

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Intelligent Project Tracking System

Juan Bellon Lopez



Mestrado em Engenharia Informática e Computação

Supervisor: Prof. João Pascoal Faria

Second Supervisor: João Miguel Araujo

September 24, 2024

Intelligent Project Tracking System

Juan Bellon Lopez

Mestrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Prof. Doutora Ana Cristina Ramada Paiva

Arguente: Prof. Doutor Miguel António Sousa Abrunhosa Brito

Vogal: Prof. Doutor João Carlos Pascoal Faria

September 24, 2024

Resumo

A estimativa de *Story Points* continua sendo um desafio principal na gestão de projetos ágeis, especialmente em projetos de desenvolvimento de software. Esta dissertação apresenta o "Sistema Inteligente de Rastreamento de Progresso", desenvolvido para aprimorar a previsão de *Story Points*, oferecendo maior precisão no planejamento de sprints. O sistema utiliza inteligência artificial com o objetivo de analisar dados históricos e identificar padrões de desempenho anteriores, permitindo previsões mais eficazes e ajustadas à realidade de cada projeto.

O foco deste sistema está na capacidade de prever o esforço necessário para a conclusão de tarefas, baseado em *Story Points*, oferecendo uma visão clara e preditiva do progresso dos projetos. Ao utilizar *Machine Learning* para analisar os dados de progresso e alocação de recursos, o sistema permite que os gestores tomem decisões mais informadas sobre prazos e sobrecarga de equipes. A capacidade de previsão de prazos e otimização da alocação de recursos melhora consideravelmente a eficiência das equipes de desenvolvimento.

A dissertação investiga a aplicação da IA na estimativa de *Story Points* e no rastreamento de projetos ágeis, mostrando como essas tecnologias podem complementar e melhorar os métodos existentes de planejamento. O resultado é um sistema que não apenas automatiza as previsões de esforço, mas também permite ajustes em tempo real conforme o andamento do projeto, otimizando os recursos e por consequência, reduzir os riscos.

O valor desta pesquisa reside na capacidade do sistema de fornecer previsões preditivas precisas, que auxiliam na tomada de decisões e no planejamento de sprints. Os principais objetivos desta tese incluem o desenvolvimento de uma ferramenta que ofereça previsões detalhadas baseadas em *Story Points*, bem como visualizações que permitam o rastreamento contínuo e a alocação eficiente de recursos.

A capacidade deste sistema foi validada mediante um caso de estudo com profissionais de Deloitte, uma empresa global de serviços profissionais em diversas áreas, incluindo consultoria tecnológica. O caso prático mostrou que este sistema tem a possibilidade de ajudar na estimativa dos projetos e por consequência reduzir significativamente os riscos.

Abstract

Estimating Story Points remains a key challenge in agile project management, especially in software development projects. This dissertation presents the "Intelligent Progress Tracking System", developed to improve the prediction of *Story Points*, offering greater precision in sprint planning. The system uses artificial intelligence with the aim of analyzing historical data and identifying previous performance patterns, allowing for more effective predictions adjusted to the reality of each project.

The focus of this system is on the ability to predict the effort required to complete tasks based on Story Points, offering a clear and predictive view of project progress. By using *Machine Learning* to analyze progress data and resource allocation, the system allows managers to make more informed decisions about deadlines and team overload. The ability to predict deadlines and optimize resource allocation considerably improves the efficiency of development teams.

The dissertation investigates the application of AI in estimating Story Points and tracking agile projects, showing how these technologies can complement and improve existing planning methods. The result is a system that not only automates effort forecasts but also allows real-time adjustments as the project progresses, optimizing resources and, consequently, reducing risks.

The value of this research lies in the system's ability to provide accurate predictive forecasts, which aid in decision-making and sprint planning. The main objectives of this thesis include the development of a tool that offers detailed predictions based on Story Points, as well as visualizations that enable continuous tracking and efficient resource allocation.

The capacity of this system was validated through a case study with professionals from Deloitte, a global professional services company in several areas, including technological consultancy. The practical case showed that this system has the possibility of helping to estimate projects and, therefore, significantly reduce risks.

Agradecimentos

First and foremost, A heartfelt thank you to Professor João Pascoal Faria for his invaluable guidance, patience, and insightful feedback, which were crucial to the development of this work.

I thank my family — my parents, Loreto and Jorge, and my sister, Loreto, for their unwavering love, encouragement, and belief in me, even during the most challenging moments.

I would also like to extend my gratitude to Deloitte and my supervisor, João Miguel Araujo, for providing me with the opportunity and resources to work on this project and for offering valuable support.

I am thankful to all my friends and colleagues who have been a source of encouragement and assistance.

Lastly, I want to express my utmost gratitude to Carlota for her unwavering support, patience, love and comprehension during every step of this process. Your constant encouragement has been vital in the conclusion of this thesis.

Juan Bellon Lopez

*“If you fulfil your obligations every day
you don’t need to worry about the future”*

Jordan Peterson

Contents

1	Introduction	1
1.1	Context and Motivation	1
1.2	Goals	2
1.3	Document Structure	2
2	Background and State of the Art	4
2.1	Stages of a Software Development Project	4
2.1.1	Inception Phase	5
2.1.2	Build Phase	5
2.1.3	Testing Phase	5
2.2	Methodologies of Software Delivery	6
2.2.1	Waterfall	6
2.2.2	Agile	7
2.2.3	Scrum	7
2.3	Businnes Intelligence	8
2.3.1	Predictive Analytics	9
2.4	Risk Management	11
2.5	AI Integration in Project Tracking	11
2.5.1	Machine Learning	11
2.5.2	Large Language Models	12
2.5.3	Generative Pre-trained Transformer	12
2.5.4	Prompt engineering	12
2.5.5	AI and ML Advancements	13
2.5.6	Challenges and Limitations in Integrating AI and ML	13
2.6	User Story Estimation	14
2.7	Existing Story Point Prediction Models	14
2.7.1	Story Points	15
2.7.2	Using Latent Dirichlet Allocation	16
2.7.3	GPT2SP	17
2.8	Conclusion	17
3	Intelligent Project Tracking System	19
3.1	Main Functionalities	19
3.1.1	User Interface and Interaction	19
3.1.2	Core Functionalities	22
3.2	Architecture	25
3.2.1	Detailed Architectural Components	26
3.2.2	Technologies Used	27

3.2.3	Data	29
3.2.4	AI Integration	29
3.2.5	Results	32
3.2.6	Workflow Overview	34
4	Validation	36
4.1	Experimental Validation	36
4.1.1	Iterative Feedback and System Enhancements	36
4.1.2	Testing with JSON Files	37
4.1.3	Testing with JIRA REST API	38
4.1.4	User-backed Enhancements	38
4.2	Use Case	39
4.2.1	Background questionnaire	39
4.2.2	Task Execution	39
4.2.3	Post-task Questionnaire	41
5	Conclusions and Future Work	46
5.1	Conclusions	46
5.2	Main Contributions	46
5.3	Future Work	47
	References	48
A	Intelligent Project Tracking System Validation	51

List of Figures

2.1	Waterfall Methodology	6
2.2	Agile Methodology	7
2.3	Scrum Methodology [21]	8
2.4	Business Intelligence cycle	9
2.5	Project Completion Time Through Sprints [31]	10
2.6	Planning Poker Cards	15
2.7	Overview architecture of GPT2SP	17
3.1	Intelligent Project Tracking Start Page	21
3.2	Intelligent Project Tracking Completion Rate	22
3.3	Intelligent Project Tracking Resource Allocation	23
3.4	Intelligent Project Tracking Project Phase Timeline	24
3.5	Intelligent Project Tracking Story Point Prediction Table	25
3.6	Intelligent Project Tracking Architecture	27
3.7	Intelligent Project Tracking Workflow	35
4.1	Current Position	40
4.2	Usage Frequency of Project Management Tools	40
4.3	Years of Experience in Project Management	40
4.4	SUS results	42
4.5	Usefulness of Features	44

List of Tables

3.1	Summary of Issues and Story Points (SP) in Projects P1 and P2	29
3.2	Performance Metrics for LLM, ML, and Naive Approaches in Project 1 (P1) . . .	32
3.3	Performance Metrics for LLM, ML, and Naive Models in Project 2 (P2)	32

Listings

Abbreviations

API	Application Programming Interface
AI	Artificial Intelligence
CSS	Cascading Style Sheets
ETL	Extract, Transform, Load
GPT	Generative Pre-trained Transformer
HTML	Hypertext Markup Language
IQR	Interquartile Range
I/O	Input/Output
JS	JavaScript
JSON	JavaScript Object Notation
KPI	Key Performance Indicator
LLM	Large Language Model
MAE	Mean Absolute Error
ML	Machine Learning
MSE	Mean Squared Error
NLP	Natural Language Processing
RMSE	Root Mean Squared Error
R ²	Coefficient of Determination
SDLC	Software Development Life Cycle
SP	Story Point
TF-IDF	Term Frequency-Inverse Document Frequency
UI	User Interface
URL	Uniform Resource Locator
XP	Extreme Programming

Chapter 1

Introduction

This chapter provides an overview of the dissertation, outlining the background, motivation, objectives, and structure of the work. The aim is to offer a comprehensive understanding of the research goals and establish a clear framework for the subsequent sections of the dissertation.

1.1 Context and Motivation

The integration of new technologies has caused a substantial transformation in project management, particularly artificial intelligence (AI) and machine learning (ML). Traditional project tracking systems have largely relied on manual data input, static reporting, and retrospective analysis, which often lack the flexibility to predict future outcomes or respond to real-time changes. As projects grow in complexity, the limitations of these conventional methods become more pronounced, necessitating the development of advanced systems capable of managing large data sets, offering predictive insights, and enabling proactive decision-making.

In modern organizations, project management involves coordinating a wide range of activities, resources, and timelines across multiple departments, all of which incur costs. The data generated from these activities is often siloed across various platforms, such as task management tools, collaboration software, and external services. The challenge lies in effectively merging these disparate data sources to produce a unified and actionable view of project progress. This dissertation addresses these challenges by proposing the development of an intelligent project-tracking application that leverages AI and ML to improve project visibility, optimize resource allocation, and forecast key project events, thus mitigating risks.

This research is motivated by practical needs identified within Deloitte, which manages a diverse portfolio of projects varying in size, complexity, and scope and often encounters the limitations of existing project management tools. The Intelligent Project Tracking system was designed to meet specific needs, offering a more agile and anticipatory approach to project management by incorporating real-time data analysis and machine learning to improve decision-making, risk management, and resource allocation.

1.2 Goals

Building on the context and motivation, this dissertation aims to design and implement an intelligent project-tracking system that applies GPT models and Machine Learning with the goal of aiding in story point prediction.

The specific goals of this dissertation are:

- Conduct a comprehensive review of the latest project management tools and techniques, focusing on the integration of machine learning and AI.
- Develop a data processing pipeline capable of handling and integrating project data, providing a unified view of predicted story points and project progress.
- Design and implement machine learning models to predict story points and estimate project end dates based on the remaining issues.
- Test and validate the Intelligent Project Tracking application through user evaluations at Deloitte, using pre-defined data and collecting feedback via pre- and post-testing forms.
- Evaluate the accuracy and performance of the predictive models using industry-standard metrics.
- Share the research findings with the academic community to contribute to the body of knowledge on AI-driven project management.

1.3 Document Structure

This dissertation is organized as follows:

- **Chapter 2: Background and State-of-the-Art Analysis**

This chapter introduces the relevant topics underpinning the research, followed by a detailed analysis of current state-of-the-art approaches in project management, particularly in the context of machine learning and AI integration. Related work and existing tools are reviewed to provide a foundation for the development of the Intelligent Project Tracking system.

- **Chapter 3: Intelligent Project Tracking Application**

This chapter outlines the design and implementation of the Intelligent Project Tracking application. It covers the system architecture, data integration processes, and the application of machine learning models for predictive analytics. The chapter concludes with an analysis of quantitative results.

- **Chapter 4: Validation and Case Study**

The validation process is described in this chapter, including both experimental validation

in controlled environments and a real-world case study at Deloitte. Feedback from Deloitte's project managers and other stakeholders is analyzed to assess the tool's usability and effectiveness.

- **Chapter 5: Conclusion and Future Work**

This chapter summarizes the research findings, discusses the implications of the Intelligent Project Tracking system, and suggests potential areas for future work.

This structure is intended to guide the reader through the systematic development and evaluation of the Intelligent Project Tracking system, ensuring that each aspect of the research is thoroughly explored and documented.

Chapter 2

Background and State of the Art

This chapter presents a comprehensive overview of the current research relevant to the dissertation, structured into multiple sections for clarity and coherence. The goal is to explore the foundational concepts, methodologies, and advanced tools that form the basis of the proposed Intelligent Project Tracking System while also situating this work within the broader context of software development and Agile methodologies.

The first section delves into the stages of software development, emphasizing the growing role of Agile methodologies and their adaptability in dynamic environments. It examines the challenges inherent in Agile practices, such as user story estimation and risk management, particularly as they relate to the integration of AI-driven solutions and predictive analytics. The discussion further examines the evolution of project tracking, with an emphasis on how AI has transformed traditional approaches by offering other solutions to sprint planning and better utilization of resources.

The second section offers a thorough review of existing studies and methodologies in Agile software development, identifying both common ground and gaps. It provides a detailed analysis of the limitations of current systems and methodologies. The chapter concludes by comparing the proposed system to other state-of-the-art solutions, illustrating how it addresses existing challenges and contributes new advancements to the field.

2.1 Stages of a Software Development Project

The Software Development Life Cycle (SDLC) is a comprehensive guide for systematically creating and maintaining software systems [3]. This framework delineates the progression of software development through a series of defined phases, each with its own set of goals and outputs, ensuring structured and disciplined execution. The SDLC's methodical approach facilitates the delivery of superior software by segmenting the development process into discrete, manageable stages that cumulatively lead to the successful completion of the project.

2.1.1 Inception Phase

The Inception phase is the initial stage where the foundation of a software project is established. It entails a thorough process of obtaining requirements, during which the scope and objectives of the project are clarified. During this phase, stakeholder meetings play a crucial role in determining the objectives of the software, while early feasibility studies are carried out to evaluate the project's potential for success. Sommerville [28] emphasises the importance of a strong Inception phase in ensuring the success of a project, as it establishes a firm groundwork for all future activities. The deliverables of this phase consist of a project charter, preliminary risk assessment, and resource estimates, which function as a blueprint for the project's progression.

2.1.2 Build Phase

Transitioning from the planning stages, the Build phase is where design and development activities come to the fore. Creating software architecture, coding, and integrating various software components into a functioning system characterizes this phase. Boehm's Spiral Model [4] introduces an iterative element to this phase, recognizing the complexity inherent in software development and the need for flexibility. Iterations within the Spiral Model allow for gradual refinement and risk management, accommodating changes in project requirements as the project's understanding deepens.

2.1.3 Testing Phase

The testing phase is a rigorous process designed to ensure that the software performs as required. As articulated in the reference [24], testing encompasses a broad spectrum of activities aimed at uncovering defects or areas of non-conformance to specifications. From unit testing, which examines individual components, to system testing, which evaluates the integrated system, the testing phase focuses on ensuring the software is reliable, secure, and user-friendly. This phase culminates in acceptance testing, where end-users verify that the software meets their needs and expectations.

Closely linked to the testing phase is the Quality Assurance (QA) phase, which extends beyond testing to ensure a continuous commitment to quality throughout the software development process. QA is not a separate phase but rather an integral sub-phase that runs parallel to testing. As described in [19], QA involves a comprehensive approach to quality planning, control, and improvement. Its goal is not only to detect defects but also to prevent them, ensuring that software consistently meets established standards and achieves the level of excellence expected by stakeholders. By integrating QA with the testing phase, both processes work together to guarantee that the final product is robust, reliable, and of high quality.

In conclusion, the SDLC is an essential framework ensuring a structured software development approach. Its clearly defined phases—Inception, Build, Testing, and Quality Assurance—contribute unique value to software development. By adhering to the principles and activities specified within each SDLC phase, project teams can deliver software products that effectively meet or exceed the highest quality standards and fulfil user requirements.

2.2 Methodologies of Software Delivery

In this subsection, I explore some of the primary methodologies of software delivery used in the industry, thereby grounding the research in practical, real-world applications.

2.2.1 Waterfall

The Waterfall model stands as a time-honoured and straightforward framework in software development. Its static, linear, and sequential approach is detailed in the reference [1]. Characterized by distinct, non-overlapping phases — requirement analysis, design, implementation (coding), testing, and operation and maintenance — the Waterfall model mandates the completion of one phase before progressing to the next. While this approach provides clear structure and milestones, it presents limitations, particularly in its rigidity and lack of flexibility to revisit or iterate previous stages.

This methodology is typically employed in projects with well-defined requirements and where changes during the development process are minimal or non-existent. It excels in environments where the project scope and deliverables are evident from the outset, allowing for a more predictable planning and execution path. The model's straightforward nature makes it easily understandable, thus often being the preferred choice in projects where clarity and simplicity are paramount.

Despite its limitations in accommodating changes, the Waterfall model remains a staple in specific contexts, mainly where simplicity, predictability, and cost-effectiveness are crucial. It is especially prevalent in projects where post-coding modifications are prohibitively costly or unfeasible.

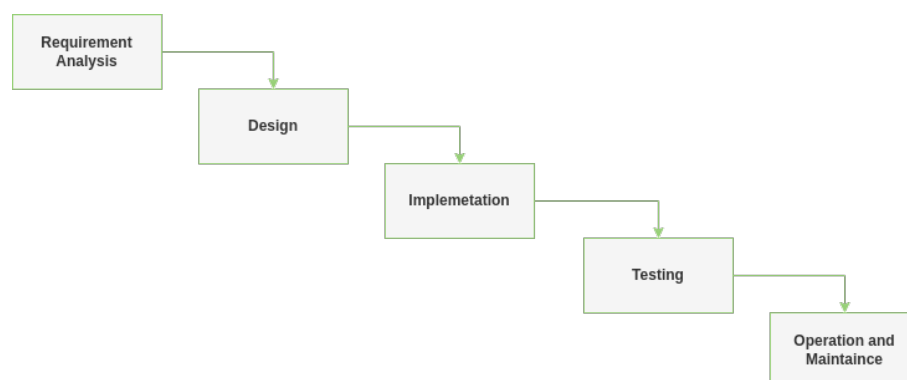


Figure 2.1: Waterfall Methodology

The non-overlapping and sequential nature of the Waterfall model phases is visually depicted in Figure 2.1. This representation underscores the model's structured approach and aids in comprehending its application in software development projects.

2.2.2 Agile

Agile approaches, such as Extreme Programming (XP) and Scrum, emerged as a response to the rigidities of the Waterfall methodology. Individuals and interactions are prioritized above processes and technologies in the Agile concept, working software over detailed documentation, customer collaboration over contract negotiation, and adaptability over rigidity.

Agile methodologies are typically employed in projects requiring flexibility, rapid iteration, and frequent reassessment of tasks and goals. They are particularly well-suited for environments with rapidly changing requirements or projects where client involvement and feedback are critical throughout development.

Fundamental principles of Agile include delivering valuable software early and continuously, welcoming evolving requirements, and promoting close collaboration between business stakeholders and developers. Agile practices support sustainable development by maintaining a consistent, manageable pace and ensuring the software remains releasable at each iteration's conclusion. The iterative nature of Agile allows for the adjustment of project scope based on evolving customer needs and feedback, making it a dynamic and adaptive approach.



Figure 2.2: Agile Methodology

As illustrated in Figure 2.2, Agile methodology necessitates high levels of communication, especially crucial in smaller projects where this close interaction facilitates planning for subsequent iterations. Agile also demands experienced developers capable of making realistic time estimates and adapting to the evolving needs of a project, posing challenges for novice programmers.

2.2.3 Scrum

Scrum, one of the most widely used Agile methods, integrates Iterative and Incremental models. It emphasizes productivity through continuous iterations, effective communication, and meticulous

planning. Scrum involves close collaboration among the Scrum team, Scrum Master, and Product Owner, clearly defining each role and activity. The process is structured around sprints, typically lasting 1 to 3 weeks, each aiming to deliver a potentially shippable product increment.

Scrum is particularly effective in contexts requiring frequent reassessment and adaptation. Its sprint-based structure allows teams to evaluate progress and adjust their approach as needed regularly. This flexibility makes Scrum ideal for projects with evolving requirements or those that benefit from regular stakeholder feedback.

The hallmark of Scrum is its emphasis on flexibility, customer satisfaction, and rapid adaptability to changes, qualities that have made it highly effective across various industries. Its adaptability aligns with the dynamic nature of modern projects, enabling teams to respond quickly to changes without disrupting the overall project trajectory.

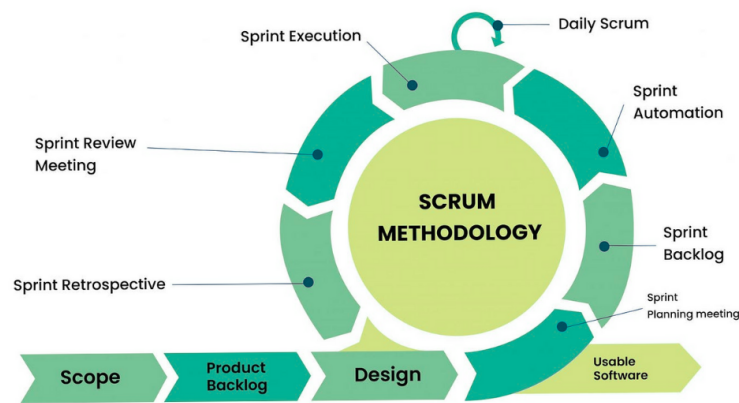


Figure 2.3: Scrum Methodology [21]

Figure 2.3 illustrates the Scrum workflow and key components, clearly representing how Scrum facilitates effective project management. This depiction highlights the roles and processes essential to Scrum's success.

In conclusion, while the Waterfall model provided a foundational structure for software development, Agile methodologies, particularly Scrum, have emerged as more adaptable and responsive to the evolving needs of modern software projects. These methodologies accommodate and embrace change as a tool for continuous improvement, thus aligning well with the development principles of modern, dynamic software environments.

2.3 Business Intelligence

Business Intelligence (BI) refers to the process where businesses collect and analyze data to drive decision-making and strategy. The image below 2.4 illustrates this cyclical process. Business generates the data, which is then extracted and processed to create intelligence. This loop provides companies with a constant, number-based opportunity for improvement.

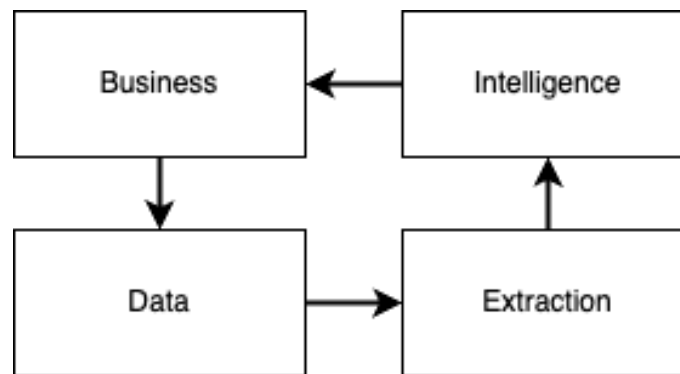


Figure 2.4: Business Intelligence cycle

The Digital Transformation Dossier by IDC and Statista (2021) [18] reports that 90% of companies globally have adopted at least one digital transformation initiative, with a projected global market value of \$3.3 trillion by 2026. The report also highlights that 65% of organizations accelerated their digital strategies in response to the COVID-19 pandemic, with cloud computing expected to grow at an annual rate of 16%. The adoption of artificial intelligence (AI) is projected to reach 50% by 2026. However, the dossier notes that 72% of businesses are concerned about cybersecurity risks, and 60% identify the need to reskill employees as a critical challenge to achieving digital transformation goals.

2.3.1 Predictive Analytics

Predictive analytics, as a key component of BI, transforms historical data into actionable insights, enabling businesses to make informed decisions and optimize strategies in real-time. It involves analyzing historical data to forecast future trends, identify potential risks, and optimize resource allocation, as detailed in the reference [14]. Adapting predictive analytics in Agile environments, characterized by iterative and flexible processes, significantly improves project management. Agile teams, equipped with predictive insights, can anticipate potential issues and make proactive adjustments, aligning more closely with project goals and stakeholder expectations.

In Agile, predictive analytics notably contributes to sprint planning and backlog prioritization. Agile's short cycles of delivering workable software produce valuable quantitative data, which, when analyzed, provides insights into future task requirements. This enhances sprint planning efficiency, allowing teams to allocate efforts accurately and mitigate project delays or failures. Predictive analytics also plays a role in understanding user behaviours and market trends, enabling teams to align product development more closely with user needs and market demands.

2.3.1.1 Impact of Predictive Analytics

This section delves into the impact of predictive analytics in forecasting future trends, optimising operations, and managing risks across various industries [20].

With the goal of integrating AI into agile development, the article [31] presents a "systematic and strategic approach" on how to achieve this. The authors present the following 5-step strategy:

1. **Identify Agility Gaps:** AI is applied in sprint planning, resource allocation, risk management, and progress tracking to enhance decision-making.
2. **Data Collection:** Historical project data, team skillsets, and external sources are gathered and prepared for model training.
3. **AI Technique Selection:** Techniques such as supervised learning for task estimation, unsupervised learning for risk identification, and NLP for analyzing user stories are chosen based on project needs.
4. **Model Training and Integration:** AI models are trained on collected data and integrated into project management tools and communication platforms.
5. **Evaluation:** The AI solution is continuously monitored and improved using metrics like accuracy and resource utilization.

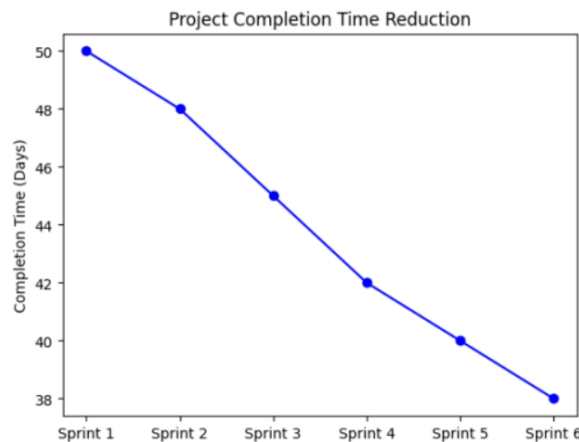


Figure 2.5: Project Completion Time Through Sprints [31]

The values obtained are derived from a comparison of 12 projects that did not utilise AI and 3 projects that incorporated AI for agile project management. The findings revealed a notable enhancement in the precision of estimation and the efficient utilisation of resources following the integration of AI, as shown in figure 2.5. The accuracy of estimation increased by 20% after a single iteration, resulting in more realistic sprint planning and a decrease in schedule overruns. In addition, the implementation of AI-driven resource allocation led to a 15% reduction in team member idle time. This improvement resulted in more efficient task distribution and ultimately increased overall productivity.

2.4 Risk Management

Risk management, defined by the Cambridge English Dictionary, is "the job of deciding what possible financial risks are involved in a planned activity and how best to avoid or deal with them" [26]. A definition more commonly related to the financial industry, such as banks and investment firms, also plays a crucial role in project management.

Given the inherent uncertainties and dynamic nature of technology projects, effective risk management is indispensable in Agile software development. Agile methodologies, characterized by their adaptability and iterative nature, often lack structured risk management approaches, leading to improvised strategies that may not always be effective. This underscores the need for a more systematic approach to risk management within Agile frameworks to prevent project failures [5]. The core challenge in Agile risk management is balancing the methodology's flexibility with the need for a systematic and structured approach. Integrating risk management practices that align with the principles of the Agile Manifesto is crucial, as traditional approaches may prove too cumbersome for Agile's lightweight and fast-paced nature. Consequently, there is a growing demand for tools and practices specifically adapted to Agile's unique dynamics.

Recent research, such as the survey conducted on 54 Agile practitioners [16], highlights significant gaps and challenges in adapting traditional risk management frameworks to Agile environments. The survey finds that while Agile's flexibility and iterative processes often conflict with the rigid structures of traditional risk management, there has been progress in developing more adaptive approaches. Notably, over 80% of participants reported that their companies did not have a designated role or position specifically for risk management. Another relevant figure presented in this survey was that 52% of the respondents do not communicate any risk management. This reliance on informal and ad-hoc risk management procedures can result in missed hazards and project failures. However, the survey acknowledges the rise of real-time risk identification and mitigation using predictive analytics and machine learning. These tools are increasingly being integrated into Agile workflows, emphasizing the need for continuous risk management techniques that become an intrinsic part of daily Agile activities rather than a distinct or sporadic effort.

2.5 AI Integration in Project Tracking

Integrating Generative AI and Machine Learning (ML) into Agile project management is, as in many other areas, fundamentally transforming the field, introducing innovative tools and methodologies that enhance various aspects of project management. These advancements bring significant improvements in efficiency and decision-making but also introduce new complexities and challenges that must be carefully managed.

2.5.1 Machine Learning

ML is a critical subset of artificial intelligence focused on developing algorithms and statistical models that enable computers to perform tasks by identifying patterns and making inferences

from data, rather than following explicit instructions. In project management, ML is used to build analytical models that can predict outcomes, optimize processes, and provide insights based on historical data. [2]

2.5.2 Large Language Models

LLMs, such as Generative Pre-trained Transformer (GPT) models, are specialized forms of ML that are trained on extensive datasets of text to understand, generate, and interact with human language. LLMs can perform a wide range of language-related tasks, including translation, summarization, question-answering, and text generation. These capabilities make LLMs particularly useful in project management for tasks such as automating documentation, generating reports, and enhancing communication between team members [7].

2.5.3 Generative Pre-trained Transformer

GPT models, a subset of LLMs, are specifically designed for text generation. They can produce human-like text across diverse topics and styles, making them useful for generating content such as project updates, meeting summaries, and even predictive insights in project management systems. The continual improvement of GPT models through more extensive training data has significantly increased their accuracy and utility in professional settings [27].

2.5.4 Prompt engineering

A prompt is defined as an "instruction given to an artificial intelligence by a human using natural language rather than computer language"[25]. On the other hand, Prompt engineering is an emerging field that focuses on creating and refining prompts to maximise the performance of large language models, specifically in activities related to natural language processing[15].

Prompt engineering, as previously mentioned, has a key role in the creation of results and utilizing the full power of GPT models. With the rapid improvement of GPT models, many techniques of prompting have been created, such as Zero-shot prompting, which involves giving a task without prior knowledge, or Negative Prompting, where the user tells the model what not to do instead of what to do. For this dissertation, the main two solutions analysed were few-shot prompting and chain-of-thought prompting.

2.5.4.1 Few-Shot Prompting

Few-shot prompting is a method where the model is provided alongside the output format, with some examples of input-output pairs within the prompt itself. An example of this would be the following:

This is awesome: positive
That was horrible: negative
Everything was good:

These examples help the model to better understand the task by demonstrating how it should respond to similar inputs. By showing the model a handful of relevant examples, it can generalize and make accurate predictions on unseen data. This approach has proven effective in tasks like story point estimation, where past examples can guide the model's understanding of new descriptions. [7].

2.5.4.2 Chain of Thought Prompting

Chain of Thought (CoT) prompting encourages the model to break down a complex problem into intermediate steps before arriving at a solution. This technique is especially useful for tasks requiring logical reasoning or multi-step processes.

Q1: (QUESTION)

A1: Because of X then Y thus, the answer is Z

Q2: (QUESTION)

A2:

This method applies the Few-shot technique to aid with CoT prompting, as the model is given previous examples of how to delineate its thought process. This leads to simpler comprehension responses by the model, which enhances the precision of intricate predictions.[30].

Prompting techniques continue to emerge as essential tools for improving the performance and interpretability of LLMs. Techniques like Few-Shot Prompting and CoT exemplify how structured inputs can significantly enhance a model's ability to perform complex tasks. These methods allow for greater precision in task execution. However, the optimal use of these techniques often depends on the specific task and context, requiring continued research to refine and adapt them to new applications. It is not a one-size-fits-all scenario where every problem may be solved with the same technique. The testing of various different models is an essential step to improve the model's output. As the field evolves, the development of more nuanced prompting methods will likely be a key factor in fully grasping the potential of GPT models.

2.5.5 AI and ML Advancements

The role of AI and ML in Agile methodologies extends beyond traditional technological tools, serving as collaborative partners in project management. These technologies automate routine tasks and provide advanced analytics that enhances estimation accuracy, improve risk management, and support more informed decision-making [11]. For instance, ML models can predict project delays by analyzing past project data, while LLMs can automatically generate user stories or refine task descriptions to ensure consistency across the project lifecycle.

2.5.6 Challenges and Limitations in Integrating AI and ML

Despite the significant advancements brought by AI and ML in Agile software development, their integration is not without challenges. A primary issue is the current lack of advanced analytical

capabilities in many Agile project management tools, which still rely heavily on human judgment for critical decision-making tasks [11]. This reliance can limit the effectiveness of AI integration, as the full potential of AI tools is not realized without the proper analytical infrastructure.

Security concerns are also heightened with the introduction of AI systems, especially in the development of safety-critical products. The risk of cyber-attacks, particularly those targeting deep neural networks and LLMs, poses a significant challenge. Additionally, training AI models with appropriate and relevant data remains a persistent hurdle. While there is potential in combining software-related data from various contexts to improve predictions, the vulnerability of AI to adversarial attacks through data manipulation remains a concern [8].

Moreover, the integration of AI in Agile environments requires specialized technical expertise and careful contextual consideration. Challenges such as standardized processes, socio-technical limitations, and the scarcity of skilled AI engineers must be addressed to fully leverage AI's potential in Agile project management. Failure to adequately navigate these challenges could result in suboptimal use of AI technologies and potential setbacks in project outcomes [8].

In conclusion, while the integration of AI and ML into Agile project management offers substantial benefits, including improved efficiency and enhanced decision-making, it also introduces new challenges that must be carefully managed. The successful integration of these technologies requires a balanced approach that considers both the innovative capabilities of AI and the practical realities of Agile project environments.

2.6 User Story Estimation

A User Story is a brief description of a feature from the perspective of the end-user or customer. The primary goal of a User Story is to articulate how a specific feature will provide value to the user. The typical structure for a User Story follows: "As a [who], I want [what], so that [why].", which was first mentioned by Mike Cohn [9]. This format ensures that development focuses on delivering features that meet the business goals.

State-of-the-art user story estimation has had significant development in recent years, mainly due to the increase in the maturity of users of such methods and the new upcoming support of GenAI technologies. In his book, *User Stories Applied for Agile Software Development*, Mike Cohn explains, "User stories are not about writing; they are about conversations" [9]. This emphasizes the importance of communication between stakeholders and development teams, where the user story serves as a starting point for deeper discussions about functionality and design.

2.7 Existing Story Point Prediction Models

This final section aims to examine the existing research and studies that are relevant to the topic of this thesis. The main goal is to identify any similarities and differences between them. By doing this, the intention is to emphasise shared aspects in research methods, results, and approaches,

while also acknowledging any notable differences or areas that need further exploration. In addition, this section offers a comparative analysis to situate the current work within the wider context of the field.

2.7.1 Story Points

Story Points represent a more abstract unit of measure used to estimate the size, complexity, and effort required for implementing a user story in Agile frameworks like Scrum. Unlike time-based estimates, Story Points offer a high-level understanding of the effort involved, focusing on the relative complexity of tasks. This approach helps teams better manage their sprint planning and resource allocation, as it allows for a comparison of effort across different tasks[22].

2.7.1.1 T-shirt Sizing

T-shirt sizing is a popular and intuitive approach for estimating user stories in Agile software development. This technique involves categorizing stories into sizes that reflect their relative complexity or effort required, much like the sizes of T-shirts: XS (Extra Small), S (Small), M (Medium), L (Large), and XL (Extra Large). T-shirt sizing is beneficial for quickly estimating many stories, providing a high-level, relative measure of complexity rather than a precise metric. It facilitates rapid alignment among team members, allowing for quick prioritization and planning in the early stages of a project [22].

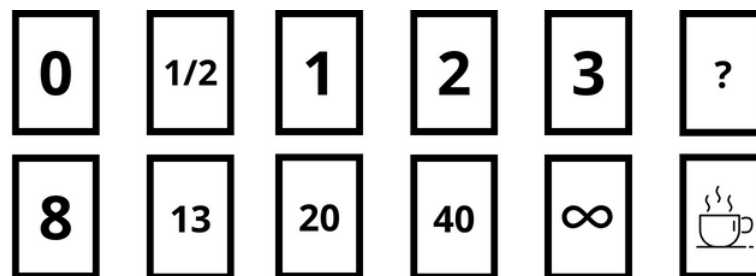


Figure 2.6: Planning Poker Cards

2.7.1.2 Planning Poker

Planning Poker is a method that involves gathering expert opinions and engaging in collaborative discussions to estimate user stories. During this process, team members utilise cards that have pre-assigned story point values to indicate their estimates for a user story. Following a comprehensive discussion facilitated by either the product owner or scrum master, every team member chooses a card that corresponds to their estimation. The estimates are disclosed simultaneously to prevent any influence from early estimates on subsequent ones, thus avoiding anchoring. Developers who provide the highest and lowest estimates offer justifications, which are then followed by additional discussion in order to reach a collective agreement. This iterative process ensures that

all viewpoints are considered, leading to more balanced and accurate estimates. Empirical studies on Planning Poker have demonstrated its effectiveness in improving estimation performance, fostering team involvement, and making the estimation process more enjoyable and engaging, as mentioned in the reference [17]. Planning Poker also addresses common issues observed in unstructured group estimation processes, such as unequal participation and the influence of dominant personalities, by ensuring that all team members contribute to the estimation process. The Planning Poker cards and the estimation process are depicted in Figure 2.6, which visually represents the cards typically used in this methodology.

2.7.1.3 Dot Voting

Dot Voting is a technique employed by agile teams to streamline the process of prioritising or selecting items, with input from all team members. Every participant is given an equal amount of dot stickers to place on the items they consider most significant. Users have the option to distribute their dots among various items or allocate multiple dots to a single item. Priority is given to items with the highest number of dots. This method is often used during sprint retrospectives to prioritise possible enhancements [10].

The article *Study on Agile Story Point Estimation Techniques and Challenges* [23], after a thorough analysis of various story point estimation techniques, concluded that "further work has to carry out to arrive at better effort estimation techniques to overcome the challenges in existing Agile Scrum Story Point estimate." The need for ongoing research and development in the field is evident, as current methods have limitations in accuracy and consistency. This suggests that continuous improvement in estimation techniques is crucial to enhance the effectiveness of Agile project management.

2.7.2 Using Latent Dirichlet Allocation

The authors of [29] introduce a new approach for estimating Story Points (SP) in agile development by analyzing textual features of software issues. Using latent Dirichlet allocation (LDA), which is a type of Bayesian network, to represent issues in topic space and hierarchical clustering, issues are grouped by topic similarity. Estimation models are built for each cluster, and predictions are made based on the closest cluster. Tested on a dataset of 26 open-source projects, the results are comparable to state-of-the-art methods but not significantly better than simpler models.

Relevant conclusions that were achieved from this paper[29] include the discovery that while the LHC-SE model offers comparable performance to state-of-the-art techniques, its additional complexity is often unnecessary compared to simpler, naive estimators like using the median Story Points. The study also recommends additional investigation into advanced text analysis techniques, enhanced data preprocessing, and alternative clustering and distance metrics. It emphasises the necessity for more precise and efficient methodologies in the field of SP estimation.

2.7.3 GPT2SP

In the landscape of Agile software development methodologies, both this dissertation and the work presented in [13] explore the application of artificial intelligence (AI) to enhance Story Point estimation. While both works focus on leveraging AI to improve Agile practices, their approaches and contributions differ in several key aspects.

This dissertation introduces an Intelligent Project Tracking System that utilizes machine learning (ML) to improve the accuracy of Story Point predictions by analyzing historical project data. The system's predictive capabilities are integrated into a broader project management framework, allowing for real-time adjustments to effort estimates as the project evolves. By offering ongoing insights into Story Point predictions alongside project tracking and resource allocation, the system aims to provide a more comprehensive tool for managing the complexities of Agile projects.

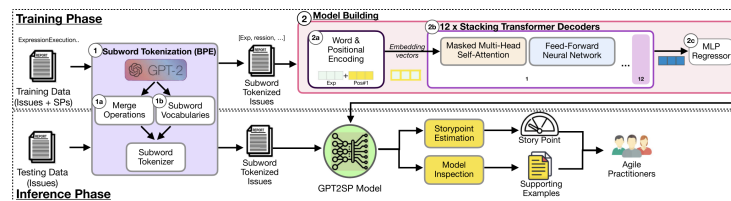


Figure 2.7: Overview architecture of GPT2SP

On the other hand, the GPT2SP model presented by [13] is specifically focused on improving Story Point estimation through the use of GPT-2, a pre-trained language model which is shown in the figure above 2.7. The GPT2SP approach emphasizes the linguistic analysis of user stories to capture their semantic nuances, which enables precise estimation of Story Points. This method addresses some of the limitations inherent in traditional estimation techniques, such as Planning Poker, by providing a more data-driven and interpretable approach to Story Point prediction across various projects.

Although both works share the goal of integrating AI into Agile methodologies, the scope of application differs. This dissertation extends beyond estimation by embedding Story Point predictions within a more expansive framework that supports project tracking and resource management. In contrast, the GPT2SP model focuses exclusively on refining the process of Story Point estimation by improving the interpretability and accuracy of individual task estimations through natural language processing.

2.8 Conclusion

After reviewing the state-of-the-art techniques in story point estimation, it's clear that while progress has been made, significant gaps remain. Current models often struggle to provide consistent and

accurate estimates, particularly in the dynamic context of Agile development. Many existing solutions fail to provide an improvement in traditional techniques, leading to inconsistent predictions and limited real-time adaptability.

The identified shortcomings underscore the necessity for further advancement of the Intelligent Project Tracking system, which will be discussed in the subsequent chapter. The integration of machine learning may enhance prediction accuracy through the utilisation of historical data. Advancing this work could substantially improve story point estimation and project tracking, offering more reliable and adaptive tools for Agile project management.

Chapter 3

Intelligent Project Tracking System

This chapter provides a comprehensive examination of the Intelligent Project Tracking system, developed to integrate predictive analytics with standard task-tracking methodologies in the context of project management. The application is constructed using Electron Forge, which allows for the creation of cross-platform desktop applications with web technologies. The front end is developed using HTML, CSS, and JavaScript, ensuring a responsive and interactive user interface. The backend is also JavaScript-based, facilitating seamless communication with Jira for the retrieval and processing of project-related data.

In **Section 3.1**, the specific requirements of the Intelligent Project Tracking system are outlined, detailing the essential functionalities and user interactions that the system is designed to support. This section covers the integration with Jira, the user interface design, and the core features that enable effective project tracking and management.

Section 3.2 provides an in-depth analysis of the application's architecture, discussing both the front-end and back-end components, the data flow within the system, and the integration of machine learning models. This section also examines the modularity, scalability, and maintainability of the system, highlighting how these architectural principles support the application's ability to adapt to varying project management needs.

3.1 Main Functionalities

The Intelligent Project Tracking system is precisely designed to streamline project management operations. It allows for easy data integration from Jira, effective task tracking, resource allocation, and advanced analytics using machine learning. This section defines the essential components and features, giving a thorough understanding of how each contributes to the application's overall performance.

3.1.1 User Interface and Interaction

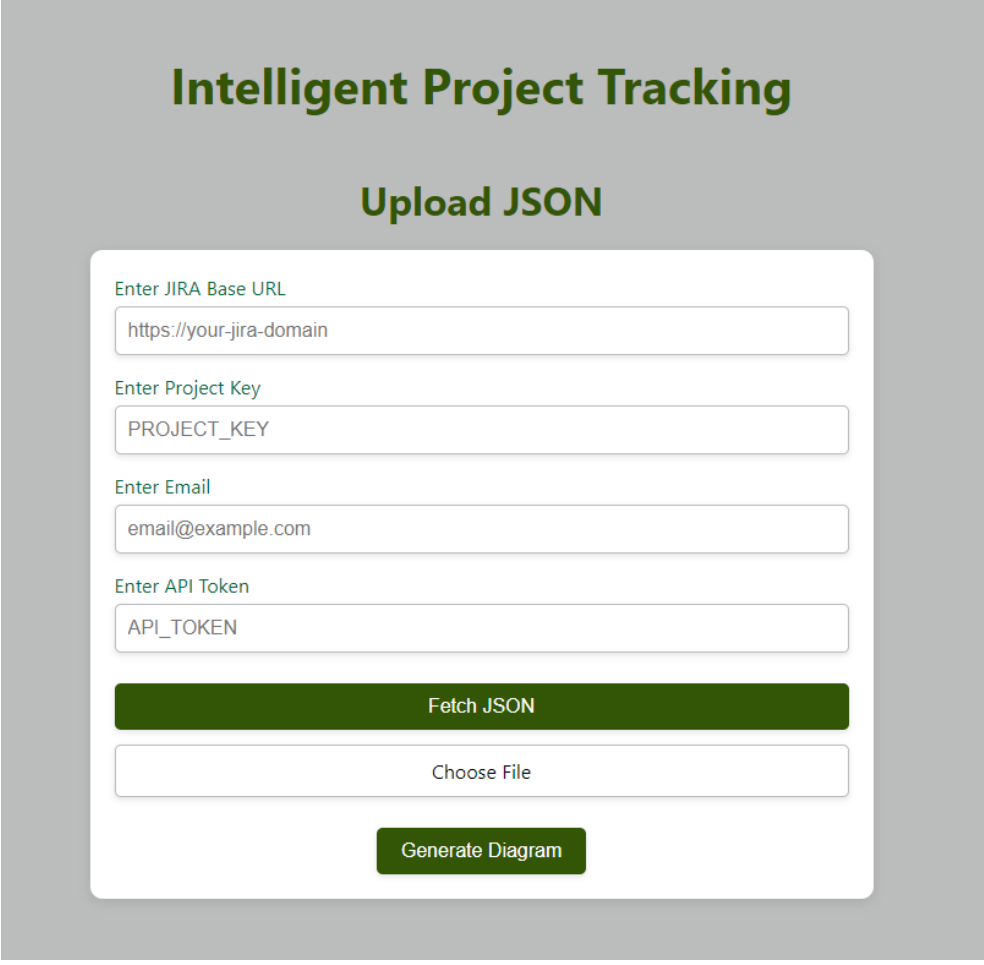
The user interface (UI) of the Intelligent Project Tracking system is designed with a focus on usability and efficiency, ensuring a seamless experience for project managers and team members

alike. The UI is constructed using HTML for structure, CSS for styling, and JavaScript for interactivity, creating a responsive and dynamic environment that adapts to the user's needs.

3.1.1.1 Jira Integration Form

A pivotal feature of the UI is the Jira integration form 3.1, which allows users to establish a connection between the application and their Jira instance. The form is composed of several fields that are essential for retrieving project data:

- **Jira Base URL:** This input field requires the user to specify the base URL of their Jira domain. This URL acts as the endpoint for all subsequent API calls, serving as the primary access point to the Jira server.
- **Project Key:** Users must input the unique project key associated with the Jira project they wish to track. This key is used to filter the project-specific data that is retrieved from Jira, ensuring that the application only processes relevant information.
- **Email:** The user's email address is required for authentication purposes. This input is used in conjunction with the API token to authenticate the user's requests to the Jira API securely.
- **API Token:** The API token field is where users input their Jira API token. This token, which is encoded to protect the user's credentials, is essential for authenticating API requests and ensuring secure communication with the Jira server.

The image shows a web form titled "Intelligent Project Tracking" with a subtitle "Upload JSON". The form is set against a light gray background. It contains four input fields, each with a label above it: "Enter JIRA Base URL" (with placeholder "https://your-jira-domain"), "Enter Project Key" (with placeholder "PROJECT_KEY"), "Enter Email" (with placeholder "email@example.com"), and "Enter API Token" (with placeholder "API_TOKEN"). Below these fields are three buttons: a green "Fetch JSON" button, a white "Choose File" button, and a green "Generate Diagram" button.

Intelligent Project Tracking

Upload JSON

Enter JIRA Base URL
https://your-jira-domain

Enter Project Key
PROJECT_KEY

Enter Email
email@example.com

Enter API Token
API_TOKEN

Fetch JSON

Choose File

Generate Diagram

Figure 3.1: Intelligent Project Tracking Start Page

3.1.1.2 Data Retrieval from Jira

Upon completing the form, users can trigger the data retrieval process by clicking the *Fetch JSON* button. This action initiates an asynchronous JavaScript function that sends a request to the Jira API using the provided credentials and project key. The API responds with a JSON object containing the project issues, which is then parsed and stored in the application's session storage. This data is used to generate visualizations and support further analysis within the application.

3.1.1.3 File Upload Functionality

In addition to direct integration with Jira, the application offers an alternative method for data input through a file upload feature. Users can upload a JSON file containing project data, which the application then reads and processes. The content of the file is stored in session storage, allowing users to generate visualisations and perform analyses without requiring a live connection to Jira.

3.1.2 Core Functionalities

The Intelligent Project Tracking system is equipped with a range of functionalities designed to enhance project management. After acquiring the necessary project data, either through Jira API integration or manual file upload, the application provides users with a variety of analytical tools. These tools include visualisations and predictive analytics that are crucial for evaluating project performance, tracking progress, and making informed decisions.

3.1.2.1 Completion Rate Visualisation

The **Completion Rate** feature offers a visual representation of the project's progress by comparing the number of completed tasks to those that remain unfinished. As depicted in the screenshot (Figure 3.2), this metric is displayed as a pie chart, providing project managers with a quick and clear overview of the current state of the project. The Completion Rate visualization is crucial for monitoring overall project advancement and ensuring that milestones and deadlines are met effectively.

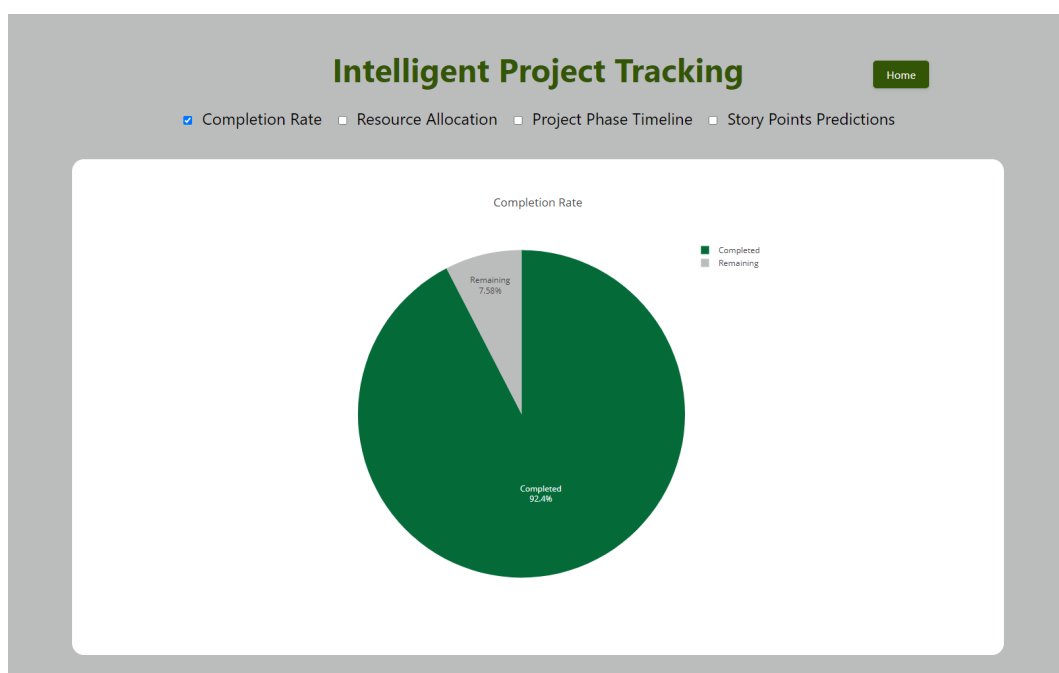


Figure 3.2: Intelligent Project Tracking Completion Rate

3.1.2.2 Resource Allocation Chart

The *Resource Allocation* chart provides a clear visualization of how tasks are distributed among developers. The bar chart, shown in Figure 3.3, displays the names of developers on the x-axis and the number of issues—both current and completed—associated with each developer on the y-axis. This visualization helps project managers identify imbalances in task distribution, allowing them to make informed adjustments to ensure a more even workload.

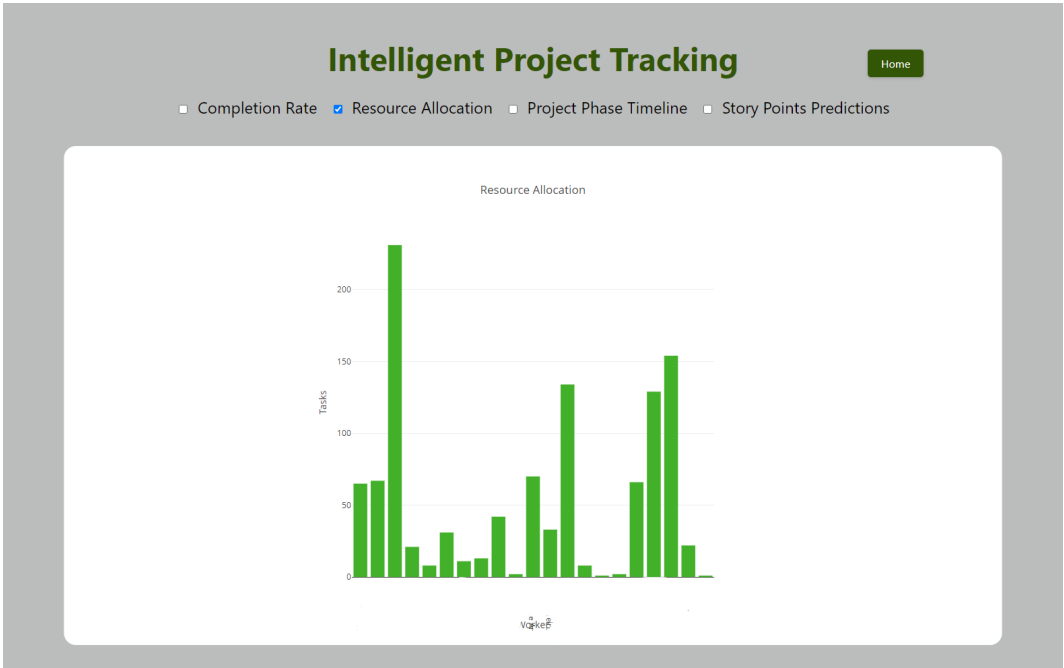


Figure 3.3: Intelligent Project Tracking Resource Allocation

3.1.2.3 Project Phase Timeline

The **Project Phase Timeline** feature provides a chronological overview of the project’s progress, integrating both data-driven predictions and actual performance metrics. As illustrated in Figure 3.4, this timeline is particularly valuable for identifying potential delays and forecasting future milestones. The timeline incorporates predictions from Machine Learning (ML) and Large Language Models (LLM), which are further refined using statistical measures such as the Interquartile Range (IQR). By analyzing the spread of data through the IQR, the system identifies outliers and offers a more robust prediction model, enabling project managers to anticipate challenges and adjust their strategies proactively.

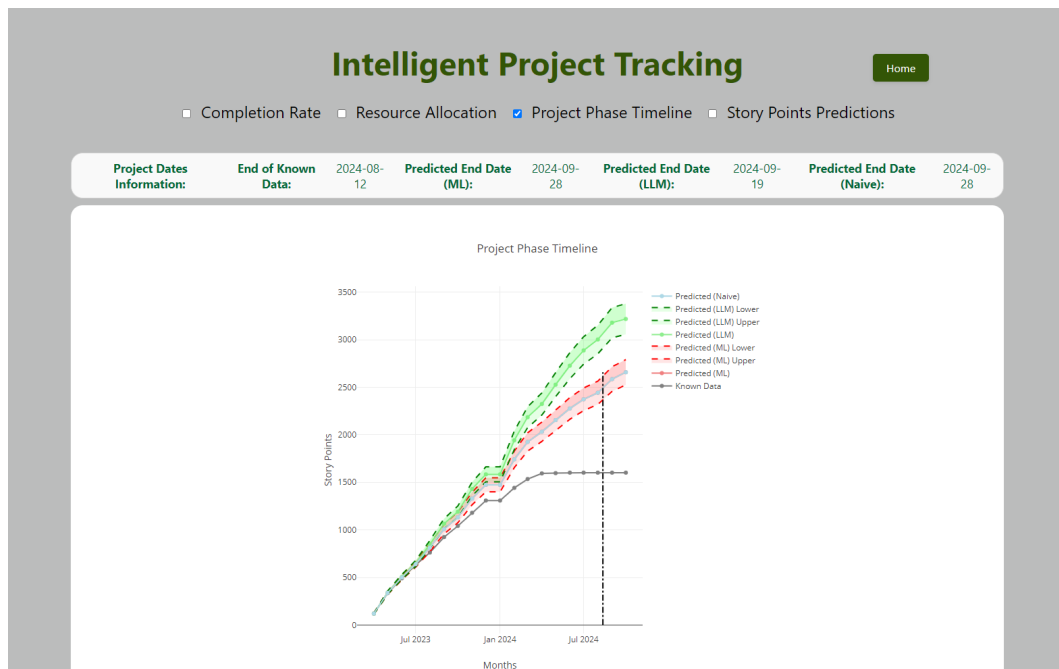


Figure 3.4: Intelligent Project Tracking Project Phase Timeline

3.1.2.4 Story Points Prediction

The Story Points Prediction feature utilises advanced machine learning methods to assess task complexity quantified in story points. The application evaluates predictions produced by various models, including ML, LLM, and a Naive approach, as illustrated in Figure 3.4. The graph displays the outputs of various prediction models, accompanied by two additional lines that illustrate a quartile range above and below the predicted values, thereby visualising the associated uncertainty.

The x-axis of the graph denotes time in months, with each completed issue assigned a story point aggregated into its corresponding completion month. The y-axis is the Total number of story points. The bottom line, labelled as *Known Data*, indicates completed issues that had story points assigned by the project team.

The red line represents the machine learning (ML) model, where each point for a given month is the cumulative sum of story points from completed issues. For issues with user-assigned story points, the known values are used, while for completed issues without assigned story points, the ML model predicts the values. This results in a progressive line combining both known and predicted story points. Similarly, the green line represents the LLM model, where known story points are combined with predictions made by the language model for issues without assigned values. The blue line, representing the Naive approach, assigns a fixed story point value of 3 to all issues without an assigned value. Each line is cumulative, adding the values from previous months, and a vertical dashed line marks the current date, distinguishing between completed and future data.

The upper table, labelled *Project Dates Information*, in 3.4, shows predicted end dates for each model by estimating the time needed to complete the remaining story points. The dates are derived from the average monthly completion of story points, with the projected values adjusted for each model accordingly.

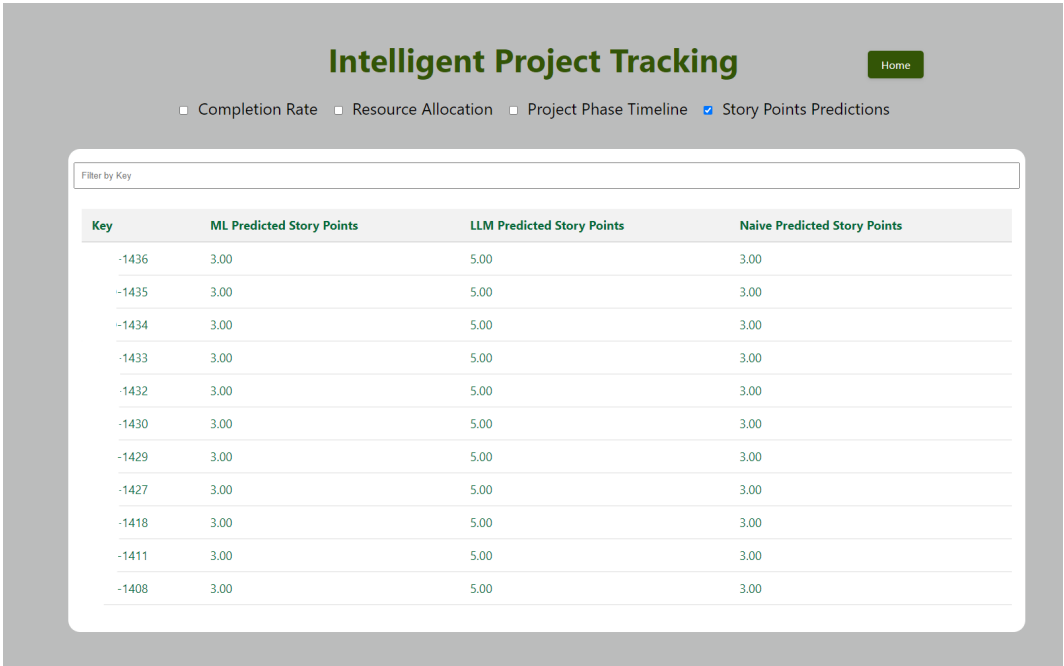


Figure 3.5: Intelligent Project Tracking Story Point Prediction Table

This last diagram, labelled *Story Points Predictions* 3.5, is a search table that allows user to filter their issues by the Key, which is a value JIRA associates with each issue. These are the same values that are used in figure 3.5, just a different visualization that allows project managers who need to understand the expected effort required for each specific task.

3.2 Architecture

The Intelligent Project Tracking system is designed for modularity, scalability, and maintainability, ensuring it can adapt and evolve over time. The architecture efficiently manages complex project data, integrates with services like Jira, and utilizes machine learning for predictive analytics. Each component is designed to work seamlessly with others, allowing the system to remain flexible and meet new requirements.

This section provides an overview of the architecture, covering both front-end and back-end components and the data flow between them. The modular design allows each component to operate independently while interacting with others as needed, simplifying development and maintenance. The architecture’s scalability enables the application to handle growing data and user demands with minimal infrastructure changes, making it suitable for organizations of varying sizes.

3.2.1 Detailed Architectural Components

The back-end, powered by Node.js, manages data processing, Jira integration, and machine learning tasks. It retrieves project data from Jira via API calls, processes it, and stores it in session storage for use across the application. TensorFlow.js integration allows for real-time machine learning tasks, such as story point prediction and natural language processing using OpenAI's API, enhancing the system's predictive capabilities.

The front end, built with HTML, CSS, and JavaScript, offers a user-friendly interface for project managers and team members. It enables data input, displays processing results, and machine learning predictions. The front end communicates smoothly with the back end, ensuring real-time updates and interactive data experiences. This separation of front-end and back-end concerns is vital for maintaining modularity and ease of maintenance.

Data flow within the system is carefully managed to ensure efficiency and reliability. Data moves from the user interface to the back-end for processing, then is either stored locally or used for real-time predictions. Local and session storage support offline functionality and intermittent connectivity, while robust error handling ensures data integrity throughout the process.

The architecture supports maintainability through well-defined interfaces and modular components, allowing developers to update or expand parts of the system without affecting the whole. The machine learning models are separated from core application logic, enabling independent updates as better models or new data become available.

Figure 3.6 illustrates the architecture, showing key components, data flow, and interactions with external services like Jira and OpenAI.

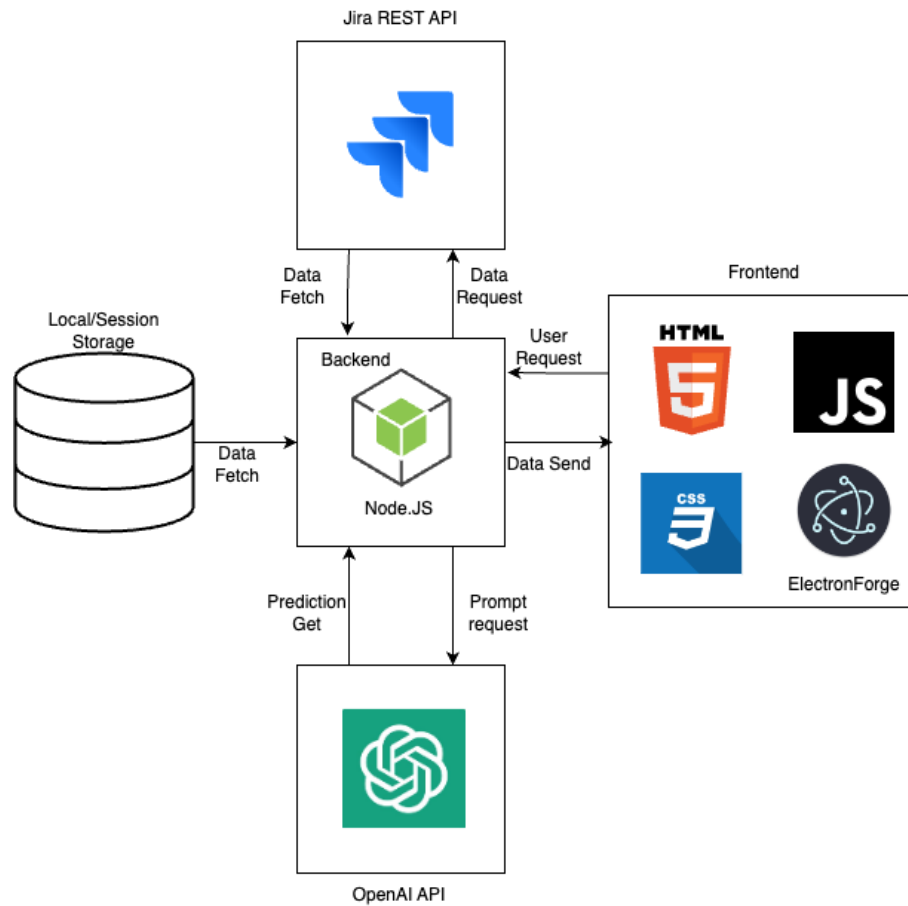


Figure 3.6: Intelligent Project Tracking Architecture

3.2.2 Technologies Used

The Intelligent Project Tracking system integrates machine learning, natural language processing, and real-time project management to meet project objectives. For efficient data processing, user-friendly interface design, scalability, and cross-platform capability, the following technologies were chosen.

3.2.2.1 ElectronForge

ElectronForge¹ is a framework designed to create desktop applications using web technologies such as JavaScript, HTML, and CSS. It is particularly useful for building cross-platform applications that need to function on Windows, macOS, and Linux. In the Intelligent Project Tracking system, ElectronForge is used to package and deploy the dashboard interface, ensuring consistency across different operating systems. The use of ElectronForge simplifies the development process, allowing for faster iterations and updates while ensuring that the application remains responsive and interactive. Its support for integrating native JavaScript libraries further enhances the flexibility of the application [12].

3.2.2.2 Node.js

Node.js² is utilized as the backend platform for the Intelligent Project Tracking system. It facilitates efficient communication between the frontend and the backend, as well as between the application and external services such as the Jira API. Node.js is known for its event-driven, non-blocking I/O model, making it ideal for handling the asynchronous operations required by the system. By using Node.js, the application can efficiently process user requests, fetch data from project management tools, and send updates to the frontend in real-time. Its extensive ecosystem of libraries, including React for UI development, supports rapid prototyping and integration with other essential technologies.

3.2.2.3 TensorFlow

TensorFlow³ plays a crucial role in the AI-driven components of the project. As an open-source machine learning framework, TensorFlow is employed to build and deploy the predictive models that estimate project timelines, assess risks, and provide real-time insights based on historical project data. TensorFlow's versatility allows the system to handle large datasets and perform complex calculations, making it suitable for Agile project management environments where accurate data-driven insights are critical. The machine learning models developed with TensorFlow enable project managers to make informed decisions about resource allocation, task prioritization, and risk management.

3.2.2.4 OpenAI API

The integration of the OpenAI API⁴ enhances the system's capacity for processing natural language data, particularly within the domain of project management tasks. By incorporating advanced natural language processing (NLP) functionalities, the Intelligent Project Tracking system is able to obtain SP predictions with greater precision. This integration ensures that the system remains dynamic, continuously improving its predictive accuracy and adaptability as new data is introduced. The continuous updates on their models allow for better results without the need to change the prompts. As some issues may have incomplete descriptions, this feature became more relevant.

3.2.2.5 Jira

Jira⁵ is the primary project management tool integrated into the Intelligent Project Tracking system. As a widely used platform for tracking issues, managing tasks, and monitoring project

progress, Jira serves as the main data source for the system. Through its REST API functionality, the application fetches real-time issues, which are then processed by the backend to generate the visualisations.

3.2.3 Data

The data used for the analysis of the GPT model and ML were derived from two ongoing projects at Deloitte, referred to from now on as P1 and P2. As shown in Table 3.1, P1 consisted of 350 total issues, of which 309 had been completed, leaving 41 remaining issues. Among the remaining issues, only 12 had assigned Story Points (SP), whereas 198 of the completed issues had SPs, accounting for approximately 64% of completed issues with SP.

In contrast, P2, a larger project, had 1298 total issues, with 1193 completed and 105 remaining. Out of the remaining issues, 10 had SPs, while 872 of the completed issues had SPs, which represents around 73% of the total completed issues.

This distribution of issues and Story Points shows that while both projects have a significant number of completed issues with SPs, a much smaller proportion of the remaining issues in both P1 and P2 have SPs. This suggests that the training data for the ML and GPT models primarily relies on the completed issues, limiting the potential influence of the remaining issues in making accurate future predictions. Additionally, the relatively higher percentage of completed issues with SPs in P2 (73%) compared to P1 (64%) might offer a more robust dataset for training the models in the larger project.

Threats to Validity: The limited sample size and project-specific datasets may restrict the generalizability of these findings. Both P1 and P2 represent unique contexts, and variations in story point assignments across teams introduce noise into the data.

Project	Total Issues	Completed Issues	Remaining w/ SP	Completed w/ SP
P1	350	309	12	198
P2	1298	1193	10	872

Table 3.1: Summary of Issues and Story Points (SP) in Projects P1 and P2

3.2.4 AI Integration

The predictive capabilities of the Intelligent Project Tracking system are driven by a combination of advanced machine learning models and natural language processing (NLP) techniques powered by a Large Language Model (LLM). This section provides an in-depth explanation of the processes involved in both the ML and LLM workflows, detailing how they contribute to accurate story point predictions and overall project management.

¹<https://www.electronforge.io/>

²<https://nodejs.org/>

³<https://www.tensorflow.org/>

⁴<https://openai.com/>

⁵<https://www.atlassian.com/software/jira>

3.2.4.1 Data Preprocessing

The first crucial step in the machine learning pipeline is data preprocessing. This process involves converting raw textual data, such as task descriptions, into a structured format that can be used for training predictive models. The application uses a **TF-IDF Vectorizer** to transform task descriptions into numerical vectors, which reflect the importance of each word within the corpus. This transformation is essential for enabling the machine learning model to learn meaningful patterns from the data.

- **Term Frequency-Inverse Document Frequency:** The TF-IDF Vectorizer calculates term frequency (TF) and inverse document frequency (IDF) for each word in the task descriptions. TF quantifies the frequency of a word within a document, whereas IDF assesses the importance of the word throughout the entire corpus. The product of these values, TF-IDF, provides a weight that represents the word's relevance within a specific task description, enabling the model to focus on the most informative features.

3.2.4.2 Model Training and Prediction

Once the data has been preprocessed, the machine learning model, known as the **StoryPoint Predictor**, is trained on these vectorized task descriptions. The model is designed to learn from historical data, identifying patterns that correlate with task complexity, which are then used to predict story points for new tasks.

- **Training Process:** The StoryPointPredictor is trained using a supervised learning approach. During training, the model is provided with pairs of task descriptions and their corresponding story points. The model adjusts its internal parameters iteratively, with the goal of minimizing the error between its predictions and the actual story points. This is achieved using an optimizer that tweaks the weights of the neural network to improve prediction accuracy over time.
- **Activation Function - ReLU:** Within the neural network, the Rectified Linear Unit (ReLU) activation function is employed. ReLU introduces non-linearity into the model, allowing it to learn complex patterns within the data. By converting negative values to zero and retaining positive values, ReLU helps speed up the training process and mitigate issues like vanishing gradients, which are common in deep learning models.
- **Prediction Process:** After the model is trained, it is applied to new, unseen task descriptions. The StoryPointPredictor generates predictions that estimate the story points required for each task, based on the learned patterns. These predictions are crucial for project managers who need to allocate resources efficiently and plan effectively.

3.2.4.3 Large Language Model (LLM) Prompts and Logic

In addition to the machine learning model, the application integrates a **Large Language Model (LLM)** to enhance prediction accuracy, especially for tasks with complex descriptions where traditional models might struggle. The LLM uses natural language processing (NLP) techniques to interpret and predict story points based on task descriptions.

LLM Prompt Structure: The prompts used in the LLM workflow are carefully crafted to guide the model in generating accurate predictions. The structure and logic of the prompts are as follows:

- **Initial Prompt:** The LLM is initialized with a comprehensive instruction set. For example:

"You are an AI assistant tasked with predicting story points for software development tasks. Use the provided examples to guide your predictions. If no clear pattern is discernible, make an educated guess based on the similarity of descriptions. The values should be similar to the examples given. DO NOT SKIP ANY VALUES. Predict all the story points. Check if you have the same number of predictions as the number of descriptions."

This prompt instructs the LLM to utilize both the training data and its understanding of task descriptions to generate story point predictions. It emphasizes the importance of consistency and completeness in the projections, ensuring that no tasks are left unpredicted.

- **Prompt Logic and Token Management:** To manage the LLM's token limit effectively, the prompt dynamically adjusts based on the length of the task descriptions and examples provided. The initial prompt is followed by specific examples of tasks and their corresponding story points, which serve as a reference for the LLM when predicting new tasks. The examples are carefully selected to balance the inclusion of enough context while staying within the token limits.
- **Batch Processing:** The LLM processes task descriptions in batches to optimize resource use and ensure that the predictions remain within computational constraints. For each batch of task descriptions, the LLM generates predictions by comparing the new tasks with the examples provided, following the logic embedded in the prompt.
- **Prompt Adjustment:** If the generated prompt exceeds the token limit, the system dynamically reduces the number of examples or task descriptions in the batch. This adjustment ensures that the LLM operates within its token limits while still providing accurate predictions. The flexibility of the prompt structure is crucial for handling a wide range of input sizes and complexities.

The integration of the LLM with the machine learning model allows the application to harness the strengths of both approaches. While the ML model excels at learning from structured

data and historical patterns, the LLM offers advanced natural language understanding, making it particularly effective for complex or nuanced task descriptions.

3.2.5 Results

3.2.5.1 Evaluation Metrics

To evaluate the effectiveness of both the machine learning (ML) model and the large language model (LLM), several key performance metrics were calculated for two projects, referred to as Project 1 (P1) and Project 2 (P2). These metrics provide insights into how accurately each model predicts story points and their potential utility in real-world project management scenarios.

- **Mean Absolute Error (MAE):** This metric measures the average magnitude of errors between the predicted and actual story points, providing a straightforward gauge of prediction accuracy.
- **Mean Squared Error (MSE):** This metric calculates the average of the squared differences between predicted and actual values, with larger errors having a more significant impact on the final score.
- **Root Mean Squared Error (RMSE):** As the square root of MSE, RMSE offers an error metric in the same units as the original data, making it easier to interpret the overall model performance.
- **R-Squared (R²):** This metric indicates how well the model’s predictions align with the actual data, with values closer to 1 denoting a better fit.

Metric	LLM	ML	Naive
MAE	1.000	0.846	0.83
MSE	2.750	1.923	1.83
RMSE	1.658	1.387	1.35
R ²	0.419	0.257	-0.015

Table 3.2: Performance Metrics for LLM, ML, and Naive Approaches in Project 1 (P1)

Metric	LLM	ML	Naive
MAE	1.063	0.833	1.33
MSE	2.188	1.833	3.00
RMSE	1.479	1.354	1.73
R ²	0.571	0.015	0.00

Table 3.3: Performance Metrics for LLM, ML, and Naive Models in Project 2 (P2)

3.2.5.2 Analysis

The performance metrics in Tables 3.2 and 3.3 offer a comparative analysis of the LLM and ML models in the two separate projects, P1 and P2. For the ML model, an 80-20 training-test split was applied using the completed issues with SP. The training was done using the model in 3.2.4.2. In contrast, for both the Naive and LLM methods, the same 20% of issues were used for testing. In the case of the LLM, the prompt outlined in section 3.2.4.3 was employed, along with three randomly selected examples from the training data. These examples helped the GPT model better understand the prediction style while maintaining contextual relevance.

The machine learning model in Project 1 (P1) shows slightly improved accuracy, as evidenced by lower MAE and RMSE values. This indicates that its predictions tend to be closer to the actual values when compared to the LLM. In addition, the ML model demonstrates superior performance in handling larger errors, as evidenced by its lower Mean Squared Error (MSE). Nevertheless, the R^2 value for the ML model indicates that it fits the data reasonably well. However, the LLM, with a higher R^2 of 0.419, more accurately captures the variability in the data. One possible reason for this could be the LLM's ability to analyse at least three well-organised user stories or issues per prompt, which allows it to gather more information and make more accurate predictions.

In P2 a consistent trend is observed, with the ML model once again demonstrating superior performance in terms of MAE and RMSE. On the other hand, the LLM demonstrates a higher R^2 value of 0.571, suggesting a more robust capability to capture the underlying patterns and variability in the data for this project. The R^2 value of the ML model in P2, while positive, is relatively low (0.015), indicating a less robust fit compared to the LLM.

The ML models in both projects have demonstrated slightly better average accuracy in terms of MAE and RMSE, making them more suitable for precise story point predictions. However, the LLMs show stronger capabilities in capturing data variability, as reflected in higher R^2 values. This suggests that LLMs may be more effective for identifying broader patterns and trends, whereas ML models excel in specific, precise estimations.

When compared to the results from the GPT2SP model in [13], both studies reveal that LLMs provide better generalization, though not consistently outperforming simpler models like the Naive approach. In fact, the Naive method performs reasonably well in terms of MAE and MSE, raising the question of whether the complexity of ML and LM models is always necessary. For certain datasets, especially when variability is low, the Naive method might suffice.

It's important to acknowledge that story point estimations are inherently influenced by human input, introducing potential human error and subjectivity into the training data. This variability, based on how teams define story points, impacts model predictions and may limit their accuracy across different projects.

In conclusion, while both ML and LLM models show promise, the results suggest that simpler models, like the Naive approach, can sometimes perform adequately without added complexity. Continued research into refining model architectures and improving the consistency of training data is required to further enhance the prediction of Story Points.

3.2.6 Workflow Overview

The Intelligent Project Tracking system follows a structured workflow designed to optimize user efficiency and provide continuous insights into project progress. This workflow, illustrated in Figure 3.7, is composed of several key stages:

1. **Data Acquisition:** The workflow begins with acquiring project data, either through direct integration with Jira using user credentials or by uploading a JSON file containing project information. This data forms the foundation for all subsequent analysis and visualization.
2. **Data Processing and Storage:** Once acquired, the data is validated, sanitized, and stored in session storage for immediate use. If necessary, the data is also stored locally, enabling quick access in future sessions without the need for reprocessing.
3. **Metric Selection and Visualization:** Users select from various metrics, such as Completion Rate or Resource Allocation. The selected metrics are visualized in real time through dynamically generated charts and graphs, providing a clear understanding of the project's current status.
4. **Predictive Analysis:** The application employs integrated machine learning models and natural language processing (NLP) via the OpenAI API to predict story points for tasks lacking this information. These predictions are based on historical data and are stored locally for reuse in subsequent interactions.
5. **User Interaction and Navigation:** Users can interact with visualizations by filtering and sorting data. After initial processing, results are stored locally to facilitate faster loading in future sessions. Users also have the option to return to the home screen to input new credentials or upload different data.
6. **Ongoing Monitoring and Feedback:** The workflow is iterative, allowing users to continuously revisit the application to monitor progress, adjust project plans, and refine predictions based on the latest data. This iterative process ensures that project managers remain informed and can respond swiftly to emerging issues, keeping the project on track.

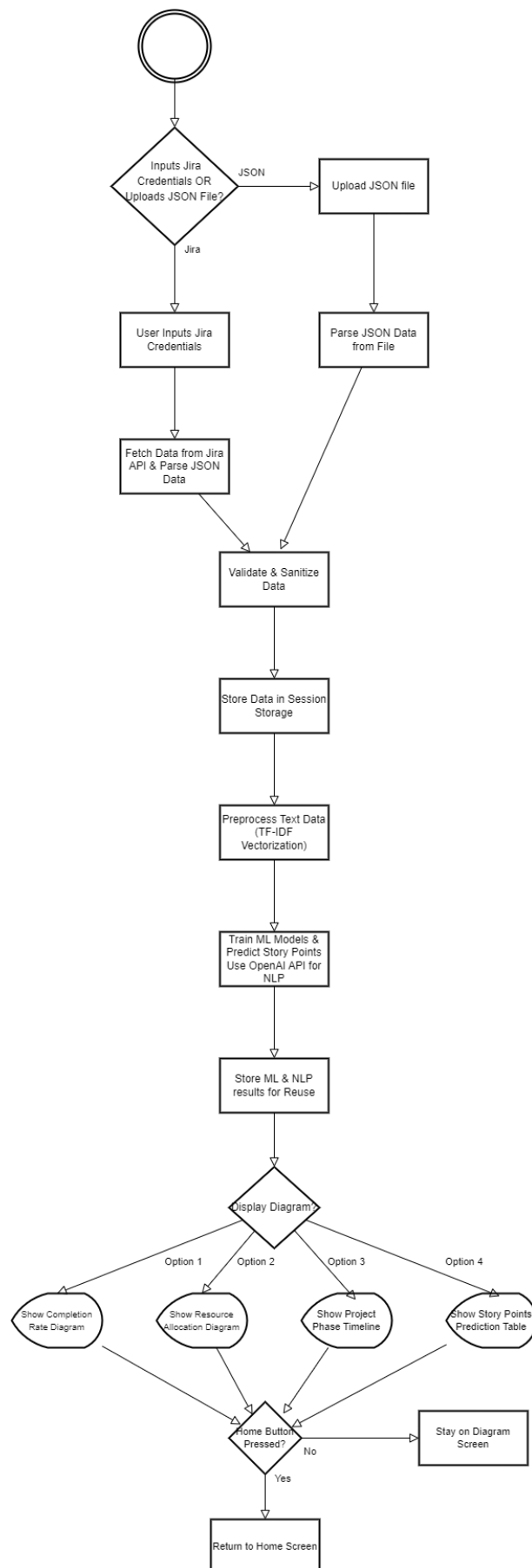


Figure 3.7: Intelligent Project Tracking Workflow

Chapter 4

Validation

Validating a framework is an essential phase to verify its efficiency, user-friendliness, and overall applicability. This section outlines the rigorous validation process undertaken for the proposed Intelligent Project Tracking System. To ensure comprehensive testing, a two-pronged approach was employed.

Initially, an experimental validation was performed to assess the applications's functionality and performance. This stage helped identify any potential shortcomings, which were addressed to enhance the system's resilience and reliability.

Subsequently, a real-world case study was conducted to assess the system's practical application. In collaboration with a set of project managers and developers, feedback was collected through a questionnaire, offering valuable perspectives on how the system could perform in real-life project management scenarios.

4.1 Experimental Validation

To evaluate the robustness and adaptability of the Intelligent Project Tracking System, a series of tests were carried out using *JSON* files as the initial data input. These tests were then expanded to incorporate direct communication with the *RestAPI* of JIRA, allowing for the dynamic retrieval of project data. This approach aimed to assess the system's ability to handle both static data from files and live data from an external system, ensuring flexibility in obtaining and processing project information. Key features, such as the conclusion day estimation, were developed iteratively as they emerged during testing, improving the system's overall functionality.

4.1.1 Iterative Feedback and System Enhancements

During its early stage, the system used a bar graph to display project data and show the progress of the project. Upon receiving feedback from the supervisor, it became clear that a line graph would offer a more concise representation of the data trends over time. This modification enhanced the system's capacity to present project dynamics in a more efficient manner.

In addition, earlier versions of the system mainly concentrated on extracting data from JSON files. After further testing, it was found that using JIRA's RestAPI for direct communication would be more efficient in tracking projects in real time. This change enabled the system to better adjust to a constantly changing environment. It allowed for the continuous retrieval and updating of project information, resulting in a more precise and current overview of project progress.

Throughout the testing process, feedback highlighted the importance of having a user-friendly interface, particularly for individuals who are not familiar with technical procedures. Consequently, improvements were made to enhance usability, making interactions with the system easier and streamlining the process of retrieving and analysing data. Through a series of iterative adjustments, the system's performance was greatly enhanced, ensuring it effectively met the practical needs of project managers and other stakeholders.

By tackling these important usability and functionality issues, the system was able to better meet the needs of its users, resulting in improved practicality and effectiveness in real-world project management situations.

4.1.2 Testing with JSON Files

In the initial phase of testing, the Intelligent Project Tracking System was evaluated using static JSON files as input data. The goal was to verify the system's ability to correctly process, map, and analyze data from a variety of predefined JSON structures. This phase was critical in ensuring that the system could handle diverse input formats and maintain accuracy in its outputs.

The testing involved the following steps:

1. **Initial Data Input:** Several JSON files were used to simulate data from various project management tools. These files contained a mix of standard and complex fields, including nested structures. The system's ability to extract relevant information and organize it into a consistent output structure was closely observed.
2. **Feedback and Adjustments:** During the testing, users highlighted areas where the system encountered difficulties, such as handling missing or incomplete fields. These challenges were addressed by refining the parsing mechanisms to better handle null values and inconsistencies in the data.
3. **Improved Visualization:** Initially, project progress was visualized using bar charts. However, after feedback from supervisors, the bar charts were replaced with line graphs, as the latter provided a clearer and more continuous representation of project dynamics.

This phase of testing validated the system's capability to handle static data from JSON files, setting a strong foundation for more dynamic data integration in the subsequent phase.

4.1.3 Testing with JIRA REST API

Following the successful testing with JSON files, we extended the system's capabilities by integrating JIRA's RestAPI to enable real-time data retrieval from live project management environments. The primary objective was to ensure that the system could dynamically extract and process live data while maintaining the flexibility and accuracy demonstrated with static files.

The testing with JIRA involved several key improvements here presented:

1. **Initial Data Mapping:** Live project data was retrieved from JIRA using the RestAPI, and the system was set up to handle potential discrepancies such as null values or missing fields through a background pipeline. This ensured a smooth flow of data despite possible inconsistencies in real-time updates as the fetch of files.
2. **User Feedback and Iteration:** Through continuous testing and feedback, users indicated that manual field mapping, which was required in the earlier phase, was too complex and time-consuming. This led to the need for automation to make the system more user-friendly, particularly for non-technical users.
3. **Automated Mapping Enhancements:** To simplify the process, an automated mapping feature was introduced. This feature automatically identified and matched fields between JIRA and the system's internal structure when fields had similar names, significantly reducing manual intervention and improving efficiency.

By integrating JIRA's RestAPI and automating the data mapping process, the system became more adaptable to real-time environments, offering enhanced usability and more effective project tracking. This phase demonstrated the system's ability to handle live data while minimizing errors, ultimately supporting better decision-making through up-to-date project insights.

4.1.4 User-backed Enhancements

During a meeting with the supervisors, the idea of an end-date prediction feature was suggested. This feature would estimate the project's completion based on the remaining issues and their predicted story points, giving project managers a clearer view of the timeline. The following steps were taken to implement this enhancement:

- **Feature Suggestion:** The need for an end-date prediction based on remaining tasks and story points was raised during a supervisor meeting.
- **Data Collection:** The system was updated to track remaining issues and their associated story points, which would form the basis for the end-date calculation.
- **Visualization Addition:** A visual representation of the predicted end date was added to the dashboard, allowing users to see an estimated completion date based on current uncompleted issues.

- **Testing and Feedback:** Initial testing was performed, and feedback was incorporated to ensure the feature provided clear and useful information to users.

This visualization feature now provides project managers with a quick and intuitive way to monitor project timelines based on remaining work.

4.2 Use Case

In order to assess the viability and effectiveness of the Intelligent Project Tracking application, we carried out a user validation using real-world data and participants. Our study specifically targeted project managers and developers from Deloitte, who played a crucial role in the evaluation process. This evaluation aimed to assess the system's efficiency in tracking project progress and visualising key metrics, ensuring that it met the needs of both technical and managerial users. For this validation, we enlisted the help of seven project managers and 3 developers to take part in the testing phase.

Participants were given a thorough introduction to the system's main features, including its ability to merge data from JIRA and predict project completion dates. This introduction was important to make sure all users understood how the system worked and how it could improve project oversight and decision-making.

4.2.1 Background questionnaire

The purpose of the background questionnaire was to gather initial information on the participants, providing an understanding of their demographic and professional characteristics. This information was crucial for providing context to their responses and gaining a comprehensive grasp of the varied participant pool.

Participants were asked about their current position, as illustrated in 4.1, how frequently they use project management tools (Figure 4.2), and their years of experience in project management (Figure 4.3). The questions offered helpful insights into the participants' backgrounds and their knowledge of project management systems, which helped provide context for their feedback on the system's usability and functionality.

4.2.2 Task Execution

Participants were provided with the application and given explicit instructions on how to engage with the system by following a sequence of actions. The tasks involved in this process encompassed inputting project data, producing visual representations such as line graphs, and examining displays that showed resource allocation and task accomplishment. The entire process was supposed to be finished within a 10-minute duration. Participants were instructed to complete the activities autonomously in order to obtain an impartial evaluation of the system's usability and functionality.

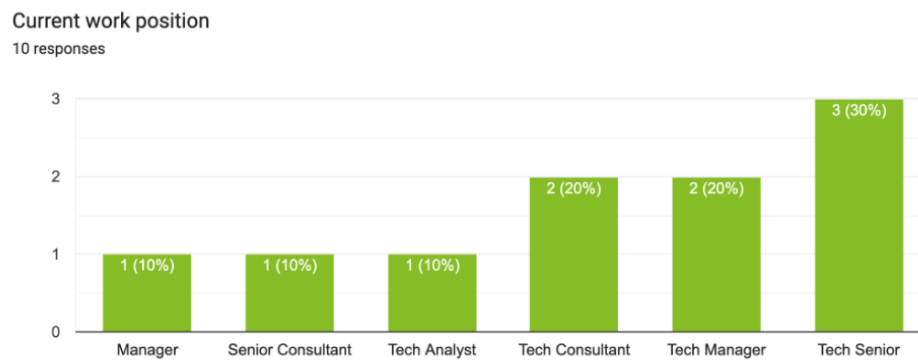


Figure 4.1: Current Position

How often do you use project management tools in your daily work?

10 responses

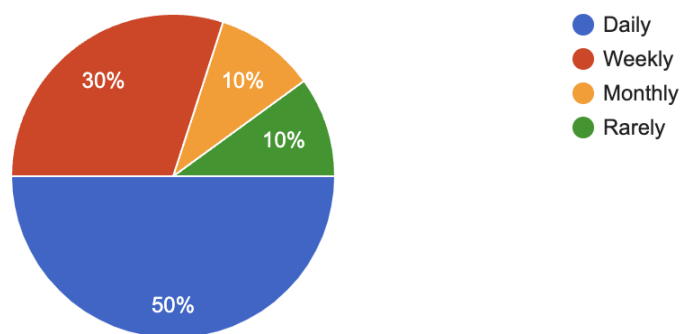


Figure 4.2: Usage Frequency of Project Management Tools

How many years of experience do you have in project management?

10 responses

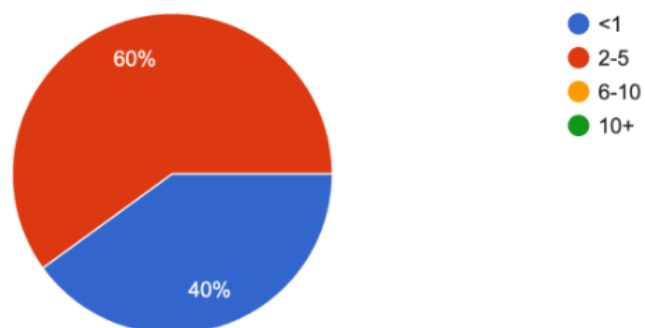


Figure 4.3: Years of Experience in Project Management

4.2.3 Post-task Questionnaire

After the conclusion of the task, users were asked to answer a series of questions on their experience and overall application feedback.

4.2.3.1 SUS (System Usability Scale)

Participants were asked to evaluate the system's usability by answering the following set of standardized questions [6]:

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

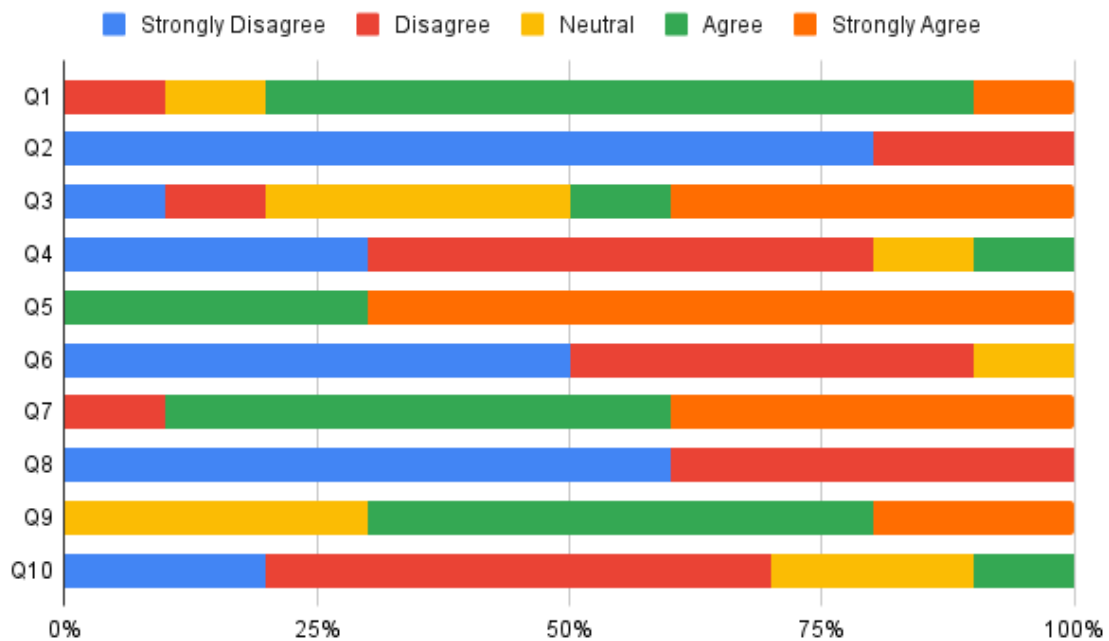


Figure 4.4: SUS results

Follows an in-depth analysis of the SUS Results 4.4 obtained.

- Question 1:** The majority of participants (70%) selected "Agree," while 10% opted for "Strongly Agree," indicating a generally positive attitude toward frequent system usage. However, 20% expressed a more neutral or negative stance, suggesting that while most users are satisfied, a minority may not find the system as appealing for regular use.
- Question 2:** A significant portion (80%) strongly disagreed or disagreed that the system was unnecessarily complex, showing that users overwhelmingly found the system straightforward. Only 10% offered a neutral stance, and no participants rated the system as highly complex, affirming its simplicity in use.
- Question 3:** Half of the participants (40%) rated the system as "Strongly Agree," with an additional 10% marking "Agree." However, 30% indicated neutrality, reflecting that while many found the system easy to use, a small group was less certain about its usability.
- Question 4:** The feedback on requiring technical support was varied, with 40% disagreeing or strongly disagreeing and 10% being neutral. Still, 30% agreed they might need help, suggesting that certain users may require more technical support or guidance when using the system.
- Question 5:** Most participants (70%) strongly agreed that the system's features were well-integrated, while no one found significant issues with its integration. This shows a strong positive perception regarding the system's coherence and functionality.

- **Question 6:** Half of the users (50%) strongly disagreed that there were inconsistencies within the system, while 40% disagreed or remained neutral. This highlights that most participants found the system reliable, with only a few experiencing minor inconsistencies.
- **Question 7:** A majority (90%) agreed or strongly agreed that the system had a smooth learning curve. Only 10% offered a neutral response, indicating that the system is generally easy to adopt for new users with few challenges in learning.
- **Question 8:** Most participants (60%) strongly disagreed that the system was cumbersome, and an additional 30% disagreed. Only 10% expressed neutrality, showing that users found the system user-friendly and easy to navigate.
- **Question 9:** The majority (90%) felt confident using the system, with 50% selecting "Strongly Agree" and 40% choosing "Agree." This suggests that the system inspires confidence among users, contributing to a smooth and effective experience.
- **Question 10:** Half of the respondents (50%) indicated that minimal learning was required before using the system, although 30% disagreed. This points to the system's accessibility but highlights that some users may benefit from additional training or onboarding.

$$\text{SUS Score} = 2.5 \times \sum_{i=1}^{10} \text{Adjusted Score}_i$$

The SUS score, calculated using the formula above, was 79.5, reflecting a favourable user perception of the system, surpassing the average usability benchmark of 68. The findings indicate that most users perceived the system as user-friendly and expressed overall satisfaction with its functionality. Nonetheless, opportunities for enhancements remain to improve user experience and satisfaction. The sample size of 10 is inadequate for drawing definitive conclusions; additional testing is necessary.

4.2.3.2 Feedback

Participants provided specific feedback by ranking various features based on their usefulness in managing project tasks. These features included the Completion Pie Chart, End-date Prediction Feature, Line Graph Visualization, Uncertainty Feature, and Issue Key Search. They were also asked to evaluate how easy it was to search for and retrieve specific issue information using the Issue Key.

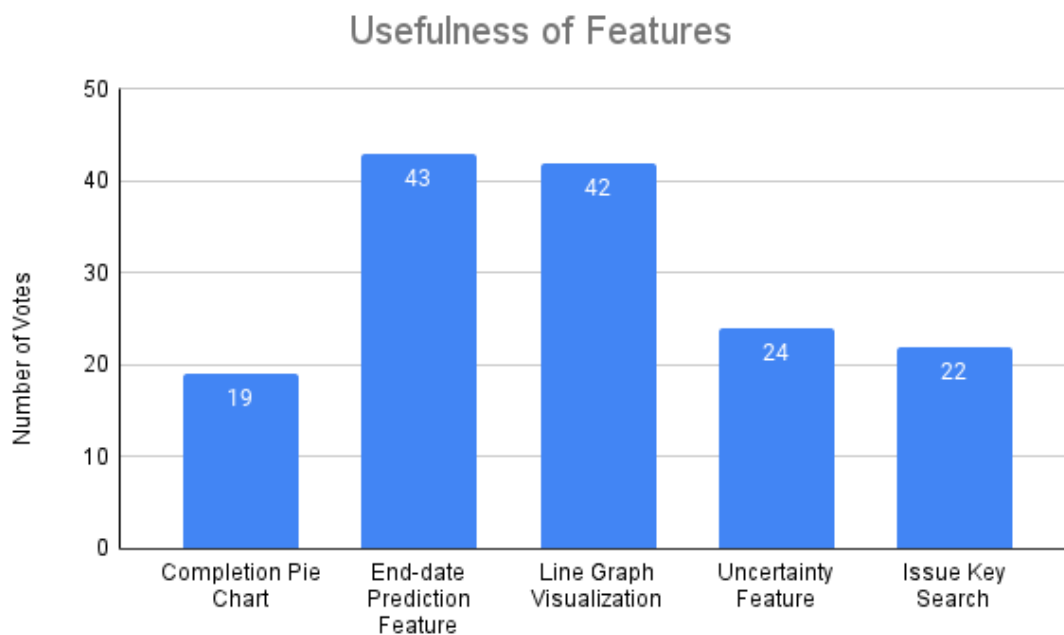


Figure 4.5: Usefulness of Features

The analysis of the chart 4.5 shows that participants found the End-date Prediction Feature and Line Graph Visualization to be the most useful tools, emphasizing the importance of predictive and visual aids. These features were likely the most valuable in assisting users in monitoring their progress and predicting the outcomes of their projects. On the other hand, the Uncertainty Feature and Issue Key Search were of moderate usefulness, suggesting that although they were beneficial, they were not as essential to the entire workflow. The Completion Pie Chart was deemed the least valuable, indicating that static depictions of data were less appreciated in comparison to more dynamic and future-oriented instruments. This underscores the participants' inclination towards characteristics that provide immediate and accurate observations and forecasts.

4.2.3.3 Additional Comments

Participants were prompted to provide suggestions for enhancing the user interface, report any encountered faults while using the app, and express their likelihood of recommending the app to other project managers. In addition, they have the ability to offer optional justification for their suggestion.

The feedback from participants revealed key areas for improvement, including interface adaptability, feature enhancements, and minor usability issues. Users expressed a desire for better compatibility with larger screens, additional features like individual issue visualization, and more varied graph types. Navigation improvements, such as scroll and zoom options, were also frequently suggested. Despite positive feedback on the app's potential to save time and improve

sprint planning, some users noted loading delays and the need for clearer guidance and support within the app.

One participant recommended "better adaptation to larger screens, or even full-screen," while others suggested features like "individual issue visualization" and additional graph types to offer a broader perspective on project data. Several users also mentioned adding scroll and zoom functions to improve graph interaction. Loading issues, especially with the line graph, were noted, with a suggestion to "adapt the loading circle to the line graph" to help manage expectations. A lack of in-app guidance and support features was highlighted, with requests for "bigger in size/scalable features" to improve usability.

Most participants expressed willingness to recommend the app, citing its potential to "save time in sprint planning" and its strong predictive capabilities. However, some users suggested that a full project test would be necessary "to compare results" in real-world settings. For broader adoption, participants recommended additional features such as "Excel file input" and sprint-specific views to fully meet project managers' needs.

Chapter 5

Conclusions and Future Work

5.1 Conclusions

The integration of large language models (LLM) and machine learning (ML) into project management provides an opportunity to alter the decision-making methodologies used currently. This work acknowledges the transition from traditional management practices to more data-driven ways and highlights the challenge of being able to look into the future based on past data, which, as a consequence, teams and companies benefit greatly.

This dissertation developed an Intelligent Project Tracking System that leverages LLM and ML to improve the accuracy and efficiency of project management processes. The framework was designed with the main goal of providing managers with their much-needed estimations and reducing the need for manual estimations.

The framework was validated through a case study within Deloitte, where its real-world application demonstrated its effectiveness in enhancing project tracking and management. The results were promising, illustrating the framework's ability to Story Points and support better decision-making accurately. While these findings are encouraging, it is recognized that further iterations and refinements are necessary to realize its full potential. Compared to other State-of-the-Art systems in this area, the Intelligent Project Tracking system widened the range of technologies used, yet compared to more Traditional, human-input methods, still has some improvements to make. In terms of utility in real-world scenarios, based on the input provided by the test users, it could be a solid Proof of Concept to integrate into everyday work and sprint planning sessions.

5.2 Main Contributions

This dissertation provides contributions to the field of project management:

- **Intelligent Project Tracking System:** The development of a framework that integrates AI and ML to enhance the efficiency of project management processes. This framework was rigorously tested and validated in real-world scenarios.

- **Case Study Implementation:** An application of the framework within Deloitte, demonstrating its practical utility in managing complex project data and contributing to more informed decision-making.
- **Knowledge Advancement:** Offering insights into the practical challenges and solutions involved in integrating LLM and ML into project management
- **Scientific Contribution:** The preparation of research findings for submission to a peer-reviewed journal, aiming to share the lessons learned and progress made with the broader academic and professional communities.

In summary, this work successfully developed and validated an LLM and ML-based framework for intelligent project management. While the contributions have practical and academic value, this research lays the groundwork for further refinement and exploration.

5.3 Future Work

This dissertation establishes a foundation for future research and development in intelligent project management systems. Enhancing the user interface is a priority, focusing on improved screen adaptation, scalability, and user support features. Additionally, integrating functionality to split data by sprints or weeks will further align the framework with Agile project methodologies, making it more user-friendly.

Future work should also focus on refining data analytics and enhancing machine learning models to increase prediction accuracy and system adaptability. Incorporating more sophisticated algorithms and expanding the dataset will reduce manual intervention and improve the system's ability to handle complex, real-time project management scenarios.

Finally, the framework would benefit from long-term deployment in real-world project cycles. Continuous use, feedback, and iteration will allow the system to mature, fostering wider adoption and ensuring it evolves to meet the changing demands of modern project management practices.

References

- [1] Adetokunbo Adenowo and Basirat Adenowo. Software Engineering Methodologies: A Review of the Waterfall Model and Object- Oriented Approach. *International Journal of Scientific and Engineering Research*, 4:427–434, September 2020.
- [2] Ethem Alpaydin. *Introduction to Machine Learning, fourth edition*. MIT Press, March 2020. Google-Books-ID: tZnSDwAAQBAJ.
- [3] Adel Alshamrani and Abdullah Bahattab. A comparison between three sdlc models waterfall model, spiral model, and incremental/iterative model. *International Journal of Computer Science Issues (IJCSI)*, 12(1):106, 2015.
- [4] B. W. Boehm. A spiral model of software development and enhancement. *Computer*, 21(5):61–72, May 1988. Conference Name: Computer.
- [5] Carlos Eduardo Sanches da Silva Breno Gontijo Tavares, Mark Keil and Adler Diniz de Souza. A risk management tool for agile software development. *Journal of Computer Information Systems*, 61(6):561–570, 2021.
- [6] John Brooke. Sus: A quick and dirty usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & I. L. McClelland (Eds.), *Usability evaluation in industry*. London: Taylor and Francis, 1996.
- [7] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020.
- [8] Beatriz Cabrero-Daniel. Ai for agile development: a meta-analysis, 2023.
- [9] Mike Cohn. *User Stories Applied: For Agile Software Development*. Addison-Wesley Professional, Boston, MA, 2004.
- [10] Jeff Dalton and Jeff Dalton. Dot voting. *Great Big Agile: An OS for Agile Leaders*, pages 165–166, 2019.

- [11] Hoa Khanh Dam, Truyen Tran, John Grundy, Aditya Ghose, and Yasutaka Kamei. Towards Effective AI-Powered Agile Project Management. *2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 41–44, May 2019. Conference Name: 2019 IEEE/ACM 41st International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER) ISBN: 9781728117584 Place: Montreal, QC, Canada Publisher: IEEE.
- [12] Electron Forge. Why electron forge. Accessed: 2024-08-11.
- [13] Michael Fu and Chakkrit Tantithamthavorn. GPT2SP: A Transformer-Based Agile Story Point Estimation Approach. *IEEE Transactions on Software Engineering*, 49(2):611–625, February 2023. Conference Name: IEEE Transactions on Software Engineering.
- [14] Kamran Ghane. Quantitative planning and risk management of agile software development. In *2017 IEEE Technology & Engineering Management Conference (TEMSCON)*, pages 109–112, June 2017.
- [15] Louie Giray. Prompt engineering with chatgpt: a guide for academic writers. *Annals of biomedical engineering*, 51(12):2629–2633, 2023.
- [16] Muhammad Hammad, Irum Inayat, and Maryam Zahid. Risk management in agile software development: A survey. In *Risk Management in Agile Software Development: A Survey*, 11 2019.
- [17] N.C. Haugen. An empirical study of using planning poker for user story estimation. In *AGILE 2006 (AGILE’06)*, pages 9 pp.–34, July 2006.
- [18] Statista IDC. Digital transformation, 2021. Retrieved May 2024, from https://drive.google.com/file/d/1X0ltNT9jFA9l8cUkZ_1p4jNjF7_vVWcV/view.
- [19] J.M. Juran and A.B. Godfrey. *Juran’s Quality Handbook*. McGraw-Hill International Editions. McGraw Hill, 2010.
- [20] Musarath Jahan Karamthulla, Sanjeev Prakash, Anish Tadimarri, and Manish Tomar. Efficiency unleashed: Harnessing ai for agile project management. *International Journal For Multidisciplinary Research*, 6(2):1–13, 2024.
- [21] Tokay Korkut. Excelling in software development with scrum methodology part 2.
- [22] Ravi Kiran Mallidi and Manmohan Sharma. Study on Agile Story Point Estimation Techniques and Challenges. *International Journal of Computer Applications*, 174:9–14, January 2021.
- [23] Ravi Kiran Mallidi and Manmohan Sharma. Study on agile story point estimation techniques and challenges. *Int. J. Comput. Appl*, 174(13):9–14, 2021.
- [24] Glenford J. Myers, Corey Sandler, and Tom Badgett. *The art of software testing*. John Wiley & Sons, Hoboken and N.J, 3rd ed edition, 2012.
- [25] Cambridge University Press. prompt, September 2024.
- [26] Cambridge University Press. risk management, September 2024.
- [27] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners, 2019.

- [28] Ian Sommerville. *Software engineering*. Pearson, Boston, 9th ed edition, 2011. OCLC: ocn462909026.
- [29] Vali Tawosi, Afnan Al-Subaihin, and Federica Sarro. Investigating the effectiveness of clustering for story point estimation. In *2022 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 827–838, 2022.
- [30] Jason Wei, Xuezhi Wang, Dale Schuurmans, et al. Chain of thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*, 35:24824–24837, 2022.
- [31] Elham Karim Zadeh, Ali Bagheri Khoulenjani, and Mohammad Safaei. Integrating ai for agile project management: Innovations, challenges, and benefits. *International Journal of Industrial Engineering and Construction Management (IJIECM)*, 1(1):1–10, 2024.

Appendix A

Intelligent Project Tracking System Validation

Intelligent Delivery Progress Tracking System

This feedback form is part of a research study for the master's dissertation titled "Intelligent Delivery Progress Tracking System" at FEUP. The purpose of this form is to collect valuable insights from users like you who have interacted with the newly developed project management application. Your feedback is crucial in helping understand how well the application meets the needs of project managers and how its features can be improved.

The application you tested is designed to integrate seamlessly with JIRA and provides enhanced tracking and predictive insights for project management. It features a completion Pie Chart for monitoring project progress, a breakdown of User Stories completed by each worker, predictive analysis for project completion dates, and a functional interface for issue-specific queries.

Structure of the Form:

1. **User Background:** Questions about your experience with project management and familiarity with JIRA.
2. **System Usability Scale (SUS):** A standard questionnaire to assess the usability of the system.
3. **App-Specific Feedback:** Questions targeting specific functionalities and your experience with the application.

This survey should take approximately 10-15 minutes to complete. Your responses will remain confidential and will only be used for academic purposes related to this dissertation. We value your honest and detailed feedback.

Thank you for contributing to this important phase of our research. Your insights are vital to improving the effectiveness and usability of our application.

* Indicates required question

User Background

Questions about your experience with project management and familiarity with JIRA.

1. **Current work position ***

2. **Number of years in that position ***

Mark only one oval.

- ☐ <1 year
- ☐ 2-3 years
- ☐ 4-5 years
- ☐ >5 years

3. **How many years of experience do you have in project management? ***

Mark only one oval.

- ☐ <1
- ☐ 2-5
- ☐ 6-10
- ☐ 10+

4. **Familiarity with JIRA ***

Mark only one oval.

1 2 3 4 5

Not ☐ ☐ ☐ ☐ ☐ Very Familiar

5. **How often do you use project management tools in your daily work? ***

Mark only one oval.

- ☐ Daily
☐ Weekly
☐ Monthly
☐ Rarely

6. **How comfortable are you with using technology to manage projects? ***

Mark only one oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very Comfortable

7. **Proficiency with Project Management Software ***

Mark only one oval.

	1	2	3	4	5	
No	☐	☐	☐	☐	☐	Very Proficient

8. **What features do you value most in project management software? (List 1 or more)**

9. Have you ever participated in a project management software beta test before? *

Mark only one oval.

☐ Yes

☐ No

System Usability Scale (SUS) Questions (10 Questions)

The SUS provides a reliable tool for measuring the usability. It consists of a 10-item questionnaire with five response options for respondents; from Strongly agree to Strongly disagree.

10. I think that I would like to use this system frequently. *

Mark only one oval.

1 2 3 4 5

Strongly disagree ☐ ☐ ☐ ☐ ☐ Strongly agree

11. I found the system unnecessarily complex. *

Mark only one oval.

1 2 3 4 5

Strongly disagree ☐ ☐ ☐ ☐ ☐ Strongly agree

12. I thought the system was easy to use. *

Mark only one oval.

1 2 3 4 5

Strongly disagree ☐ ☐ ☐ ☐ ☐ Strongly agree

13. I think that I would need the support of a technical person to be able to use this system. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

14. I found the various functions in this system were well integrated. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

15. I thought there was too much inconsistency in this system. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

16. I would imagine that most people would learn to use this system very quickly. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

17. I found the system very cumbersome to use. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

18. I felt very confident using the system. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

19. I needed to learn a lot of things before I could get going with this system. *

Mark only one oval.

1 2 3 4 5

Strc ☐ ☐ ☐ ☐ ☐ Strongly agree

App-Specific Questions (6 Questions)

This section should focus on the specific features and functionalities of your app. It should gather insights on what works well and what could be improved.

20. Please rank the following features in order of usefulness in managing your project tasks

1- Least Useful, 5 - Most Useful

Mark only one oval per row.

	1	2	3	4	5
Completion Pie Chart	☐	☐	☐	☐	☐
End-date Prediction Feature	☐	☐	☐	☐	☐
Line Graph Visualization	☐	☐	☐	☐	☐
Uncertainty Feature	☐	☐	☐	☐	☐
Issue Key Search	☐	☐	☐	☐	☐

21. How easy is it to search for and retrieve specific issue information using the Issue Key? *

Mark only one oval.

	1	2	3	4	5	
Very	☐	☐	☐	☐	☐	Very Easy

22. What improvements would you suggest for the user interface? (Optional)

23. Did you encounter any issues while using the app? If so, please describe.

24. Would you recommend this app to other project managers? *

Mark only one oval.

1 2 3 4 5

Defi ☐ ☐ ☐ ☐ ☐ Definitely Yes

25. Why or why not? (Optional)

This content is neither created nor endorsed by Google.

Google Forms

Tasks

Task 1

1. **Unzip the Application:** Unzip the provided application package.
2. **Launch the Application:** Open the application from the unzipped folder.
3. **Open `credentials.txt`:** Locate the `credentials.txt` file in the unzipped folder.
4. **Fill in Jira Base URL:** Copy the Jira Base URL from `credentials.txt` and paste it into the corresponding field in the application.
5. **Enter Project Key:** Copy the Project Key from `credentials.txt` and enter it into the application.
6. **Input Email:** Copy the Email from `credentials.txt` and paste it into the relevant field in the application.
7. **Provide API Token:** Copy the API Token from `credentials.txt` and enter it into the application.
8. **Generate Diagrams:** Click on the button to generate diagrams.
9. **Review Task Completion Display:** Assess the Task Completion display for correctness and clarity.
10. **Check Resource Allocation:** Examine the Resource Allocation display to ensure it accurately reflects the project data.
11. **Evaluate Line Graph:** Review the line graph display for its accuracy and relevance.