: excel in specific sectors or industries but may lack innovation or performance compared to other vendors. They may also have a reasonably broad product and, at the same

[1] https://www.gartner.com/en/research/methodologies/magic-quadrants-research. Accessed on 2024-05-16



Figure 2.8: Gartner Magic Quadrant for Analytics and Business Intelligence technologies in the year 2023 [47]

time, limited support capabilities or a relatively small customer base—only a specific region or industry, for example;

- *Challengers*: are well-positioned to succeed in the market but may face limits in specific areas or lack a cohesive approach. They might also lag behind leaders in marketing, sales, geographical reach, industry content, and innovation.

Among all the displayed companies, it is noticeable that Microsoft is nearly isolated at the Leaders quadrant while being the best in "Ability to execute" and "Completeness of vision". This notable result is mainly due to its product, *Power BI*, one of the most widely used BI tools nowadays.

An overview of these solutions is displayed in Table 2.2.

2.3 Dashboards and Data Visualization

As a result of the daily volume accumulation of data, organizations are increasingly experiencing the phenomenon called information overload. It corresponds to circumstances in which management bodies are constantly overwhelmed with disorganized and badly presented information that distracts them rather than guiding and assisting them [41].

For that reason, data visualization is becoming progressively more important to BI businesses. Considered an interdisciplinary field that combines technology, psychology, decision-making, and art, it consists of the visual representation and assessment of data in any size, origin, or organization. It is the last step of the BI pipeline, providing the user with a visual and interactive data exploration experience.

Since direct visualization of data improves understanding, problem-solving, and decision-making abilities, key indicators in BI systems should be displayed with visual suggestions that easily guide users to areas of interest or contrast and allow them to explore the human visual system by extracting additional information [41] [48].

In recent years, the feature known as self-service has been increasingly important in the BI market. It refers to the ability of casual users to prepare and analyze data using easy-to-use tools without relying on professionals and their more complex analyses [49]. The more users can interact with the BI tools, the more evenly distributed information is within a company, which enables the generation of different interpretations and allows more entities within the company to acquire a common understanding of these same interpretations [50].

First developed in the 1980s, today's dashboards are one of the most noteworthy visual communication mediums. They provide a robust and distinctive solution that centralizes the most relevant information onto a single screen, delivering an overview of the company's present status to all stakeholders and allowing for easy monitoring at a glance [51]. As a result of using dashboards, the organization transmits an atmosphere of openness and transparency [23].

Since dashboards gather, summarize, and present information from various sources, it is expected that dashboard implementation will improve decision-making by strengthening users'

Table 2.2: Overview of current Business Intelligence solutions listed in Gartner Magic Quadrant [47]

Tool name	Company	Magic Quadrant	Characteristics	Advantages	Disadvantages
Quick BI	Alibaba	Challenger	Largest public cloud platform provider in the Asia/Pacific market. Offers data preparation, visual-based data discovery, interactive dashboards, and reports. Available as a SaaS on Alibaba Cloud's infrastructure.	• Digital workplace-oriented analytics and collaboration • Composable analytics • Data literacy program	• Not an investment focus • Complex competition environment • Geographical presence and lower market momentum
Amazon QuickSight	Amazon Web Services	Challenger	Largely sold into the AWS customer ecosystem. It has mature APIs and software development kits (SDKs), and serves clients of all sizes, vertical industries and from different regions.	• Competitive pricing • Integrated global cloud service • Serverless architecture enables scalability	• Limited emphasis on emerging critical capabilities • Feedback on customer experience • Lack of business application ecosystem
Domo	Domo	Challenger	Appeals to executive business users, analytic developers, business analysts, and data scientists. It has a niche in marketing analytics and is popular with organizations without a robust data warehouse, especially in small and midsize enterprises. Allows fast time to value by accelerating the data ingestion, integration, and transformation process.	• Multipersona support • Scale with control and security • Driving innovation	• Lack of a broader application ecosystem channels • Limited geographic presence • Premium pricing model
GoodData	GoodData	Niche Player	Strong niche in customers interested in building and integrating a metrics store as a universal semantic layer in their environment. Offers robust APIs and SDKs for front-end customization and integration to third-party ABI, data science tools, and other applications.	• Headless vision • Composability pioneers • Availability of third-party resources	• Limited to mature buyers • Uncertain evolution of metrics stores • Traditional ABI deployments are limited
Looker	Google	Challenger	Cloud-architected platform that offers highly governed analytics (self-service visualizations and dashboards) and a code-first semantic modeling layer based on LookML. Its developer-focused platform provides a version-controlled, collaborative framework to build BI applications. Direct access to Google Sheets, Salesforce (Tableau) and Microsoft Power BI.	• Governed self-service semantic layer • Decision support and workflow resources • Google Cloud Platform (GCP) integration	• Developer/data engineer focus • Incomplete analytical modeling loop • Limited native augmentation
IBM Cognos Analytics	IBM	Visionary	Offers consistent core capabilities across hybrid, multicloud, public, and on-premises deployments. Can be deployed on AWS, Azure, GCP, and IBM cloud. Enhancements, like time entity filtering improved, multivariate forecasting, time series library, natural language generation, and search. Integration with Slack and Microsoft Teams.	• Comprehensive offering and supported roles • Flexible deployment options • Roadmap for applying analytics everywhere	• Diminishing market awareness • Lack of sales adoption drivers´ • Price versus cloud vendor alternatives
Incorta	Incorta	Niche Player	Capable of quickly model data from enterprise application (SAP, Oracle, Salesforce, and JD Edwards). It has data warehouse automation, data acquisition, management, and visualization/reporting. Added data destinations, which provide the ability to set up third-party to systems to receive data outputs.	• Agile operational reporting • Security and scalability • Strong appeal for low-maturity customers	• Lack of functionality for augmented consumers • Competition from packaged analytic applications • Lack of market awareness
Power BI	Microsoft	Leader	Well-above-average functionality. It has new features, like a metric-tracking capability that enables teams to align their goals and key priorities in a visually collaboration, and low-code data marts for easy access to self-serve managed relational analytic solutions.	• Alignment with Microsoft 365, Teams and Azure Synapse • Price/value combination • Power portfolio and product ambition	• Governance of content creation and publication • Limited headless, open architecture • Azure as the only deployment option
MicroStrategy	MicroStrategy	Challenger	Rich BI and reporting functionality, including hyperIntelligence that dynamically identifies predefined insights within existing applications. Excels at scalability, manageability, and governance. Wide geographic presence, present in 37 countries.	• Stability of integrated product • Open platform • Enterprise reporting and governance	• Absence of surrounding data or application ecosystem • Poor augmented analytics capabilities • Lack of product differentiation
Oracle Analytics Cloud	Oracle	Visionary	Leadership in database management and comprehensive cloud business applications. It has 17 prebuilt data science models that reduce customers' advanced analytics delivery. Leading-edge vision with innovative new features.	• Enterprise cloud data and analytics • Augmented capabilities throughout • Comprehensive data management	• Insufficient momentum in a crowded market • Oracle application-centric • Pricing
Pyramid Analytics	Pyramid Analytics	Visionary	Completely browser-based and cloud-platform-agnostic, allowing for hosting in AWS, GCP, Oracle Cloud or Alibaba. Integrated suite across the data life cycle, out of the box. ML-based data preparation and wrangling. Sharing via business-user-generated dashboards and reports in a low-code/no-code environment.	• Accessibility for nontechnical users • Augmented analytics at its core • Investment in community building	• Lack of market awareness • Less focus on developer-centric capabilities • Limited catalog interoperability and user enrichment

Table 2.2: Overview of current Business Intelligence solutions listed in Gartner Magic Quadrant [47]

Tool name	Company	Magic Quadrant	Characteristics	Advantages	Disadvantages
Qlik Sense Enterprise SaaS	Qlik	Leader	Cloud-agnostic vendor that has the highest level of partnership with AWS, Microsoft and Google, along with partnerships with Databricks and Snowflake. Client-managed product that can be deployed on-premises or in the customer's cloud. Provides automatic forecasting, clustering, and key driver analysis. Creates AI-generated dashboards with natural language inputs and automated dashboard layouts.	• Composable analytics • Comprehensive data and analytics capabilities • Cloud-agnostic	• Absence of surrounding data or application ecosystem • Prioritizing data integration • Product licensing complexity
Tableau	Salesforce	Leader	Mainly focused on visual-based exploration that enables business users to access, prepare, analyze, and present findings in their data. Provides augmented analytics capabilities for analysts and citizen data scientists. It has global operations and serves clients of all sizes.	• Tableau is considered a skill • Agnostic and flexible position • Analytics experience	• Market momentum and support concerns • Salesforce-centric innovation • Uncertain impact on-premises clients
SAP Analytics Cloud	SAP	Visionary	Cloud-native multitenant platform with a broad set of capabilities across data visualization, reporting, and augmented analytics. Improved reporting, data storytelling, analytics developer capabilities, UX, data integration, and modeling capabilities. Offers composable analytics apps to lines of business.	• SAP data integration and connectivity • Augmented capabilities and closed-loop analytics • Analytics accelerators for SAP business apps	• Average product capabilities • Limited adoption outside of SAP ecosystem • Cloud-only deployments
SAS Visual Analytics	SAS	Visionary	Offers visual and augmented data preparation, ABI, data science and machine learning, and AI solutions. Integration with Microsoft Azure, Databricks, AWS, and Synapse.	• Unified platform • Market-leading data visualization and automated insights´ • Global reach with industry solutions	• Limited adoption • Bundled pricing • Limited natural language query (NLQ)
Sisense	Sisense	Visionary	One of the only nonhyperscaler vendors that can equally support both self-service augmented analytics and embedded use cases in a hybrid cloud environment. Support the ramp-up in demand for embedded analytics.	• Strong embedded offering • DevOps-first principles • Innovation speed	• No metrics store integration • Lacks a surrounding data or application ecosystem • Community support
Tellius	Tellius	Visionary	Used by consumers, analysts, and citizen data scientists. Insights using "What?", "Why?", and "How?" interfaces. It has industry-specific market share calculations directly from its´ search interface for consumer packaged goods and pharmaceutical customers.	• Rapid time to insight with NLQ and automated insights • Performance and scalability • Multipersona support	• Lack of market momentum • Product gaps • Emerging geographic strategy
ThoughtSpot	ThoughSpot	Visionary	Consumer-friendly UX, ability to handle complex analysis, and scalability to analyze large datasets. Its software guides the user to more granular insights. Continuously monitoring of data to produce more personalized insights. Runs its search and automated insight workloads via direct queries of databases to minimize data movement. Targeted for smaller teams and organizations with a consumption-based pricing model.	• Market recognition • Augmented consumer-centric product offering´ • Embedded product offering strategy	• Narrowing functional differentiation • As a pure-play, it lacks a full data and analytics stack solution • Smaller customer installed base
TIBCO Spotfire	TIBCO Software	Visionary	Combines visual exploration and data science, with strong support for streaming, including the capability to combine real-time and static data. It can also perform predictive models on streaming data. Provides 45 prebuild data functions for business users to perform advanced analytics	• Support for data scientists and citizen data scientists • Deployment flexibility and autoscaling • Prebuilt domain application	• Limited market momentum • Steep learning curve • Product feature gaps
Zoho Analytics	Zoho	Niche Player	Mainly focused on data preparation, data visualization, and its marketplace of prebuilt analytical apps for business analysts. Geographically diverse operations and clients tend to be small to midsize companies.	• Unified business analytics • Composable analytics vision • Zoho oopportunities	• Product capability gaps • Lower market momentum • Lack of advanced analytics

cognition and leveraging their perceptual capacities [52]. Additional advantages of dashboards are [41] [53]:

- Allow performance monitoring;
- Provide a holistic knowledge of an organization;
- Can be used to establish objectives and strategies to be followed;
- Communicate information to important stakeholders.

However, every piece of information displayed in the dashboard must be associated with an action plan in case a metric falls outside the predicted range, so the user knows how to act [23].

The design of a dashboard is an essential component of its functionality since only a well-organized arrangement of components can effectively convey essential information to users and provide effortless access to the organization's details [54]. To develop an effective design, both functional and visual characteristics must be considered. The former refers to the visualization features, such as whether a dashboard has drill-down abilities, automated alerts, update frequency, and adaptability. Meanwhile, the visual characteristics relates to the concepts of data visualization, i.e., how efficiently and effectively information is presented to the user [51].

The difference between efficient and effective data visualization is that while efficient data visualization corresponds to the maximum amount of data absorbed in the shortest amount of time, effective visualization is about whether the perceived information reflects real information [52]. To obtain both, literature cites some suggestions:

- Appropriate use of visuals for the desired metrics, as certain graphics can bias the decision by producing visual illusions [54];
- Sparingly use of colors to highlight information, as excessive color can distract the user [52];
- Maximization of data-ink ratio, which measures the proportion between ink or pixels for data display and the overall amount of ink or pixels for the whole dashboard [52];
- Flexibility to implement improvements [52]
- Ability to display a variety of data formats and various aggregation levels [41];
- *Drill-down* feature, as it lets the user engage with the dashboard by selecting a number, a portion of a chart, or a filter and be returned with tickets sourced by the element pressed [18];
- Size must fit onto a single computer screen [55];
- Automated alerts to warn the user if a metric is out of range [55];
- Data as charts, traffic lights, relative rankings, lists (top, most recent, oldest), or absolute scores and metrics [55];

2.4 Issue Tracking Systems

An issue tracking system (ITS), also known as a support ticket, incident ticket system, trouble ticket system, or support ticket, is a type of computer program capable of handling and storing a list of issues and development tasks [56][57]. Additionally, these systems can keep track of changes between the various issue statuses and record the information during its transitions [18].

These platforms can be considered more than a centralized issue registry, as they incorporate time accounting, priority management, communication channels, supervisory procedures, and resource allocation. For this reason, they are commonly employed in collaborative environments, such as large-scale organizations, to share information across the team and result in a more productive performance.

In 2006, a study of developers' work habits revealed that developers at Microsoft fixed bugs and issues for nearly half of their working time [58]. Therefore, a system that can coordinate issue resolution among employees and teams would be extremely valuable for any company. Other benefits from issue tracking systems are software quality refinement; user satisfaction and team productivity enlargement; request accountability assurance; internal and external communication improvement; and cost reduction.

The life cycle of an issue represents all statuses, from creation to closure, that a ticket can own and is, thus, related to its workflow. Figure 2.9 illustrates a simplified form of it. The cycle starts with the creation of a ticket that includes essential issue information (who experienced it, the date of submission, a detailed description, attempted solutions, and other pertinent information), as well as a unique reference number for better tracking. Besides that, the ticket must have an urgency rating based on its overall importance, with low or zero rating as minor things that can be postponed as long as necessary [59].

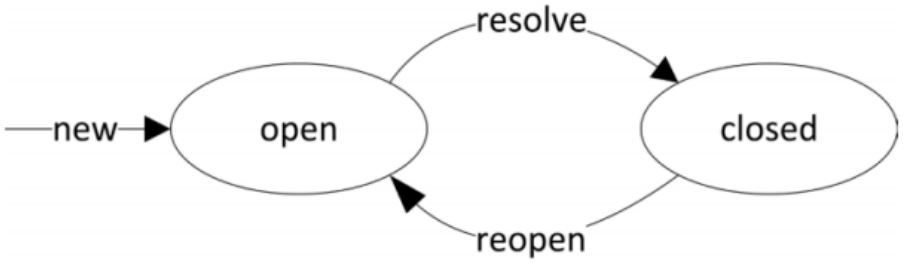


Figure 2.9: Simplified life cycles of issues [56]

2.4.1 Current Solutions

This subsection provides a brief description of the most popular ITS, highlighting the pros and cons of each solution. After assessing the available options and contrasting them with our needs, the objective is to have a holistic view of the ITS landscape and determine how this project might benefit from it.

Table 2.3 shows a summary of the analyzed solutions.

Table 2.3: Brief overview of four current Issue Tracking Systems

System	Creator	Launch date	Documentation/Reports	Customize workflow	Multiple projects
Bugzilla	Mozilla Foundation	1989	Yes, integrated reports and charts, scheduled reports by mail	Yes	Yes
Trac	Edgewall Software	2006	Yes, integrated wiki	Yes	Yes
Redmine	Jean-Philippe Lang	2006	Yes, integrated wiki, discussion forums, news blogs, e-mail integration, calendars, Gantt charts, export to PDF, export to Excel/CSV	Yes	Yes
JIRA	Atlassian	2002	Yes, dashboard with reporting widgets, Excel, Word, customizable charts, Confluence	Yes	Yes
Microsoft Planner	Microsoft	2016	Yes, visual charts and task boards, integration with other Microsoft 365 applications	No	Yes

2.4.1.1 Bugzilla

Considered one of the most well-known open-source ITS tool, Bugzilla[2] is a web-based bug tracking system and testing tool. It was released in 1998 by Netscape Communications to be employed originally on the Mozilla project. Even though its user interface (UI) is customizable, it is considered outdated and lacks usability and aesthetics.

Among its several features, some are particularly noteworthy, such as e-mail notification of new or updated issues, workflow customization, basic time monitoring, and a voting system in which users can vote for issues or features they want to see adopted[3]. In addition, Bugzilla is suitable for large organizations with multiple projects and users since it can manage enormous amounts of bug reports [60].

In addition to its unfriendly UI, Bugzilla is perceived as resource-intensive and complex to set up and configure, and has limited integration with well-known development and project management applications.

[2] https://www.bugzilla.org/
[3] https://wiki.mozilla.org/Bugzilla:Home. Accessed on 2024-05-13.

2.4.1.2 Trac

Another example of an open-source, web-based issue tracking system and project management tool is Trac[4]. It is compatible with all major operating systems and can be integrated with major version control systems like Subversion and Git.

Unlike Bugzilla, Trac is lightweight and delivers fast performance while its user interface combines simplicity with usability [60]. However, it does not offer as many features as Bugzilla. Furthermore, its personalization is deemed complex since it requires some technical expertise.

2.4.1.3 Redmine

Similar to the aforementioned tools, Redmine[5] is an open-source, web-based issue tracking and project management tool. It is cross-platform and cross-database, with a design resembling Trac, described previously.

Redmine's features contribute greatly to its use, such as multiple project tracking; configurable role-based access control; a Gantt chart and calendar; news, documents, and files management; time tracking; and various version control system integration. Moreover, it is available in 49 languages and also includes plug-ins. However, its customization complexity is seen as one of its most significant downsides[6].

2.4.1.4 JIRA

Developed by Atlassian in 2002, JIRA[7] is a bug tracking, issue tracking, and agile project management tool. It is particularly famous for the large number of extensions and plug-ins, as well as for the extensive quantity of learning resources and documentation, both created by its community.

Regarding the user interface, it remains a user-friendly interface despite its acknowledged complexity. To make it even easier to use, plug-ins and easy customizations are still available to the user. Furthermore, JIRA offers fine-grained enterprise-level security, automatic creation of requests from e-mails, division into subtasks, time tracking, and the capacity to handle large amounts of reports [60].

The expense is a major setback in this case, since only companies with up to 10 users can use it for free[8].

2.4.1.5 Microsoft Planner

Even though it is not considered a traditional Issue Tracking System, Microsoft Planner[9] offers a straightforward, visual approach for teams to organize and track work, and hence issues. It is the chosen platform for Adapttech's feedback system, as explained in Section 2.1.3.

[4] https://trac.edgewall.org/
[5] https://www.redmine.org/
[6] https://www.redmine.org/. Accessed on 2024-05-13.
[7] https://www.atlassian.com/software/jira
[8] https://www.atlassian.com/software/jira/pricing. Accessed on 2024-05-13.
[9] https://tasks.office.com/

In essence, Planner is based on task containers, called plans. It provides a clear overview of the workflow with a Kanban board interface, where tasks are represented as cards that can be moved between customizable columns.

The integration of Planner with other Microsoft 365 apps, including Teams, Sharepoint, and Outlook, is unquestionably one of its strongest points. However, it lacks intricate workflow management features and sophisticated customization choices in contrast to feature-rich tools like Jira[10].

2.5 Summary

This chapter contained a more in-depth introduction to the Motio StepWatch System and its components, in which it is possible to highlight the storage of the Motio App data in an SQL database and the storage of feedback data in both Microsoft Sharepoint List and Microsoft Planner using a common ID.

From a more theoretical standpoint, it reinforced the importance of a Business Intelligence application for any company that wishes to improve its decision-making and, consequently, its performance. Moreover, there was also an introduction to Issue Tracking Systems (ITS), as the project's intended platform will also monitor Adapttech's ITS ticket information.

All these concepts are necessary during the implementation of the pipeline, described in the following chapter.

[10] https://www.nuclino.com/solutions/microsoft-planner-vs-jira. Accessed on 2024-05-21.

Chapter 3

Methodologies for Application Development

After introducing the concepts related to the Motio StepWatch™ System monitoring app, this chapter focuses on describing a typical application development pipeline, employed in this project. Figure 3.1 outlines its steps.

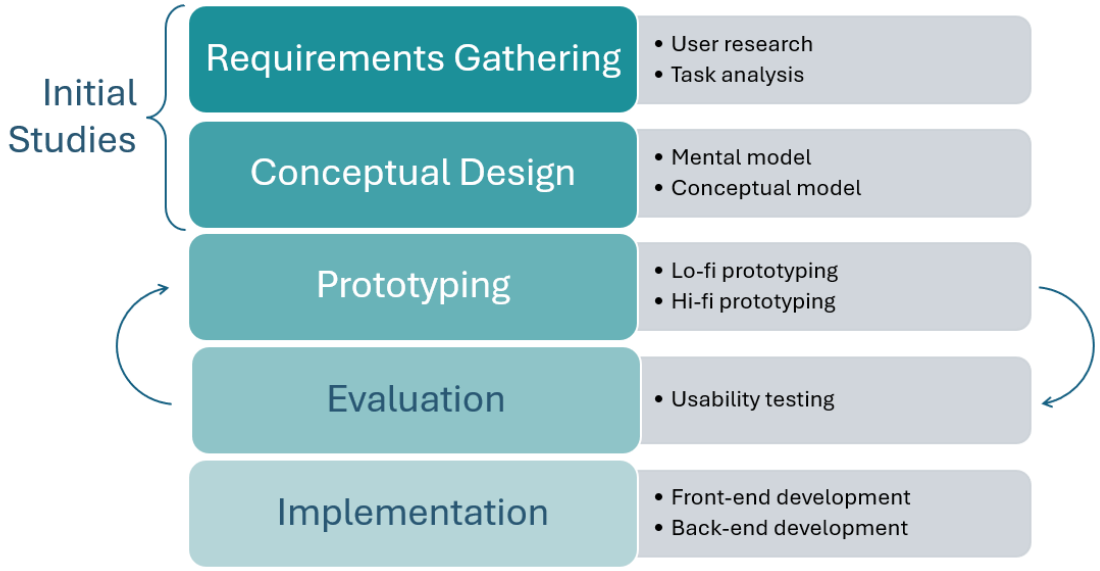


Figure 3.1: Typical application development pipeline

3.1 Design Principles and Practices

3.1.1 User-Centered Design (UCD)

User-centered design (UCD) is one of the most effective techniques for enhancing a system's usability. It requires an understanding of the system's users, the tasks they will carry out, and the

organizational, social, and physical environment where the system is placed [61]. Additionally, it emphasizes user involvement at every stage of the process, such as:

- Early in the design process, during the requirements gathering;
- During prototyping to test the design and alternatives;
- Just before the delivery of the product;
- During training/after delivery of the system.

The concept of user-centered design is so valuable that it has its own ISO standard: ISO 9241-210, also known as *Human-Centered Design Processes for Interactive Systems* [62]. This standard outlines and identifies the main principles and essential actions to achieve system usability in the human-(user-)centered design. In summary, the five main human-centered design principles are:

1. The active involvement of users and a clear understanding of user and task requirements;
2. User-centered evaluation drives and refines the design;
3. The design addresses the whole user experience;
4. Iteration of design solutions;
5. Multi-disciplinary design.

Moreover, the essential actions for human-centered design consist of:

1. Understand and specify the context of use;
2. Specify the user and organizational requirements;
3. Produce design solutions (prototypes);
4. Evaluate designs against requirements.

Even though it helps achieve a better design, involving users brings along a few challenges, including the potential to disrupt the design process, the possibility of conflicting objectives, and the high cost and time required [63]. In addition, the design of a system must consider user and technology limitations when trying to satisfy functional objectives.

3.1.2 Iterative Design

In the context of UCD, the iterative design would be an appropriate approach since it also prioritizes the user throughout the development process, as discussed above. It involves repeated cycles of designing, prototyping, testing, and refining the system, as Figure 3.2 illustrates. A typical workflow of iterative design is described below:

1. Analysis of users and their tasks;

2. Development of a low-fidelity (Lo-Fi) prototype;
3. Collection of user feedback for this Lo-Fi prototype;
4. Development of a high-fidelity (Hi-Fi) prototype with design refinements based on the previous feedback;
5. Feedback gathering and usability testing on this Hi-Fi prototype;
6. Continuous design refinement, feedback gathering, and testing until the UI achieves usability standards and user satisfaction.

Unfortunately, iterative design has some drawbacks, such as the struggle to locate users who fit the ideal profile and the lack of a method for designing screens and fixing usability issues once identified [64].

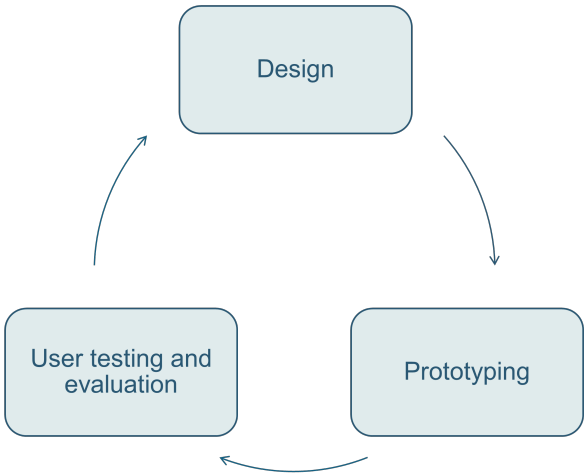


Figure 3.2: Iterative user interface design and evaluation process

3.1.3 Usability

As stated earlier, a user-centered and iterative design can improve the usability of a system, which is becoming increasingly necessary as technology becomes more affordable and the number of users grows. Part 11 of the ISO 9241 standard defines usability as "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use" [65]. In this standard, the terms effectiveness, efficiency, and satisfaction have the following meanings:

- *Effectiveness*: the accuracy and completeness with which specified users can achieve specified goals;
- *Efficiency*: the resources expended concerning the accuracy and completeness of the goals achieved

- *Satisfaction*: the comfort and acceptability of the work system to its users and other stakeholders.

Therefore, usability encompasses two main factors:

- **Fit for use** (or *functionality*): the system can accommodate the user's task in real life;
- **Ease of use**: composed of ease of learning, task efficiency, ease of remembering, subjective satisfaction, and understandability.

Having a good interface design with increased usability should lead to faster learning, fewer errors, less demand for hotlines, better motivation, faster daily work, less stress, more users, wider work areas, and, consequently, lower operating costs and economic loss for the company [64].

3.2 Initial Studies

Requirements gathering and conceptual design are components of the initial studies of application development. Section 3.2.1 describes the step of requirements gathering, which includes stakeholder identification, user research, and task analysis. Conceptual design using mental models and conceptual models is covered in Section 3.2.2.

3.2.1 Requirements Gathering

According to a 2000 article that investigated the causes of IT project failure, one of the main contributors to their lack of success is "unclear objectives and requirements" [66]. Therefore, the starting point of developing an application should be requirements gathering. However, first, the stakeholders, their activities, and how and where they work must be defined. This step is essential for understanding how stakeholders already interact with their work and pinpointing current difficulties and dependencies with their existing system.

Stakeholders include everyone who will use the system—in the case of an application, the users—and can be divided into primary and secondary. The former encompasses individuals or groups within an organization who will use the application directly. Meanwhile, secondary users refer to individuals or groups who affect or influence its development or who are affected or influenced in some way by the system [61].

Following their identification, research will be conducted on the stakeholders, also known as the step of *User Research*. There are two methods of user research: qualitative and quantitative [63].

- **Qualitative Research**: comprehension of user behavior and the reasons behind it.
  - **User interviews:** one-on-one talks with users to learn about their experiences, needs, and frustrations. They may have structured, semi-structured, or flexible format;
  - **Group discussions:** gathering of multiple perspectives from different users;

  - **Ethnographic studies:** observation of users in their natural environment to understand their interactions with the system and the circumstances around its use.

- **Quantitative Research**: numerical data that can be statistically analyzed. It helps in recognizing trends and patterns in user activity.

  - **Surveys and questionnaires:** useful for collecting demographic data and quantifying user preferences. The questions may be open or closed;
  - **Analytics:** tools or methods that track user behavior on an application. It outputs metrics that offer insights into how users engage with the system;
  - **A/B testing:** comparison of two versions of an application or a feature to determine which performs better.

After identifying the stakeholders and conducting user research, an in-depth look at users' activities and tasks is necessary. This step is called *Task Analysis* and also comprises a thorough understanding of the knowledge required to complete those tasks and accomplish the goals [61].

Cognitive task analysis and hierarchical task analysis are two frequently used methodologies. Cognitive Task Analysis seeks to comprehend tasks that require decision-making, problem-solving, memory, attention, and judgment. In contrast, Hierarchical Task Analysis focuses on breaking down a high-level task into subtasks. Moreover, there are some rules for gathering valuable user tasks, namely [64]:

- **Closure rule**: every user task must end with a meaningful goal. A proper task should run from trigger to closure, preferably without interruptions;
- **Session rule**: small, related tasks performed without breaks in the same work session should be grouped under a single description;
- **Domain level**: describe what the user and computer do together;
- **Don't program**: don't go into detail with how the task is performed, the specific sequence of steps required, under which conditions something is done, etc.

The resulting data from the task analysis can be translated into useful knowledge, as shown in Table 3.1.

At last, the requirements are easily identified after user research and task analysis, and they will serve as a guide of what the product should do. However, during the gathering of requirements, several constraints and trade-offs may emerge, affecting the progress. They are sometimes referred to as nonfunctional requirements and can relate to a variety of factors, such as costs, budgets, timescales, individual stakeholders' agendas, contradictory requirements, organizational policies, and available technology and its interoperability with other hardware and software [61]. Alongside that, there are other types of requirements, which are listed below [67].

- **Data requirements**: type, volatility, size/amount, persistence, accuracy, and value of required data;

Table 3.1: Knowledge obtained from the task analysis data

Data outputs	Converted knowledge
Description of physical, perceptual, and cognitive activities involved with each task	What experiences (personal, social, and cultural) users bring to the tasks
Task duration and variability	What users actually do to achieve their goals
Task frequency	
Task sequence allocation	
Task complexity	
Environmental conditions	How users are influenced by their physical environment
Data and information dependencies	What a computer system must do
Tools required for the task	What functionality the system must provide to support users in their goals and tasks
User skills, education, and training	How users' previous knowledge and experience influence

- **Environmental requirements** (or *context of use*): intended operating circumstances of the product. They consider four aspects, namely physical, social, organizational, and technical environments;
- **User requirements**: characteristics of the targeted user group;
- **Usability requirements**: usability objectives and associated measures for a particular system.

3.2.2 Conceptual Design

Designing screen layouts before deciding what functions they should support makes little sense. Therefore, a *Conceptual Design* of the system is necessary before the prototyping begins, as it refers to the process of establishing the underlying organization and structure of a user interface [61].

Before the conceptual design phase, the user already has a mental model in mind. Mental models reflect the user's understanding of what the system is and how it works based on their prior experiences and knowledge [61]. They are characterized by their constant evolution, lack of accuracy, ability to simplify a complex phenomenon, and occasional use of if-then-else logic in their representation. These models are important since users typically rely on them when using the system. Additionally, they can be a source of usability problems if the mental model does not match what the system actually does [64].

In contrast, a conceptual design is something the designer does to define the intended mental model while concealing the system technology. Alongside the mental model, other models are considered during conceptual design, such as [63]:

- **Conceptual model**;

- **System model**: representation of how the system works, in detail;
- **System image**: what the user "sees" and results from the physical structure of the development (documentation, instructions, labels, etc).

Focusing more on conceptual models, they function as a broader explanation of how the application works, as well as a plan for designing the system's details [64]. Conceptual models only refer to objects, properties, and actions in the task domain, which means they are expressed in terms of the intended users' tasks. Additionally, they should be as simple as possible while providing the required functionality and being task-focused, with a direct mapping between system and task operations. Some conceptual modeling strategies and techniques include entity-relationship diagrams, object-role modeling, and the Unified Modeling Language (UML).

3.3 Prototyping and Designing

While testing usability is easier in the final system, making modifications at that stage is more costly; hence, the necessity of the prototyping phase. Prototypes are experimental, primitive versions that are not intended for real use but rather for tests seeking to solve challenging problems [64]. A study revealed that conducting early usability testing on a simple prototype can reduce usability defects by 70% [68]. In addition, they present other purposes, including [61]:

- Check the application's usefulness;
- Allow users to test ideas and contribute to the design of the application;
- Test the viability of ideas with users;
- Validate and negotiate the requirements (i.e., to identify contradictory or lacking requirements).

There are two types of prototyping, low-fidelity and high-fidelity, as detailed in the following subsections.

3.3.1 Low-Fidelity (Lo-Fi) Prototyping

Low-fidelity, or Lo-Fi, prototypes are typically paper-based and can be used to illustrate design concepts, screen layouts, and design alternatives. Even though they might give users an idea of how the UI looks and feels, they can only provide limited detail regarding its functionality. Despite this, they still offer numerous advantages, such as affordability, ease of creation, and adaptability. Hence, issues found in this stage are less complicated to rectify as the design has not advanced too far.

The least complex low-fidelity prototype is the paper prototype, which consists of sketches or drawings depicting basic user interfaces and their interactions. It is useful for exploring and crystallizing design ideas in the early stages, as well as facilitating communication between the

designer and the users [61]. Since it is hand-drawn, this prototyping style has a significant disadvantage: it is time-consuming[1].

Another popular type of low-fidelity prototyping is the wireframe, as seen in Figure 3.3. It concerns rudimentary blueprints of the user interface that capture the system layout and structure while ignoring its visual and interactive aspects. Since wireframes can portray user interactions in a user flow, usability testing can be conducted on them[2].

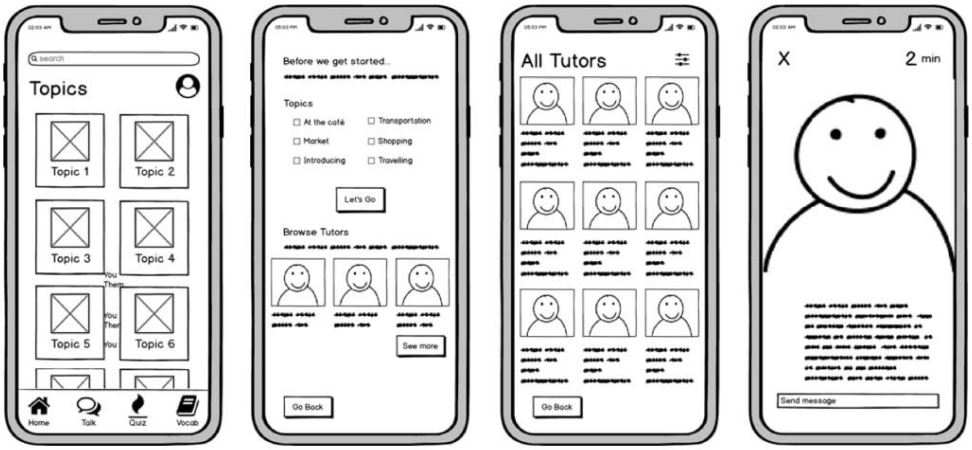


Figure 3.3: Example of wireframe[3]

3.3.2 High-Fidelity (Hi-Fi) Prototyping

In general, high-fidelity, or Hi-Fi, prototypes are functional, polished simulations of the system. They allow the user to experience the look and feel of the final system by displaying visual design details and genuine content. Moreover, its robust interactivity and functionality provide a more realistic user experience and facilitate advanced usability testing. Figure 3.4 illustrates an example of a high-fidelity prototype.

Among the current tools for creating a high-fidelity prototype, Figma, InVision, and Adobe XD are a few of the most popular ones.

Figma

A flexible and web-based application, Figma[4] allows the creation of several prototypes, from detailed wireframes to interactive ones, without needing any coding experience. It simplifies workflows by offering an extensive library of components, a sophisticated plugin ecosystem, powerful vector editing capabilities, and responsive layouts. Due to its robust collaborative aspect, Figma

[1] https://www.interaction-design.org/literature/article/prototyping-learn-eight-common-methods-and-best-practices. Accessed on 2024-06-02

[2] https://www.figma.com/resource-library/low-fidelity-prototyping. Accessed on 2024-02-06.

[3] https://bootcamp.uxdesign.cc/wireframes-5d63f9d760a8. Extracted on 2024-06-05.

[4] https://www.figma.com

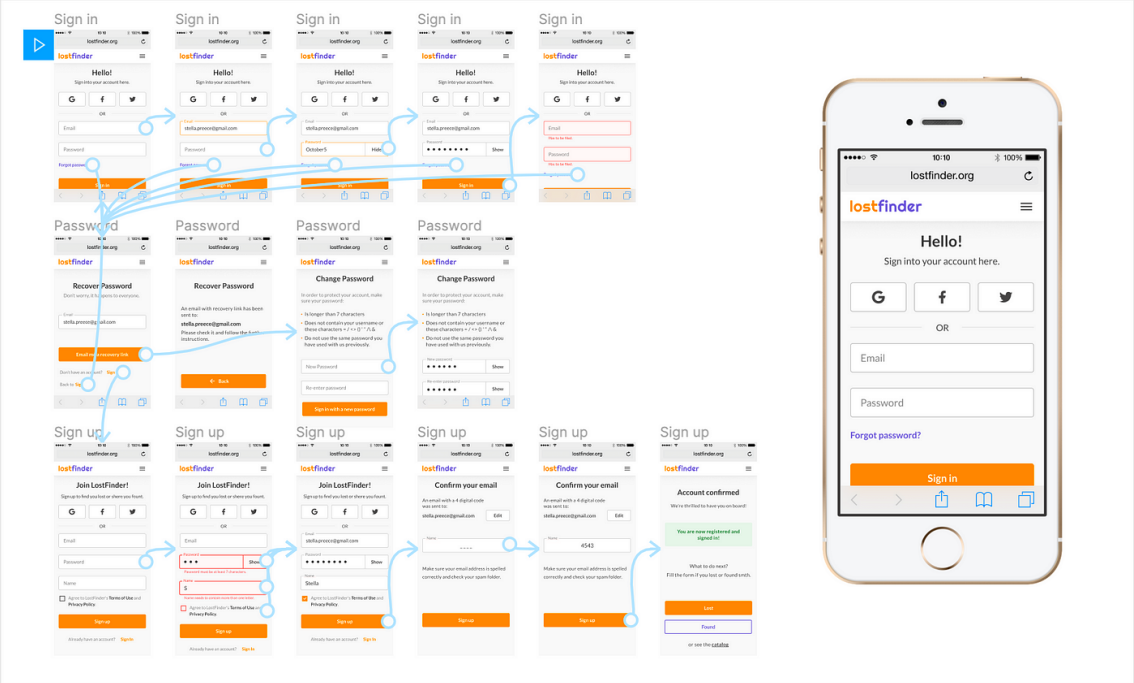


Figure 3.4: Scheme of a Hi-Fi prototype[5]

is ideal for remote or distributed teams, allowing multiple people to manage and work on projects simultaneously.

Figma was the chosen tool for the prototyping of the Motio StepWatch™ System monitoring platform.

InVision

Similar to Figma, InVision[6] is a no-code prototyping tool that lets users develop high-fidelity functional prototypes as well as simple wireframes. Since InVision can seamlessly convert Sketch or Photoshop projects into interactive prototypes, many designers prefer it over other tools. Furthermore, its superior version control enables effortless monitoring of changes. However, one of its main downsides is that it is not as collaborative as Figma, even though it still allows team members and stakeholders to comment and tag each other directly on projects.

Adobe XD

Part of the Adobe Creative Cloud suite, Adobe XD[7] is a vector-based, no-code prototyping tool that allows designers to build low-fidelity wireframes to high-fidelity prototypes. It supports responsive resizing and projects with micro-interactions while offering a vast library of plugins and UI kits. The main differential of Adobe XD is its seamless integration with other Adobe Creative

[5] https://bootcamp.uxdesign.cc/what-is-the-difference-between-low-and-high-fidelity-prototypes-b1f3612f85f7. Extracted on 2024-06-05.

[6] https://www.invisionapp.com/

[7] https://adobexdplatform.com/

Cloud products, which allows direct transfer from one application to another, making the entire process more efficient. At last, its collaborative nature is comparable to InVision's, as it mainly comprises commentaries from other members or stakeholders.

However, in May 2023, Adobe XD entered maintenance mode[8], which implies that there will be no further development or new features within it.

3.3.3 Style Guide for Designing

Prototyping alone is not enough to improve usability if the application's design is confusing or unorganized. Therefore, user interface design knowledge is also required when developing a system. This knowledge is based on both theory, such as the field of cognitive psychology, and experience, like researching currently effective designs [61].

Regarding the psychological field, there are four principles for UI development, which are [61]:

1. Users see what they expect to see;
2. Users have difficulty focusing on more than one activity at a time;
3. It is easier to perceive a structured layout;
4. It is easier to recognize something than to recall it.

Principle 1 bases itself on the fact that humans are not particularly good at dealing with unexpected circumstances after they become used to a pattern. Thus, the design must follow two guidelines: the *consistency* and the *exploiting prior knowledge*. The former refers to the consistent use of colors, styles, and fonts, whilst the latter involves investigating users' prior knowledge, as they usually perceive the screen using its experience.

Even though most people can divide their attention satisfactorily, this skill varies considerably from person to person. Therefore, the UI should provide support in case this situation happens, as stated by Principle 2. From this, two other guidelines that the designer must take into account are the *perceptual organization* and the *importance*. The first guideline indicates that items that belong together should be grouped, and the second one expresses that if something is significant to the user, it should be displayed prominently.

Next, Principle 3 involves Gestalt psychology, which encompasses five laws that define how we perceive the world, as listed below.

1. *Law of* ***proximity***: closely spaced elements appear as groups rather than as random elements;
2. *Law of* ***similarity***: elements of the same shape or color seem to belong together;
3. *Law of* ***closure***: where possible, an incomplete element is seen as complete;
4. *Law of* ***continuity***: dots in a row are viewed as a line rather than as a random set of dots;

[8] https://helpx.adobe.com/pt/support/xd.html

5. *Law of* ***symmetry***: regions bounded by symmetrical borders are perceived as coherent figures.

Furthermore, there is the figure-ground segregation concept, which states that when a figure contains two or more distinct regions, these sections are seen as a figure and the rest as the background. Figure 3.5 exemplifies this concept, as the perceived shape depends on which color —white or black—the person considers as the figure.

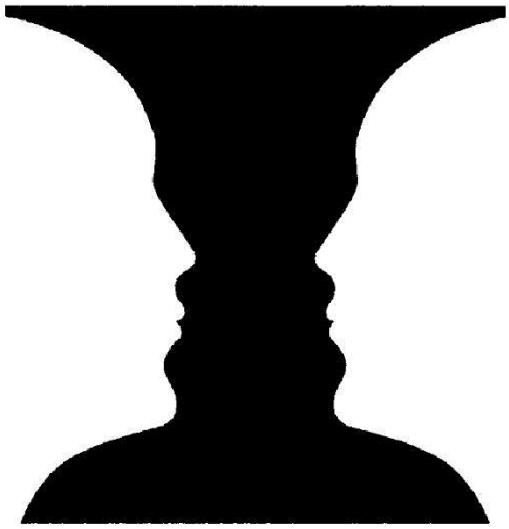


Figure 3.5: "One Vase or Two Faces?" [69] - Figure-ground segregation example

Finally, Principle 4 resonates with some studies about memory that showed that it was easier to identify information than to recollect it. This fact follows the notion of *knowledge in the head* and *knowledge in the world*, whose main idea is that people combine information stored in memory with information found in the environment when they carry out daily activities. As a result, the last guideline that the UI must obey is the *recognition*, which translates to "where possible, allow users to recognize what they want rather than forcing them to recall it".

In the book "*The Design of Everyday Things*", Don Norman established other six principles for interaction design, which are:

1. *Principle of* ***visibility***: users are more likely to know about and use a more visible element;
2. *Principle of* ***affordance***: it should be evident how to use a control;
3. *Principle of* ***feedback***: there must be a response to a performed action and its consequence;
4. *Principle of* ***constraints***: the range of possible interactions must be limited to simplify the interface and guide the user to the appropriate next action;
5. *Principle of* ***mapping***: controls and their effects must have a clear relationship;

6. *Principle of* ***consistency***: similar to Principle 1 from the psychological field, "users see what they expect to see".

3.4 Evaluation

An evaluation of the system is essential to identify existing problems, which include anything that hampers the users—from bugs and missing functionality to ease-of-use issues. It can be conducted either with the prototype or the final system.

3.4.1 Usability Testing

Usability testing is a user-based evaluation in which one or more representative users perform tasks under observation [70]. It is standardized by the ISO/EIC 25062, *Common Industry Format for Usability Test Reports*, which provides a common framework and a set of guidelines for reporting the test results to guarantee replicability, scalability, and confidence [71].

The outcome measures of this test assess the participants' effectiveness, efficiency, and satisfaction with the tasks, as described in Section 3.1.3. *Effectiveness* refers to how accurately and thoroughly the goals are achieved; thus, frequent measurements include task completion percentage and error frequency. In contrast, *efficiency* relates to how much resource was expended to complete the task, such as the mean time to accomplish it. At last, *satisfaction* describes a user's subjective response to using the product, often assessed via a questionnaire. Frequently used questionnaires are After-Scenario Questionnaire (ASQ), Computer Usability Satisfaction Inventory (CUSI), Post-Study System Usability Questionnaire (PSSUQ), Questionnaire for User Interaction Satisfaction (QUIS), Software Usability Measurement Inventory (SUMI), and System Usability Scale (SUS). Figure 3.6 illustrates the System Usability Scale to showcase an example.

3.4.2 Feedback Gathering from Stakeholders

The simplest way to evaluate a system is by gathering feedback from stakeholders, which includes any input, idea, opinion, or viewpoint from users or anyone affected by the system. There are several methods for conducting this evaluation, from focus groups to anonymous surveys. Table 3.2 shows the benefits and challenges involved with stakeholder feedback.

3.5 Implementation

The implementation step of an application pipeline consists essentially of converting the validated prototype into a fully functional and interactive interface. It represents the development of the application's *front end* and its integration with the *back end*.

The term *front end* refers to the graphical user interface (GUI) with which the users interact, such as menus, buttons, images, and graphs. In web applications, the main front-end languages used are:

	The System Usability Scale Standard Version	Strongly Disagree					Strongly Agree
			1	2	3	4	5
1	I think that I would like to use this system frequently.		O	O	O	O	O
2	I found the system unnecessarily complex.		O	O	O	O	O
3	I thought the system was easy to use.		O	O	O	O	O
4	I think that I would need the support of a technical person to be able to use this system.		O	O	O	O	O
5	I found the various functions in this system were well integrated.		O	O	O	O	O
6	I thought there was too much inconsistency in this system.		O	O	O	O	O
7	I would imagine that most people would learn to use this system very quickly.		O	O	O	O	O
8	I found the system very awkward to use.		O	O	O	O	O
9	I felt very confident using the system.		O	O	O	O	O
10	I needed to learn a lot of things before I could get going with this system.		O	O	O	O	O

Figure 3.6: Standard System Usability Scale [72]

Table 3.2: Benefits and challenges of gathering stakeholder feedback

Benefits	**Challenges**
Easier to manage stakeholder expectations	Complex identification of stakeholders
Improves alignment between stakeholders	Irregular feedback quality and reliability
Facilitates the identification of process improvements	Stakeholders' availability
Leads to greater innovation	Several outreach methods required
Promotes trust and engagement	Too much feedback

- **HyperText Markup Language (HTML)**: application structure and content
- **Cascading Style Sheets (CSS)**: design features, including layout, fonts, colors, and visual style
- **JavaScript**: more interactive features

Meanwhile, the *back end* (or *server side*) handles the application's general functionality. It usually encompasses its operations, databases, and application programming interfaces (APIs). Frequently used programming languages for back end development are Java, Python, and Ruby.

3.6 Summary

In this chapter, the concepts of user-centered design, iterative design, and usability were introduced. Then, the proposed methodology for this project was presented, which is based on these concepts and follows a classic application development pipeline.

The methodology starts with requirements gathering, which comprises stakeholder identification, user research, and task analysis. The next step is the conceptual design using mental and conceptual models. Based on this gathered information, a high-fidelity prototype is developed with the low-fidelity prototype as its foundation. Subsequently, usability testing and feedback collecting are conducted to reduce issues in the final system. Finally, after validating the prototype, its implementation will be carried out to provide a thoroughly working version.

The following chapter addresses the first two steps, requirements gathering and conceptual design.

Chapter 4

Initial Studies

Following the overview of all stages in the application development, this chapter will address the first two steps, Requirements Gathering and Conceptual Design, combined into Initial Studies. Section 4.1 describes the stakeholders involved in the Motio System Monitoring Platform and their tasks, and establishes the requirements. Next, in Section 4.2, the Conceptual Design of the application is built on two sides: a requirements one and a technical one. First, a Conceptual Model was created using a UML Use Case to represent the requirements gathered and adopted. As for the technical side, a System Model was produced by describing the system components and their interactions, then focusing on specific components.

4.1 Stakeholder Requirements

4.1.1 Stakeholders Identification

First, as indicated by the pipeline in Chapter 3, the stakeholders and their tasks inside Adapttech were identified, as described below.

- **Business Support**: responsible for implementation of a Quality Management System compliant with 21 CFR part 820; development and maintenance of technical documentation for class I devices; registration and listing of FDA devices; development of risk management activities; management of customer complaints and CAPA; leadership on data privacy compliance under GDPR and HIPAA; and provision of internal training on Good Documentation Practices and Product Lifecycle Management;

- **Clinical Feedback and Support**: responsible for support of customers and end-users, as well as support for the other areas inside Adapttech (Business Support, R&D, and Marketing);

- **R&D**: responsible for software development (back-end and front-end development), data science, and management of IT resources - internally and cloud hosted;

- **Marketing and Communications**: responsible for management of social media, blog content, marketing campaigns, press releases, and Award applications, as well as content creation, improvement of SEO, strategies development for product launches, and participation management in national and international events and tradeshows.

More specifically, their tasks concerning the Motio StepWatch™ System and its monitoring platform are:

- **Business Support**: compliance and safety monitoring of the Motio StepWatch™ System;
- **Clinical Feedback**: recognition of Motio StepWatch™ System usage patterns for improvement or issues identification;
- **R&D**: identification of system usage frequency and inference of infrastructure requirements;
- **Marketing and Communications**: recognition of Motio StepWatch™ System's reach to determine how it could be advertised.

Moreover, some demographic information about the stakeholders and, by extension, future users is that their ages range from 30 to 40, and they all have a higher education.

4.1.2 Requirements Gathering

To simplify this step and make it less time-consuming for stakeholders, the requirements were gathered through a collaborative document, in which stakeholders could write what they wanted to see in the monitoring platform and comment on others' suggestions. In case of conflicted opinions, discussions would be held internally.

In this document, the requirements were divided by subject and classified into three types: essential (E), desirable (D), or optional (O), based on their level of priority. The requirements concerning Motio App Performance, Post-Market Surveillance, Clinical Insights, and Others are displayed in Tables 4.1, 4.2, 4.3, and 4.4, in this respective order.

4.1.3 Requirements Analysis

Since numerous requirements can make the development of an application challenging, some had to be prioritized above others. Therefore, after a discussion with the stakeholders, the decision was first to address all the essential requirements. A desirable requirement would also be considered if the time allows or when it is easily feasible, like, for example, *Requirements ORC1, ORO3, ORO4, and ORO5*.

Some requirements, such as *ORP1*, *ORP2*, *ORP6*, and *ORP7*, were excluded since they either fall out of the project scope or may be met by other existing technology. For example, requirements about interaction evaluation could be satisfied using a tool like Microsoft Clarity and hence do not necessitate to be implemented into the platform.

Table 4.1: Functional requirements related to Motio App Performance

ID	Description	Type	Relevant Stakeholders
ERM1	Be able to view how many new patients were added into the system in a Monthly, Quarterly, or Yearly view, with the option to group by location over time.	E	Marketing and Communications
ERM2	Be able to view how many new appointments (with at least an outcome test) were created in a Monthly, Quarterly, or Yearly view, with the option to group by location over time. Present in tabular and graphic form.	E	Clinical and Feedback Support; Business Support
ERM3	Be able to view the amount (absolute and relative to the total number of appointments) of outcome tests performed in the Motio App in a Monthly, Quarterly, or Yearly view, with the option to group by location.	E	Clinical and Feedback Support; Business Support
ERM4	Be able to export any data being viewed (through graphics or table) as a .csv or other similar format.	E	Clinical and Feedback Support; Business Support
ERM5	Be able to export viewed graphics as a .png or .pdf.	E	Clinical and Feedback Support; Business Support
ERM6	Be able to monitor the subscription license state per clinic, for example, the date of last purchase and available stock of reports.	E	Clinical and Feedback Support; Business Support
ORM1	Be able to monitor the report purchase/license history of a specific clinic.	D	Clinical and Feedback Support; Business Support

Even though they will be considered during the application development, the Others requirements (Table 4.4) refer to more technical aspects of the Motio System Monitoring Platform, such as information about the application access or color requirements. Therefore, they will be regarded differently from the rest, with the majority only being addressed during implementation.

Table 4.2: Functional requirements related to Post-Market Surveillance of Motio App

ID	Description	Type	Relevant Stakeholders
ERP1	Be able to view how many new issues, improvements, complaints, or CAPAs have been reported in a monthly, quarterly, or yearly view, with the option to group by location and by product/system.	E	Business Support
ERP2	Be able to view how many issues, improvements, complaints, or CAPAs related to safety were reported in a monthly, quarterly or yearly view, with the option to group by location and by product/system.	E	Business Support
ERP3	Calculate the closing times of issues, improvements, complaints, or CAPAs and the average for each month, quarter, or year, with the option to group by product/system.	E	Business Support
ERP4	Show the number of open complaints and CAPAs per month, quarter, and year, with the option to group by product/system and severity.	E	Business Support
ERP5	Calculate the percentage of complaints closed in less than 1 week and less than 1 month. View monthly, quarterly and yearly and group by severity, safety-related, and product/system.	E	Business Support
ORP1	Count the number of times that the user canceled the outcome measure process	O	-
ORP2	Track time of execution for each outcome measure	O	-
ORP3	Show which complaints and CAPAs are still open per month, quarter, and year, with the option to group by product/system and severity.	D	Business Support
ORP4	Detect how many appointments quarterly and yearly were not closed after 30 days.	D	Business Support
ORP5	Detect how many patients do not have an appointment associated, monthly, quarterly, and yearly.	D	Business Support
ORP6	Detect how many times the button for the Instructions for Use was clicked on monthly, quarterly, and yearly.	O	Business Support
ORP7	Be able to detect how many unsuccessful login attempts were made monthly, quarterly, and yearly.	O	Business Support

Table 4.3: Functional requirements related to Clinical Insights

ID	Description	Type	Relevant Stakeholders
ERC1	Be able to download in tabular form the PLUS-M results performed during a given time frame, together with the test date and some information about the patient, such as gender, date of birth, amputation type and etiology, AMPPRO score, TUG duration, K level, 10-meter walk test time, and 2-minute or 6-minute walk test distance.	E	R&D
ERC2	Be able to download in tabular form the AMPPRO results performed during a given time frame, together with the test date and some information about the patient, such as date of birth and amputation, amputation type and etiology, and K level.	E	R&D
ORC1	Be able to download in tabular form the Timed Up and Go result, date of test, and risk of fall performed during a given time frame	D	R&D
ORC2	Be able to download in tabular form the Long-term acquisition results, and date of test performed during a given time frame	D	R&D
ORC3	Be able to download the Motio Functional level in a tabular form.	D	R&D
ORC4	Be able to download the Clinical Assessment in a tabular form	D	R&D

Table 4.4: Other functional requirements

ID	Description	Type	Relevant Stakeholders
ERO1	Be able to filter collected data by a specific time frame and time resolutions (monthly, quarterly, and yearly).	E	R&D
ERO2	The application must only be accessible from within the company network.	E	R&D
ORO1	Be able to set a default time frame in which the user can view data (Last Quarter, Last 6 Month, Last Year) in this Dashboard app	D	R&D
ORO2	Settings configured by the user of the Dashboard app should be saved locally	O	R&D
ORO3	The Application should be accessible online through authentication using a Microsoft account.	D	R&D
ORO4	When using color to represent a location, severity, or product, it must be consistent within the App at that moment.	D	Clinical and Feedback Support; Business Support
ORO5	When using color to represent a location, severity, or product, it must be consistent in the App overtime.	D	Clinical and Feedback Support; Business Support

4.2 Conceptual Design

The Conceptual Design of this project was defined using a Conceptual Model, more specifically a Unified Modeling Language (UML) Use Case diagram, and a System Model, which concerns the functionality design of the platform and will also encompass other types of UML diagrams.

4.2.1 Conceptual Model

To create the Conceptual Model, the system requirements were translated into a Use Case Diagram, which depicts the relationships among actors—i.e., individuals, organizations, or external systems—and use cases within a system. Apart from providing an overview of the requirements, another role of this diagram is to convey the scope of the development [73].

Figure 4.1 displays the developed Use Case diagram for this project, which illustrates the relationships of all components of the Motio System, namely the Motio StepWatch™ System, Report Motio Feedback App, and Motio System Monitoring Platform. However, it is important to highlight that the project's focus is the Motio System Monitoring Platform; the other two are depicted to give a better understanding of the whole Motio System.

In the diagram, the actors are:

- **Stakeholders**: Business Support, Clinical Feedback and Support, Marketing and Communication, and R&D
- **Systems external to the Motio System**: SQL database, Microsoft Sharepoint List and Microsoft Planner
- **Adapttech Internal**: any employee at Adapttech who can log feedback that originated internally or externally
- **Physician**: individual who will carry out the patient's appointment
- **Patient**: prosthetic user on whom the tests will be conducted

Furthermore, as explained previously, the use cases related to the Motio System Monitoring Platform comprise the considered requirements—essential requirement plus *Requirement ORC1*—described in Tables 4.1, 4.2 and 4.3 from Section 4.1. As for the Report Motio Feedback App and Motio StepWatch™ System use cases, they relate to the procedures of logging feedback, creating appointments, and performing Long-Term Acquisition tests, described in Section 2.1.

4.2.2 System Model

Motio System Monitoring Platform's System Model covers the functionality of the whole system by first describing the system components and their interactions with each other. Next, it focuses on some elements, such as the integration of external systems and the user interface.

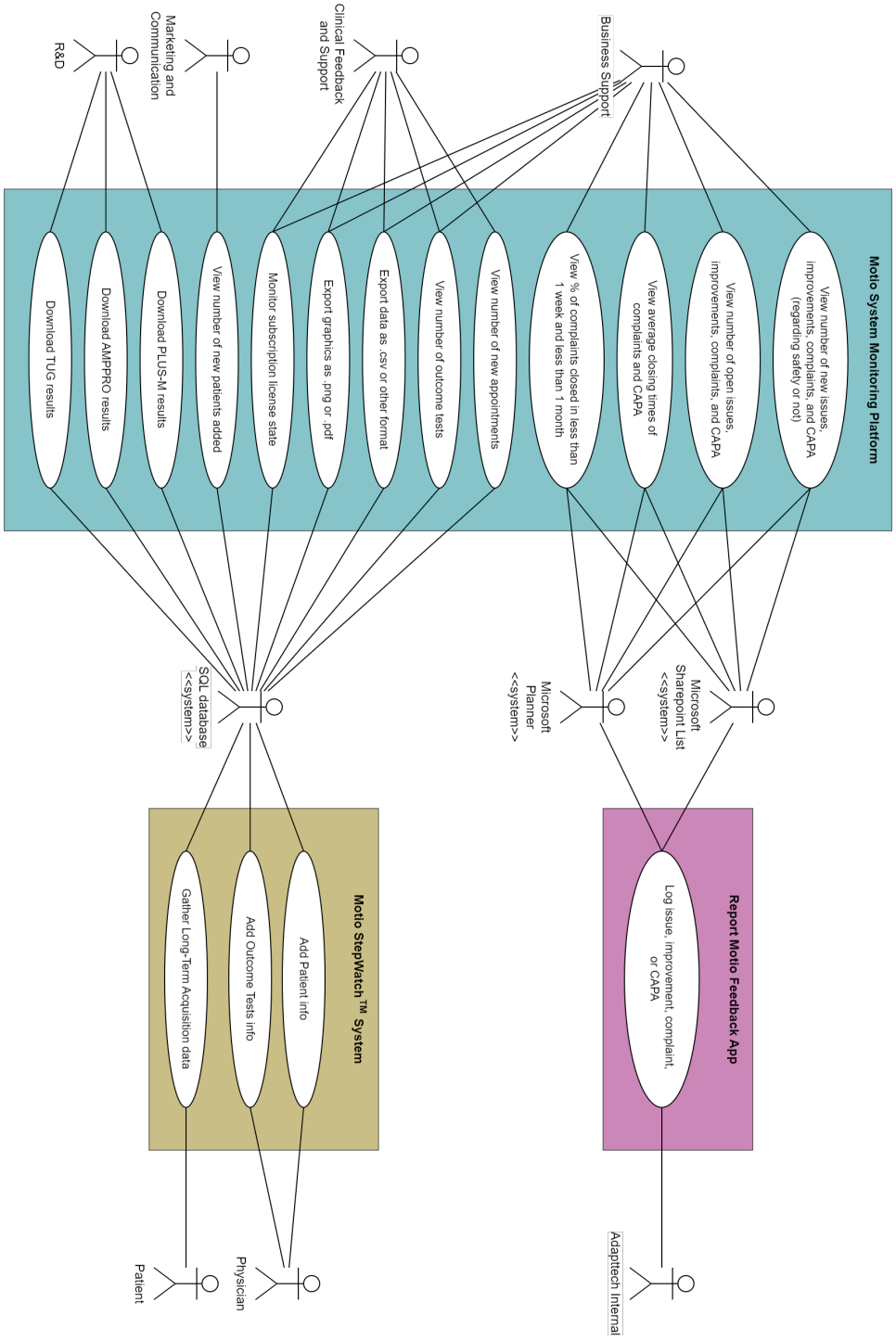


Figure 4.1: Use Case diagram

4.2.2.1 System Components

As a typical application, the Motio System Monitoring Platform can be divided into *front end* and *back end*. The *front end* refers to the platform's interface, which allows users to view graphs and tables and interact with filters. Meanwhile, the *back end* concerns all operations that happen behind the interface, which, in this case, corresponds to the database connection, data retrieval, and data manipulation.

Due to the database connection and data retrieval, this project's *back end* communicates to external systems through the *SQL Interface* and *MS Interface*. The *SQL Interface* handles the connection to the Motio SQL Database and the query of its data. Meanwhile, the *MS Interface* connects to the Microsoft Graph API and retrieves data through HTTP requests. Since this API has access to the Microsoft Planner and Sharepoint List, it can retrieve Adapttech's feedback ticket data, as explained in Section 2.1.3.

Figure 4.2 portrays all these system components using a Package Diagram.

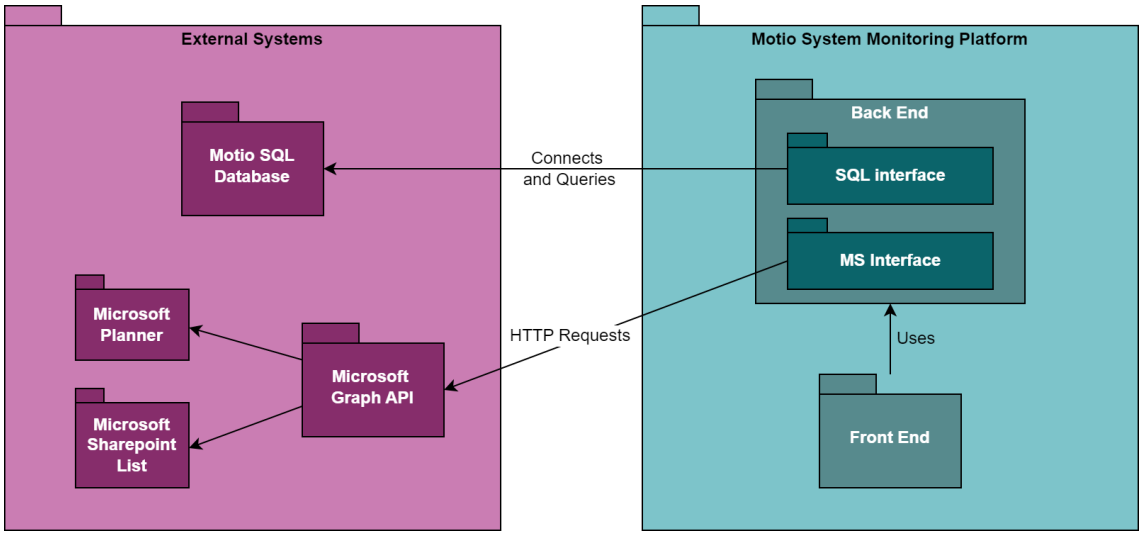


Figure 4.2: UML Package diagram - System components

4.2.2.2 Component Interaction

Motio App Data Visualization and Export

As indicated in the Use Case diagram (Figure 4.1), users must be able to see the dashboard with the Motio App data and also export them as CSV. Therefore, two UML Activity diagrams were created to outline the system's activities, one for the data visualization (Figure 4.3) and the other for the export (Figure 4.4). The second diagram, Figure 4.4, can be perceived as a continuation of Figure 4.3, which indicates that the time view and, if applicable, filters have already been selected.

Although *Requirement ERM5* also determined the export of the viewed graphics, this was deemed unnecessary since the chosen data visualization tool already provides this feature automatically when displaying the graphs, as seen in Figure 6.26 in Section 6.6.

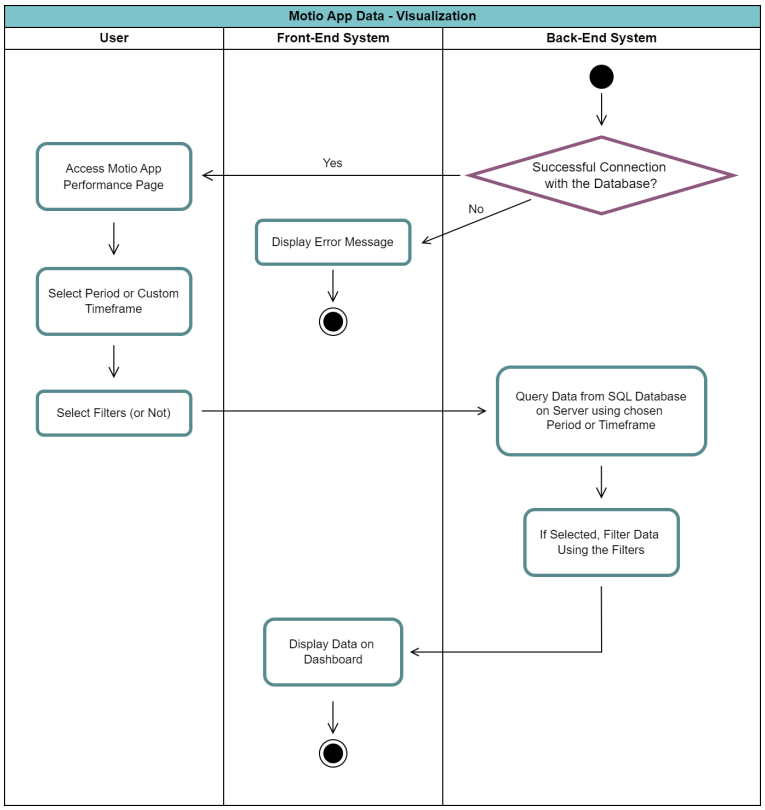


Figure 4.3: UML Activity diagram - Motio App data visualization

Furthermore, a Sequence Diagram was also created to exhibit the display and export of the Motio App data, as illustrated in Figure 4.5. It is noticeable that these diagrams are similar to the Activity diagrams; however, they detail more about what happens in the *backend*, such as the classes, functions, parameters, and messages. In contrast, the user—or stakeholders, in this case—present the same information, namely their interaction with the user interface.

To better comprehend the Sequence Diagram from Figure 4.5, its lifelines must be described since some of them were not introduced previously, such as (from left to right):

- Motio App Performance Page (UI): represents the interface itself with which the actors will interact;
- Motio App Performance Page (Controller): represents the code that generates this interface;
- DashboardingFunctions: represents the data manipulation section of the back end;

Feedback Data Visualization and Export

Since the use cases related to the Microsoft Planner and Sharepoint List also covered data visualization, their Activity Diagram and Sequence Diagram, depicted in Figures 4.6 and 4.7, respectively, were similar to the diagrams of the Motio App data, with the only main difference being the need to combine the List, Planner, and Task items. This distinction between Planner and Task will be explained in Section 4.2.2.3.

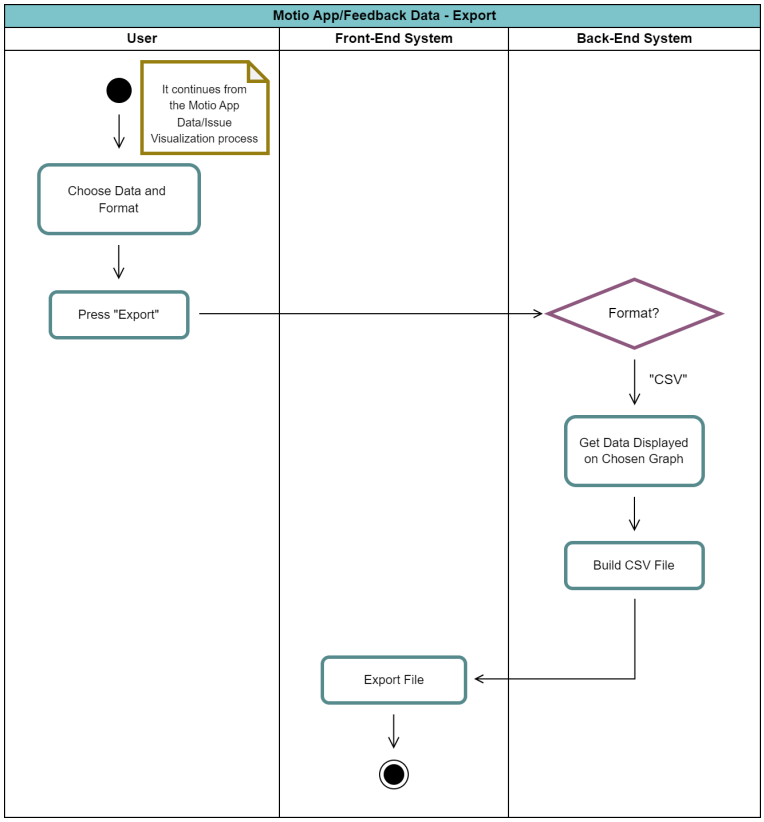


Figure 4.4: UML Activity diagram - Motio App and Feedback data export

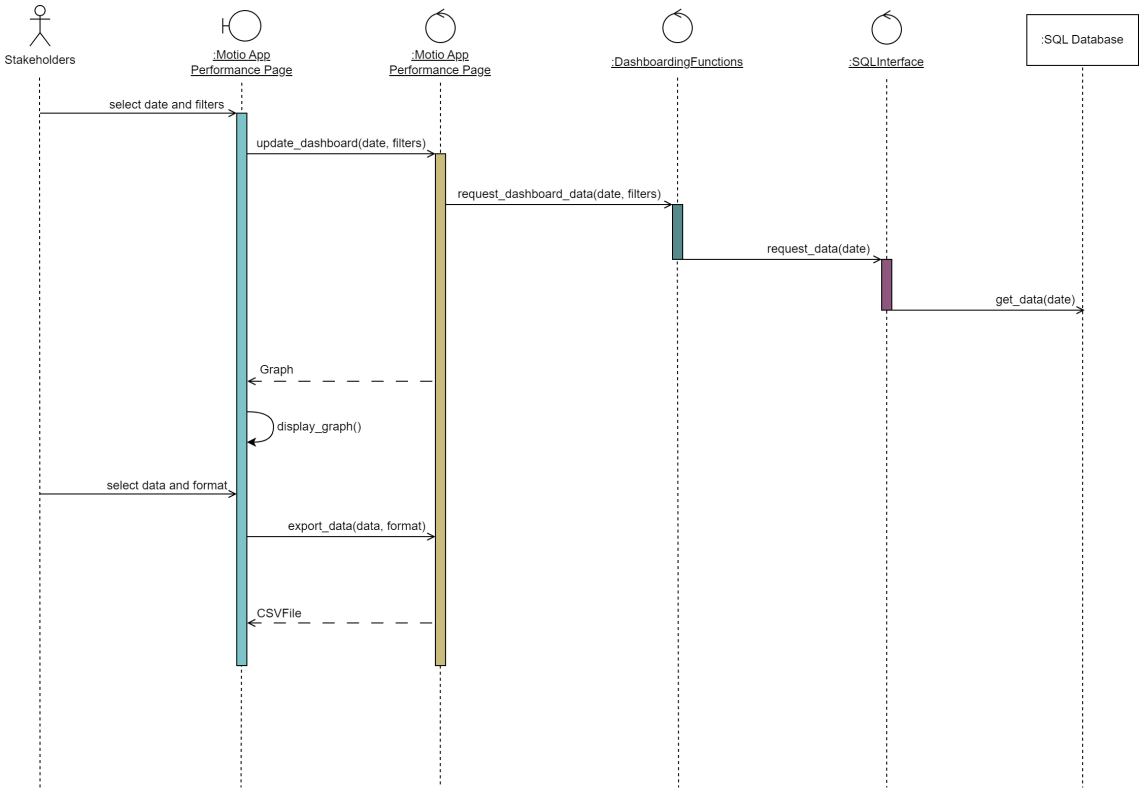


Figure 4.5: UML Sequence diagram - Motio App data visualization and export

Despite not being a requirement, the idea of having an export feature also for Feedback data was raised. Since its implementation would be straightforward, the stakeholders agreed with it. Therefore, Figure 4.4 also considers the Feedback data export.

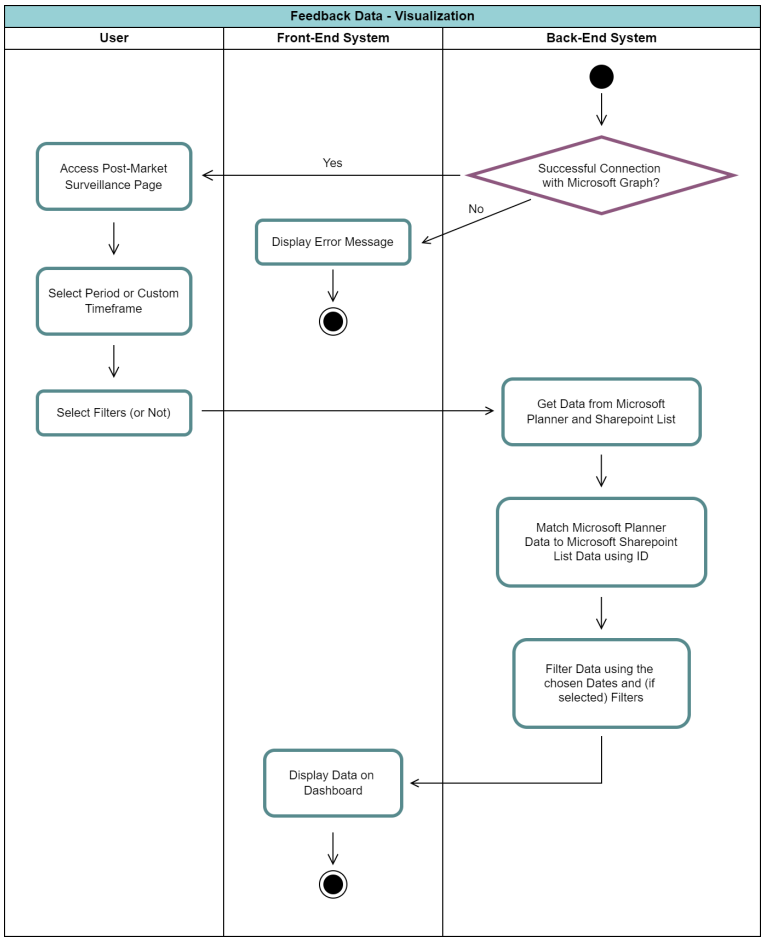


Figure 4.6: UML Activity diagram - Feedback data visualization

Outcome Test Report Export

Analogous to the previous component interactions, Figures 4.8 and 4.9 exhibit the Activity Diagram and Sequence Diagram, respectively, of the use cases regarding the download of test reports.

4.2.2.3 External Integration

As indicated in Figure 4.2, there are two external systems: Motio SQL Database and Microsoft Graph API, which will be described next.

Motio SQL Database

First, to connect to the Motio SQL database, it is required to provide the Server Address and Database Name, as well as the Username and Password of a user with at least read permissions.

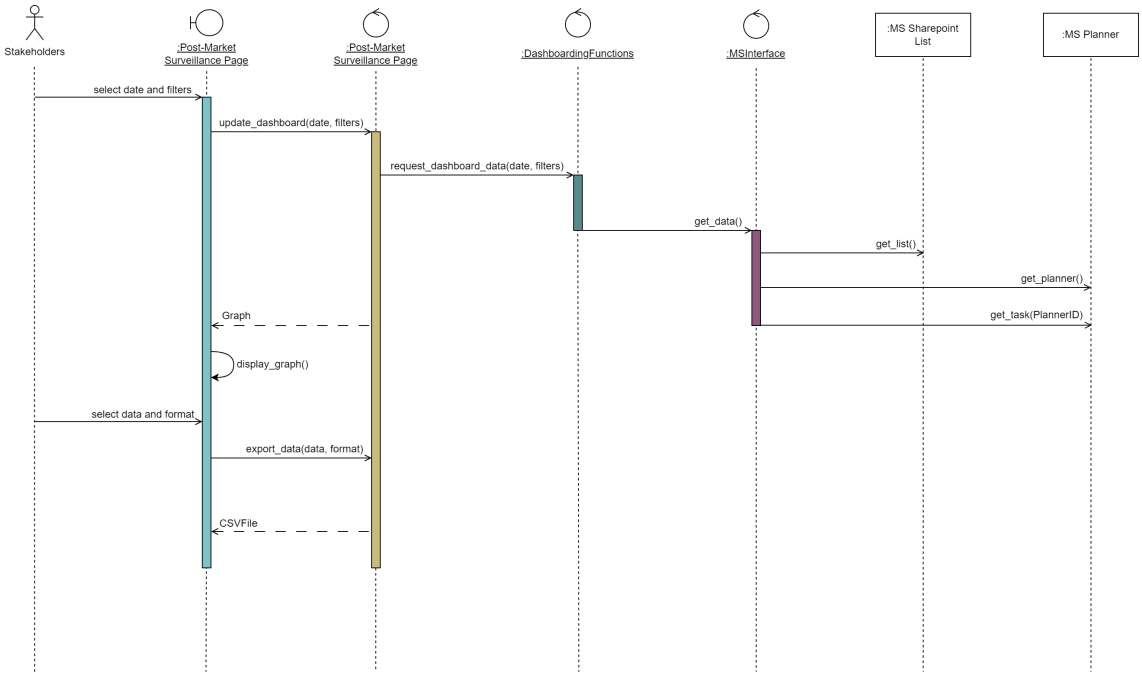


Figure 4.7: UML Sequence diagram - Feedback data visualization and export

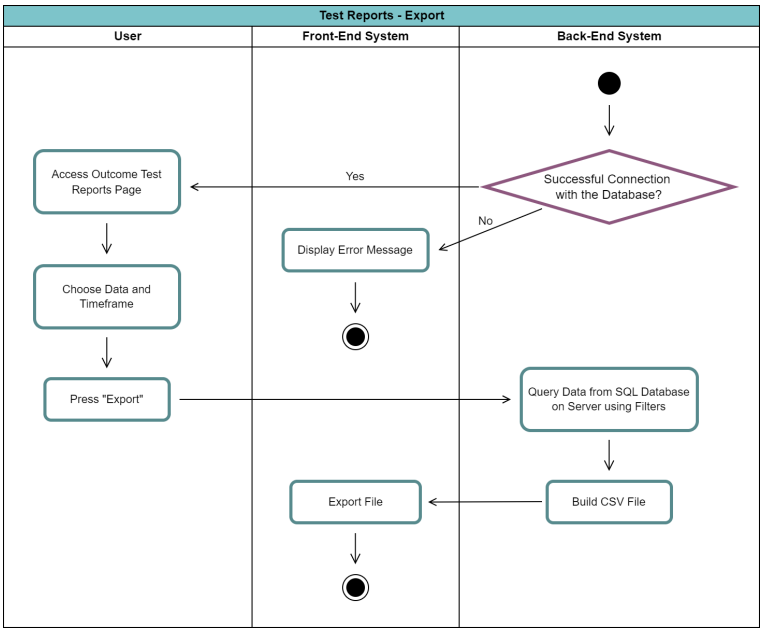


Figure 4.8: UML Activity diagram - Outcome test reports export

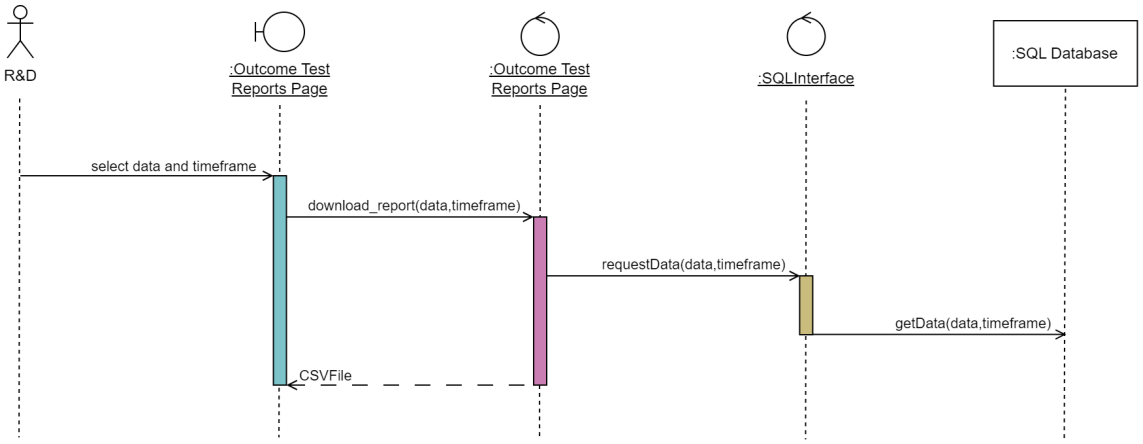


Figure 4.9: UML Sequence diagram - Outcome test reports export

When successful, the *SQL Interface* will return the connection State and Details, such as logging info. Then, to retrieve the required Data, the specified Dates are necessary alongside the Data Name, which can be:

- patients;
- appointments;
- outcome tests;
- axot (Appointments X Outcome Tests);
- plus-m (Plus-M);
- amppro (AMPPRO);
- tug (Timed Up and Go).

Figure 4.10 exhibits the Component Diagram for the Motio SQL database, which represents the database connection and data retrieval procedures. It also displays the required interfaces on the right and the provided interfaces on the left.

Now, focusing more on the Motio SQL database itself, Figure 4.11 presents a simplified version of the Class Diagram regarding the Motio SQL database that only includes the classes relevant to the Motio System Monitoring Platform and their attributes and associations. In addition, it also displays their primary and foreign keys that allow connections between these classes.

Microsoft Graph API

As for the Microsoft Graph API, its Component Diagram is depicted in Figure 4.12, which shows that the required interfaces include the Microsoft user authentication information, such as Tenant ID, Client ID, and Client Secret, alongside the information about the Microsoft Planner and Sharepoint List, namely Site ID, List ID, and Planner ID. To obtain the authentication information

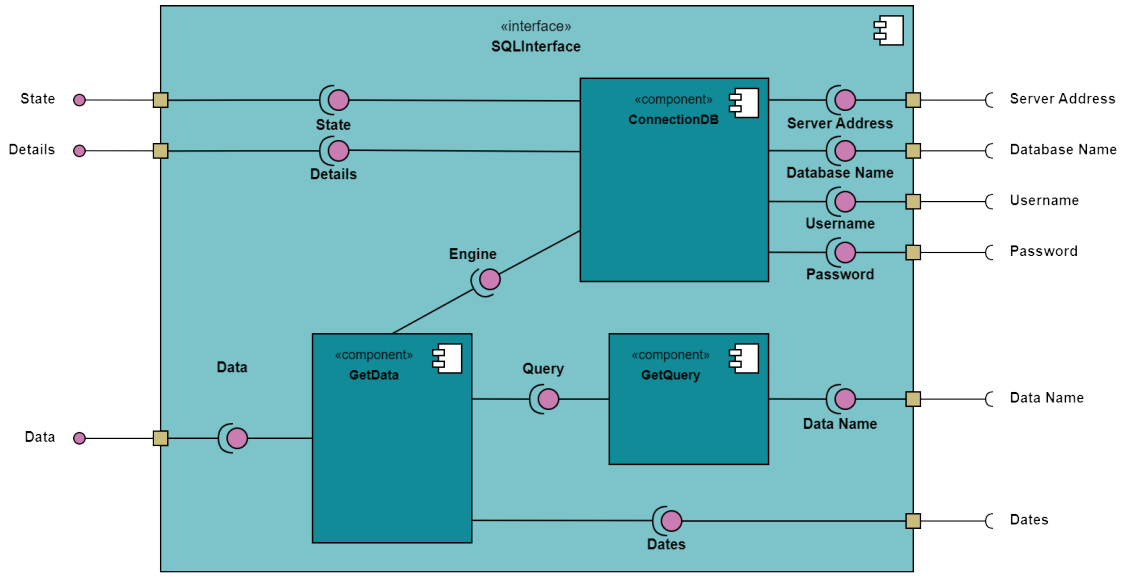


Figure 4.10: UML Component diagram - SQL Interface

about the user, a client needs to be created inside the organization using Microsoft Entra and granted read access to at least the Microsoft Planner and Sharepoint List. Then, provided interfaces encompass the connection State and Detail, returned after the established connection, and the requested Data.

Concerning the Feedback data accessed through the MS Graph API, their classes are schematized in Figure 4.13. Even though a Planner is composed of Tasks, Microsoft Planner has a distinction between what is a Planner item information and what is a Task item information. This results in the List ID being present in the Task Item, whereas the relevant feedback information, namely Feedback Type, Reported Date, Completed Date, Priority, and Status, is located in the Planner Item.

4.2.2.4 User Interface

The front-end development includes the user interface, which allows users to interact with the Motio System Monitoring Platform. In this project, the user interface was created iteratively using prototypes and following the stakeholders' requirements and feedback. The user interface prototyping process is covered in Chapter 5.

4.3 Summary

This chapter covered the first stage of application development, which encompasses gathering requirements and conceptual designing. Its outputs include a list of requirements divided by subject (Motio App Performance, Post-Market Surveillance, Clinical Insights, and Others) and priority (essential, desirable, or optional), as well as six UML diagrams (Use Case, Package, Activity,

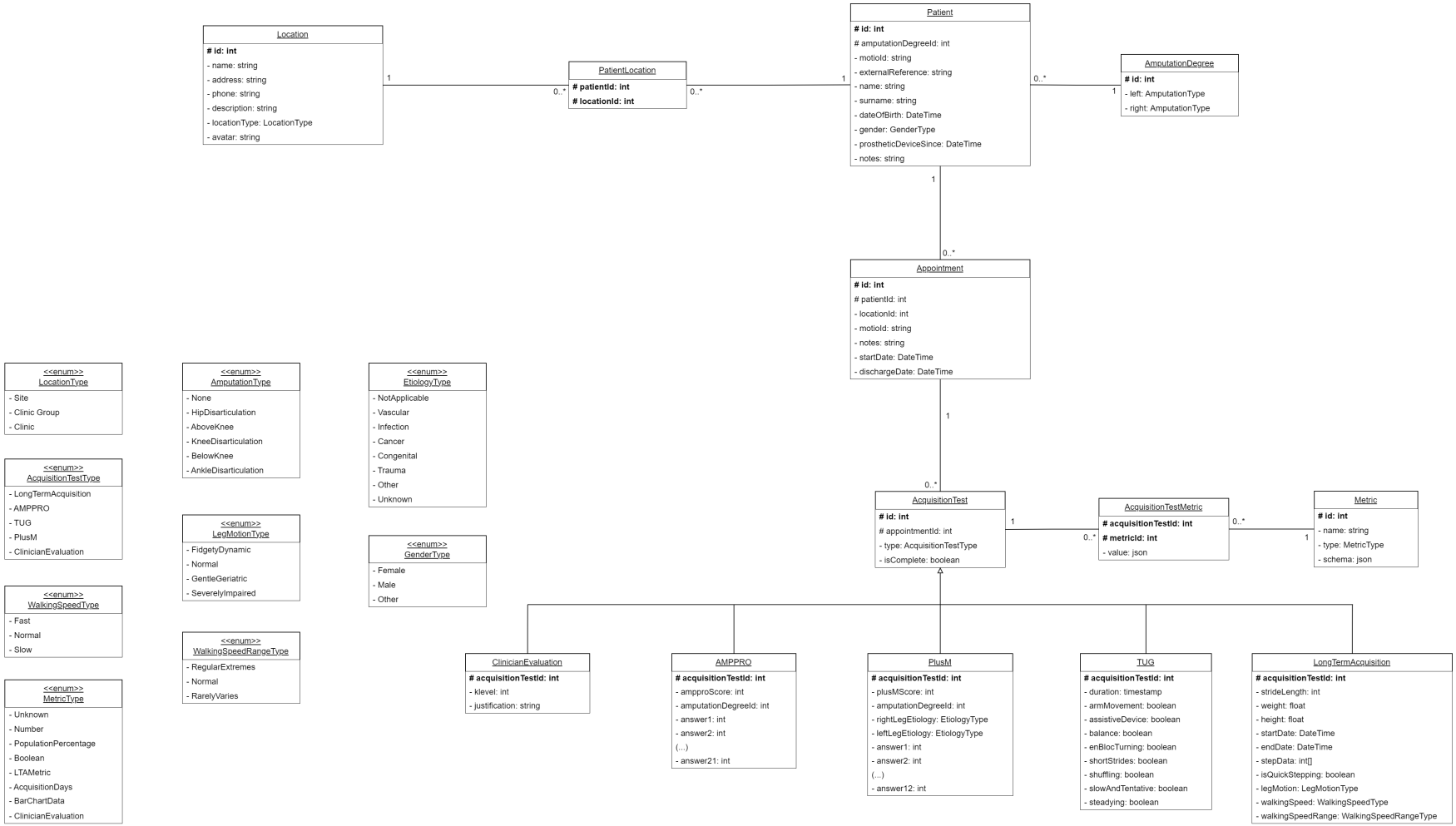


Figure 4.11: Class diagram - Motio App (SQL database)

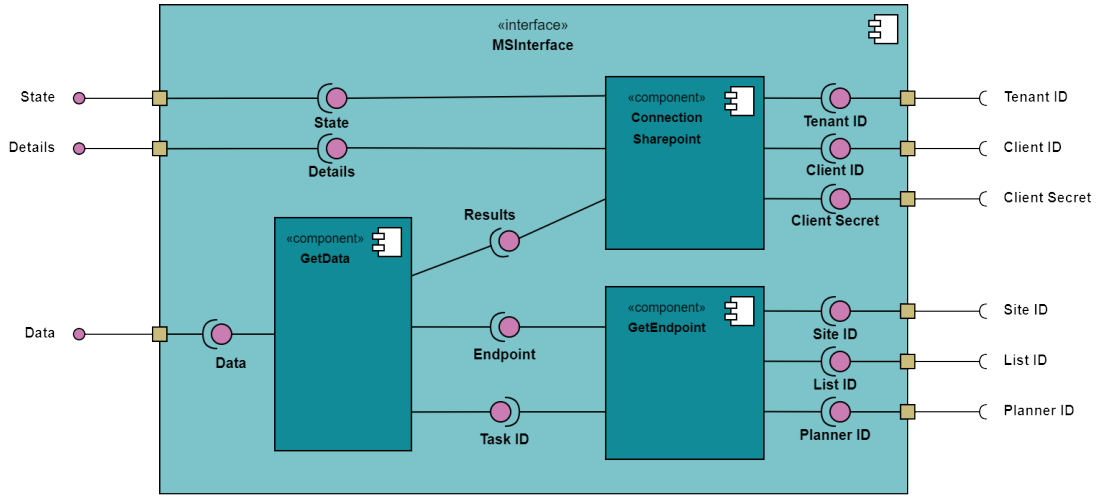


Figure 4.12: UML Component diagram - Microsoft Interface

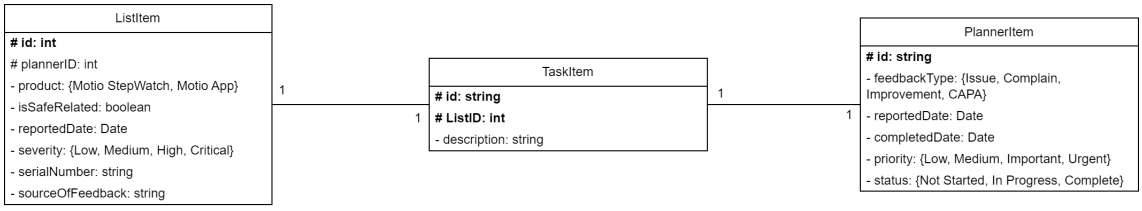


Figure 4.13: UML Class diagram - Feedback (Microsoft Planner and Sharepoint List)

Sequence, Class, and Component) that outlines the Motio System. The requirements list and the diagrams will contribute to the prototype development, discussed in the following chapter.

Chapter 5

UI Prototyping and Validation

Once the requirements are identified and a Conceptual Design is created, the next step is to develop the prototype. First, a less sophisticated version, called a low-fidelity prototype, must be created; Section 5.1 covers this process. Then, a high-fidelity prototype can be iteratively developed, addressing issues and incorporating feedback from previous versions until the final prototype is ready. In this case, three iterations were required to achieve the finished product, all of which are discussed in Section 5.2. Finally, Section 5.3 describes the final prototype validation through usability testing.

5.1 Low-Fidelity (Lo-Fi) Prototype

Concerning Lo-Fi prototypes, wireframes and data visualization sketches were developed to plan the system layout of the Motio System Monitoring Platform. First, to better organize the platform structure, pages were created based on the requirements subjects, i.e., Motio App Performance, Post-Market Surveillance, and Clinical Insights. There is no page for the other functional requirements since they are more general and most of them do not relate to the user interface.

Figure 5.1 depicts the Motio App Performance Page, with five graph cards representing:

- New Patients Added (*Requirement ERM1*);
- New Appointments (*Requirement ERM2*);
- Outcome Tests (*Requirement ERM3*);
- Outcome Tests Relative to Appointments (*Requirement ERM3*);
- Subscription License (*Requirement ERM6*).

Moreover, it includes the option of exploring data using a monthly, quarterly, or yearly view (*Requirements ERM1, ERM2, and ERM3*) and a "More Filters" button that opens the filter menu, shown in Figure 5.2, which contains a time frame (*Requirement ERO1*) and location (*Requirements ERM1, ERM2, and ERM3*) selection. At last, the reason for two Export fields (*Requirements ERM4 and ERM5*)—delineated in red—was the uncertainty regarding its ideal positioning.

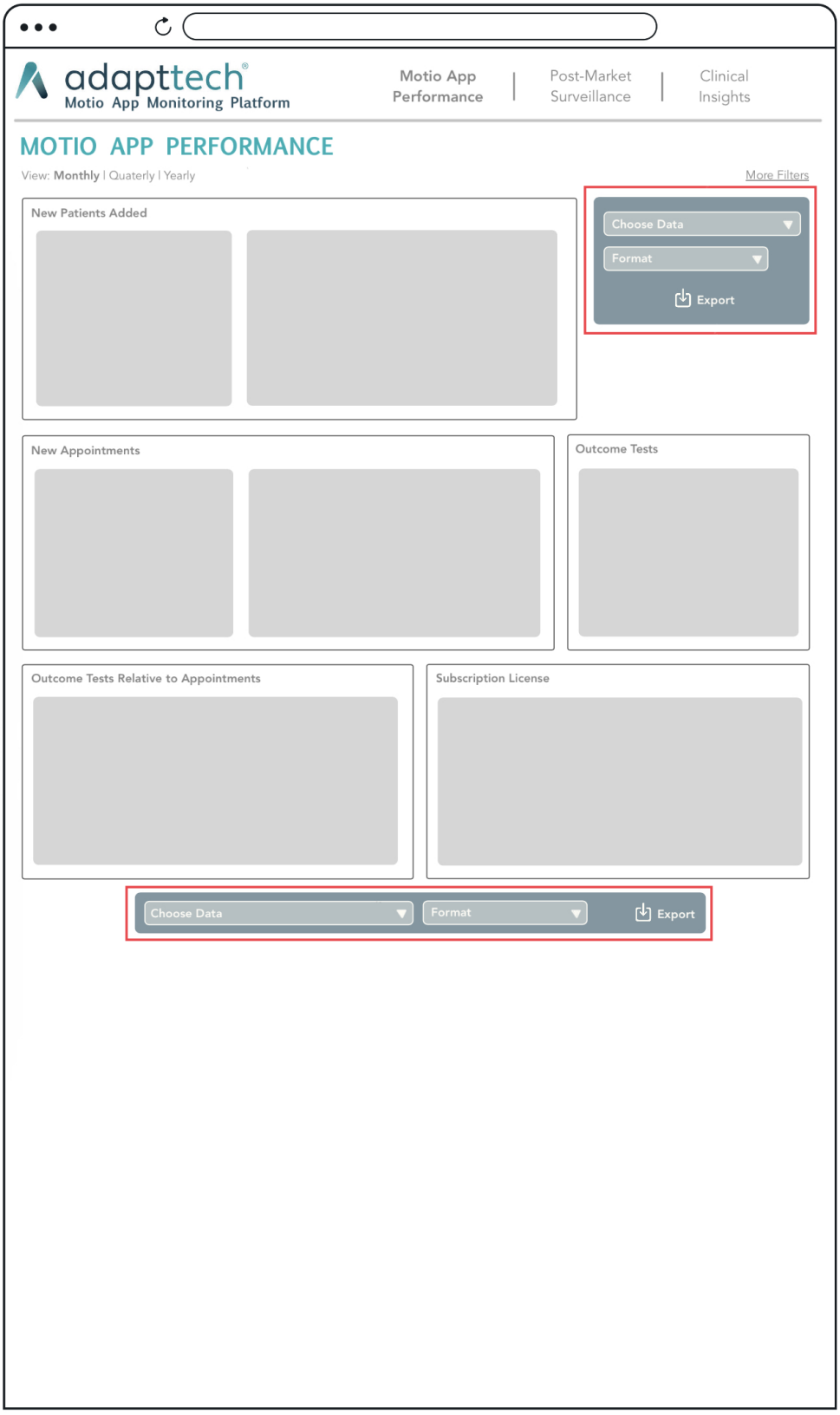


Figure 5.1: Wireframe - Motio App Performance page



Figure 5.2: Wireframe - Filter menu

The Post-Market Surveillance page is illustrated in Figure 5.3. It also comprises five graph cards:

- Overall Numbers (*Requirement ERP1*);
- Related to Safety (*Requirement ERP2*);
- Average Closing Times (*Requirement ERP3*);
- Open Complaints and CAPA (*Requirement ERP4*);
- Closed Complaints (*Requirement ERP5*).

Analogous to the Motio App Performance page, this page also has a monthly, quarterly, or yearly view *Requirements ERP1, ERP2, ERP3, ERP4, and ERP5*) and a "More Filters" button. However, in this case, the filter menu also allows the user to choose the product/system (*Requirements ERP1, ERP2, ERP3, ERP4, and ERP5*).

The gray boxes depicted in both Motio App Performance and Post-Market Surveillance pages will be filled with data in several formats. Figure 5.4 illustrates the proposed data visualization formats for each requirement identified by its ID. Additionally, note that, in these figures, the platform name is Motio App Monitoring Platform instead of Motio System Monitoring Platform since the change occurred only later, in Chapter 6; nonetheless, throughout this document, it will only be referred to as Motio System Monitoring Platform.

Finally, since all requirements related to Clinical Insights are about downloading the test results (*Requirements ERC1, ERC2, ORC1, and ORC2*), its page, represented by Figure 5.5, contains only a Download Test Results section where the user can download specific tests within a time frame.

5.1.1 Feedback Gathering - Lo-Fi Prototype

Before moving to a Hi-Fi prototype, the data visualization proposals (Figure 5.4) were presented to one of Adapttech's stakeholders to gather preliminary feedback via an unstructured interview, which resulted in the following points:

- **Requirements ERM2 and ERM3**: integration of tables into one single table with the number of appointments, the number of outcome tests, and the relative number of outcome tests;
- **Requirement ERM6**: addition of a "Time until Expiration Date" column to its table;
- **Requirements ERP1 and ERP2**: every feedback type (issue, improvement, complaints, and CAPA) must have its own number of new occurrences;
- **Requirement ERP3**: a proposed representation for this requirement is to use a boxplot graph instead of actual numbers;

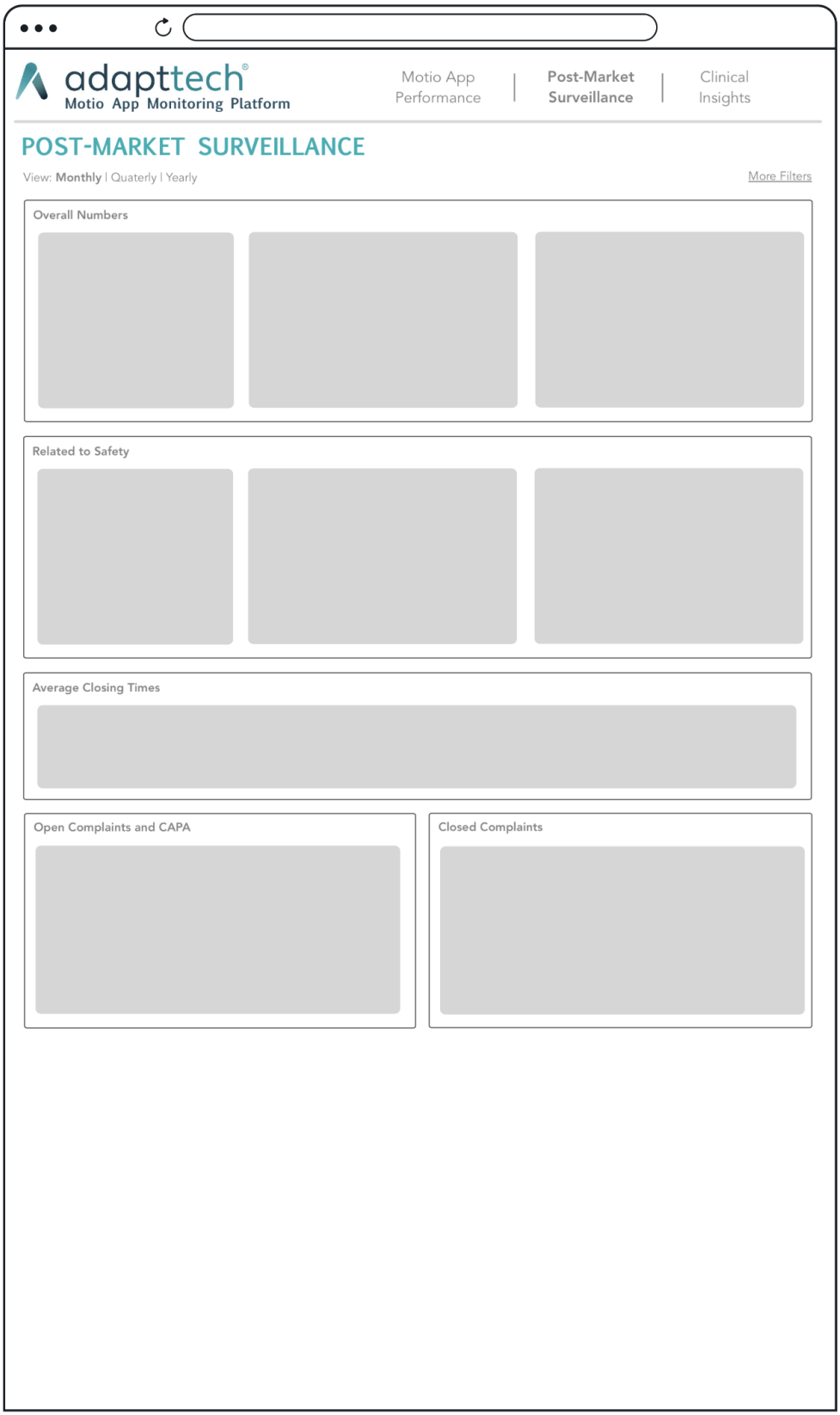


Figure 5.3: Wireframe - Post-Market Surveillance page

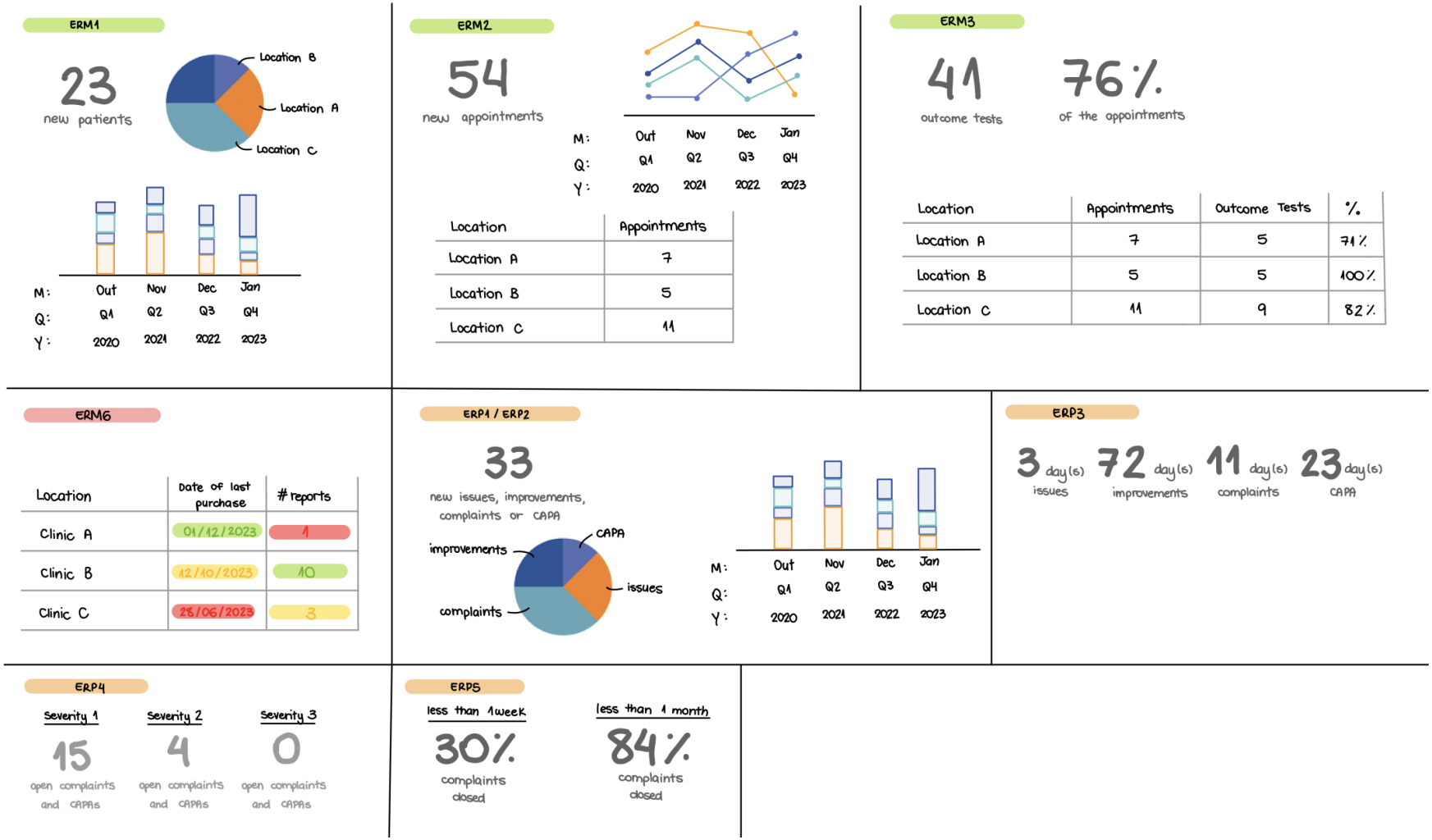


Figure 5.4: Suggested data visualization format for every requirement

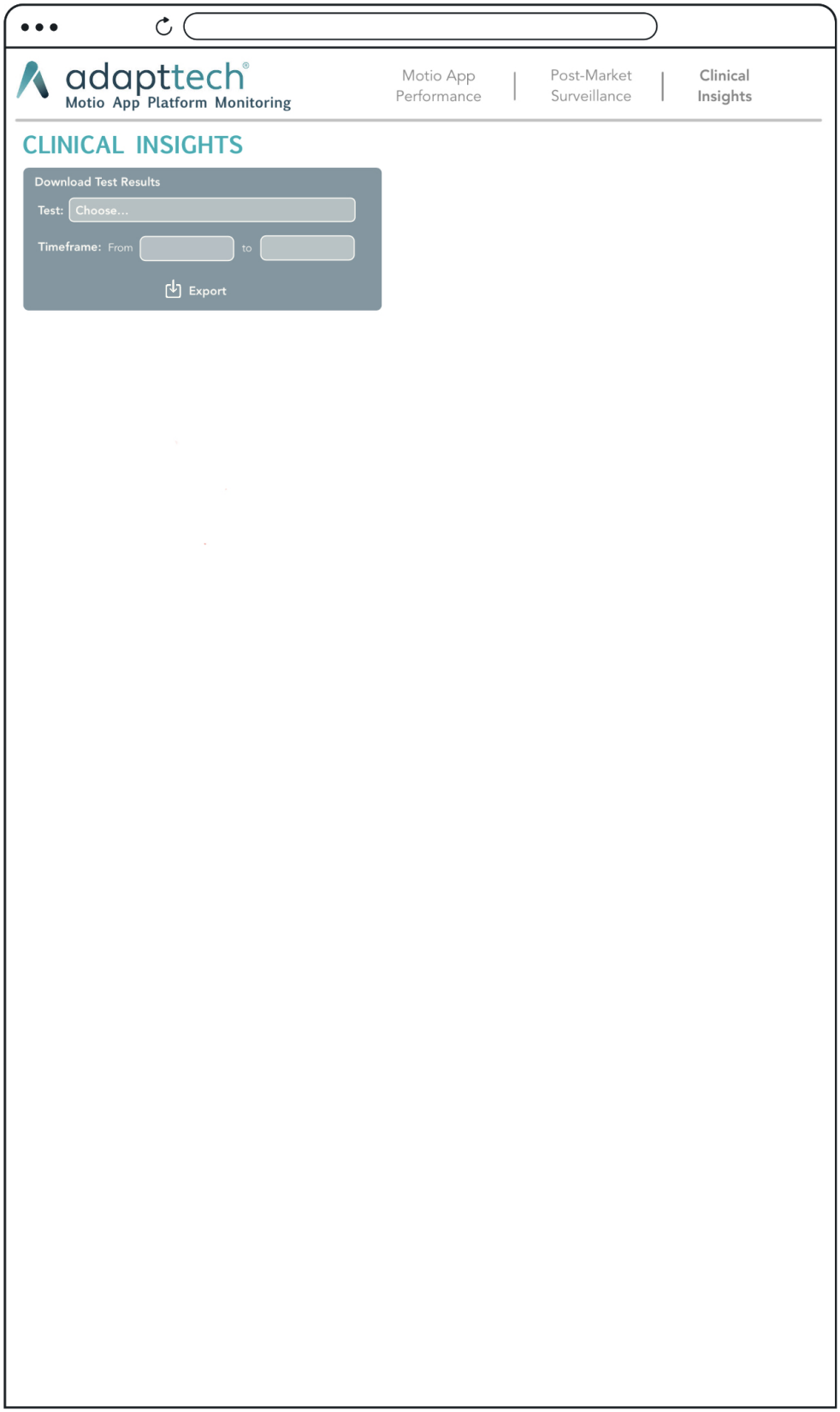


Figure 5.5: Wireframe - Clinical Insights page

- **Requirement ERP4**: the requirement can also include the remaining types of feedback (issues and improvements), and create a comparison with the new tickets;
- General
  - Graphs must have axes with scale;
  - In the case of grouping by product/system and severity, the division can be already made on the dashboard; however, for the Location grouping, it is advisable to add a filter since they are more numerous.

The same was also done for the wireframes (Figures 5.1, 5.3, 5.5. and 5.2) and the gathered comments and recommendations are listed below.

- **Placement of the Export field**: it can be chosen with A/B testing, which compares two versions of the same page and seeks the most user-friendly one;
- **Location Filter**: since there are several locations, it is more appropriate to use a dropdown menu instead of checkboxes; however, the checkboxes can be used when selecting the product/system on the Post-Market Surveillance page;
- **Clinical Insights page**: the name "Clinical Insights" is not really precise.

5.2 High-Fidelity (Hi-Fi) Prototypes

Following the concept of iterative design, the Motio System Monitoring Platform prototype was developed using Figma in three iterations, described in this Section, before reaching the final implementation version. Since there is no way to demonstrate the functional aspects of these prototypes in this document, a link to each Figma design where the prototype can be tested will provided as a footnote.

5.2.1 Hi-Fi Prototype - Prototype Version 1

Based on the previously designed wireframe and considering the received feedback, the first version of a functional prototype was developed on Figma[1]. Figures 5.6, 5.7, 5.8, 5.9, and 5.10 illustrate the platform pages. Since it is a high-fidelity prototype, it allows interaction and provides polished aesthetic elements.

Compared to the wireframe, the main differences are:

- Home page (Figure 5.6);
- All graph cards have data displayed on them (Figure 5.7 and 5.9) based on the Data Visualization proposed and its received feedback in Section 5.1;

[1] Interactive Prototype - Version 1: https://figmashort.link/k3hBHt

- Outcome Tests Relative to Appointments graph card name changed to "Appointments X Outcome Tests" (highlighted in purple in Figure 5.7);
- One of the possible Export fields, the upper one, was moved to the hidden Filter menu (highlighted in purple in Figure 5.8);
- Filter by product on the Post-Market Surveillance page (highlighted in purple in Figure 5.9);
- Test Reports page name (previously Clinical Insights page) (highlighted in purple in Figure 5.10).

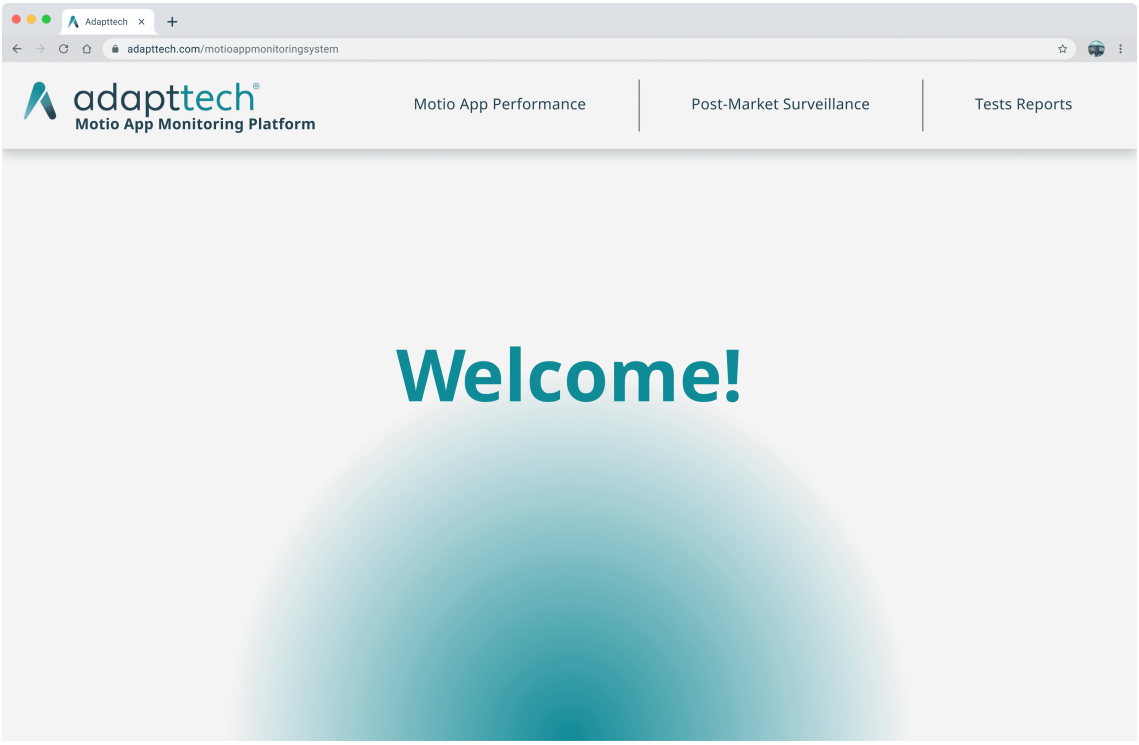


Figure 5.6: Functional prototype - Version 1 - Home page

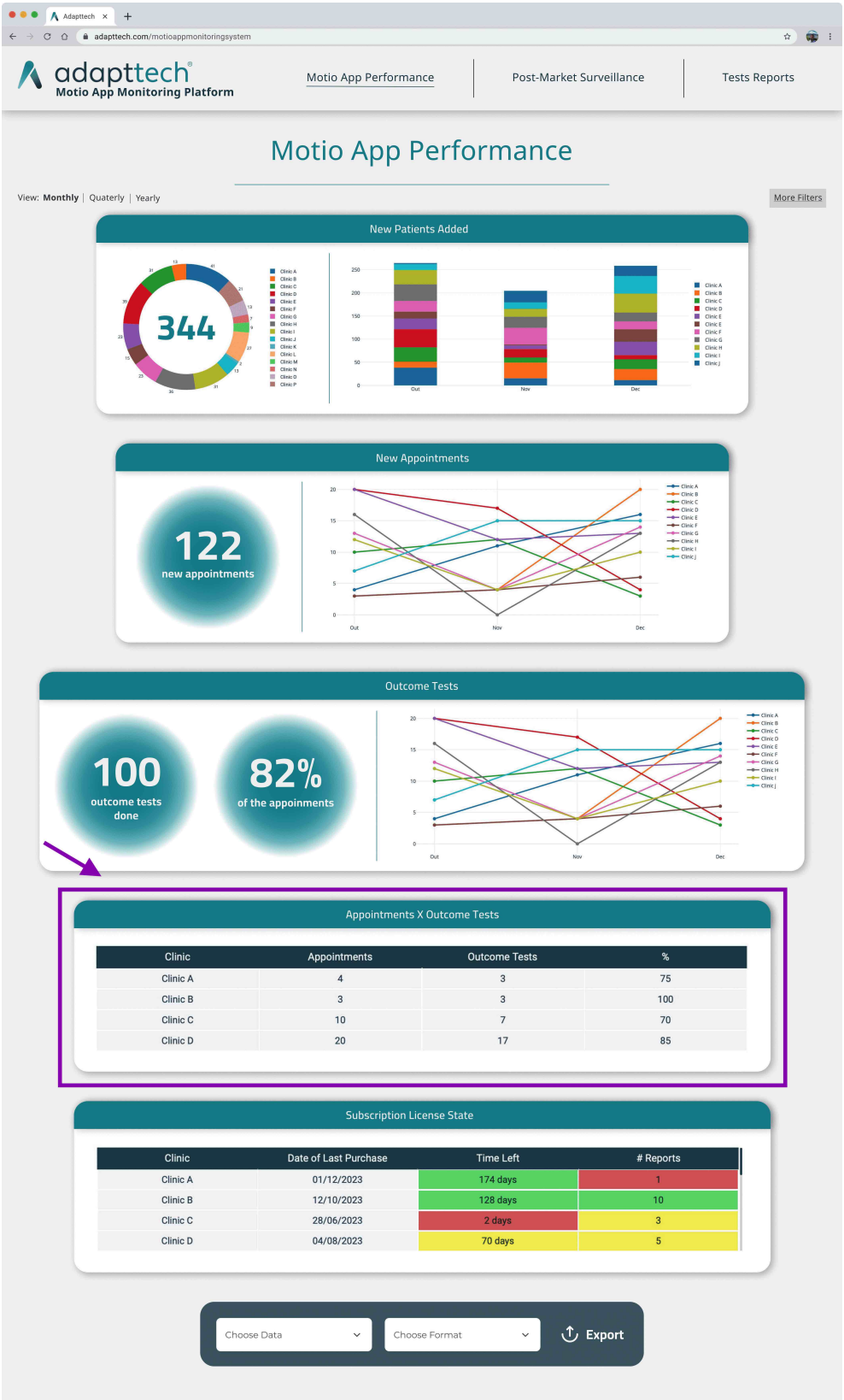


Figure 5.7: Functional prototype - Version 1 - Motio App Performance page

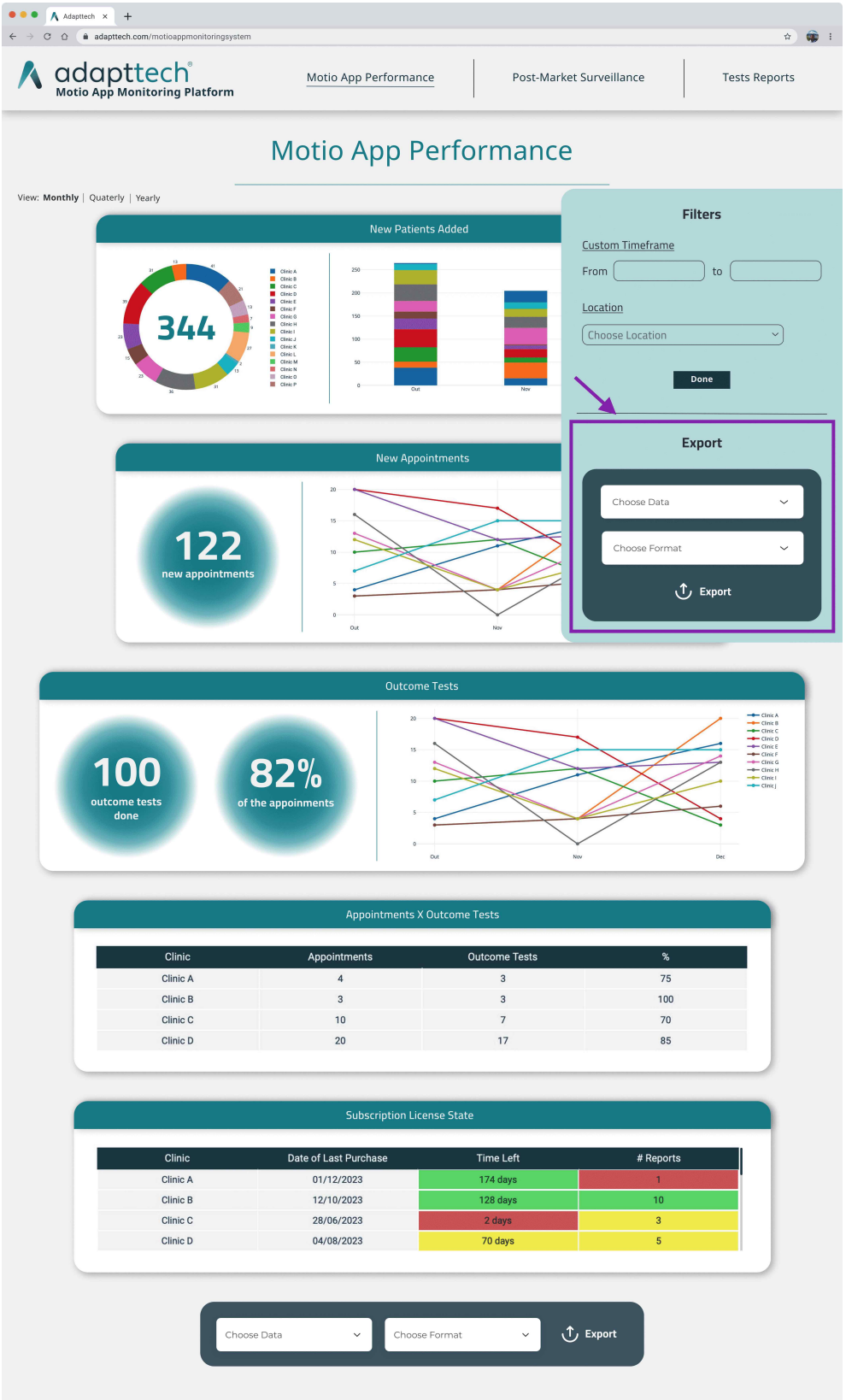


Figure 5.8: Functional prototype - Version 1 - Filter menu

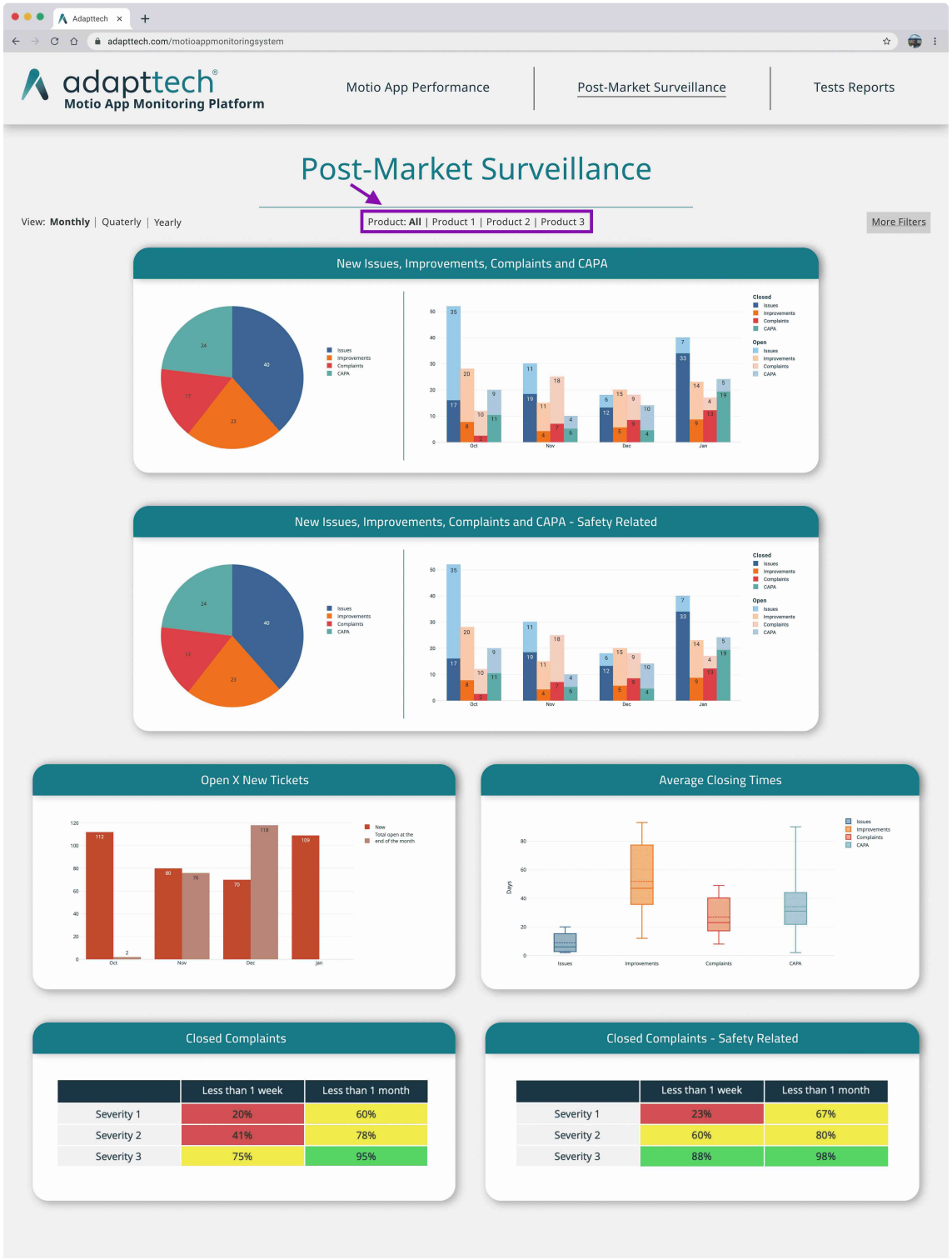


Figure 5.9: Functional prototype - Version 1 - Post-Market Surveillance page

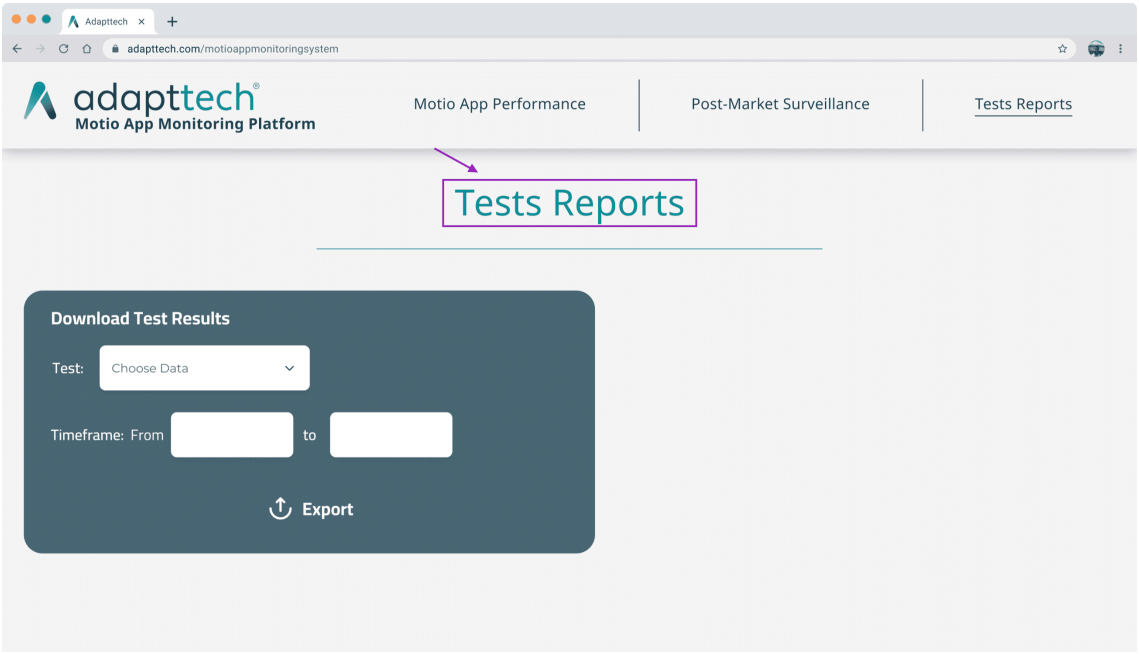


Figure 5.10: Functional prototype - Version 1 - Test Reports page

5.2.1.1 Feedback Gathering - Prototype Version 1

To evaluate Version 1 of the Hi-Fi prototype, all stakeholders were invited to participate in a virtual semi-structured meeting; however, due to scheduling conflicts, Marketing and Communications could not be present. Therefore, three out of four stakeholders participated in the meeting, in which they could view the prototype and provide feedback.

Despite that, there were still several takeaways from this discussion, listed in Tables 5.1, 5.2, and 5.3. Each table presents a type of feedback, i.e., Warnings and Advice, Ideas, and Positive Feedback, and is further divided into subjects (General, Motio App Performance, Post-Market Surveillance, and Test Reports).

Table 5.1: Stakeholder Feedback - Warnings and advice

	Warnings and advice
General	• Be careful with typos • Every graph must have an indication of the period/dates • Avoid abbreviations • Dates: the month must be indicated by its name instead of its number to avoid confusion (e.g., June instead of 6) • Replace all pie charts with absolute numbers (like the one in New Appointments)
Motio App Performance page (Figure 5.7)	• Be careful with similar colors in graphs • No need to show all clinics: Top 5 + Others is enough • Graphs: always ordered by value • Subscription License State: change "# Reports" to "Remaining Reports"
Post-Market Surveillance page (Figure 5.9)	• Always distinguish the open and closed tickets with labels • Closed complaints: add a "Total Number of complaints and a legend for the traffic light colors • New Issues, Improvements, Complaints, and CAPA: divide it into two graphs, one only for the new tickets and one only for the closed ones • Add filter for safety and CAPA
Test Reports page (Figure 5.10)	• Change the name to Outcome Test Reports

Table 5.2: Stakeholder Feedback - Ideas

	Ideas
General	• View and More Filter could follow the scroll • Filter and Export menu always visible on the side
Motio App Performance page (Figure 5.7)	• Add a graph for the Total Number of Patients • New Appointments and Outcome Tests: area chart instead of line chart • Add which clinic had the most patients added, the most appointments, and the most outcome tests • Outcome Tests: remove relative numbers and alter the graph to group by test type instead of by clinic • Appointments X Outcome Tests: display only the Top 5 in an ordered manner and include how many times each test was conducted • Subscription License State: add a View More (amount of reports purchased last time) and the option to order by "Time Left" or "Remaining Reports"
Post-Market Surveillance page (Figure 5.9)	• Add a filter for the type of feedback • Average Closing Time: maybe change to a Swarmplot since there are not many instances

Table 5.3: Stakeholder Feedback - Positive Feedback

	Ideas
General	• Contrasting colors • Option of selecting the time frame in More Filters • Absolute number gradient background • Traffic light colors on the tables • Good aesthetics

5.2.2 Hi-Fi Prototype - Prototype Version 2

A second version of the Motio System Monitoring Platform[2] was developed after considering the collected feedback from the stakeholders' meeting. The main alterations compared to version 1 are highlighted in yellow in Figures 5.11, 5.12, and 5.13.

The Motio App Performance page, shown in Figure 5.11, had the largest number of modifications, which are:

- Number 1: Filter and Export menus are always accessible on the left, rather than appearing only after pressing the "More Filters" button;
- Number 2: the total number of new patients now follows the pattern, discarding the pie chart;
- Number 3: indication of the corresponding period in all graphs ;
- Number 4: showing only the top clinics and grouping the rest in "Others" instead of displaying all of them;
- Number 5: addition of the Total Number of Patients graph card;
- Numbers 6 and 8: substitution of the line graph to an area chart for better visualization;
- Number 7: removal of outcome tests relative to appointments;
- Number 8: graph categorized by type of outcome test instead of by clinic;
- Number 9: indication of the quantity of undertaken outcome tests;
- Number 10: inclusion of the Overall Numbers graph card;
- Number 11: ability to sort by "Time Left" or "Remaining Reports".

Figure 5.12 portrays the Post-Market Surveillance page, which had six changes.

- Number 1: exact as Number 1 from the Motio App Performance page;
- Number 2: inclusion of filters for feedback type and safety-related;
- Number 3: two graph cards, one for new tickets and the other for closed tickets;
- Number 4: analogous to Number 2 from the Motio App Performance page;
- Number 5: graph type changed from boxplot to violin plot;
- Number 6: addition of "Total Number of Complaints";
- Number 7: legend for the traffic light colors.

Lastly, the only alteration the Test Reports page had was regarding its name, which changed to Outcome Test Reports, as evidenced in Figure 5.13.

[2]Interactive Prototype - Version 2: https://figmashort.link/eFN2hG

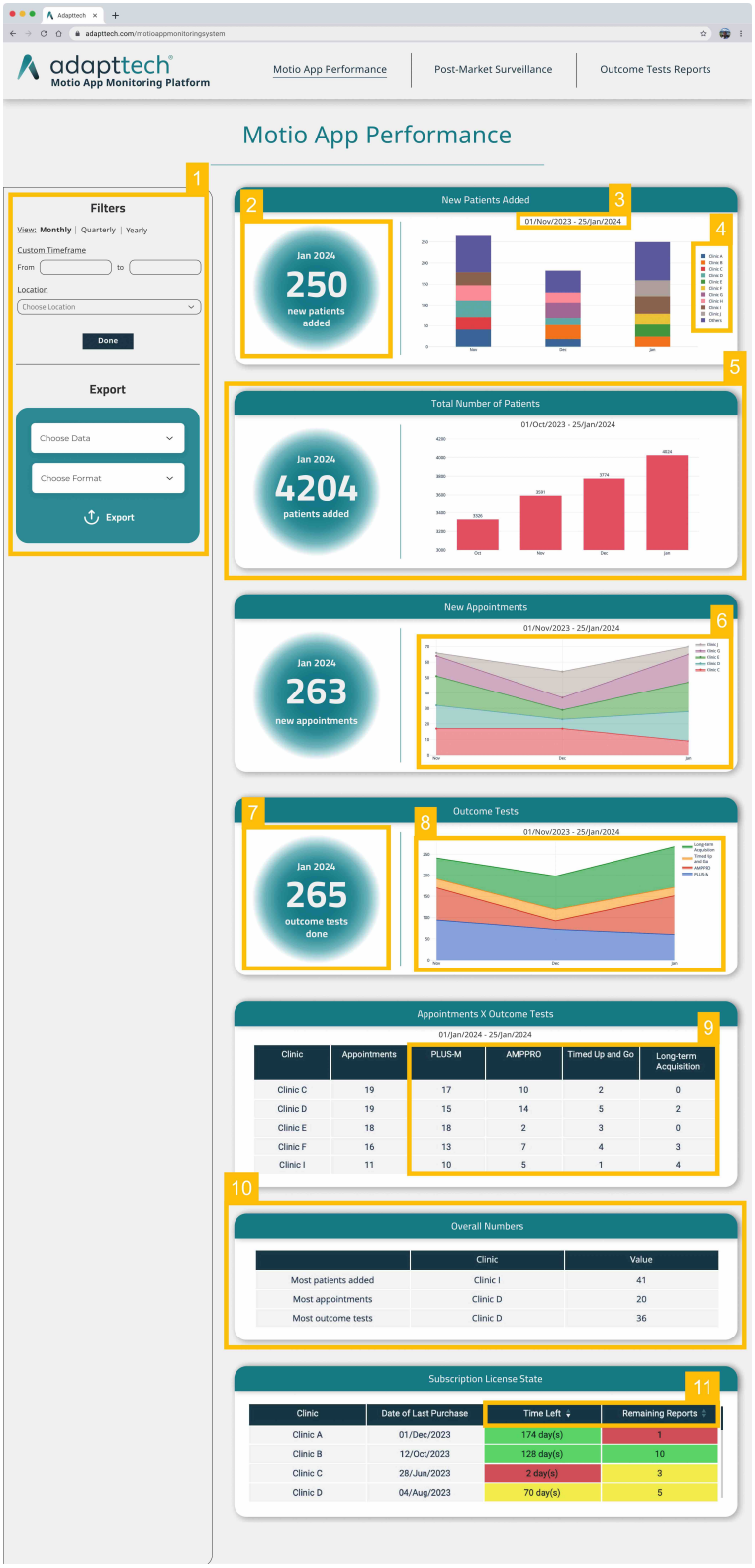


Figure 5.11: Functional prototype - Version 2 - Motio App Performance page



Figure 5.12: Functional prototype - Version 2 - Post-Market Surveillance page

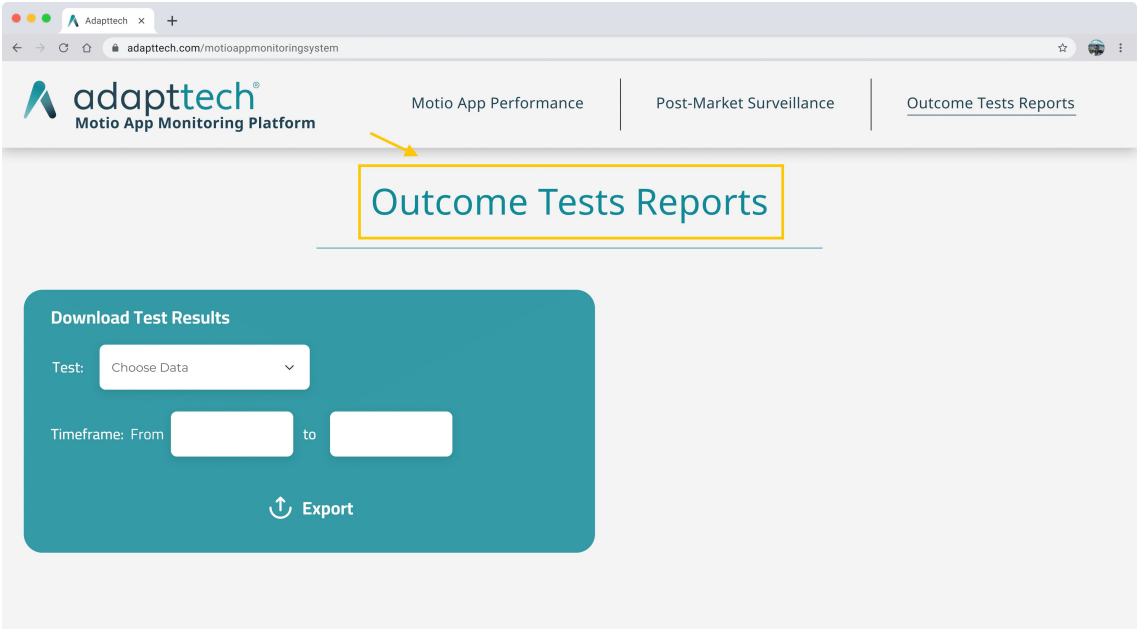


Figure 5.13: Functional prototype - Version 2 - Outcome Test Reports page

5.2.3 Hi-Fi Prototype - Prototype Version 3

From Version 2 to Version 3[3], only minor changes happened. None of them came from the stakeholders, but rather due to implementation requirements, author's suggestions, or aesthetics refinement.

Related to the implementation, it was identified a need for initial pages for both Motio App Performance and Post-Market Surveillance. That way, the system would only retrieve data requested by the user according to their specified needs through the filters, rather than querying for unnecessary data. Considering this requirement, the initial pages were designed and are shown in Figures 5.14 and 5.15.

Other modifications include the removal of the Total Number of Patients graph from the Motio App Performance page (Figure 5.16), as it was redundant information, and the Export menu incorporation to the Post-Market Surveillance page (Figure 5.17). Finally, the Download Test Results menu on the Outcome Test Reports page received some design improvements, as illustrated in Figure 5.18.

[3]Interactive Prototype - Version 3: https://figmashort.link/5zFGbN

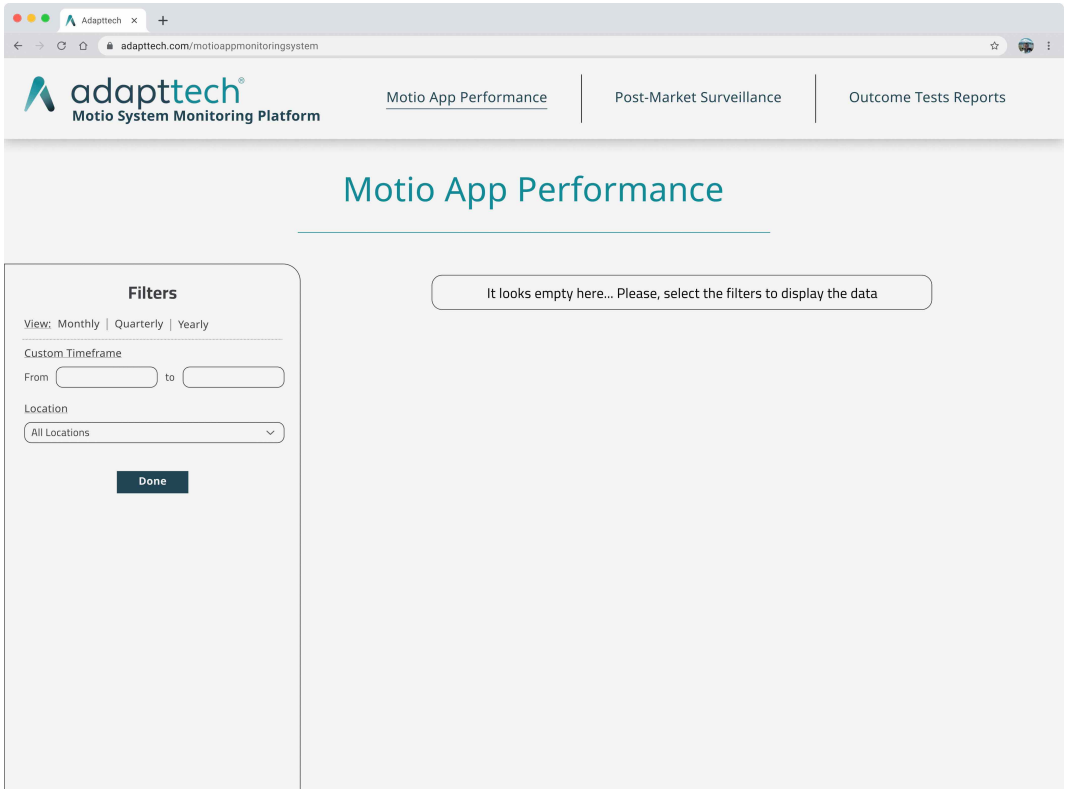


Figure 5.14: Functional prototype - Version 3 - Motio App Performance initial page

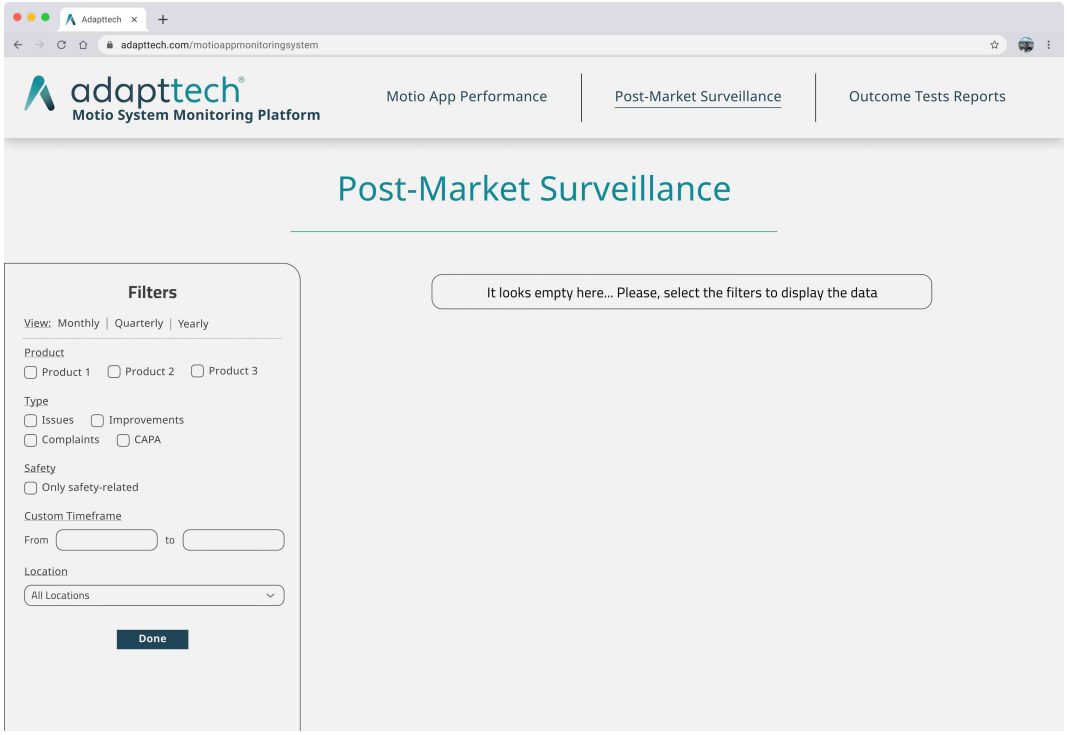


Figure 5.15: Functional prototype - Version 3 - Post-Market Surveillance initial page

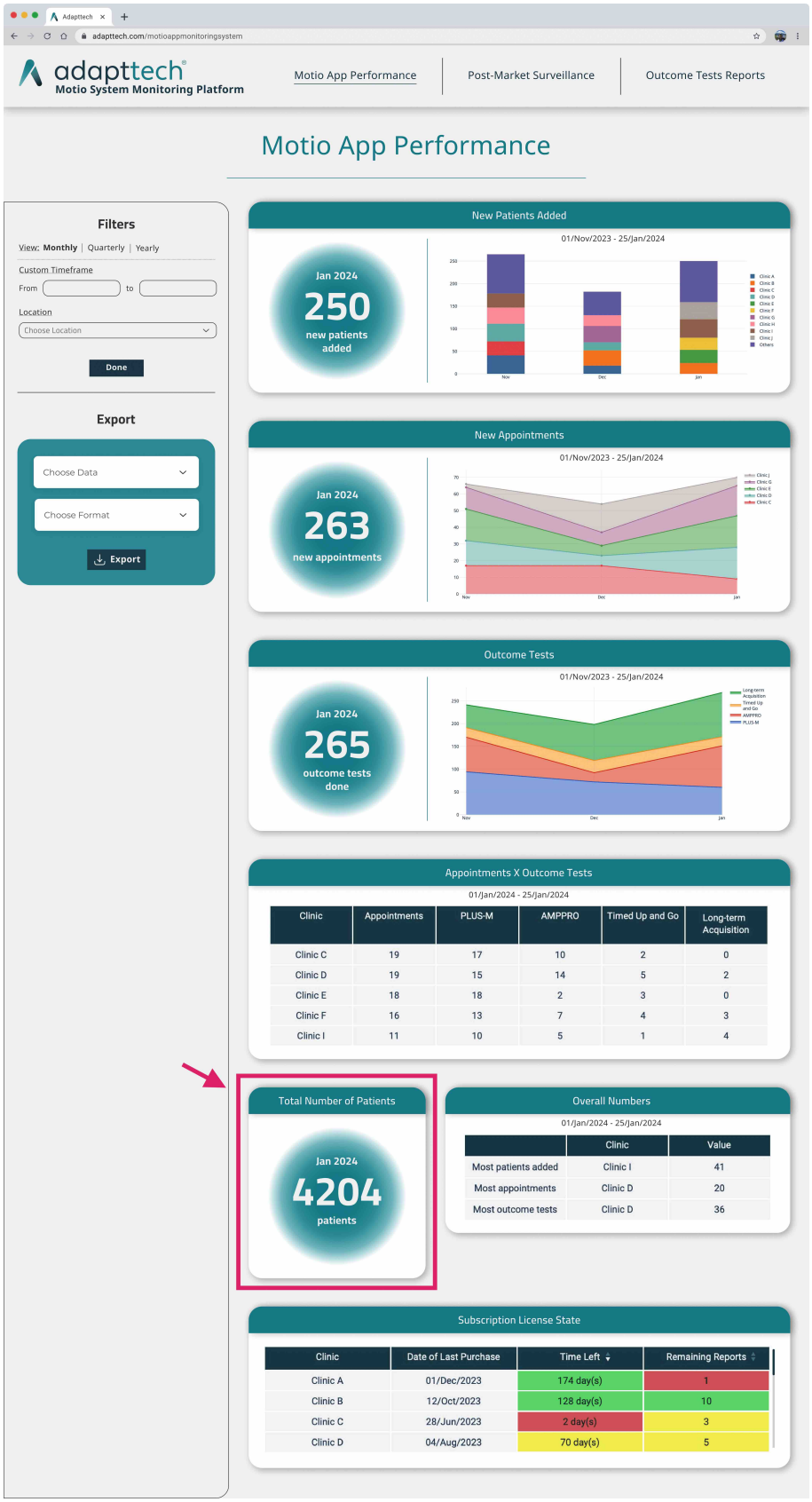


Figure 5.16: Functional prototype - Version 3 - Motio App Performance page



Figure 5.17: Functional prototype - Version 3 - Post-Market Surveillance page

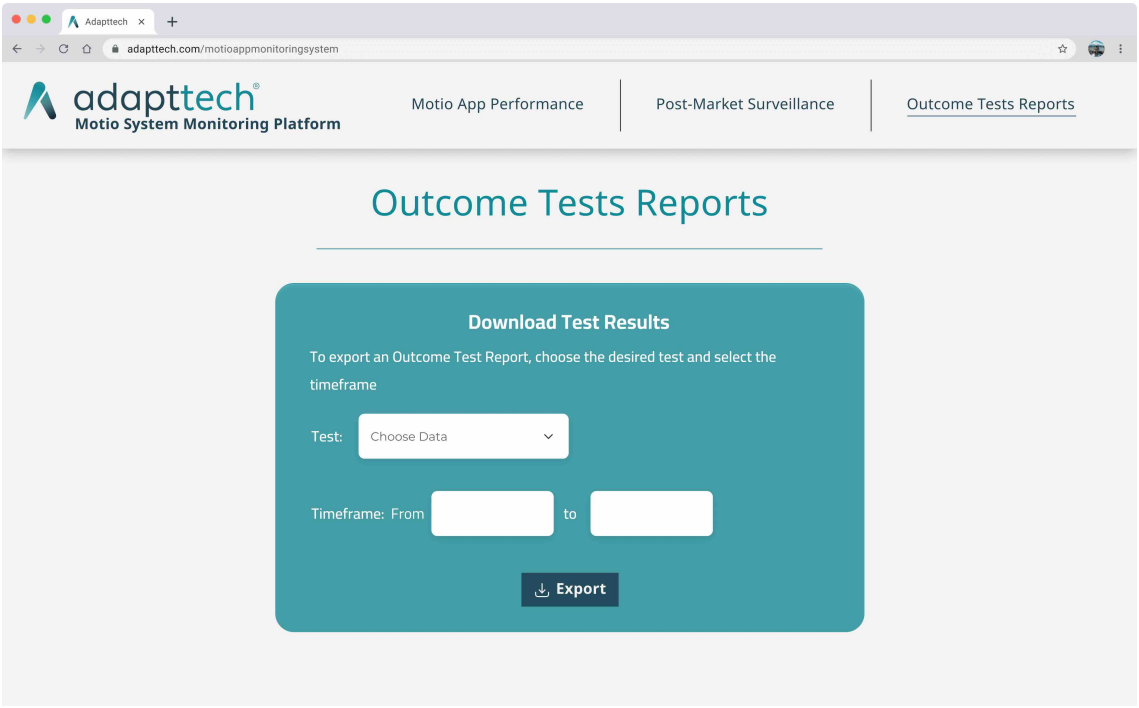


Figure 5.18: Functional prototype - Version 3 - Outcome Test Reports page

5.3 Prototype Validation - Usability Testing

5.3.1 Methodology

The Motio System Monitoring Platform Version 3 underwent usability testing based on the standard ISO/EIC 25062, Common Industry Format for Usability Test Reports. The test was conducted on 19 participants aged 20 to 30 with higher education and no previous knowledge about the Motio StepWatch™ System. Its goal was to evaluate the effectiveness, efficiency, and satisfaction of potential users regarding a few routine tasks on the platform. The effectiveness and efficiency measures were tested using a prototyping tool called Quant-UX, whereas the satisfaction was measured using a questionnaire following the standard System Usability Scale (SUS), as discussed in Section 3.4.1.

Concerning task selection, it was crucial that the tasks covered the application's overall functionality. Under these circumstances, the five chosen tasks were:

1. Check the graphics related to the Motio App Performance using the monthly, quarterly, and yearly view in this order;
2. Try to export any data in any format;
3. Choose a custom time frame and a location;
4. Check the Post-Market Surveillance Page using the monthly view or select any filter and press 'Done';
5. Export any outcome test report using any time frame.

Since the type of hardware was not significant, participants could use their personal computers to complete the test. The procedure consisted of the following three steps:

1. The participant receives two links, one for the test[4] and the other for the Satisfaction Questionnaire[5], and a small list of instructions (Appendix A.1);
2. First, the participant needs to complete the test that encompasses the aforementioned five tasks;
3. After completion, the participant needs to answer the 12-question Satisfaction Questionnaire, displayed in Figures A.1 and A.2 in Appendix A.

During the test, Quant-UX records—with consent—the participant's screen and their interactions with the application (details in Appendix A.3). After its completion, valuable information is provided automatically by the tool, such as:

- Task success rate;

[4]Usability Testing: https://shorturl.at/v60cU
[5]Satisfaction Questionnaire: https://forms.gle/kXSRLRj1RjL5TL1q8

- Task duration;
- Number of interactions;
- Scatter and box plot regarding interactions and duration for every task;
- Every task step duration and interactions;
- Overall data distribution regarding interactions and duration;
- Heatmaps;
- User journey;
- Tasks and drop off information.

5.3.2 Results - Usability Testing

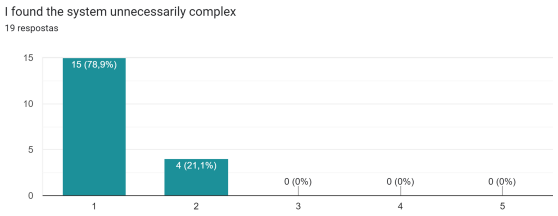
After all 19 participants had completed their tests and answered the questionnaire, the results were gathered. Table 5.4 summarizes the effectiveness and efficiency measures, while Figure 5.19 depicts the answer distribution for each question on the Satisfaction Questionnaire. For more details, Tables A.1 and A.2 in Appendix A display the results and answers per test.

Table 5.4: Summary of usability testing measures regarding effectiveness and efficiency

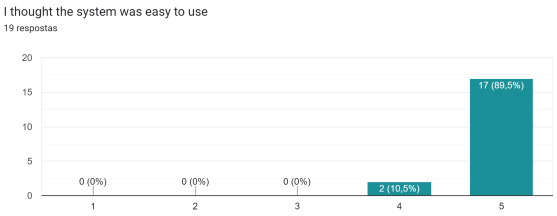
Name	Start	Success	Duration (s) (Mean)	Duration (s) (Median)	Duration (s) (STD)	Interactions (Mean)	Interactions (STD)
Task 1	19 / 19	19 / 19	49.7	33.9	27.5	19.4	17.6
Task 2	18 / 19	17 / 19	3.3	3.0	0.9	5.2	0.9
Task 3	18 / 19	16 / 19	79.2	21.5	195.0	18.0	20.7
Task 4	17 / 19	17 / 19	22.8	6.5	59.7	4.9	2.2
Task 5	16 / 19	16 / 19	13.4	11.7	8.0	12.1	6.6

In addition to the standard questions, the Satisfaction Questionnaire featured a suggestion box (Figure A.2 in Appendix A) where participants could leave commentaries or ideas. As a result, the responses were:

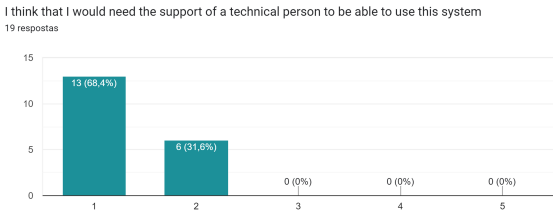
- "In the 'Outcome Reports Page', the only option provided is to Download Test Results. Perhaps the name of the page should make it clear that it's intended for downloading, e.g., 'Outcome Reports Download'." (Participant 7)
- "It would be interesting to decide which measures and in which order am I going to see. Especially, if the user will be able to export, selecting which information will be uploaded would be useful." (Participant 10)



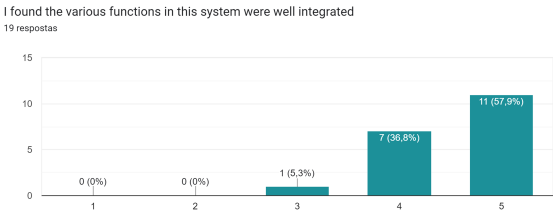
(a) Question 2



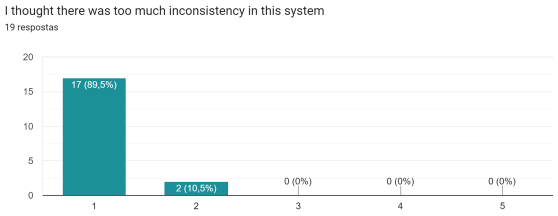
(b) Question 3



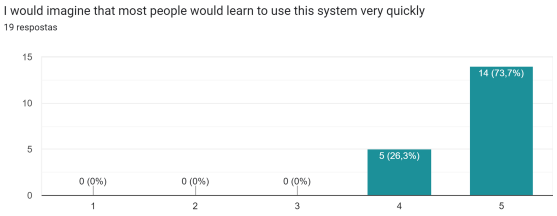
(c) Question 4



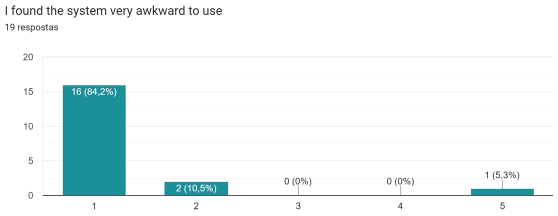
(d) Question 5



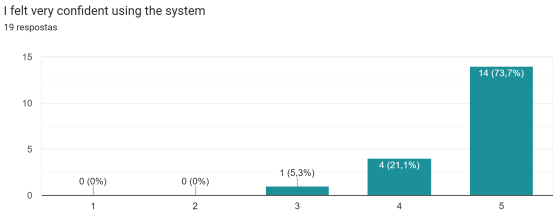
(e) Question 6



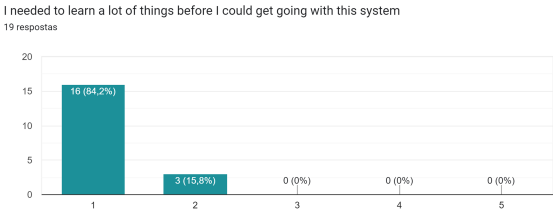
(f) Question 7



(g) Question 8



(h) Question 9



(i) Question 10

Figure 5.19: Distribution of answers per question in the Satisfaction Questionnaire

- "The dates shouldn't be able to be selected in inverse order (e.g., it is possible to select from 17/06 to 03/06)." (Participant 14)
- "When selecting custom time frames, it is allowed to choose a second date before the first one." (Participant 23)

The possibility of picking inverse dates, mentioned in the last two responses, was fixed during implementation (Chapter 6), as the chosen date picker only allows valid ranges of dates.

As indicated in Table 5.4, the majority of participants had no trouble completing the tasks. However, three participants could not start Task 5, and upon checking their records, it was discovered that the issue was finding the correct page. This struggle could be due to its name, "Outcome Test Reports", which was not sufficiently clear, as suggested by the commentary from Test 7. Almost all the other uncompleted tasks resulted from the participant's failure to follow the task descriptions or the testing platform's inflexibility since it does not mark them as completed if the user takes a different path than the one specified in the settings.

Concerning the Satisfaction Questionnaire, it is important to highlight that question 1 ("I think that I would like to use this system frequently") was excluded from the analysis since none of the participants currently work at Adapttech and thus will not have access to this platform. Then, the conclusion of this questionnaire, as evidenced by Figure 5.19, is that most participants were satisfied with the prototype's design, with 89.5% believing the system was easy to use.

Other testing outcomes provided by Quant-UX, such as box plots, data distribution, and heatmaps, can be seen in Appendix A.6. An interesting takeaway from these additional outcomes is the boxplot from Figure A.4, as it shows that Task 3 required the most interactions and time from the participants. Moreover, this information is consistent with Table 5.4 since this task had the least success when considering the number of started tasks (16 completed tasks to 18 started tasks). Figure A.7 also provides useful insights as this heatmap shows that almost all participants selected "Done" after pressing the "Quarterly" view, which is not required. This misstep should be rectified during implementation by, for example, removing the button "Done" and refreshing the data automatically.

5.4 Summary

In this chapter, the prototyping process of this project was detailed, from the wireframe creation based on the requirements to the three-iteration development of the functional prototype. After every prototype evaluation, feedback was collected and used to correct any potential issues. As for the last step, the final prototype was validated by conducting on real participants a usability test that assessed both the performance and the ease of use. This validated prototype will be then employed in the implementation as a base for the aesthetic components and overall functionality.

Tables 5.5, 5.6, 5.7, and 5.8 summarizes how the final prototype version satisfied the collected requirements.

Table 5.5: Traceability Matrix after Prototyping - Motio App Performance requirements

ID	Description	Evidence	Observations
ERM1	Be able to view how many new patients were added into the system in a Monthly, Quarterly, or Yearly view, with the option to group by location over time.	Figure 5.16	"Group by" turned into a filter
ERM2	Be able to view how many new appointments were created in a Monthly, Quarterly, or Yearly view, with the option to group by location over time. Present in tabular and graphic form.	Figure 5.16	"Group by" turned into a filter; Tabular was incorporated into Appointments X Outcome Tests (Section 5.1.1)
ERM3	Be able to view the amount (absolute and relative to the total number of appointments) of outcome tests performed in the Motio App in a Monthly, Quarterly, or Yearly view, with the option to group by location.	Figure 5.16	Relative number removed (Section 5.2.1.1); "group by" turned into a filter
ERM4	Be able to export any data being viewed (through graphics or table) as a .csv or other similar format.	Figure 5.16	-
ERM5	Be able to export viewed graphics as a .png or .pdf.	Figure 5.16	-
ERM6	Be able to monitor the subscription license state per clinic, for example, the date of last purchase and available stock of reports.	Figure 5.16	"Time Left" added
ORM1	Be able to monitor the report purchase/license history of a specific clinic.	-	Not considered initially

Table 5.6: Traceability Matrix after Prototyping - Post-Market Surveillance requirements

ID	Description	Evidence	Observations
ERP1	Be able to view how many new issues, improvements, complaints, or CAPAs have been reported in a monthly, quarterly, or yearly view, with the option to group by location and by product/system.	Figure 5.17	"Group by" turned into a filter
ERP2	Be able to view how many issues, improvements, complaints, or CAPAs related to safety were reported in a monthly, quarterly, or yearly view, with the option to group by location and by product/system.	Figure 5.17	Turned into a filter
ERP3	Calculate the closing times of issues, improvements, complaints, or CAPAs and the average for each month, quarter, or year, with the option to group by product/system.	Figure 5.17	"Group by" turned into a filter
ERP4	Show the number of open complaints and CAPAs per month, quarter, and year, with the option to group by product/system and severity.	Figure 5.17	Included all feedback types and comparison with new tickets (Section 5.1.1); "Group by" turned into a filter
ERP5	Calculate the percentage of complaints closed in less than 1 week and less than 1 month. View monthly, quarterly, and yearly and group by severity, safety-related, and product/system.	Figure 5.17	"Total number of Complaints" added (Section 5.2.1.1
ORP1	Count the number of times that the user canceled the outcome measure process	-	Not considered initially
ORP2	Track time of execution for each outcome measure	-	Not considered initially
ORP3	Show which complaints and CAPAs are still open per month, quarter, and year, with the option to group by product/system and severity.	-	Not considered initially
ORP4	Detect how many appointments quarterly and yearly were not closed after 30 days.	-	Not considered initially
ORP5	Detect how many patients do not have an appointment associated, monthly, quarterly, and yearly.	-	Not considered initially
ORP6	Detect how many times the button for the Instructions for Use was clicked on monthly, quarterly, and yearly.	-	Not considered initially
ORP7	Be able to detect how many unsuccessful login attempts were made monthly, quarterly, and yearly.	-	Not considered initially

Table 5.7: Traceability Matrix after Prototyping - Clinical Insights requirements

ID	Description	Evidence	Observations
ERC1	Be able to download in tabular form the PLUS-M results performed during a given time frame, together with the test date and some information about the patient, such as gender, date of birth, amputation type and etiology, AMPPRO score, TUG duration, K level, 10-meter walk test time, and 2-minute or 6-minute walk test distance.	Figure 5.18	-
ERC2	Be able to download in tabular form the AMPPRO results performed during a given time frame, together with the test date and some information about the patient, such as date of birth and amputation, amputation type and etiology, and K level.	Figure 5.18	-
ORC1	Be able to download in tabular form the Timed Up and Go result, date of test, and risk of fall performed during a given time frame	Figure 5.18	-
ORC2	Be able to download in tabular form the Long-term acquisition results, and date of test performed during a given time frame	-	Not considered initially
ORC3	Be able to download the Motio Functional level in a tabular form.	-	Not considered initially
ORC4	Be able to download the Clinical Assessment in a tabular form	-	Not considered initially

Table 5.8: Traceability Matrix after Prototyping - Other requirements

ID	Description	Evidence	Observations
ERO1	Be able to filter collected data by a specific time frame and time resolutions (monthly, quarterly, and yearly).	Figures 5.16, 5.17	-
ERO2	The application must only be accessible from within the company network.	-	Not related to the User Interface
ORO1	Be able to set a default time frame in which the user can view data (Last Quarter, Last 6 Month, Last Year) in this Dashboard app	-	Not considered initially
ORO2	Settings configured by the user of the Dashboard app should be saved locally	-	Not considered initially
ORO3	The Application should be accessible online through authentication using a Microsoft account.	-	Not related to the User Interface
ORO4	When using color to represent a location, severity, or product, it must be consistent within the App at that moment.	Figures 5.16, 5.17	-
ORO5	When using color to represent a location, severity, or product, it must be consistent in the App overtime.	Figures 5.16, 5.17	-

Chapter 6

Implementation

After the prototyping is developed and validated, its implementation can start. But first, Section 6.1 creates a plan for the implementation. Then, Sections 6.2, 6.3, and 6.4 address each of the plan steps: Data Collection, Data Preparation and Analysis, and Data Visualization. Once the implemented version is ready, it must be deployed to make it accessible to the users, as explained in Section 6.5. At last, the chapter ends with the implementation results in Section 6.6.

6.1 Implementation Architecture

The architecture of this project's implementation was based on a typical BI application, as described in Subsection 2.2.2. Its scheme is illustrated in Figure 6.1, which divides the phases by BI steps (data collection, preparation, analysis, and visualization) and by software engineering concepts (backend and frontend).

The project primarily employed Python 3 for implementation, with the additional use of HTML and CSS during the data visualization phase. After its conclusion, the user will be able to access it through a web browser.

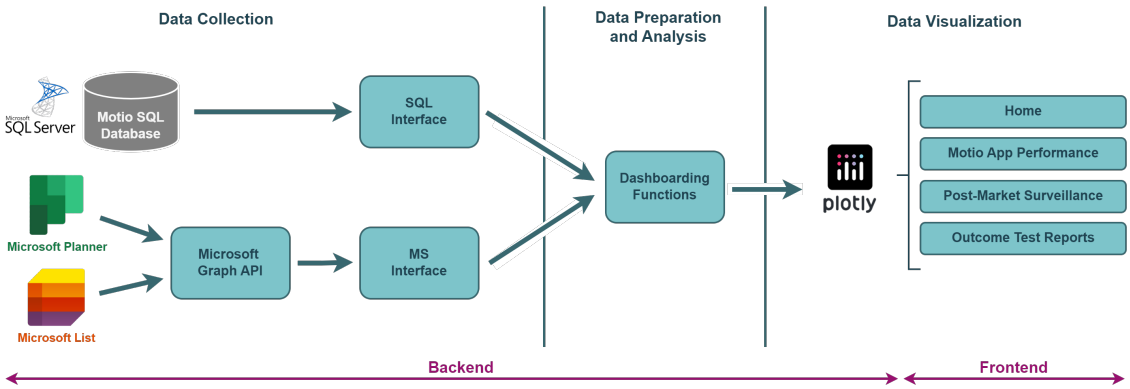


Figure 6.1: Motio System Monitoring Platform implementation architecture

6.2 Backend Implementation - Data Collection

As the name suggests, Data Collection is the phase in which data is collected from the databases—Motio SQL database, Microsoft Planner, and Microsoft Sharepoint List, in this case. Therefore, two classes were created: *SQL Interface* and *MS Interface*, illustrated as a UML Class Diagram in Figure 6.2. They are responsible for obtaining the Motio App and Feedback data, respectively, and their operation was schematized as a UML Component Diagram in Figures 4.10 and 4.12 in Section 4.2.2.3.

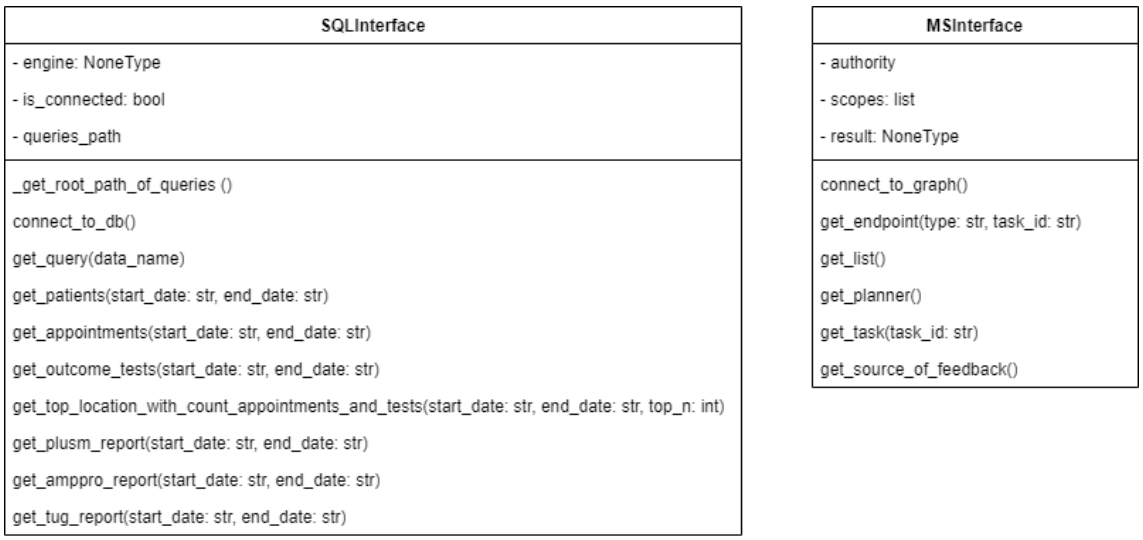


Figure 6.2: UML Class Diagram - SQL and MS Interfaces

6.2.1 SQL Interface

First, it is essential to know that the Motio App database is hosted on *Microsoft Azure SQL Server*. However, for the development, a local version was employed as it is simple to switch to the server version later. Since the connection to this server is done effortlessly using *pyODBC*, this module was chosen when creating the *SQL Interface*. Besides that, Microsoft invests its testing resources and confidence in this module, even though several other options are available. In short, *pyODBC*[1] is a Python module capable of easily connecting to ODBC (Open Database Connectivity) databases, which include Microsoft Azure SQL Server, thanks to its dedicated ODBC driver, the Microsoft ODBC Driver for SQL Server. More about the information required to establish a database connection using *pyODBC* is detailed in Section 4.2.2.3.

In addition to *pyODBC*, *SQLAlchemy*[2] is another package employed in the connection portion of *SQL Interface* as it is an Object Relation Mapper (ORM) toolkit that enhances and expands SQL database management capabilities. It helps in data conversion between relational databases and object-oriented programming languages, allowing developers to reduce the amount of database

[1] https://mkleehammer.github.io/pyodbc/
[2] https://www.sqlalchemy.org/

accesses while increasing productivity. To connect to the SQL database, SQLAlchemy employs pyODBC as its default Python Database API Specification, responsible for the bridge between the database and the Python application. In this project, *SQLAlchemy* was adopted as it allows the conversion of the collected data into *Pandas* dataframes.

Even though the phase of data preparation and analysis comes next, instead of transferring the whole dataset and transforming it to the desired format with processing libraries, such as polars or pandas, it was decided that the data would be collected already transformed using SQL Queries. Under this circumstance, the application would handle less data and, as a result, improve its efficiency. However, the queries, written beforehand and disclosed in Appendix B, are not excessively specific and only concern data subject and date of patient/appointment creation. For example, if the user wants to see the new appointments from this month, the collected appointment data only includes when and where the appointment took place filtered by this month's dates. Therefore, the available data subjects in *SQL Interface* are all related to the graph cards on the Motio App Performance page, which are:

- Patients;
- Appointments;
- Outcome tests;
- Locations with the most appointments and their performed outcome tests;
- PLUS-M™ report;
- AMPPRO® report; and
- TUG report.

Therefore, all of them can be filtered by the dates chosen by the user through the monthly, quarterly, or yearly view or through the custom time frame. Another functionality provided by *SQL Interface* is database connection logs for easier error handling when needed.

6.2.2 MS Interface

As for the *MS Interface*, the library in charge of the Microsoft Planner and Microsoft Sharepoint List connection is the Microsoft Authentication Library (MSAL) for Python[3]. It allows the acquisition of security tokens from Microsoft Entra ID to enable access to protected web APIs, such as Microsoft Graph, the data portal of several Microsoft products and services, including the Planner and the Sharepoint List. Section 4.2.2.3 provides more details about the required identifiers to access Microsoft Graph API. Then, to obtain the data from these locations, the *requests*[4] package was necessary. It enables Python to make HTTP requests, which are vehicles of communication between the client—the Motio System Monitoring Platform, in this case—and the server on the

[3] https://learn.microsoft.com/en-us/entra/msal/python/
[4] https://requests.readthedocs.io/en/latest/

web—Microsoft Planner and SharePoint List—and, by employing the method GET, it can retrieve the data from the server.

On the one hand, *MS Interface* has similar features to *SQL Interface*, such as connection logs for quickly addressing errors. On the other hand, in contrast, it collects the whole data by storing them in *Pandas* dataframes for easy manipulation afterward. Another important detail, as explained in Section 4.2.2.3, is that the Planner items do not contain their description field, where the IDs shared with the Sharepoint List items are. This information is, in reality, located in the Task items. For this reason, three data retrievals are required: the first for the Sharepoint List items, the second for the Planner items, and the third and last for the Task Items.

6.3 Backend Implementation - Data Preparation and Analysis

The steps of the Data Preparation and Analysis phase depend on the required data. Therefore, to make it simpler, the graph cards were grouped by their necessary preparation and analysis, as listed below.

- **Group 1**: New Patients Added, New Appointments Created, Outcome Tests Performed;
- **Group 2**: Appointments x Outcome Tests;
- **Group 3**: Overall Numbers;
- **Group 4**: New Issues, Improvements, Complaints, and CAPA; Closed Issues, Improvements, Complaints, and CAPA; Open X New Tickets; and Average Closing Times.

Some graph cards, namely Subscription License State from the Motio App Performance Page (*Requirement ERM6*) and Closed Complaints from the Post-Market Surveillance (*Requirement ERP5*), were not addressed during implementation. The former is due to the still ongoing development of license management capability at Adapttech, whereas the latter is due to the lack of complaint data. As of June 15th, 2024, there were only three closed complaint tickets, which is insufficient to build the proposed table, and mock data could not be created due to time constraints. Additionally, the data relative to the outcome tests, namely PLUS-M, AMPPRO, and TUG, did not require transformation since the queried data was already in the required format, i.e., filtered by test type and date, as shown in Appendix B.

For Group 1, Table 6.1 summarizes its Data Preparation and Analysis steps. The first step is to verify whether the retrieved dataframe from the *SQL Interface* is not empty for the dates specified by the user. If not, all date and time data is converted to *Pandas datetime* object and to Coordinated Universal Time (UTC) timezone to allow future filtering. Next, if the user chooses a location, the data is filtered to just include that location. At last, it calculates the absolute count of Patients/Appointments/Outcome Tests and produces the graph data. In the case of patients and appointments, the graph data corresponds to the five locations with the most patients or appointments considering the last three periods (months for the monthly view, quarters for the quarterly view, and years for

the yearly view) together or during the custom time frame and their corresponding numbers, while the other locations are grouped in "Others", as shown in Figure 6.3a. As for the outcome tests, the number of each test type taken in the last three periods or within the custom time frame is returned (Figure 6.3b).

Table 6.1: Group 1 - Steps of the Data Preparation and Analysis

Step	Description
1	Checks if the returned dataframe is empty
2	Converts date and time data to *datetime* and converts timezone
3	Filters by Location, if selected
4	Calculates the count of new patients/appointments/outcome tests
5	Processes patients/appointments/outcome tests graph data

LocationName	date	count
Location15	2024-03-31	13
Location17	2024-03-31	5
Location17	2024-06-30	2
Location2	2023-12-31	16
Location2	2024-03-31	16
Location2	2024-06-30	7
Location6	2024-03-31	8
Location6	2024-06-30	3
Location9	2023-12-31	1
Location9	2024-03-31	5
Others	2023-12-31	9
Others	2024-03-31	7
Others	2024-06-30	6

(a) Appointments graph data - Example of the Quarterly view (June 2024)

name	date	count
AMPPRO	2023-12-31	17
AMPPRO	2024-03-31	3
AMPPRO	2024-06-30	2
ClinicianEvaluation	2023-12-31	17
ClinicianEvaluation	2024-03-31	21
ClinicianEvaluation	2024-06-30	8
LTA	2023-12-31	21
LTA	2024-03-31	34
LTA	2024-06-30	12
PLUS-M	2023-12-31	15
PLUS-M	2024-03-31	3
PLUS-M	2024-06-30	3
TUG	2023-12-31	15
TUG	2024-03-31	7
TUG	2024-06-30	5

(b) Outcome Tests graph data - Example of the Quarterly view (June 2024)

Figure 6.3: Group 1 graph data

Since the query used to obtain the Appointments X Outcome Tests (Group 2) data from the *SQL Interface* already returns the required data, as seen in Appendix B.4, the only processing required is the filtering when the user selects a location. Similarly, Group 3 also needs less processing, as the Overall Numbers uses the already pre-processed data from Group 1; hence, it simply concatenates the locations with the most patients, appointments, and outcome tests into the same table.

Finally, Group 4, which encompasses all graph cards from the Post-Market Surveillance page, starts by converting the acquired data to *Pandas* dataframes for easy management while changing the column names to make them more understandable and removing unnecessary information. Analogous to Group 1, there is the date and time data transformation step, in which they were converted to *Pandas datetime* objects and their timezones converted. Then, the List, Planner, and Task dataframes are joined together using their shared IDs, as illustrated in the Class Diagram

in Figure 4.13. In the event of the user selecting any available filter (product, type of feedback, severity, safety related, or source of feedback), the data is filtered accordingly; otherwise, it is only filtered using the selected view (monthly, quarterly, yearly, or custom). As the last step, the absolute number of new, closed, and open tickets and the graph data are determined. For the New and Closed graph cards, the graph data includes the amount of each feedback type (Issue, Improvement, Complaint, and CAPA) from the last three periods, as illustrated in Figure 6.4a. In comparison, the Open X New Tickets graph data concerns the count of new and open tickets from the last three periods or between the selected time frame (Figure 6.4b), and the Average Closing Times one calculates the number of days required to close each feedback (Figure 6.4c). Table 6.2 provides an outline of this group's Data Preparation and Analysis.

Table 6.2: Group 4 - Steps of the Data Preparation and Analysis

Step	Description
1	Converts the retrieved data to dataframe
2	Converts date and time data to *datetime* and converts timezone
3	Joins List, Planner, and Task items using IDs
4	Filters using selected dates
5	Filters by Product, Type of Feedback, Severity, Safety Related, and Source of Feedback, if selected
6	Calculates the count of new/closed/open tickets
7	Processes graph data

feedback_type	createdDateTime	count
Complaint	2024-03-31	1
Complaint	2024-06-30	4
Improvement	2024-03-31	30
Issue	2024-03-31	14
Issue	2024-06-30	4

(a) New issues graph data - Example of the Quarterly view (June 2024)

Type	lastdayofperiod	count
open	2023-12-31	0
new	2024-03-31	45
open	2024-03-31	44
new	2024-06-30	8
open	2024-06-30	49

(b) Open X New graph data - Example of the Quarterly view (June 2024)

feedback_type	closing_time
Issue	67
Complaint	40
Complaint	3

(c) Average Closing Times graph data - Example of the Quarterly view (June 2024)

Figure 6.4: Group 4 graph data

6.4 Backend and Frontend Implementation - Data Visualization

6.4.1 Available Technologies - Data Visualization

The Data Visualization phase requires a tool capable of transforming complex data into user-friendly dashboard applications. With this motivation in mind, the following tools were considered:

- Dash;
- Streamlit;
- Panel;
- Shiny;
- Voila.

Figure 6.5 summarizes these all-in-one dashboarding solutions' popularities, determined by the amount of GitHub stars.

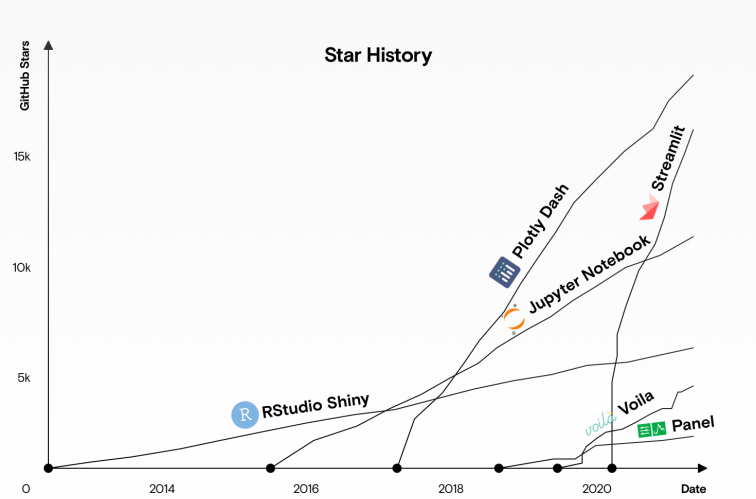


Figure 6.5: Popularity of data visualization tools[5]

6.4.1.1 Dash

Dash[6], developed by Plotly, is a low-code framework designed for building web-based data visualization applications. It allows the creation of interactive charts, graphs, and tables in complex, multi-page applications with real-time updates.

[5] https://www.datarevenue.com/en-blog/data-dashboarding-streamlit-vs-dash-vs-shiny-vs-voila. Extracted on 2024-06-16

[6] https://dash.plotly.com/

Since it supports HTML and CSS, its applications are highly customizable. Alongside the frontend aspect, Dash also engages in the application backend by managing requests and responses via Flask and handling callback functions and data processing. A major contribution to its use is the availability of comprehensive documentation with complete tutorials and guides.

Dash was the chosen tool for developing the data visualization of the Motio System Monitoring Platform.

6.4.1.2 Streamlit

Regarded as the favorite among data scientists, Streamlit[7] is an open-source Python dashboarding solution that enables the creation of interactive applications. It is compatible with most Python packages and provides free deployment using its native infrastructure.

On the one hand, Streamlit attracts developers for only requiring minimal code, making it suitable for rapid prototyping. On the other hand, this simplicity restricts its customization as it only offers a limited set of widgets.

6.4.1.3 Panel

Essentially, Panel[8] is an open-source Python library developed by HoloViz that provides straightforward features for simple applications to sophisticated, interactive multi-page projects. Its main advantages include better-than-average integration with Jupyter Notebooks and seamless cooperation with PyData tools, such as Numpy and Pandas. In contrast, Panel may become a hamper in increasingly complicated applications, even requiring a more in-depth grasp of its architecture and reactive programming model. Furthermore, when employing different visualization libraries, the dependencies can be challenging to manage and assure compatibility.

6.4.1.4 Shiny

Created by RStudio, Shiny[9] is an open-source R library that allows the building of interactive web applications. The main contrast between Shiny and the aforementioned tools is the use of R, a data analytics language.

6.4.1.5 Voila

Differently from the other, Voila[10] is a tool capable of converting Jupyter notebooks into basic dashboards in web applications. Therefore, it requires the use of Jupyter, which can become a hindrance.

[7] https://streamlit.io/
[8] https://panel.holoviz.org/
[9] https://www.rstudio.com/products/shiny/
[10] https://voila.readthedocs.io/en/stable/index.html

6.4.2 Requirements Implementation

As established in the final prototype, the Motio System Monitoring Platform is a multi-page application, more specifically with four pages:

- Home page;
- Motio App Performance page;
- Post-Market Surveillance page;
- Outcome Test Reports page.

To accomplish this in Dash, a main code `app.py` must be created. Then, the pages' code files, located in the `pages/` folder, must be registered in this main code, which is also responsible for building the navigation bar at the top of the page. The name of the Outcome Test Reports page on the navigation bar was changed to "Download Test Reports" to facilitate its location, as discussed in Section 5.3.2, and the application name was changed to Motio System Monitoring Platform, as mentioned in Section 5.1.

Since the Home page has no functional aspects other than the navigation bar created in the `app.py` file, a section describing its implementation was judged irrelevant. Another important point to highlight is that CSS was used to design all pages' elements, such as font, colors, buttons, menus, checkboxes, radio items, and dropdowns; however, this document will not go into much detail about this customization.

6.4.2.1 Motio App Performance Page Implementation

To understand the implementation of the Motio System Monitoring Platform, it is essential to first comprehend how Dash interactions work. Dash provides three types of components: interactive components (buttons, dropdowns, checklists, radio items, and so on), components whose behavior may depend on the interactions (graphs, tables, and text), and background components (state and download). When a user interacts with a button, for example, it triggers a callback that can change a behavior-dependable component. Hence, all components must have an ID to distinguish them when working as an input, state, or output of a callback. Input corresponds to components that, when interacted, trigger the callback; state components can provide data but do not trigger the callback; and output refers to the behavior-dependable components that will receive the information from the callback. In addition to the function designation, the component property must be identified, which essentially refers to the information extracted from, in the case of input and state, or delivered to, in the case of output, the component. They may include the following types: children (content), style, value, data, n_clicks (number of times the button was clicked), start_date, end_date, and figure. Table 6.3 displays all components employed in the Motio App Performance page.

As indicated in this Table, three callbacks were necessary to create this page. A scheme of the whole procedure of callbacks in the Motio System Monitoring Platform is depicted in Figure 6.6.

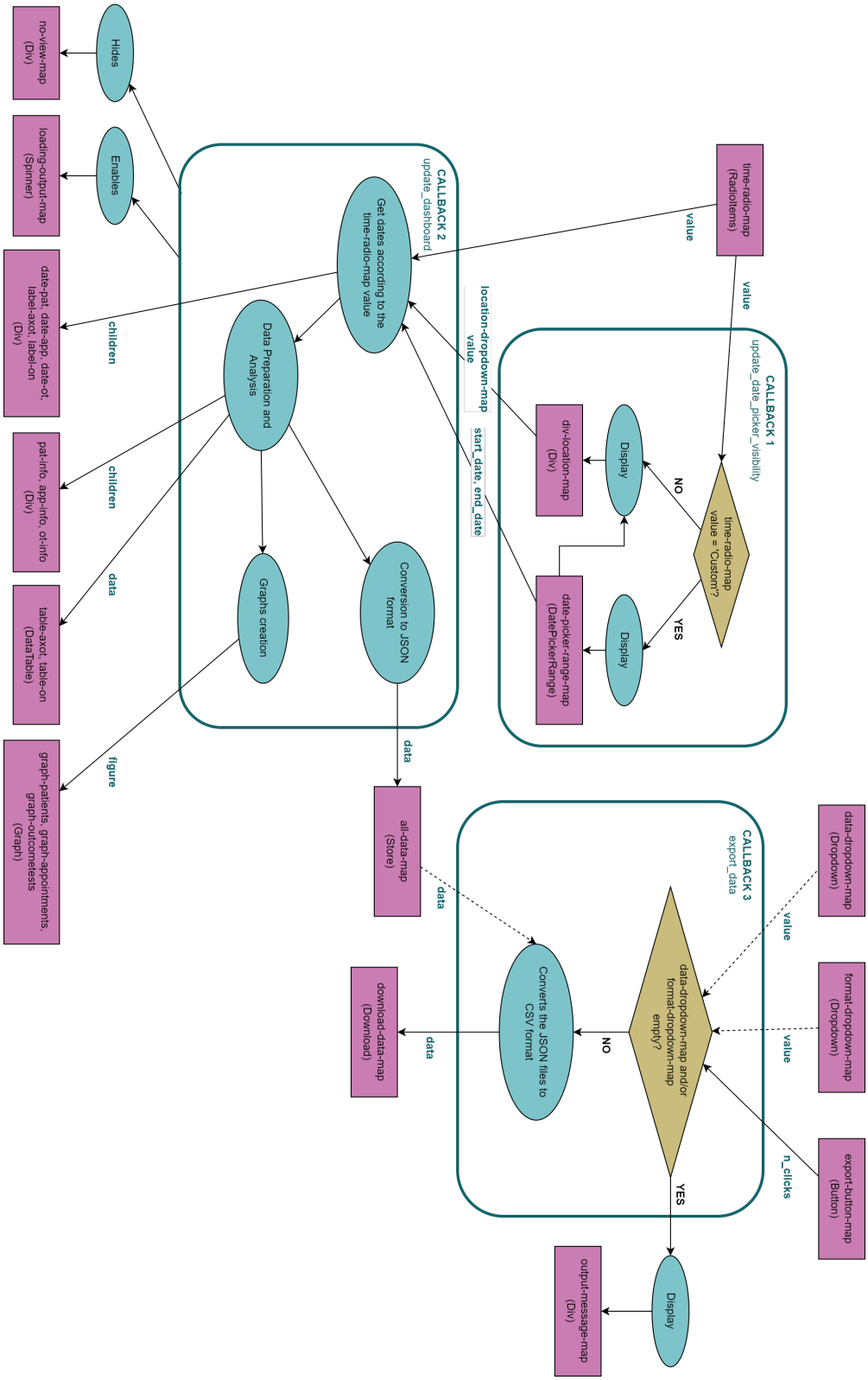


Figure 6.6: Callback diagram of the Motio App Performance page

Table 6.3: Used components and their IDs in the Motio App Performance page

ID	Type	Description	Callback 1	Callback 2	Callback 3
time-radio-map	RadioItems	Allows the user to choose from Monthly, Quarterly, Yearly or Custom view	Input (value), Output (value)	Input (value)	-
div-location-map	Div	Grouping of the label "Location:" and the location dropdown	Output (style)	-	-
date-picker-range-map	DatePickerRange	Allows the user to select the date range	Output (style)	Input (start_date, end_date)	-
location-dropdown-map	Dropdown	Allows the user to choose the location	-	Input (value)	-
loading-output-map	Spinner	Displays a spinner when content is loading	-	Output (children)	-
no-view-map	Div	Initial text with instructions for the user to select a view option	-	Output (style)	-
date-pat, date-app, date-ot, label-axot, label-on	Div	Text with the selected date	-	Output (children)	-
pat-info, app-info, ot-info	Div	Text with the total number of patients/appointments/outcome tests	-	Output (children)	-
graph-patients, graph-appointments, graph-outcometests	Graph	Graph with the patients/appointments/outcome tests data	-	Output (figure)	-
table-axot, table-on	DataTable	Data to fill the Appointments X Outcome Tests and Overall Numbers tables	-	Output (data)	-
all-data-map	Store	Stores all data between callbacks to be exported, if necessary	-	Output (data)	State (data)
export-button-map	Button	Allows the user to begin the export of their required data	-	-	Input (n_clicks)
data-dropdown-map, format-dropdown-map	Dropdown	Allows the user to choose the data and the format they want to export	-	-	State (value)
download-data-map	Download	Sends the data to export	-	-	Output (data)
output-message-map	Div	Informs the user to fill in the required fields (data or format) when exporting	-	-	Output (children)

When a `time-radio-map` RadioItems option is selected, the first callback is triggered. It displays the `div-location` Div and, if the chosen option is "Custom", the `date-picker-range-map` DatePickerRange as well. This decision to hide both components improves usability as it prevents errors from the users, such as picking a location without a chosen time view or inserting a custom time frame even when a view is already selected.

Simultaneously to Callback 1, Callback 2 is also triggered with the selection of `time-radio-map`. This callback begins by hiding the initial message (`no-view-map` Div) and setting the dates according to the `time-radio-map` RadioItems value and, if applicable, the `date-picker-range-map` DatePickerRange start and end dates. Next, the data is prepared and analyzed, following the description in Section 6.3, and sent to build the graph cards. However, this process may take some time; thus, a spinner is activated during the data loading to indicate something is happening in the background. Moreover, this same data is also transformed to JSON format and sent to the `all-data-map` Store, which, as the name implies, stores the data in case the Callback 3 is triggered.

Finally, Callback 3 concerns the export feature. It is triggered when the `export-button-map` Button is pressed, which could result in the user pressing the button without choosing a data and a format. Therefore, the first thing this callback does is check whether these Dropdowns (`data-dropdown-map` and `format-dropdown-map`) returned

any values. If not, an error message is displayed through the `output-message-map` Div; if yes, the required data is obtained from the `all-data-map` Store and delivered to the `download-data-map` Download to begin the export on the user's device.

6.4.2.2 Post-Market Surveillance Page Implementation

Regarding the Post-Market Surveillance page, Table 6.4 shows the components employed on this page. As Figure 6.7 illustrates, its callbacks are significantly similar to the callbacks from the Motio App Performance page. The main difference is the display of the div-product Div, div-feedback-type Div, div-severity Div, and div-safety Div alongside the div-location Div when an option of the time-radio-pms RadioItems is selected. Consequently, this impacts as well the data preparation and analysis since it allows the filter based on product, feedback type, severity, and safety.

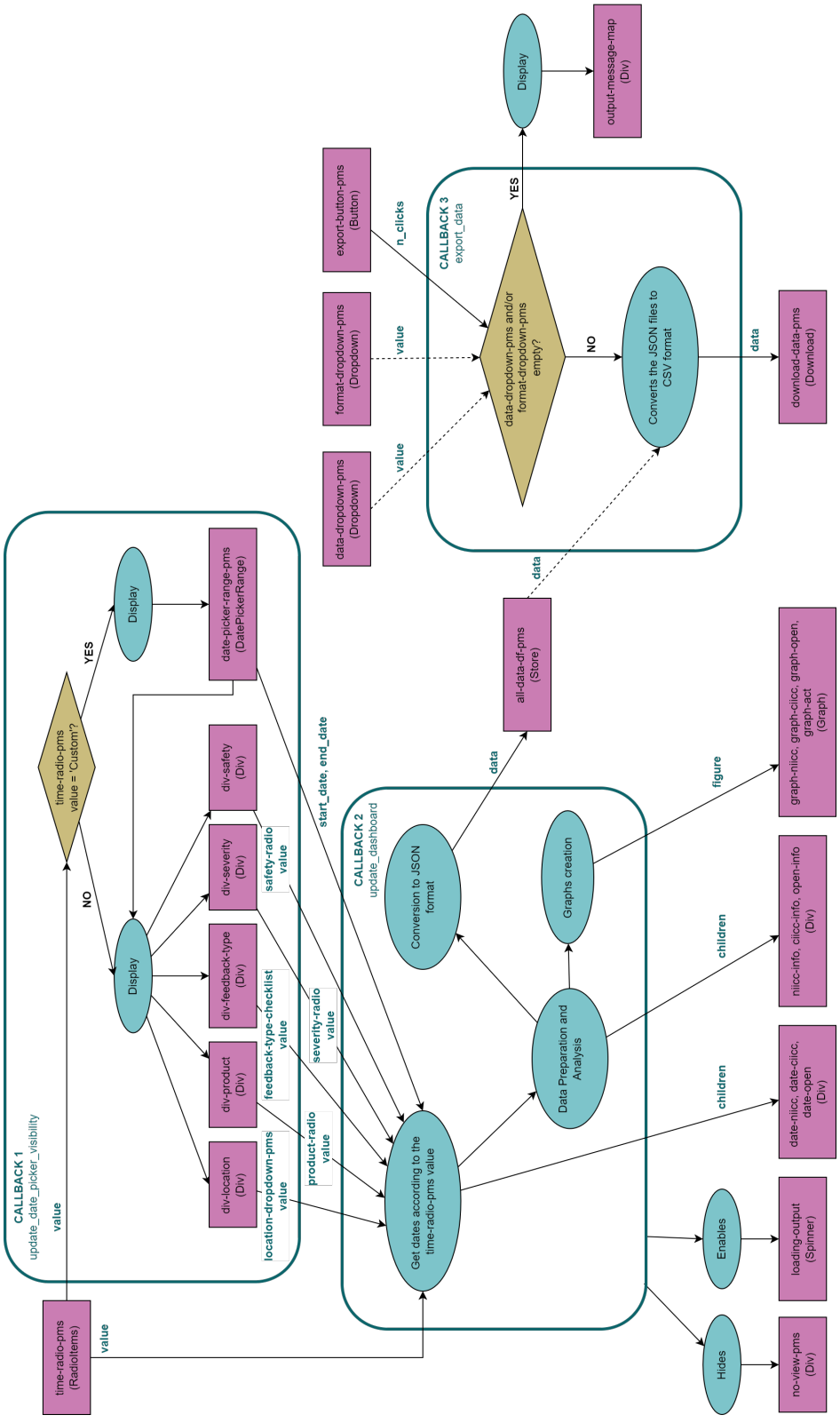


Figure 6.7: Callback diagram of the Post-Market Surveillance page

Table 6.4: Used components and their IDs in the Post-Market Surveillance page

ID	Type	Description	Callback 1	Callback 2	Callback 3
time-radio-pms	RadioItems	Allows the user to choose from Monthly, Quarterly, Yearly or Custom view	Input (value), Output (value)	Input (value)	-
div-location	Div	Grouping of the label "Source of Feedback:" and the location dropdown	Output (style)	-	-
div-product	Div	Grouping of the label "Product:" and the product radio items	Output (style)	-	-
div-feedback-type	Div	Grouping of the label "Feedback Type:" and the feedback type checklist	Output (style)	-	-
div-severity	Div	Grouping of the label "Severity:" and the severity radio items	Output (style)	-	-
div-safety	Div	Grouping of the label "Safety:" and the safety checklist	Output (style)	-	-
date-picker-range-pms	DatePickerRange	Allows the user to select the date range	Output (style)	Input (start_date, end_date)	-
location-dropdown-pms	Dropdown	Allows the user to choose the location	-	Input (value)	-
product-radio	RadioItems	Allows the user to choose the product (Motio App, Motio StepWatch, and Motio Backend)	-	Input (value)	-
feedback-type-checklist	Checklist	Allows the user to choose the feedback type (Issue, Complaint, Improvement, and CAPA)	-	Input (value)	-
severity-radio	RadioItems	Allows the user to choose the severity (Low, Medium, and High)	-	Input (value)	-
safety-checklist	Checklist	Allows the user to choose if it is safety related or not	-	Input (value)	-
loading-output	Spinner	Displays a spinner when content is loading	-	Output (children)	-
no-view-pms	Div	Initial text with instructions for the user to select a view option	-	Output (style)	-
date-niicc, date-ciicc, date-open	Div	Text with the selected date	-	Output (children)	-
niicc-info, ciicc-info, open-info	Div	Text with the total number of new/open/closed feedback tickets	-	Output (children)	-
graph-niicc, graph-ciicc, graph-open, graph-act	Graph	Graph with the new/open/closed feedback tickets data and average closing times	-	Output (figure)	-
all-data-df-pms	Store	Stores all data between callbacks to be exported, if necessary	-	Output (data)	State (data)
export-button-pms	Button	Allows the user to begin the export of their required data	-	-	Input (n_clicks)
data-dropdown-pms, format-dropdown-pms	Dropdown	Allows the user to choose the data and the format they want to export	-	-	State (value)
download-data-pms	Download	Sends the data to export	-	-	Output (data)
output-message-pms	Div	Informs the user to fill in the required fields (data or format) when exporting	-	-	Output (children)

6.4.2.3 Outcome Test Reports Page Implementation

In contrast to the previous two pages, the Outcome Test Reports page has only one callback, which is schematized in Figure 6.8. Furthermore, the used components are displayed in Table 6.5.

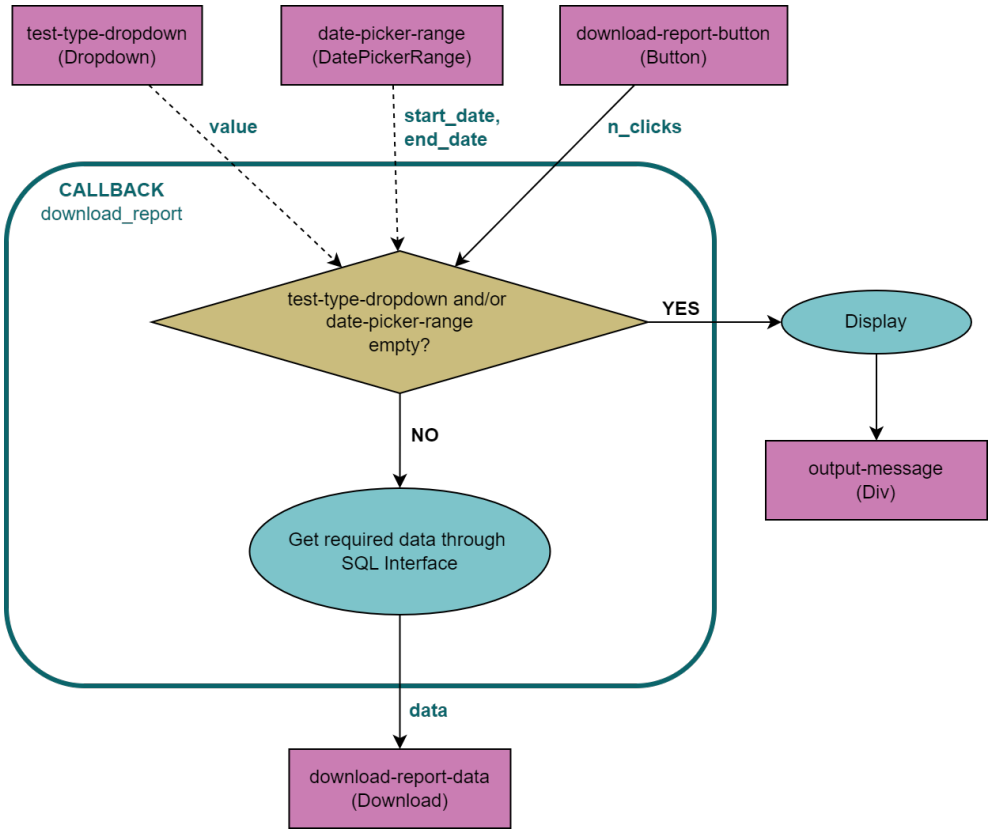


Figure 6.8: Callback diagram of the Outcome Test Reports page

Table 6.5: Used components and their IDs in the Outcome Test Reports page

ID	Type	Description	Callback
download-report-button	Button	Allows the user to begin the export of their required data	Input (n_clicks)
test-type-dropdown	Dropdown	Allows the user to choose the test type (PLUS-M, AMPPRO, and TUG)	State (value)
date-picker-range	DatePickerRange	Allows the user to select the date range	State (start_date, end_date)
download-report-data	Download	Sends the data to export	Output (data)
output-message	Div	Informs the user to fill in the required fields (test type or dates) when exporting	Output (children)

This page's callback greatly resembles the Callback 3 from the Motio App Performance and Post-Market Surveillance pages. Like them, it is triggered by the `download-report-button` Button, yet it ensures that the `test-type-dropdown`

`Dropdown` and the `date-picker-range` DatePickerRange are not empty to avoid any bug. In case of any missing required fields, it exhibits the error message (`output-message` Div). However, differently from the previously mentioned pages, the required data is obtained directly through the *SQL Interface*, as mentioned in Section 6.3, rather than from another component. At last, the data is sent to the `download-report-data` Download for export.

6.4.2.4 Other Functional Requirements Implementation

Furthermore, according to the *Requirements ORO4 and ORO5* from Table 4.4, the colors used in the graph labels must be consistent throughout the application. Therefore, the ColorsMap class was created to assign a color to each location, outcome test, and type of feedback ticket displayed. The colors were applied to the graphs while building them through the parameter `color_discrete_map`. However, due to the large number of locations that could even increase hereafter, consistency could only be achieved for one page at once since creating a suitable color palette for more than 20 locations is considered challenging. To solve this issue, it was decided along with stakeholders that only the displayed locations needed consistency; hence, an 11-color palette was developed as this number covers the maximum amount of displayed locations.

Additionally, a color analysis was planned to verify how accessible the employed colors are to people with visual impairments; however, this was judged unnecessary because Plotly charts display the label when the user hovers the cursor over them, as displayed in Figure 6.9.

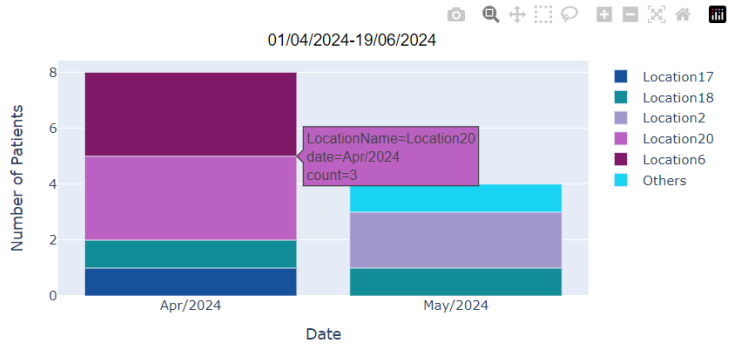


Figure 6.9: Label display when the user hovers the cursor over the chart

6.5 Deployment and Environment

Following the implementation, the platform was containerized using Docker to guarantee consistency, isolation, and portability, which are essential for deployment. The process began by building a Docker image from a created *Dockerfile*. Once the image was created, it was pushed to Adapttech's container registry. At this point, any cloud or local server could pull the docker image and start an instance of the web app.

Following that, the application could be hosted both in a local environment (server at Adapttech facilities) and as a Web App by performing the necessary configuration to connect to each

location. For both cases, the environment variables were changed to refer to the database on the server instead of the local database used in development, as mentioned in Section 6.2.1. In the case of the on-premises deployment, the variables were specified in a `.env` file (Listing 6.1) and a `docker-compose` file.

```
SERVER=localhost,1433
DATABASE=MotioDB
SQL_USERNAME=your read-only user
SQL_PASSWORD=your password
SQL_ENCRYPTED=yes
SQL_TRUSTED_SERVER_CERTIFICATE=yes
```

Listing 6.1: Environment variables in the .env file

After hosting the Motio System Monitoring Platform at Adapttech's server, its access was only possible to those locally present at Adapttech or using the organization's VPN, as *Requirement ERO2* indicated in Table 4.4. As for the Web App deployment, Azure Web App service from Microsoft was chosen as Adapttech already had an active subscription to the platform. It can restrict access through authentication using a Microsoft account, which also satisfies a requirement, *Requirement ORO3* indicated in Table 4.4. Additionally, within the Azure offer, it is possible to deploy a web app for free, with up to 60 minutes of usage per day, which is appropriate given the use case.

Figure 6.10 schematizes the whole process of deployment.

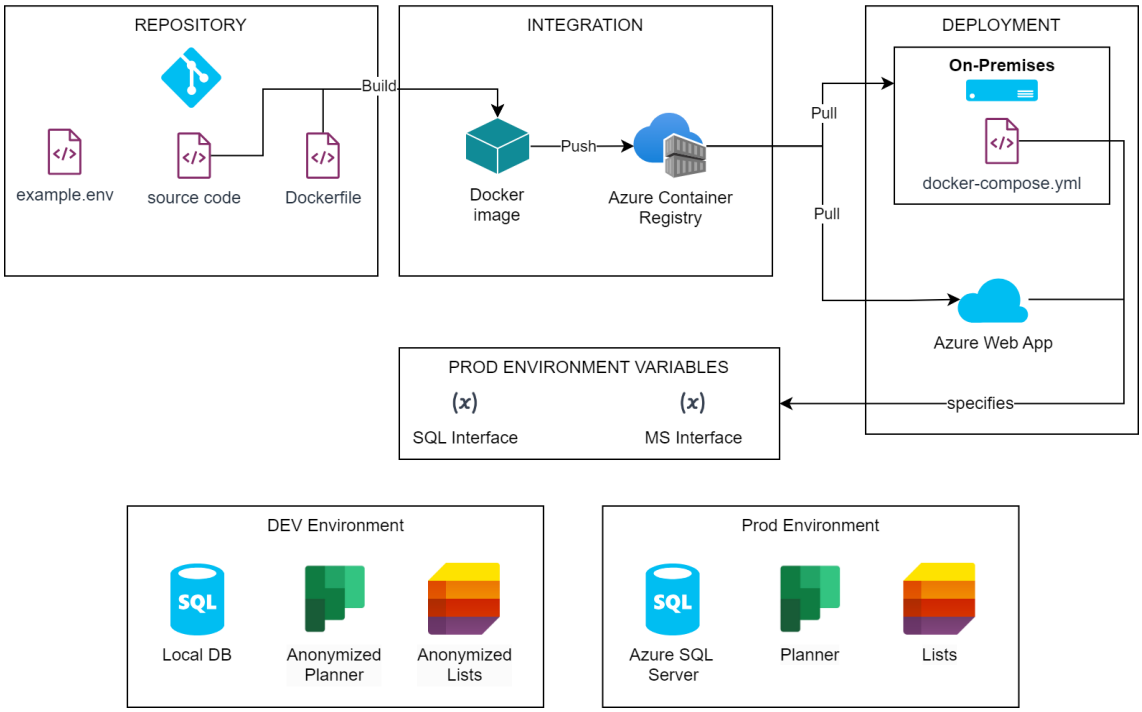


Figure 6.10: Scheme of Motio System Monitoring Platform deployment

6.6 Results

In this section, screenshots of the final implemented version are presented. First, the Home page is illustrated in Figure 6.11. Then, Figures 6.12, 6.13, 6.14, 6.15, 6.16, and 6.17 depict the final look and functionality, such as the different views (Monthly, Quarterly, Yearly, and Custom) and filters, of the Motio App Performance page. The same is represented by Figures 6.18, 6.19, 6.20, 6.21, and 6.22, but for the Post-Market Surveillance page. Finally, the Outcome Test Reports page is exhibited in Figures 6.23 and 6.24. The overall functionality of the implemented version of the Motio System Monitoring Platform was verified by a stakeholder. A video showcasing the application functionality is also available for viewing through the link[11].

In summary, Tables 6.6, 6.7, 6.8, and 6.9 show Traceability Matrices that associates the accomplished requirements with their corresponding figure.

Additionally, if a user, for example, forgets to fill in the required data in the export menus, the application still functions as usual since verifications and error messages were implemented, as mentioned in Section 6.4.2, to avoid system failures and increase user satisfaction. The figures indicated below illustrate how the messages are displayed:

- Figure 6.27a: displayed message when the user forgets to select a View before exporting;
- Figure 6.27b: displayed message when the user forgets a required field in the Export menu in the Motio App Performance page or Post-Market Surveillance page, in this case, it was the format;
- Figure 6.27c: displayed message when the user forgets a required field in the Outcome Test Reports page, in this case, it was the date;

After the implementation, a drawback was encountered after selecting a View option on the Post-Market Surveillance page: the interaction with Microsoft Graph API was significantly slow, taking around 25 seconds to load the details of 67 list items, 76 planner items, and 76 task items.

6.7 Summary

This chapter addressed the entire implementation process of the Motio System Monitoring Platform. The first phase, Data Collection, demonstrated the functionality of the SQL Interface and MS Interface, which are responsible for connecting to the database and retrieving data. Then, the data was processed during Data Preparation and Analysis to ensure it had the required format and information. Finally, Data Visualization was carried out employing Dash, culminating in the final fully functional version and its deployment. It is possible to conclude that the Motio System Monitoring Platform was developed successfully since it functions as planned and the considered requirements were satisfied as indicated by the Traceability Matrices in Tables 6.6, 6.7, 6.8, and 6.9.

[11] Implemented Version - Video link: https://drive.google.com/file/d/1DTdX3-tI8TYD3HoM4t2caJ15iCf1Oor0/view?usp=sharing

Table 6.6: Traceability Matrix after Implementation - Motio App Performance requirements

ID	Description	Evidence	Observations
ERM1	Be able to view how many new patients were added into the system in a Monthly, Quarterly, or Yearly view, with the option to group by location over time.	Figures 6.13, 6.14, 6.15, 6.17	"Group by" turned into a filter
ERM2	Be able to view how many new appointments were created in a Monthly, Quarterly, or Yearly view, with the option to group by location over time. Present in tabular and graphic form.	Figures 6.13, 6.14, 6.15, 6.17	"Group by" turned into a filter; Tabular was incorporated into Appointments X Outcome Tests (Section 5.1.1)
ERM3	Be able to view the amount (absolute and relative to the total number of appointments) of outcome tests performed in the Motio App in a Monthly, Quarterly, or Yearly view, with the option to group by location.	Figures 6.13, 6.14, 6.15, 6.17	Relative number removed (Section 5.2.1.1); "group by" turned into a filter
ERM4	Be able to export any data being viewed (through graphics or table) as a .csv or other similar format.	Figures 6.13, 6.14, 6.15, 6.25	-
ERM5	Be able to export viewed graphics as a .png or .pdf.	Figure 6.26	Plotly built-in feature
ERM6	Be able to monitor the subscription license state per clinic, for example, the date of last purchase and available stock of reports.	-	Ongoing development of license management (Section 6.3)
ORM1	Be able to monitor the report purchase/license history of a specific clinic.	-	Not considered initially

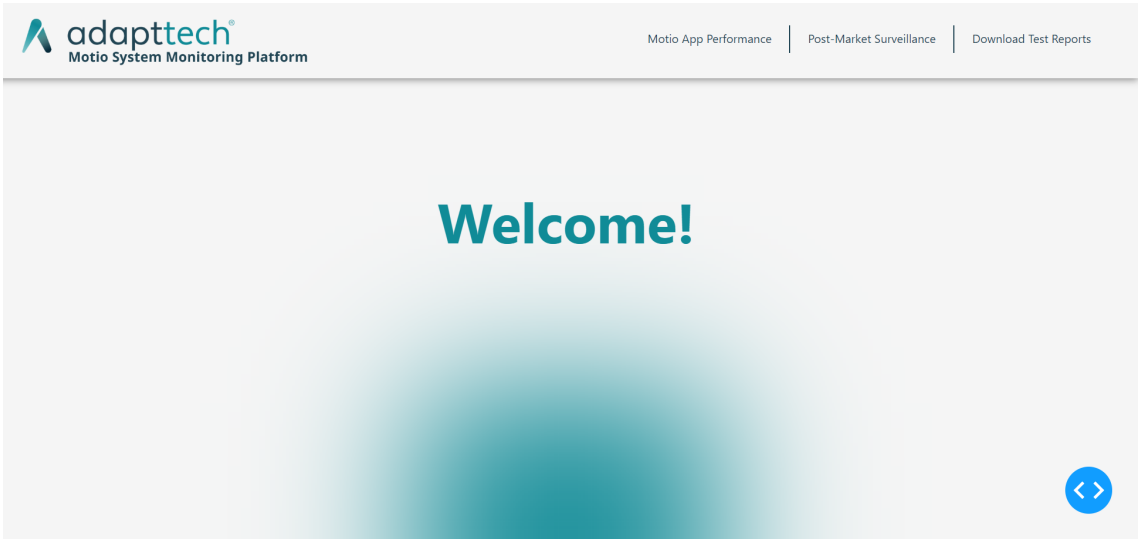


Figure 6.11: Implemented version - Home page

Table 6.7: Traceability Matrix after Implementation - Post-Market Surveillance requirements

ID	Description	Evidence	Observations
ERP1	Be able to view how many new issues, improvements, complaints, or CAPAs have been reported in a monthly, quarterly, or yearly view, with the option to group by location and by product/system.	Figures 6.19, 6.20, 6.21, 6.22	"Group by" turned into a filter
ERP2	Be able to view how many issues, improvements, complaints, or CAPAs related to safety were reported in a monthly, quarterly, or yearly view, with the option to group by location and by product/system.	Figures 6.22	Turned into a filter
ERP3	Calculate the closing times of issues, improvements, complaints, or CAPAs and the average for each month, quarter, or year, with the option to group by product/system.	Figures 6.19, 6.20, 6.21, 6.22	"Group by" turned into a filter
ERP4	Show the number of open complaints and CAPAs per month, quarter, and year, with the option to group by product/system and severity.	Figures 6.19, 6.20, 6.21, 6.22	Included all feedback types and comparison with new tickets (Section 5.1.1); "Group by" turned into a filter
ERP5	Calculate the percentage of complaints closed in less than 1 week and less than 1 month. View monthly, quarterly, and yearly and group by severity, safety-related, and product/system.	-	Lack of data (Section 6.3)
ORP1	Count the number of times that the user canceled the outcome measure process	-	Not considered initially
ORP2	Track time of execution for each outcome measure	-	Not considered initially
ORP3	Show which complaints and CAPAs are still open per month, quarter, and year, with the option to group by product/system and severity.	-	Not considered initially
ORP4	Detect how many appointments quarterly and yearly were not closed after 30 days.	-	Not considered initially
ORP5	Detect how many patients do not have an appointment associated, monthly, quarterly, and yearly.	-	Not considered initially
ORP6	Detect how many times the button for the Instructions for Use was clicked on monthly, quarterly, and yearly.	-	Not considered initially
ORP7	Be able to detect how many unsuccessful login attempts were made monthly, quarterly, and yearly.	-	Not considered initially

Table 6.8: Traceability Matrix after Implementation - Clinical Insights requirements

ID	Description	Evidence	Observations
ERC1	Be able to download in tabular form the PLUS-M results performed during a given time frame, together with the test date and some information about the patient, such as gender, date of birth, amputation type and etiology, AMPPRO score, TUG duration, K level, 10-meter walk test time, and 2-minute or 6-minute walk test distance.	Figures 6.23, 6.24	-
ERC2	Be able to download in tabular form the AMPPRO results performed during a given time frame, together with the test date and some information about the patient, such as date of birth and amputation, amputation type and etiology, and K level.	Figures 6.23, 6.24	-
ORC1	Be able to download in tabular form the Timed Up and Go result, date of test, and risk of fall performed during a given time frame	Figures 6.23, 6.24	-
ORC2	Be able to download in tabular form the Long-term acquisition results, and date of test performed during a given time frame	-	Not considered initially
ORC3	Be able to download the Motio Functional level in a tabular form.	-	Not considered initially
ORC4	Be able to download the Clinical Assessment in a tabular form	-	Not considered initially

Table 6.9: Traceability Matrix after Implementation - Other requirements

ID	Description	Evidence	Observations
ERO1	Be able to filter collected data by a specific time frame and time resolutions (monthly, quarterly, and yearly).	Figures 6.16, 6.22	-
ERO2	The application must only be accessible from within the company network.	Section 6.5	-
ORO1	Be able to set a default time frame in which the user can view data (Last Quarter, Last 6 Month, Last Year) in this Dashboard app	-	Not considered initially
ORO2	Settings configured by the user of the Dashboard app should be saved locally	-	Not considered initially
ORO3	The Application should be accessible online through authentication using a Microsoft account.	Section 6.5	-
ORO4	When using color to represent a location, severity, or product, it must be consistent within the App at that moment.	Figures 6.13, 6.19	-
ORO5	When using color to represent a location, severity, or product, it must be consistent in the App overtime.	Figures 6.13, 6.19	Except for Location colors (Section 6.4.2.4)



Figure 6.12: Implemented version - Motio App Performance page - Initial page



Figure 6.13: Implemented version - Motio App Performance page - Monthly View (Viewed in June 2024)

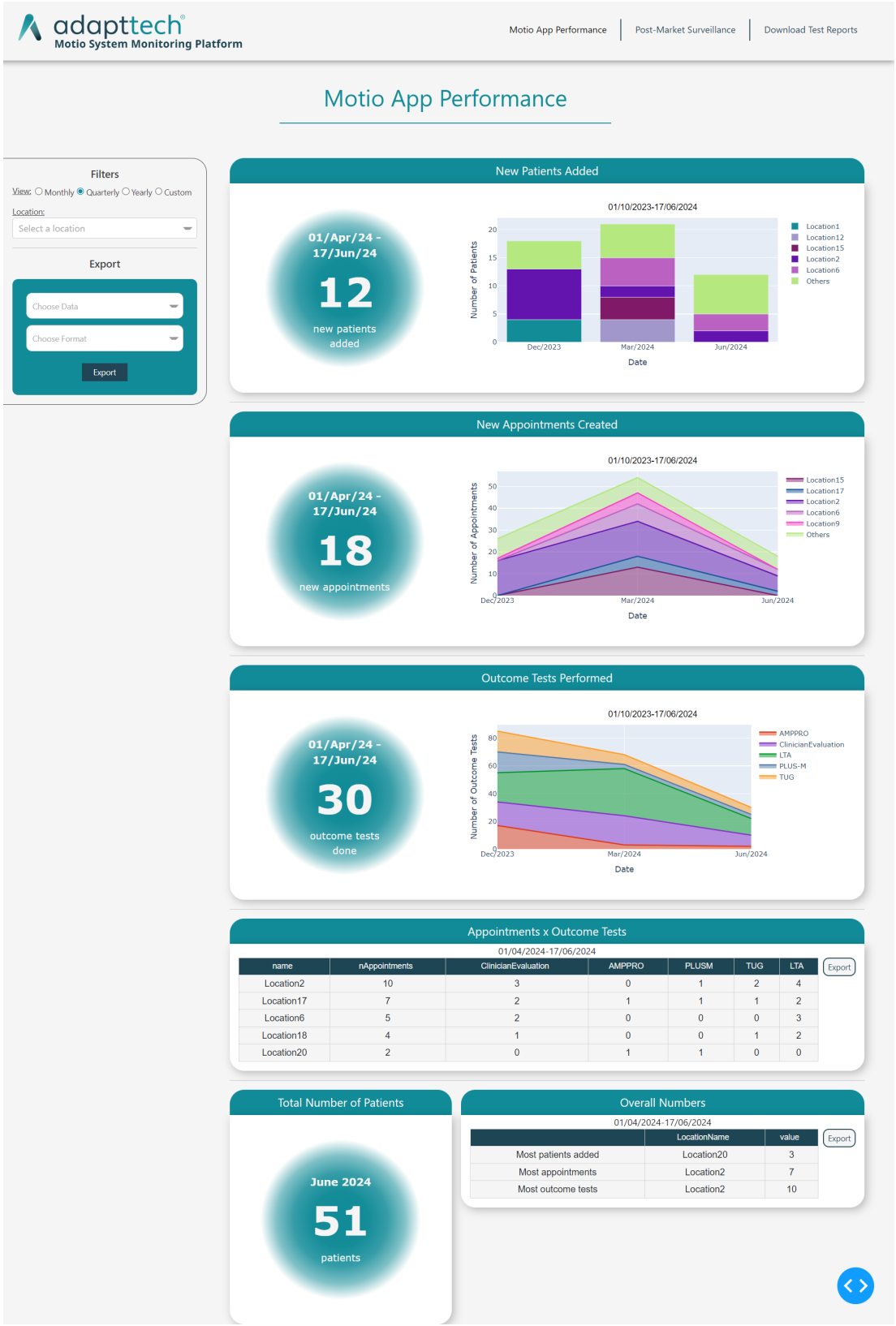


Figure 6.14: Implemented version - Motio App Performance page - Quarterly View (Viewed in June 2024)

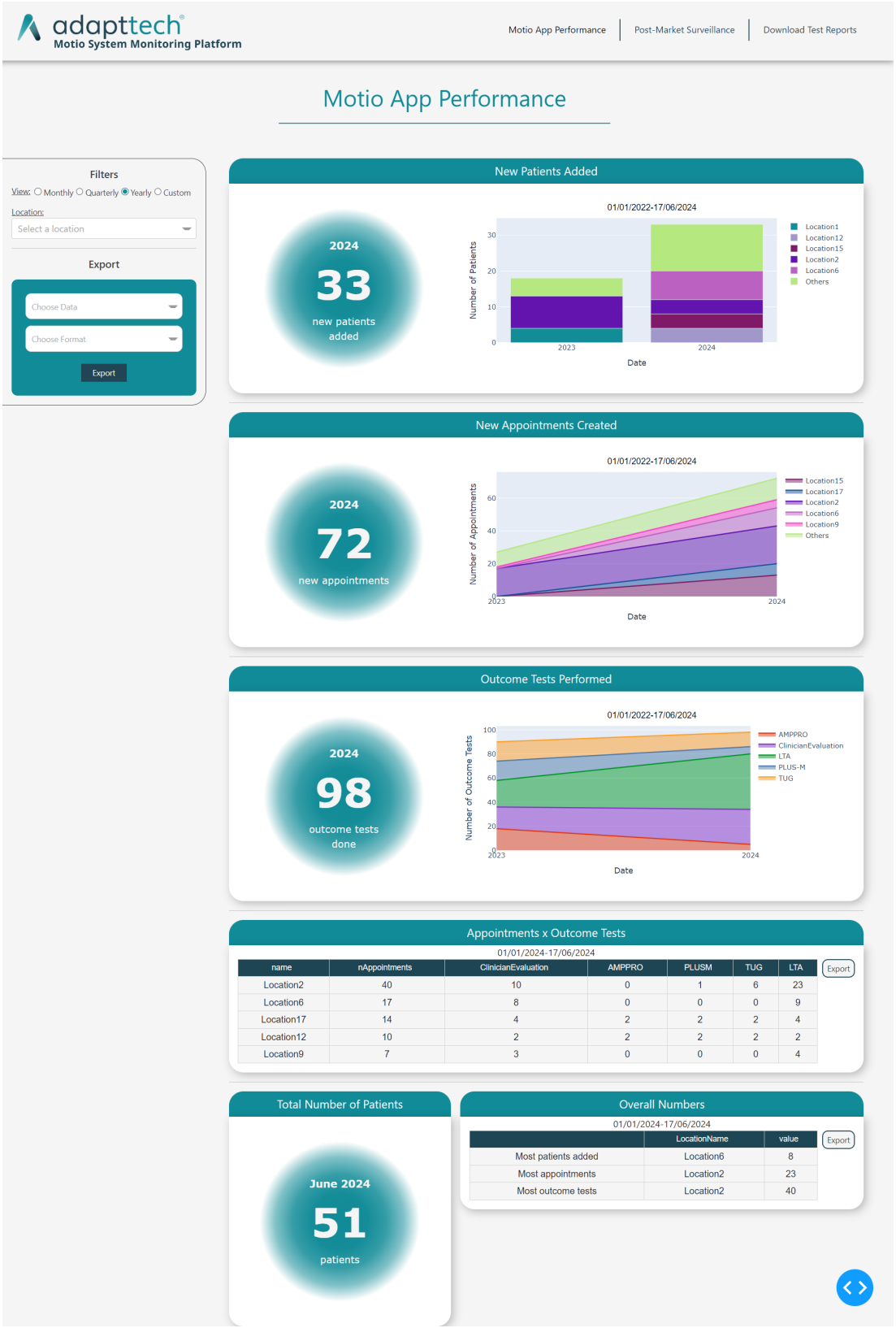


Figure 6.15: Implemented version - Motio App Performance page - Yearly View (Viewed in June 2024)

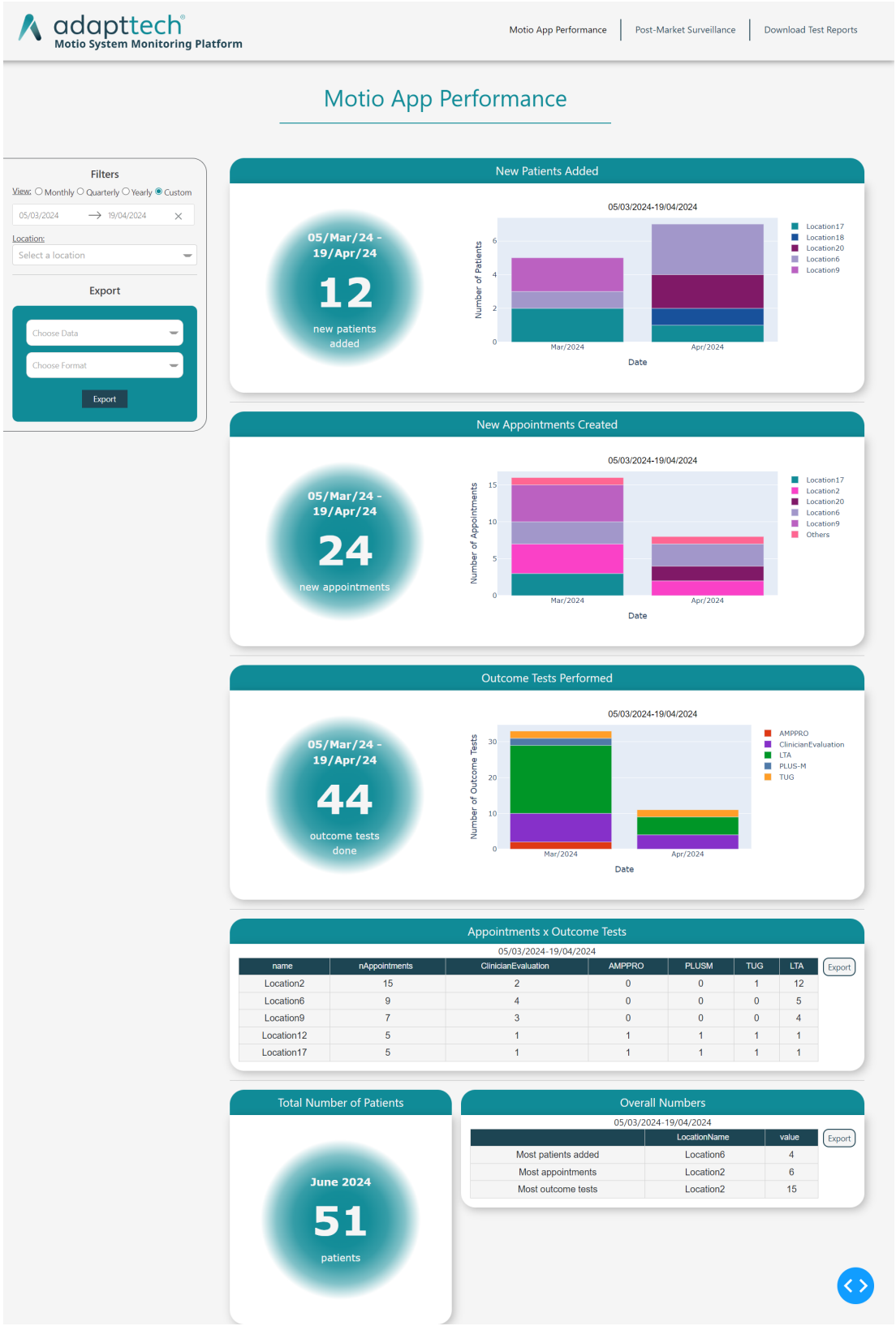


Figure 6.16: Implemented version - Motio App Performance page - Custom View (05/Mar/2024-19/Apr/2024 as selected dates)

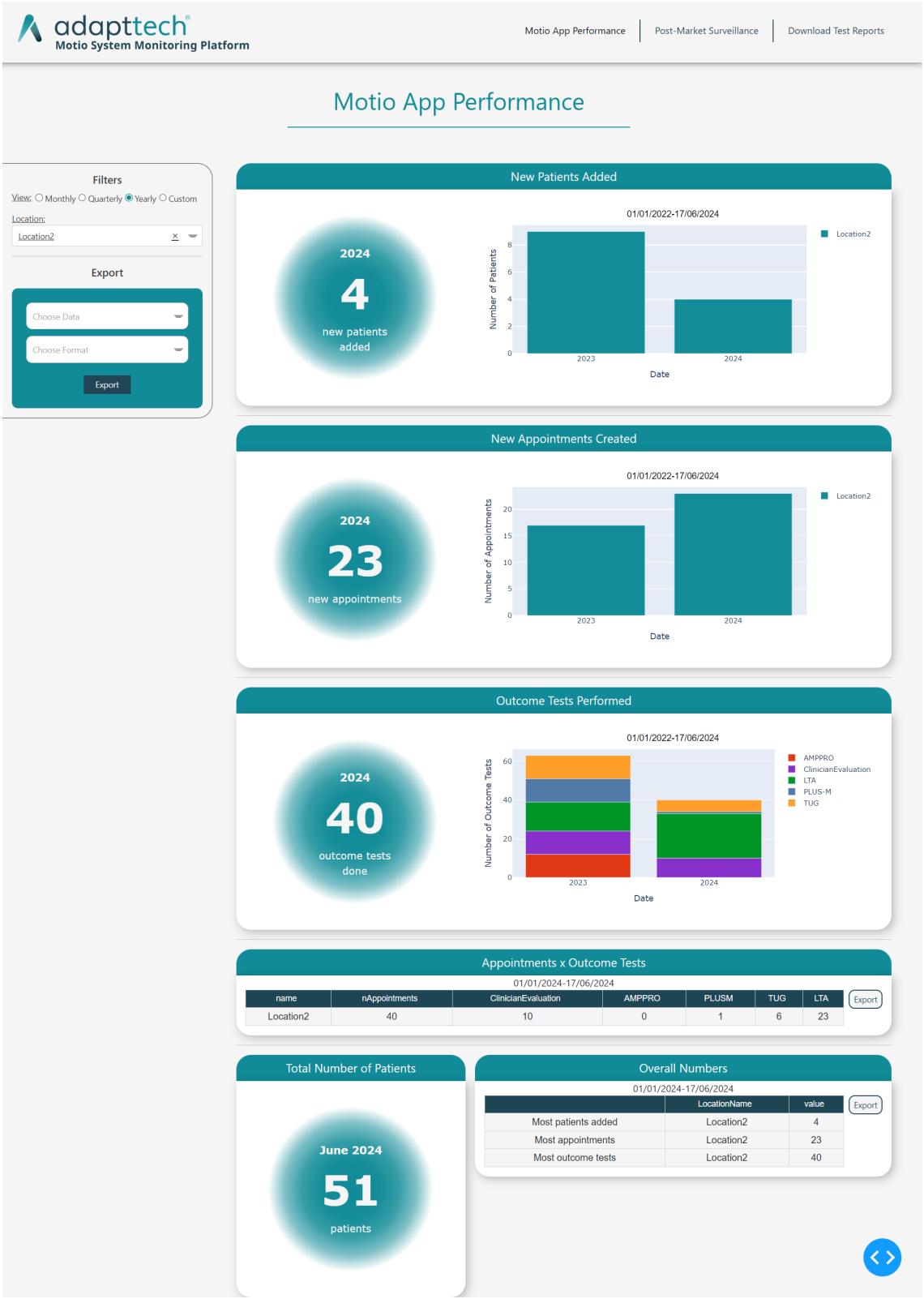


Figure 6.17: Implemented version - Motio App Performance page - Location filter (Location 2 selected)

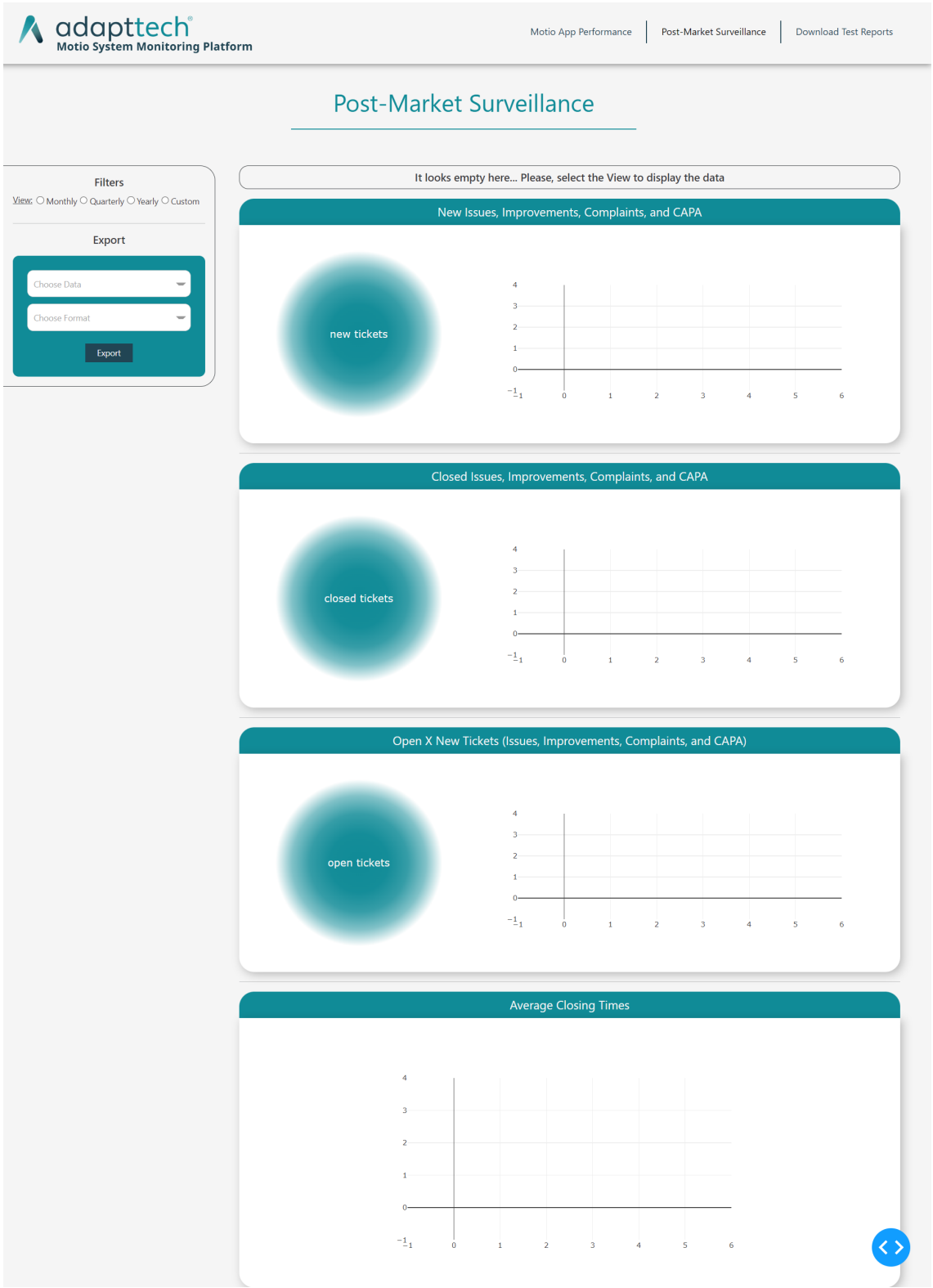


Figure 6.18: Implemented version - Post-Market Surveillance page - Initial page

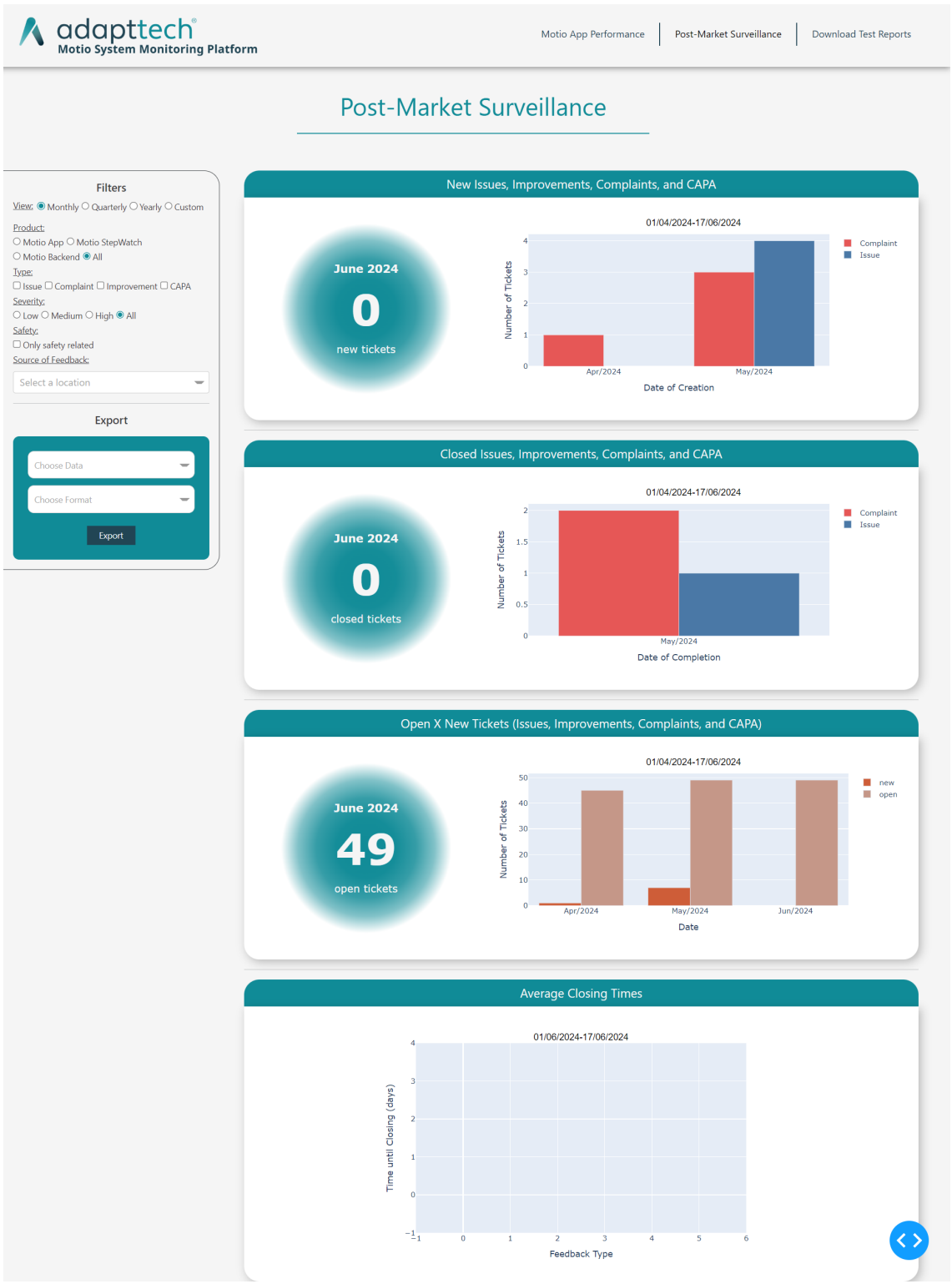


Figure 6.19: Implemented version - Post-Market Surveillance page - Monthly view (Viewed in June 2024)

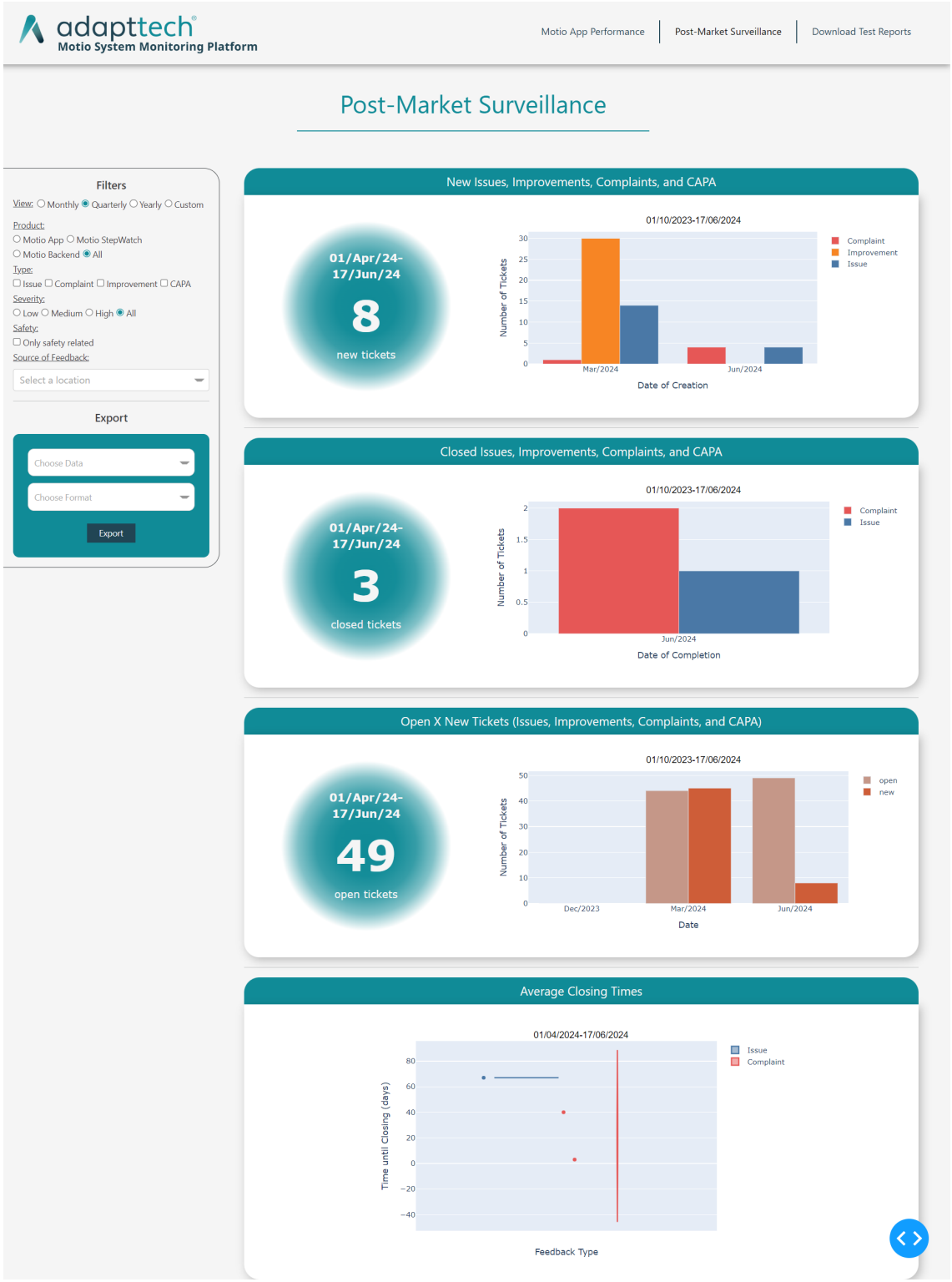


Figure 6.20: Implemented version - Post-Market Surveillance page - Quarterly view (Viewed in June 2024)

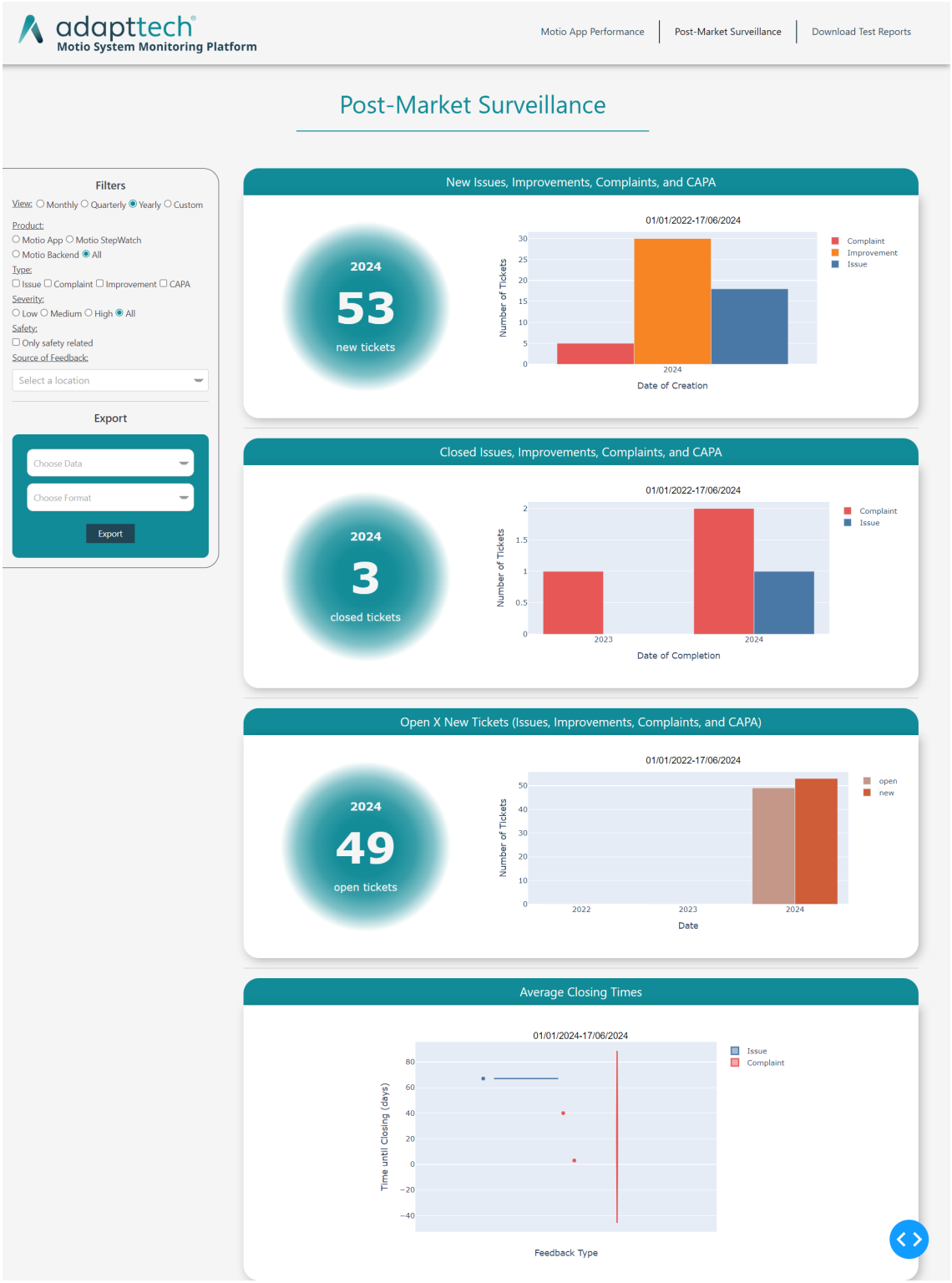


Figure 6.21: Implemented version - Post-Market Surveillance page - Yearly view (Viewed in June 2024)

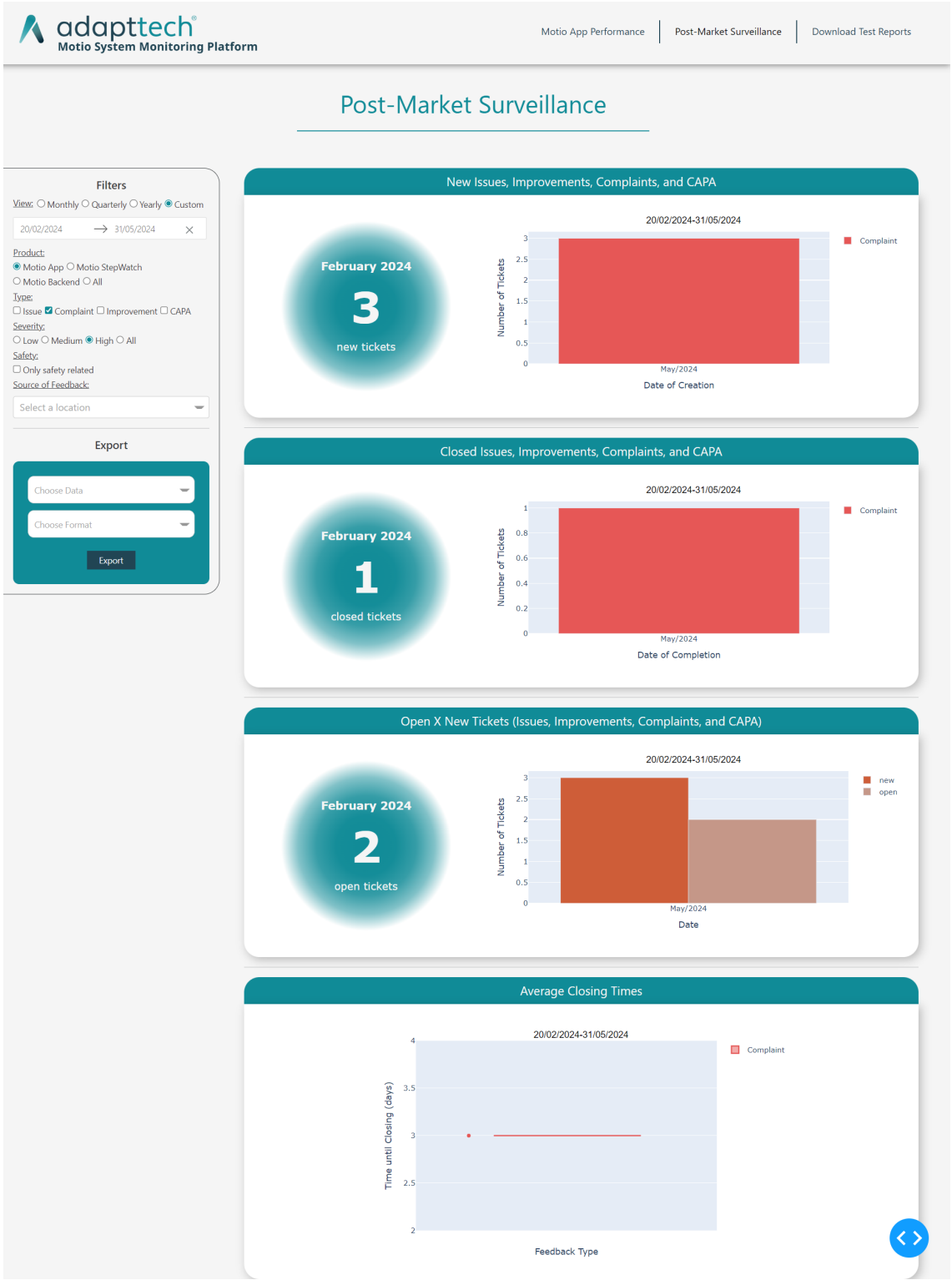


Figure 6.22: Implemented version - Post-Market Surveillance page - All filters selected

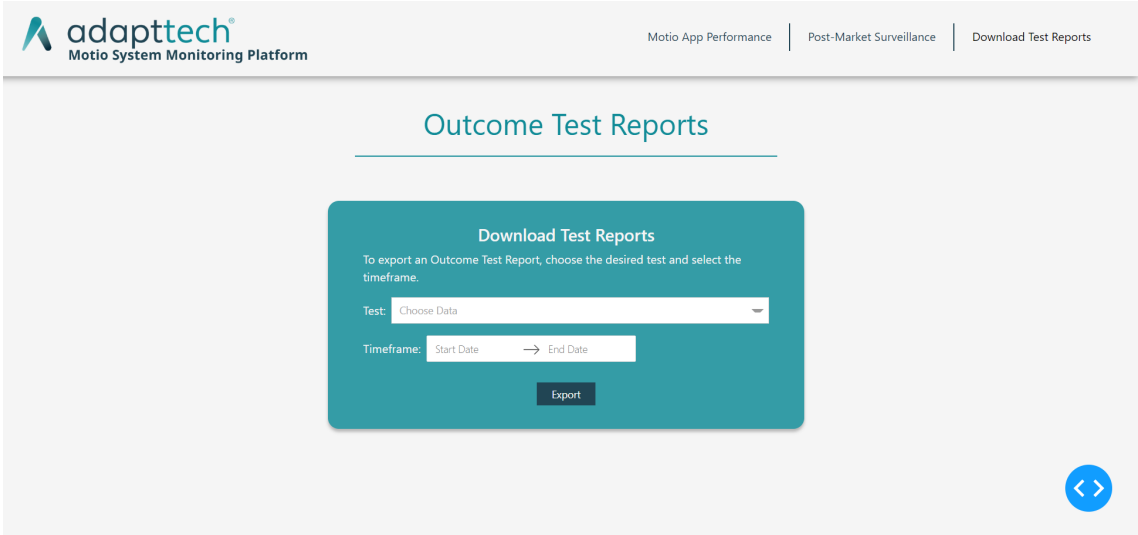


Figure 6.23: Implemented version - Outcome Test Reports



Figure 6.24: Implemented version - Test Reports options

01/04/2024-20/06/2024

name	nAppointments	ClinicianEvaluation	AMPPRO	PLUSM	TUG	LTA
Location2	10	3	0	1	2	4
Location17	7	2	1	1	1	2
Location6	5	2	0	0	0	3
Location18	4	1	0	0	1	2
Location20	2	0	1	1	0	0

Export

Figure 6.25: Implemented version - Table export button

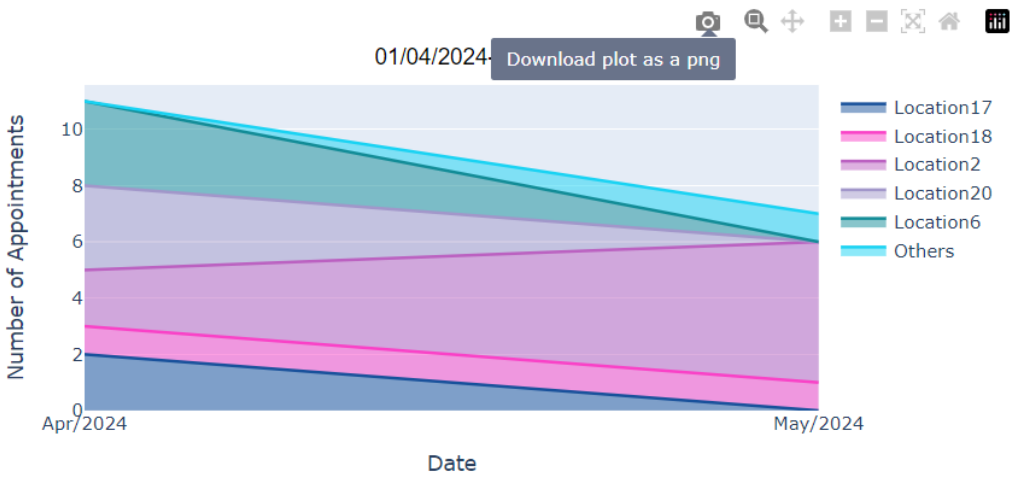
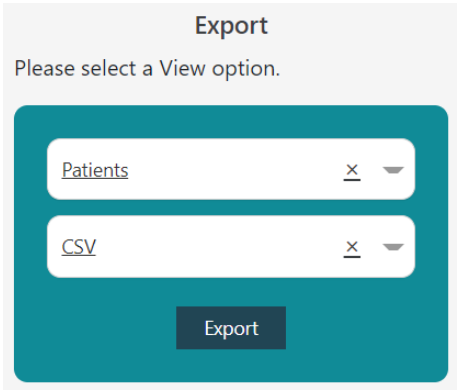
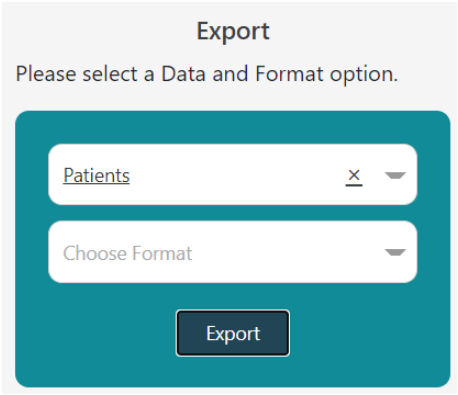


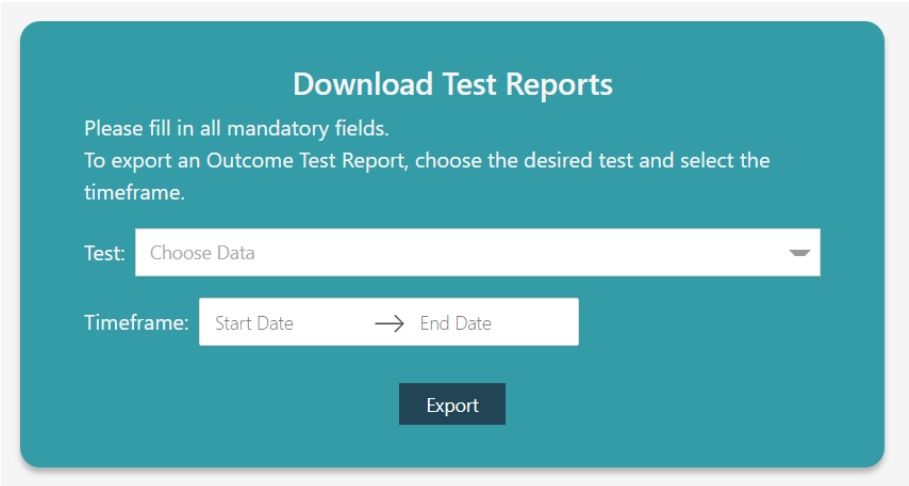
Figure 6.26: Implemented version - Export graphics as png option



(a) MAP and PMS pages - Missing View error message



(b) MAP and PMS pages - Missing field error message



(c) Outcome Test Reports page - Missing field error message

Figure 6.27: Group 4 graph data

Chapter 7

Conclusion and Future Work

People often refer to nowadays as the Data Era since there is a constant generation, collection, and analysis of data. This data-driven behavior of society has profoundly impacted various areas, allowing for more informed decision-making and fostering innovation. Therefore, it is essential that companies invest their time and money in data analytics to optimize operations, recognize trends, and prevent problems from occurring.

With that in mind, the current project focused on developing a monitoring application for the Motio StepWatch™ System, Adapttech's product capable of collecting real-world data about prosthetic users to provide them with a more accurate functional level evaluation. The developed application, called Motio System Monitoring Platform, is composed of three pages:

- **Motio App Performance:** users can view usage details of the Motio App (Motio System's application), i.e., how many patients were added, how many appointments were created, and how many outcome tests were performed.
- **Post-Market Surveillance:** users can check the performance of the Motio StepWatch™ System itself, i.e., how many issues were encountered, how many complaints were received, or how many improvements were recognized, divided into whether they are new, open, or closed. In addition, it displays information regarding their closing times.
- **Outcome Test Reports:** users can download outcome test reports (PLUS-M™, AMPPRO®, and Timed Up and Go) with all the required information.

Furthermore, in the Motio App Performance and the Post-Market Surveillance pages, users can filter the information to their liking and export any data in graphic or tabular format.

As a result of the insights provided by the developed Motio System Monitoring Platform, Adapttech may better understand the operational aspects of its main product, leading to improved decision-making, resource management, issue handling, and efficiency.

The process of developing the Motio System Monitoring Platform began by gathering requirements and assessing them to determine which ones would be addressed, followed by the creation of a Conceptual Design composed of a Conceptual Model, representing the requirements, and

a System Model that details the operational elements of the application prior to implementing them. Next, the prototype was developed in four iterations—a wireframe and three Figma prototypes—based on the requirements, stakeholder feedback, and usability testing. After the validated prototype, the application was implemented in three stages: Data Collection, Data Preparation and Analysis, and Data Visualization. Data Collection comprised the connection to the Motio SQL database and Microsoft Planner and Sharepoint List through Microsoft Graph API to retrieve the Motio App and Feedback data, respectively. Then, Data Preparation and Analysis covered all necessary data processing to generate the information required to create the charts. Finally, Data Visualization was done using Dash and encompassed the user interface with the interactive components and charts, as well as the functionality behind them. The application deployment was done in a local environment, a server at Adapttech's premises, and as a Web App, both with restricted access.

In that manner, the final product achieved its goals, as it satisfied the considered requirements and a stakeholder validated its functionality.

7.1 Acquired Knowledge

Since this thesis combined several fields, there were many takeaways from it, as listed below.

1. Understanding of the prosthetics industry: history of prosthesis, reimbursement system, prosthetic users' functional levels, and standard outcome measures and clinical tests;
2. Knowledge about web app development: required steps, HTTP requests, database integration, and deployment;
3. Knowledge about user interface design: design principles, and prototyping evaluation;
4. Business Intelligence concepts: definition of Business Intelligence, BI applications, and benefits of employing BI;
5. Knowledge about Microsoft Products: Microsoft Sharepoint List, Microsoft Planner, Microsoft Entra, Microsoft Graph API, Azure Data Studio, and Azure App Service.

7.2 Challenges Encountered

As with any project, the Motio System Monitoring Platform had a few hardships during its development, such as:

- **Microsoft Graph API access authorization:** as mentioned in Section 4.2.2.3, a client needs to be granted access to the organization's Microsoft Graph API in order to connect to the Microsoft Planner and Microsoft Sharepoint List and collect the feedback data. However, since this step required Adapttech's directors to approve it, which was an unknown procedure for them, it took a while to complete, disrupting the application development workflow;

- **Stakeholders' availability:** during the prototyping, some stakeholders could not engage or partially engaged in the feedback gathering and prototyping evaluation due to time constraints;
- **Lack of closed feedback tickets:** there were only three completed feedback tickets available, making it harder to verify charts concerning closed tickets.

7.3 Future Work

Considering the project's continuation, a few points can be addressed to improve even more the Motio System Monitoring Platform, namely:

- **Evaluate the implemented version with stakeholders**: due to time constraints, the implemented version could not be validated by the stakeholders;
- **Inclusion of remaining requirements**: only essential and a few desirable requirements were considered during the development and, even then, not all of them were implemented;
- **Colors when exporting graphics**: as stated in Section 6.4.2.4, the color analysis for colorblindness was not carried out since Plotly charts display the label when the cursor hovers over them; however, during the export, this is not possible, and thus needs to be addressed;
- **Investigation of Microsoft Graph API data retrieval delay**: as described in Section 6.6, the feedback data required some time to be retrieved; hence, this delay could be investigated to find a solution;
- **Verify and correct any existing time limits when connecting to databases**: the delay presented by offline databases could lead to an error and, consequently, a system failure.

References

[1] Kathryn Ziegler-Graham, Ellen J. MacKenzie, Patti L. Ephraim, Thomas G. Travison, and Ron Brookmeyer. Estimating the prevalence of limb loss in the United States: 2005 to 2050. *Archives of Physical Medicine and Rehabilitation*, 89(3):422–429, March 2008.

[2] Jacqueline Louise Finch, Glyn Harvey Heath, Ann Rosalie David, and Jai Kulkarni. Biomechanical Assessment of Two Artificial Big Toe Restorations From Ancient Egypt and Their Significance to the History of Prosthetics. *JPO Journal of Prosthetics and Orthotics*, 24(4):181–191, October 2012.

[3] Prosthetic through the ages: A look at how prosthetics have changed over time and how NIH supports innovation. *NIH MedlinePlus Magazine*, 18(2).

[4] Alan J. Thurston. Paré and Prosthetics: The Early History of Artificial Limbs. *ANZ Journal of Surgery*, 77(12):1114–1119, December 2007.

[5] Filipe Daniel Martins Rodrigues. Gait Characterization and Classification in Lower-Limb Amputees. Master's thesis, University of Porto, June 2018.

[6] Ethan M. Balk, Abhilash Gazula, Georgios Markozannes, Hanna J. Kimmel, Ian J. Saldanha, Linda J. Resnik, and Thomas A. Trikalinos. Lower Limb Prostheses: Measurement Instruments, Comparison of Component Effects by Subgroups, and Long-Term Outcomes. Technical report, Agency for Healthcare Research and Quality (AHRQ), September 2018.

[7] Michael S Orendurff, Silvia U Raschke, Lorne Winder, David Moe, David A Boone, and Toshiki Kobayashi. Functional level assessment of individuals with transtibial limb loss: Evaluation in the clinical setting versus objective community ambulatory activity. *Journal of Rehabilitation and Assistive Technologies Engineering*, 3:205566831663631, January 2016.

[8] Adapttech. Motio StepWatch System Clinical Guide.

[9] Hannah Ritchie, Edouard Mathieu, Max Roser, and Esteban Ortiz-Ospina. Internet. *Our World in Data*, 2023.

[10] Jason Mander and Katie Young. Digital vs. Traditional Media Consumption. Insight Report, Global Web Index, 2017.

[11] Martin Aruldoss, Miranda Lakshmi Travis, and V. Prasanna Venkatesan. A survey on recent research in business intelligence. *Journal of Enterprise Information Management*, 27(6):831–866, October 2014.

[12] Jiarui Yin and Vicenc Fernandez. A systematic review on business analytics. *Journal of Industrial Engineering and Management*, 13(2):283, May 2020.

[13] Öykü Işık, Mary C. Jones, and Anna Sidorova. Business intelligence success: The roles of BI capabilities and decision environments. *Information & Management*, 50(1):13–23, January 2013.

[14] Bringham and Women's Hospital. Standard of Care: Lower Extremity Amputation, 2011.

[15] Robert S. Gailey, Kathryn E. Roach, E.Brooks Applegate, Brandon Cho, Bridgid Cunniffe, Stephanie Licht, Melanie Maguire, and Mark S. Nash. The Amputee Mobility Predictor: An instrument to assess determinants of the lower-limb amputee's ability to ambulate. *Archives of Physical Medicine and Rehabilitation*, 83(5):613–627, May 2002.

[16] Prosthetic Limb Users Survey of Mobility (PLUS-M™) Version 1.2 Short Forms Users Guide, September 2022.

[17] Centers for Disease Control and Prevention (CDC). Assessment Timed Up & Go (TUG).

[18] David Manuel Santana Santos. Development of a Dashboard for Monitoring Cross-Team Issues. Master's thesis, University of Porto, January 2017.

[19] Adapttech. Post Market Surveillance Activities.

[20] Richard M. Devens. *Cyclopædia of Commercial and Business Anecdotes: Comprising Interesting Reminiscences and Facts, Remarkable Traits and Humors ... of Merchants, Traders, Bankers ... Etc. in All Ages and Countries ...* D. Appleton, 1868.

[21] D. M. J. Power. A brief history of decision support systems. In *The Web Conference*, 2003.

[22] Catarina Silva. Influência do Business Intelligence no Controlo de Gestão e no Desempenho Organizacional: um Estudo de Caso. Master's thesis, University of Porto, 2020.

[23] Filipa Prisco. A framework to support the creation of a Business Intelligence tool- application in an industrial context. Master's thesis, University of Porto, July 2021.

[24] Martin Kowalczyk. *The Support of Decision Processes with Business Intelligence and Analytics*. Springer Fachmedien Wiesbaden, Wiesbaden, 2017.

[25] Bert Brijs. *Business analysis for business intelligence*. CRC Press, 2013.

[26] Steve Lavalle, Eric Lesser, Rebecca Shockley, Michael S Hopkins, and Nina Kruschwitz. How the smartest organizations are embedding analytics to transform information into insight and then action. Findings and recommendations from the first annual New Intelligent Enterprise Global Executive study. *MIT Sloan Management Review*, December 2010.

[27] Gloria J. Miller, Dagmar Bräutigam, and Stefanie Virginia Gerlach. *Business intelligence competency centers: a team approach to maximizing competitive advantage*. Wiley, Hoboken, N.J, 2006.

[28] Marcelo Bronzo, Paulo Tarso Vilela De Resende, Marcos Paulo Valadares De Oliveira, Kevin P. McCormack, Paulo Renato De Sousa, and Reinaldo Lopes Ferreira. Improving performance aligning business analytics with process orientation. *International Journal of Information Management*, 33(2):300–307, April 2013.

[29] Gregory Richards, William Yeoh, Alain Yee Loong Chong, and Aleš Popovič. Business Intelligence Effectiveness and Corporate Performance Management: An Empirical Analysis. *Journal of Computer Information Systems*, 59(2):188–196, March 2019.

[30] Celina M. Olszak and Ewa Ziemba. Approach to Building and Implementing Business Intelligence Systems. *Interdisciplinary Journal of Information, Knowledge, and Management*, 2:135–148, 2007.

[31] Hugh J Watson and Barbara H Wixom. Enterprise agility and mature bi capabilities. *Business Intelligence Journal*, 12(3):4, 2007.

[32] Steve Williams and Nancy Williams. *The profit impact of business intelligence*. Elsevier, 2010.

[33] Andrei Hagiu and Julian Wright. When Data Creates Competitive Advantage. *Harvard Business Review*, (January-February 2020), 2020.

[34] Mokhamad Hendayun, Erwin Yulianto, Jack Febrian Rusdi, Awan Setiawan, and Benie Ilman. Extract transform load process in banking reporting system. *MethodsX*, 8:101260, 2021.

[35] Sonali Vyas and Pragya Vaishnav. A comparative study of various ETL process and their testing techniques in data warehouse. *Journal of Statistics and Management Systems*, 20(4):753–763, July 2017.

[36] Alejandro Vaisman and Esteban Zimányi. Database Concepts. In *Data Warehouse Systems*, pages 13–52. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[37] Carlo Vercellis. *Business intelligence: data mining and optimization for decision making*. Wiley, 2009.

[38] In Ong, Pei Siew, and Siew Wong. A Five-Layered Business Intelligence Architecture. *Communications of the IBIMA*, pages 1–11, March 2011.

[39] Thomas H. Davenport and Jeanne G. Harris. *Competing on analytics: the new science of winning*. Harvard Business School Press, Boston, Mass, 2007.

[40] Martin Kowalczyk and Peter Buxmann. Big Data and Information Processing in Organizational Decision Processes: A Multiple Case Study. *Business & Information Systems Engineering*, 6(5):267–278, October 2014.

[41] Ângela Silva. Desenvolvimento de um dashboard de indicadores de gestão científica numa Instituição de Ensino Superior. Master's thesis, University of Porto, July 2021.

[42] A Martin, D Maladhy, and V Prasanna Venkatesan. A framework for business intelligence application using ontological classification. *arXiv preprint arXiv:1109.1088*, 2011.

[43] Pedro Lemos. Reformulação e Implementação de um Business Intelligence. Master's thesis, University of Porto, July 2014.

[44] Yung-Cheng Liao and Mei-Su Chen. Research on Big Data Ad Hoc Query Technology Based on an Accident Insurance Campaign. *Engineering Proceedings*, 38(1):8, 2023. Number: 1 Publisher: Multidisciplinary Digital Publishing Institute.

[45] William Yeoh and Andy Koronios. Critical Success Factors for Business Intelligence Systems. *Journal of Computer Information Systems*, 50(3):23–32, March 2010. Publisher: Taylor & Francis _eprint: https://www.tandfonline.com/doi/pdf/10.1080/08874417.2010.11645404.

[46] Alejandro Díaz, Kayvaun Rowshankish, and Tamim Saleh. Why data culture matters. *McKinsey Quarterly*, 3(1):36–53, 2018.

[47] Kurt Schlegel, Julian Sun, David Pidsley, Anirudh Ganeshan, Fay Fei, Aura Popa, Radu Miclaus, Edgar Macari, Kevin Quinn, and Christopher Long. Magic Quadrant for Analytics and Business Intelligence Platforms. Technical report, Gartner, April 2023.

[48] Jack G. Zheng. Data Visualization in Business Intelligence. In *Global Business Intelligence*. Routledge, 2017.

[49] Sven Michalczyk, Mario Nadj, Darius Azarfar, Alexander Maedche, and Christoph Gröger. A state-of-the-art overview and future research avenues of self-service business intelligence and analytics. 2020.

[50] Richard Brath and Michael Peters. Dashboard design: Why design is important. *DM Direct*, 85:1011285–1, 2004.

[51] Stephen Few. *Information dashboard design: The effective visual communication of data*. O'Reilly Media, Inc., 2006.

[52] Ogan M. Yigitbasioglu and Oana Velcu. A review of dashboards in performance management: Implications for design and research. *International Journal of Accounting Information Systems*, 13(1):41–59, March 2012.

[53] Koen Pauwels, Tim Ambler, Bruce H. Clark, Pat LaPointe, David Reibstein, Bernd Skiera, Berend Wierenga, and Thorsten Wiesel. Dashboards as a Service: Why, What, How, and What Research Is Needed? *Journal of Service Research*, 12(2):175–189, November 2009.

[54] Matt D. Peters, Bernhard Wieder, Steve G. Sutton, and James Wakefield. Business intelligence systems use in performance measurement capabilities: Implications for enhanced competitive advantage. *International Journal of Accounting Information Systems*, 21:1–17, June 2016.

[55] Brian Ballou, Dan L. Heitger, and Laura Donnell. Creating effective dashboards: how companies can improve executive decision making and board oversight. *Strategic Finance*, 91(9):27+, March 2010.

[56] Dane Bertram, Amy Voida, Saul Greenberg, and Robert Walker. Communication, collaboration, and bugs: the social nature of issue tracking in small, collocated teams. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, CSCW '10, page 291–300, New York, NY, USA, 2010. Association for Computing Machinery.

[57] Thorsten Merten, Matus Falis, Paul Hubner, Thomas Quirchmayr, Simone Bursner, and Barbara Paech. Software Feature Request Detection in Issue Tracking Systems. In *2016 IEEE 24th International Requirements Engineering Conference (RE)*, pages 166–175. IEEE, September 2016.

[58] Thomas D. LaToza, Gina Venolia, and Robert DeLine. Maintaining mental models: a study of developer work habits. In *Proceedings of the 28th international conference on Software engineering*, pages 492–501, Shanghai China, May 2006. ACM.

[59] Shihab Hassan Hamid, Matthew Quail, Graeme David Smith, Olli Risto Nevalainen, and Paul Slade. Method and Apparatus for Performing Remote Operations in an Issue Tracking Environment, August 2014.

[60] Jiří Janák. Issue Tracking Systems. Master's thesis, Masaryk University, Brno, 2006.

[61] Debbie Stone, Caroline Jarrett, Mark Woodroffe, and Shailey Minocha. *User Interface Design and Evaluation*. Morgan Kaufmann series in interactive technologies. Elsevier : Morgan Kaufmann, Amsterdam ; Boston, Mass, 2005.

[62] International Organization for Standardization. Ergonomics of human-system interaction — part 210: Human-centred design for interactive systems, 2019. ISO Standard No. 9241-210:2019.

[63] Dongwook Yoon. Introduction to HCI Methods.

[64] Soren Lauesen. *User Interface Design: a Software Engineering Perspective*. Pearson/Addison-Wesley, Harlow Munich, 2007.

[65] International Organization for Standardization. Ergonomics of human-system interaction — part 11: Usability: Definitions and concepts, 2018. ISO Standard No. 9241-11:2018.

[66] A. Taylor. IT projects: sink or swim. *The Computer Bulletin*, 42(1):24–26, January 2000.

[67] J. Preece, Y. Rogers, and H. Sharp. *Interaction Design: Beyond Human-computer Interaction*. Interaction Design: Beyond Human-computer Interaction. J. Wiley & Sons, 2002.

[68] Soren Lauesen and Otto Vinter. Preventing Requirement Defects: An Experiment in Process Improvement. *Requirements Engineering*, 6(1):37–50, February 2001.

[69] Jonathan Schooler. Bridging the objective/subjective divide. *Towards a meta-perspective of science and experienceMetzinger T, Windt JM Open MIND*, 34, 2015.

[70] Sirpa Riihiaho. Usability Testing. In Kent L. Norman and Jurek Kirakowski, editors, *The Wiley Handbook of Human Computer Interaction*, pages 255–275. Wiley, 1 edition, February 2018.

[71] International Organization for Standardization and International Electrotechnical Commission. Software engineering — software product quality requirements and evaluation (square) — common industry format (cif) for usability test reports, 2006. ISO/IEC Standard No. 25062:2006.

[72] James Lewis and Jeff Sauro. Item Benchmarks for the System Usability Scale. 13:158–167, May 2018.

[73] Scott W. Ambler. *The elements of UML 2.0 style*. Cambridge University Press, 2005.

Appendix A

Usability Testing Details

A.1 Test Instructions

Along with the test and questionnaire links, the participants received a short list of instructions, described below.

1. Do the test on a computer or laptop;
2. This test is conducted on a prototype; hence, not everything is functional (e.g., the export percentage won't load);
3. Complete the tasks following the order (from 1 to 5), which will be marked when completed;
4. If a task is still incomplete and you are unable to finish it, you can leave it as it is.

A.2 Satisfaction Questionnaire

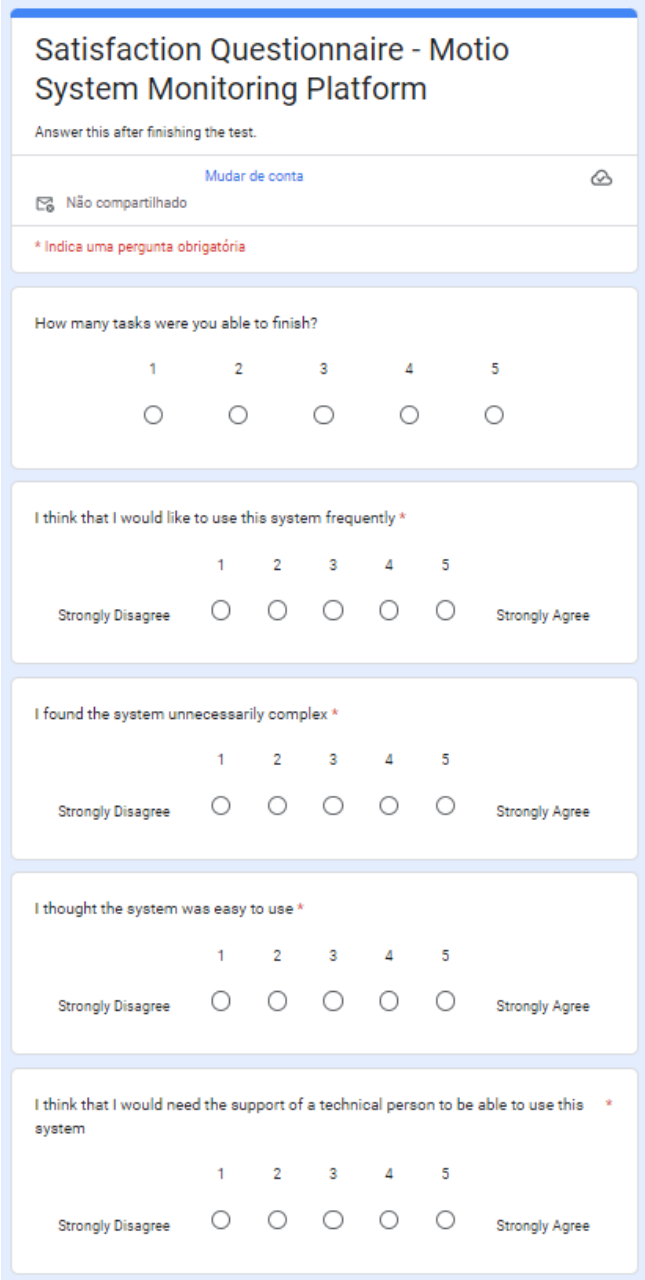


Figure A.1: Satisfaction Questionnaire - Part 1

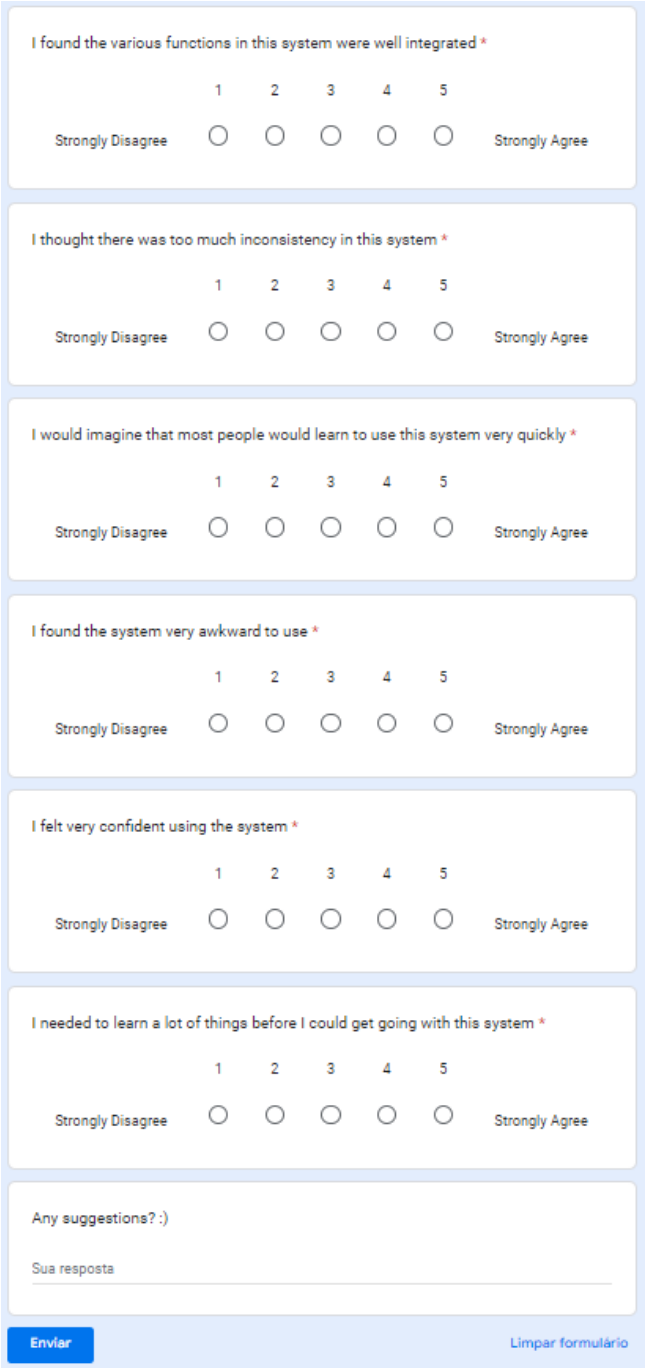


Figure A.2: Satisfaction Questionnaire - Part 2

A.3 Quant-UX Testing Consent

Since Quant-UX records the participant's screen and interaction, it displays an initial message (Figure A.3) elucidating its purpose and asking for their consent. Only after the permission, the participant can start the test.

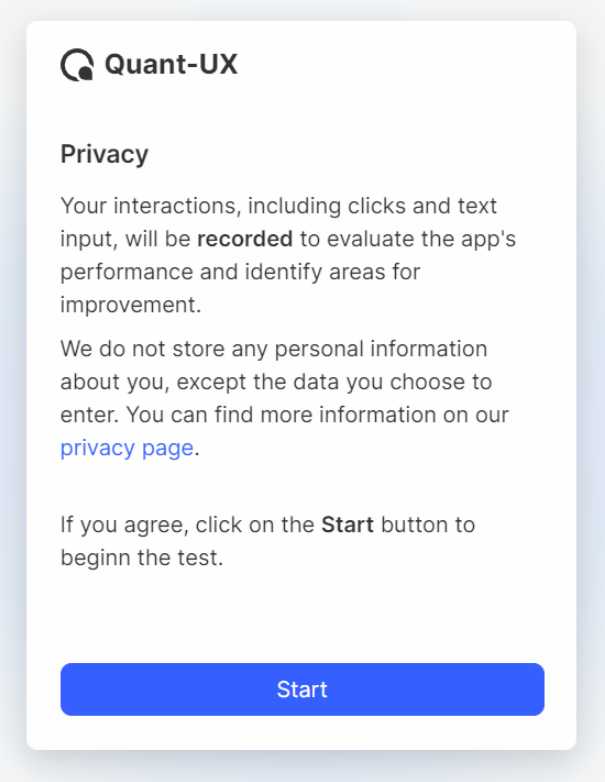


Figure A.3: Quant-UX's initial message about privacy

A.4 Effectiveness and Efficiency Measures - All Results

Table A.1 shows the number of interactions and duration of each completed task from every test.

A.5 Satisfaction Questionnaire - All Answers

Every participant who completed the test also answered the Satisfaction Questionnaire. All answers are displayed in Table A.2.

Table A.1: Usability testing measures per each test and each task

Participant	Task	Interactions	Duration (s)
P01	Task 1	64	125.0
	Task 2	4	2.5
	Task 3	22	17.4
	Task 4	2	6.1
	Task 5	11	11.9
P02	Task 1	4	26.5
	Task 2	4	3.1
	Task 3	11	22.1
	Task 4	2	6.5
	Task 5	10	17.7
P03	Task 1	4	50.1
	Task 2	4	3.0
	Task 3	7	18.1
	Task 4	2	4.5
	Task 5	7	9.7
P04	Task 1	4	79.8
	Task 2	4	3.3
	Task 3	7	17.5
	Task 4	7	11.7
	Task 5	7	7.8
P05	Task 1	4	33.9
	Task 2	4	4.9
	Task 3	8	30.0
	Task 4	2	12.0
	Task 5	7	17.2
P06	Task 1	4	24.2
	Task 2	4	2.5
	Task 3	8	18.8
	Task 4	2	2.5
	Task 5	11	17.8
P07	Task 1	12	56.0
	Task 2	4	3.6
	Task 4	2	6.8
P08	Task 1	8	31.3
	Task 2	4	5.2
	Task 3	24	27.6
	Task 4	10	21.8
P09	Task 1	5	57.2
	Task 2	4	4.5
	Task 3	13	69.9
	Task 4	2	4.5
	Task 5	7	12.7
P10	Task 1	23	24.7
	Task 5	7	6.0
	Task 3	80	833.0
P11	Task 1	5	67.1
	Task 2	4	2.2
	Task 3	9	57.3
	Task 4	2	6.8
	Task 5	7	12.1
P12	Task 1	4	16.4
	Task 2	4	2.5
	Task 3	10	19.8
	Task 4	4	11.9
	Task 5	8	11.7
P13	Task 1	6	17.4
	Task 2	4	3.2
	Task 3	7	17.7
	Task 4	2	7.1
P14	Task 1	4	31.1
	Task 5	10	8.6
P15	Task 1	5	33.9
	Task 2	4	2.8
	Task 3	10	15.8
	Task 4	2	4.5
	Task 5	8	8.3
P16	Task 2	4	2.5
	Task 1	29	91.7
	Task 4	2	12.6
	Task 5	10	10.8
P17	Task 1	6	51.0
	Task 2	4	4.1
	Task 3	12	38.0
	Task 4	2	3.0
	Task 5	9	13.4
P18	Task 1	19	69.8
	Task 2	4	2.6
	Task 3	7	42.3
	Task 4	2	4.2
	Task 5	33	41.1
P19	Task 1	53	57.0
	Task 2	8	4.2
	Task 3	7	21.5
	Task 4	2	260.0
	Task 5	7	7.3

Table A.2: Satisfaction Questionnaire answers per test

Participant	I think that I would like to use this system frequently	I found the system unnecessarily complex	I thought the system was easy to use	I think that I would need the support of a technical person to be able to use this system	I found the various functions in this system were well integrated	I thought there was too much inconsistency in this system	I would imagine that most people would learn to use this system very quickly	I found the system very awkward to use	I felt very confident using the system	I needed to learn a lot of things before I could get going with this system	Completed Tasks
P01	3	2	4	1	4	1	4	1	3	2	5
P02	5	1	5	1	5	1	4	1	5	1	5
P03	4	1	5	1	4	1	5	1	4	1	5
P04	4	1	5	1	5	1	5	1	5	2	5
P05	5	1	5	1	5	1	5	1	5	1	5
P06	5	1	5	1	5	1	5	1	5	1	5
P07	5	1	5	2	4	1	4	2	5	1	3
P08	5	2	5	2	5	1	5	1	5	1	4
P09	5	1	5	2	5	1	4	1	5	2	5
P10	5	1	5	1	4	2	5	1	5	1	3
P11	4	2	5	2	4	1	5	1	4	1	5
P12	5	1	5	1	5	1	5	1	5	1	5
P13	5	1	5	1	5	1	5	1	5	1	4
P14	4	2	5	2	4	2	5	2	4	1	2
P15	5	1	5	1	5	1	5	1	5	1	5
P16	4	1	4	2	3	1	5	1	4	1	4
P17	4	1	5	1	4	1	5	5	5	1	5
P18	5	1	5	1	5	1	4	1	5	1	5
P19	5	1	5	1	5	1	5	1	5	1	5

A.6 Other Outcomes from the Usability Testing

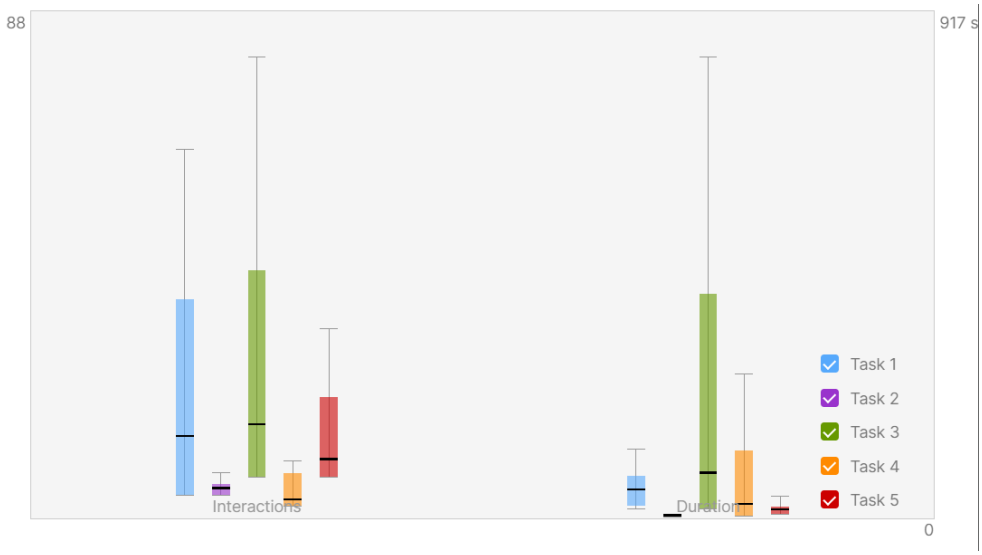


Figure A.4: Boxplot concerning interactions and duration for each task

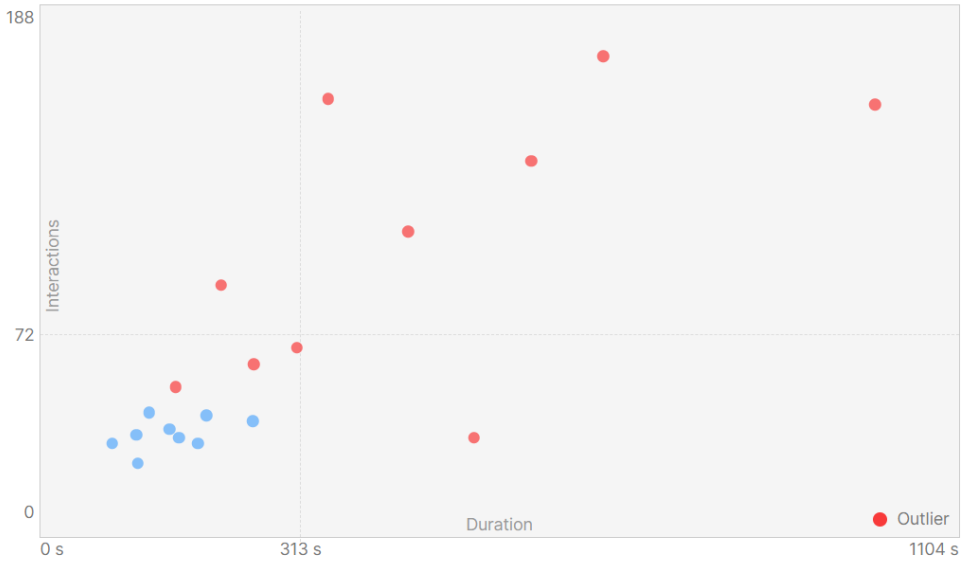


Figure A.5: Data distribution of tests - Interactions X Duration

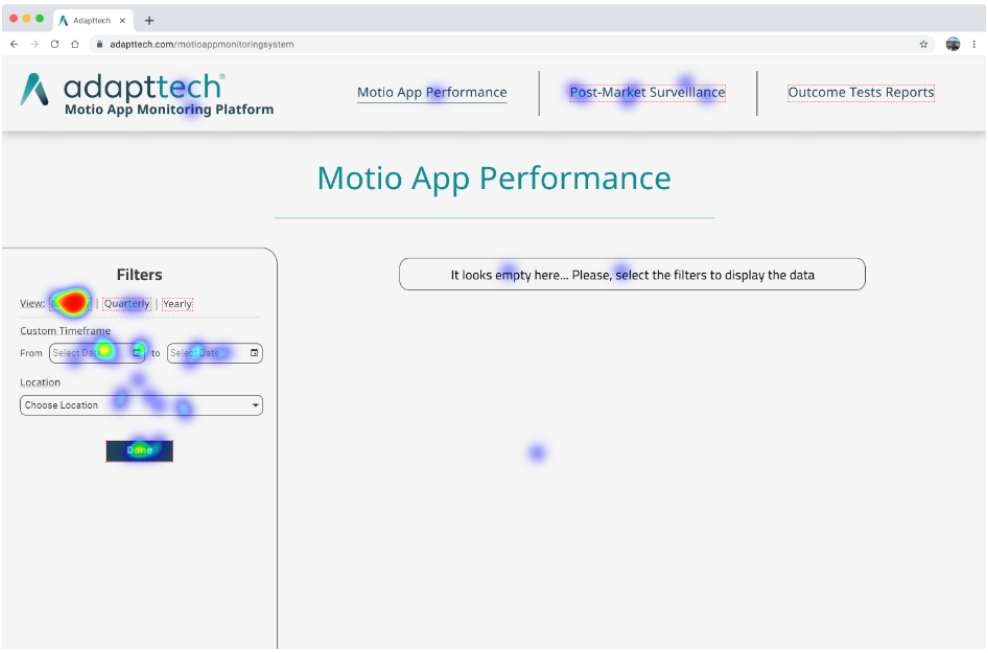


Figure A.6: Heatmap - Motio App Performance - Initial page

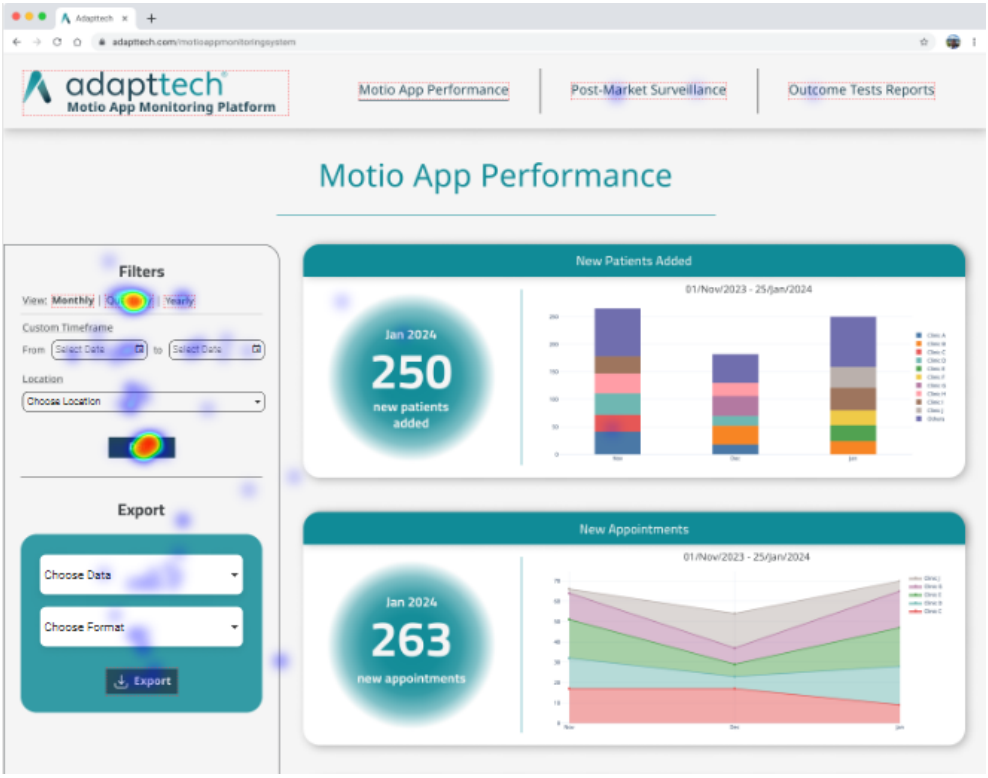


Figure A.7: Heatmap - Motio App Performance - Selecting filters

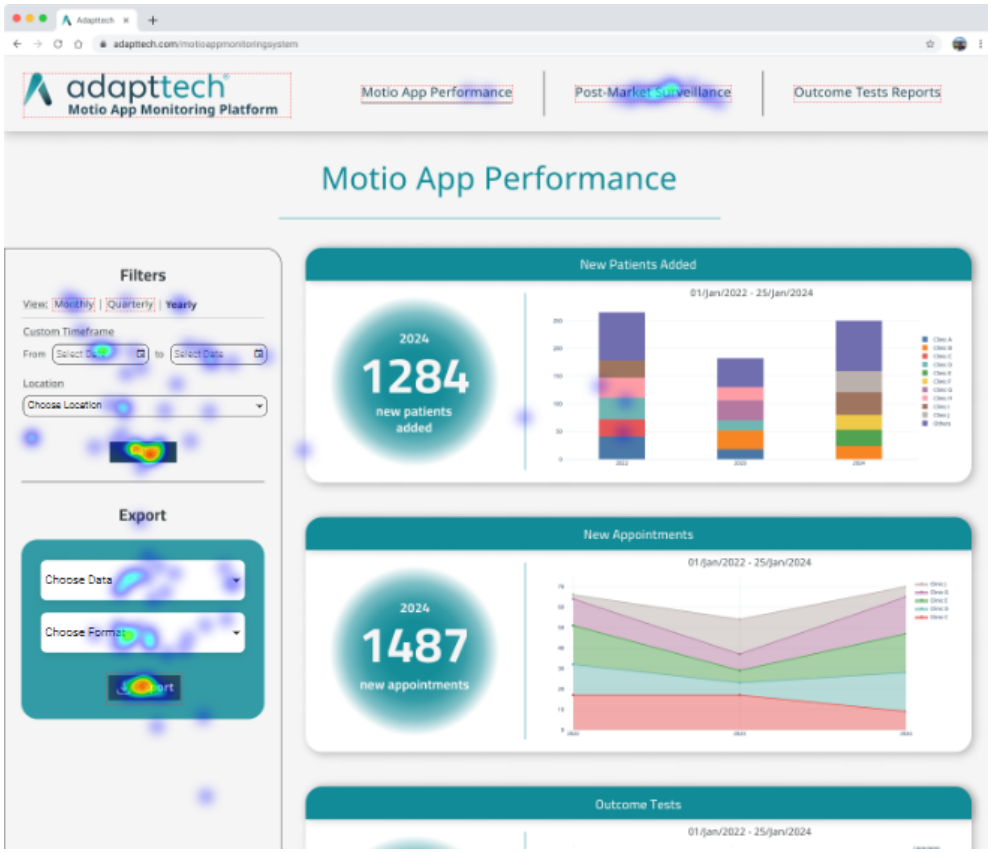


Figure A.8: Heatmap - Motio App Performance - Exporting data

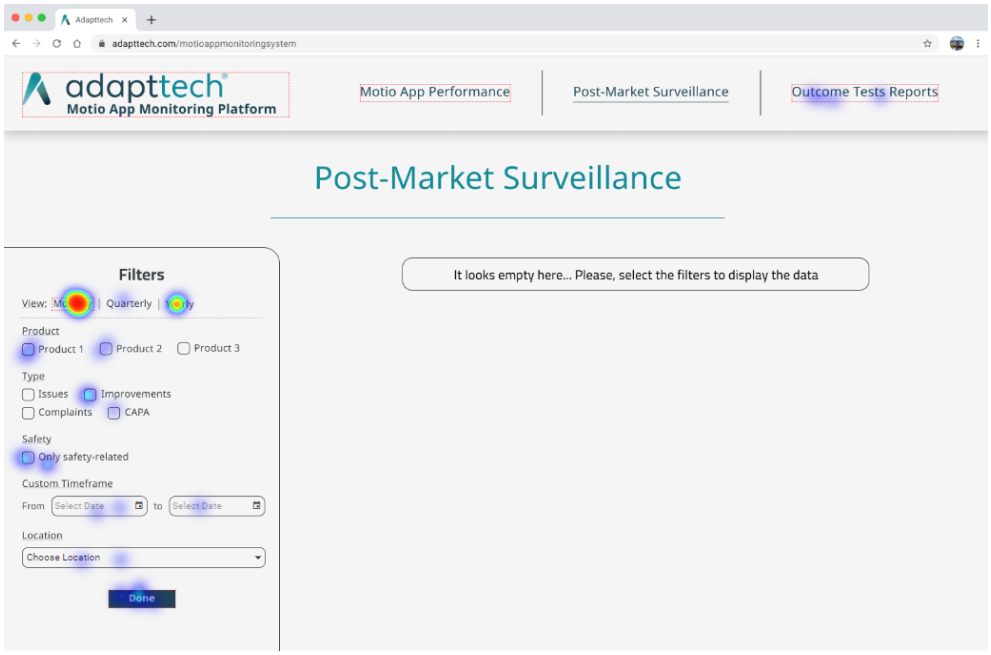


Figure A.9: Heatmap - Post-Market Surveillance - Initial page

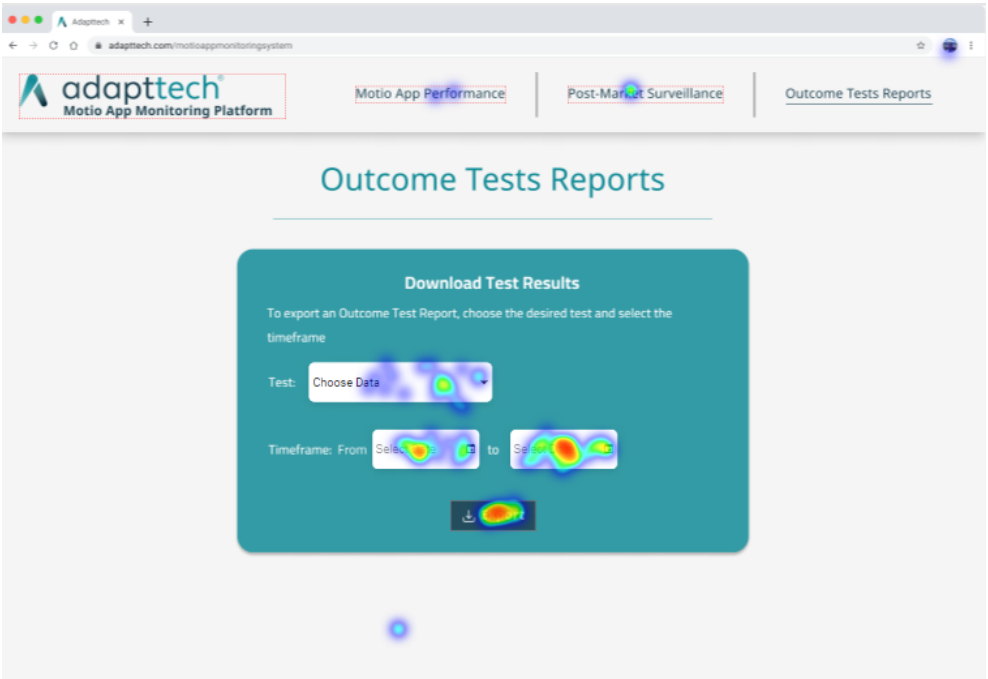


Figure A.10: Heatmap - Outcome Test Reports - Downloading data

Appendix B

Motio App Performance Queries

B.1 Patient Query

```
--Patients without a Location were considered

SELECT Location.name AS LocationName, Patient.createdAt AS PatientCreatedAt
FROM Patient AS Patient
LEFT JOIN PatientLocation AS PatientLocation ON Patient.id = PatientLocation.
    patientId
LEFT JOIN Location AS Location ON PatientLocation.locationId = Location.id
WHERE Patient.createdAt BETWEEN ? AND ? AND Patient.deletedAt IS NULL
```

B.2 Appointment Query

```
SELECT Location.name AS LocationName, Appointment.startDate AS
    AppointmentStartDate
FROM Appointment AS Appointment
JOIN Location AS Location ON Appointment.locationId = Location.id
WHERE Appointment.startDate BETWEEN ? AND ? AND Appointment.deletedAt IS NULL
```

B.3 Outcome Tests Query

```
-- Only Acquisition Tests with Appointments were considered
SELECT Location.name AS LocationName, AcquisitionTestType.name, Appointment.
    startDate AS AppointmentStartDate
FROM AcquisitionTest AS AcquisitionTest
JOIN Appointment AS Appointment ON AcquisitionTest.appointmentId = Appointment.id
```

```
5 JOIN Location AS Location ON Appointment.locationId = Location.id
6 JOIN AcquisitionTestType AS AcquisitionTestType ON AcquisitionTest.
      AcquisitionTestTypeId = AcquisitionTestType.id
7 WHERE Appointment.startDate BETWEEN ? AND ? AND Appointment.deletedAt IS NULL AND
       AcquisitionTest.deletedAt IS NULL
```

B.4 Appointments Related to the Outcome Tests Query

```
1  SELECT top (?) Location.name, COUNT(Appointment.id) AS nAppointments, SUM (
2     CASE WHEN AcquisitionTest.AcquisitionTestTypeId = 0 THEN 1 ELSE 0 END
3  ) AS ClinicianEvaluation,
4  SUM (
5     CASE WHEN AcquisitionTest.AcquisitionTestTypeId = 1 THEN 1 ELSE 0 END
6  ) AS AMPPRO,
7  SUM (
8     CASE WHEN AcquisitionTest.AcquisitionTestTypeId = 2 THEN 1 ELSE 0 END
9  ) AS PLUSM,
10 SUM (
11    CASE WHEN AcquisitionTest.AcquisitionTestTypeId = 3 THEN 1 ELSE 0 END
12 ) AS TUG,
13 SUM (
14    CASE WHEN AcquisitionTest.AcquisitionTestTypeId = 4 THEN 1 ELSE 0 END
15 ) AS LTA
16 FROM Appointment AS Appointment
17 JOIN AcquisitionTest AS AcquisitionTest ON Appointment.id = AcquisitionTest.
       appointmentId
18 JOIN PatientLocation AS PatientLocation ON Appointment.patientId =
       PatientLocation.patientId
19 JOIN Location AS Location ON PatientLocation.locationId = Location.id
20 WHERE Appointment.startDate BETWEEN ? AND ? AND Appointment.deletedAt IS NULL AND
       AcquisitionTest.deletedAt IS NULL
21 GROUP BY Location.name
22 ORDER BY COUNT(Appointment.id) DESC
```

B.5 PLUS-M™ Query

```
1 SELECT PlusM.score AS PlusMScore,
2       Appointment.startDate,
3       Gender.type, year(Patient.dateOfBirth) AS YearOfBirth,
4       AmputationType.type AS leftLegAmputationLevel,
5       AmputationType2.type AS rightLegAmputationLevel,
6       EtiologyType.type AS leftLegEtiologyType,
```

```
7       EtiologyType2.type AS rightLegEtiologyType,
8       (
9          SELECT AMPPRO.score
10         FROM Appointment AS AppointmentAMPPRO
11         JOIN AcquisitionTest AS AcquisitionTestAMPPRO ON AppointmentAMPPRO.id =
               AcquisitionTestAMPPRO.appointmentId
12         FULL JOIN AMPPRO AS AMPPRO ON AcquisitionTestAMPPRO.id = AMPPRO.
               acquisitionTestId
13         WHERE AMPPRO.score IS NOT NULL AND AcquisitionTestAMPPRO.deletedAt IS
               NULL and AppointmentAMPPRO.id = Appointment.id
14      ) AS AMPPROScore,
15      (
16         SELECT TUG.duration
17         FROM Appointment AS AppointmentTUG
18         JOIN AcquisitionTest AS AcquisitionTestTUG ON AppointmentTUG.id =
               AcquisitionTestTUG.appointmentId
19         FULL JOIN TUG AS TUG ON AcquisitionTestTUG.id = TUG.acquisitionTestId
20         WHERE TUG.duration IS NOT NULL AND AcquisitionTestTUG.deletedAt IS NULL
               AND AppointmentTUG.id = Appointment.id
21      ) AS TUGDuration,
22      (
23         SELECT ClinicianEvaluation.kLevel
24         FROM Appointment AS AppointmentCE
25         JOIN AcquisitionTest AS AcquisitionTestCE ON AppointmentCE.id =
               AcquisitionTestCE.appointmentId
26         FULL JOIN ClinicianEvaluation AS ClinicianEvaluation ON
               AcquisitionTestCE.id = ClinicianEvaluation.acquisitionTestId
27         WHERE ClinicianEvaluation.kLevel IS NOT NULL AND AcquisitionTestCE.
               deletedAt IS NULL AND AppointmentCE.id = Appointment.id
28      ) AS kLevel,
29      (
30         SELECT AcquisitionTestMetric.value
31         FROM Appointment AS Appointment10M
32         JOIN AcquisitionTest AS AcquisitionTest10M ON Appointment10M.id =
               AcquisitionTest10M.appointmentId
33         FULL JOIN AcquisitionTestMetric AS AcquisitionTestMetric ON
               AcquisitionTest10M.id = AcquisitionTestMetric.acquisitionTestId
34         WHERE AcquisitionTestMetric.value IS NOT NULL AND AcquisitionTest10M.
               deletedAt IS NULL AND AcquisitionTestMetric.metricId = 24 AND
               Appointment10M.id = Appointment.id
35      ) AS TenMeterWalkTestsAverageStd,
36      (
37         SELECT AcquisitionTestMetric.value
38         FROM Appointment AS Appointment10M
39         JOIN AcquisitionTest AS AcquisitionTest10M ON Appointment10M.id =
               AcquisitionTest10M.appointmentId
40         FULL JOIN AcquisitionTestMetric AS AcquisitionTestMetric ON
               AcquisitionTest10M.id = AcquisitionTestMetric.acquisitionTestId
```

```
41          WHERE AcquisitionTestMetric.value IS NOT NULL AND AcquisitionTest10M.
                deletedAt IS NULL AND AcquisitionTestMetric.metricId = 22 AND
                Appointment10M.id = Appointment.id
42      ) AS TwoMinuteWalkTestsAverageStd,
43      (
44          SELECT AcquisitionTestMetric.value
45          FROM Appointment AS Appointment10M
46          JOIN AcquisitionTest AS AcquisitionTest10M ON Appointment10M.id =
                AcquisitionTest10M.appointmentId
47          FULL JOIN AcquisitionTestMetric AS AcquisitionTestMetric ON
                AcquisitionTest10M.id = AcquisitionTestMetric.acquisitionTestId
48          WHERE AcquisitionTestMetric.value IS NOT NULL AND AcquisitionTest10M.
                deletedAt IS NULL AND AcquisitionTestMetric.metricId = 23 AND
                Appointment10M.id = Appointment.id
49      ) AS SixMinuteWalkTestsAverageStd
50  FROM PlusM AS PlusM
51  JOIN AcquisitionTest AS AcquisitionTest ON PlusM.acquisitionTestId =
        AcquisitionTest.id
52  JOIN Appointment AS Appointment ON AcquisitionTest.appointmentId = Appointment.id
53  JOIN Patient AS Patient ON Patient.id = Appointment.patientId
54  JOIN GenderType AS Gender ON Patient.genderTypeId = Gender.id
55  JOIN AmputationDegree AS AmputationDegree ON PlusM.amputationDegreeId =
        AmputationDegree.id
56  JOIN AmputationType AS AmputationType ON AmputationDegree.leftAmputationTypeId =
        AmputationType.id
57  JOIN AmputationType AS AmputationType2 ON AmputationDegree.rightAmputationTypeId
        = AmputationType2.id
58  JOIN EtiologyType AS EtiologyType ON PlusM.rightLegEtiologyTypeId = EtiologyType.
        id
59  JOIN EtiologyType AS EtiologyType2 ON PlusM.leftLegEtiologyTypeId = EtiologyType2
        .id
60  WHERE AcquisitionTest.deletedAt IS NULL AND Appointment.startDate BETWEEN ? AND ?
```

B.6 AMPPRO Query

```
1   SELECT AMPPRO.score AS AMPPROScore,
2          AmputationType.type AS leftLegAmputationLevel,
3          AmputationType2.type AS rightLegAmputationLevel,
4          year(Patient.dateOfBirth) AS YearOfBirth,
5          Patient.prostheticDeviceSince,
6          Appointment.startDate,
7          (
8              SELECT EtiologyType.type
9              FROM Appointment AS AppointmentPlusM
10             JOIN AcquisitionTest AS AcquisitionTestPlusM ON AppointmentPlusM.id =
                   AcquisitionTestPlusM.appointmentId
```

```
11          FULL JOIN PlusM AS PlusM ON AcquisitionTestPlusM.id = PlusM.
                acquisitionTestId
12          JOIN EtiologyType AS EtiologyType ON PlusM.rightLegEtiologyTypeId =
                EtiologyType.id
13          WHERE AcquisitionTestPlusM.deletedAt IS NULL and AppointmentPlusM.id =
                Appointment.id
14       ) AS rightLegEtiology,
15       (
16          SELECT EtiologyType2.type
17          FROM Appointment AS AppointmentPlusM
18          JOIN AcquisitionTest AS AcquisitionTestPlusM ON AppointmentPlusM.id =
                AcquisitionTestPlusM.appointmentId
19          FULL JOIN PlusM AS PlusM ON AcquisitionTestPlusM.id = PlusM.
                acquisitionTestId
20          JOIN EtiologyType AS EtiologyType2 ON PlusM.leftLegEtiologyTypeId =
                EtiologyType2.id
21          WHERE AcquisitionTestPlusM.deletedAt IS NULL and AppointmentPlusM.id =
                Appointment.id
22       ) AS leftLegEtiology,
23       (
24          SELECT ClinicianEvaluation.kLevel
25          FROM Appointment AS AppointmentCE
26          JOIN AcquisitionTest AS AcquisitionTestCE ON AppointmentCE.id =
                AcquisitionTestCE.appointmentId
27          FULL JOIN ClinicianEvaluation AS ClinicianEvaluation ON
                AcquisitionTestCE.id = ClinicianEvaluation.acquisitionTestId
28          WHERE ClinicianEvaluation.kLevel IS NOT NULL AND AcquisitionTestCE.
                deletedAt IS NULL AND AppointmentCE.id = Appointment.id
29       ) AS kLevel
30 FROM AMPPRO AS AMPPRO
31 JOIN AcquisitionTest AS AcquisitionTest ON AMPPRO.acquisitionTestId =
       AcquisitionTest.id
32 JOIN Appointment AS Appointment ON AcquisitionTest.appointmentId = Appointment.id
33 JOIN Patient AS Patient ON Patient.id = Appointment.patientId
34 JOIN GenderType AS Gender ON Patient.genderTypeId = Gender.id
35 JOIN AmputationDegree AS AmputationDegree ON AMPPRO.amputationDegreeId =
       AmputationDegree.id
36 JOIN AmputationType AS AmputationType ON AmputationDegree.leftAmputationTypeId =
       AmputationType.id
37 JOIN AmputationType AS AmputationType2 ON AmputationDegree.rightAmputationTypeId
       = AmputationType2.id
38 WHERE AcquisitionTest.deletedAt IS NULL AND Appointment.startDate BETWEEN ? AND ?
```

B.7 TUG Query

```
SELECT TUG.duration, TUG.armMovement, TUG.assistiveDevice, TUG.balance, TUG.
    enBlocTurning, TUG.shortStrides, TUG.shuffling, TUG.slowAndTentative, TUG.
    steadying, Appointment.createdAt AS Date, AcquisitionTestMetric.value AS
    FallRisk
FROM TUG AS TUG
JOIN AcquisitionTest AS AcquisitionTest ON TUG.acquisitionTestId =
    AcquisitionTest.id
JOIN Appointment AS Appointment ON AcquisitionTest.appointmentId = Appointment.id
JOIN AcquisitionTestMetric AS AcquisitionTestMetric ON AcquisitionTest.id =
    AcquisitionTestMetric.acquisitionTestId
WHERE AcquisitionTest.deletedAt IS NULL AND AcquisitionTestMetric.metricId = 3
    AND Appointment.startDate BETWEEN ? AND ?
```