KOFORIDUA TECHNICAL UNIVERSITY

FACULTY OF APPLIED SCIENCE AND TECHNOLOGY

(DEPARTMENT OF COMPUTER SCIENCE)

FUEL MANAGEMENT SYSTEM

A PROJECT WORK SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, SCHOOL OF APPLIED SCIENCE AND TECHNOLOGY IN PARTIAL FULFILMENT OF THE AWARD OF THE BACHELOR IN TECHNOLOGY (BTECH) IN COMPUTER SCIENCE PROGRAMME.

BY

LETICIA MAMLEY KABU	B202210185
EPHRAIM GYAMFI DASEY	B202210156

CSD 402: PROJECT IN COMPUTER SCIENCE

DATE: 3rd November 2023

SUPERVISOR: DR PATRICIA GHANN

DECLARATION

We hereby certify that the project work, Fuel Management System, that we presented to Koforidua Technical University is an original work that we completed with the assistance of Dr. Patricia Ghann of the Faculty of Applied Science and Technology's Computer Science. The project work is being submitted to partially complete the requirements for the Computer Science Bachelor of Technology (BTECH) degree. The project's outcomes have not been submitted to any other university or institution for the purpose of conferring a degree or diploma.

LETICIA MAMLEY KABU

……

EPHRAIM GYAMFI DASEY

……

CERTIFICATION

I certify that the research was supervised in compliance with Koforidua Technical University's project work supervision requirements.

3rd November 2023

…… ……..……

DR PATRICIA GHANN DATE

(SUPERVISOR)

ACKNOWLEGDEMENT

Our deepest gratitude is extended to the God, who provided the wisdom and understanding necessary for the completion of this project. Throughout this endeavor, the Divine's guidance and protection ensured the removal of unforeseen obstacles, strengthening our determination even during the most challenging moments.

We extend our heartfelt appreciation to the Head of the Computer Science Department, represented by Dr. Seth Alornyo. We also express our gratitude to the entire academic staff, both seasoned and new members of the Computer Science Department. Their collective support and expertise have significantly contributed to the success of this project.

Special recognition is reserved for our supervisor, Dr. Patricia Ghann, whose invaluable advice, contributions, and constructive criticism played an instrumental role in motivating us to enhance the originality and success of this work. Under Dr. Ghann's guidance, our understanding of the subject matter has greatly expanded, and we have experienced genuine research.

We wish to acknowledge our parents, Mr. Kabu and the late Mrs. Kabu, as well as Mr. and Mrs. Dasey, for their unwavering support and prayers that enabled us to complete this academic endeavor.

Finally, our sincere thanks go to our colleague, Samuel Afotey Laryea, for the assistance and support he provided, which significantly contributed to the successful completion of this academic work.

Table of Contents

ABSTRACT

Advancements in technology have revolutionized various industries, including the management of fuel resources. A fuel management system plays a crucial role in efficiently tracking and managing fuel usage, inventory, and related operations. This project aims to propose the development of a Fuel Management System to streamline fuel management processes, enhance automation, improve security and also provide a user-friendly interface. The system will consist of interconnected modules, including a Recording System for inventory management, a Commerce System for fuel sales and transactions, and a Reporting System for generating comprehensive fuel usage reports. Additionally, the system will prioritize security measures to ensure data integrity and prevent unauthorized access.

Chapter 1

Introduction

The effective management of energy resources, particularly fuel, has become an essential concern for enterprises and organizations worldwide in the current period of rapid technological innovation and increased environmental awareness. Key objectives that need the adoption of cutting-edge systems with intelligent fuel data gathering and analysis capabilities are the optimization of fuel use, cost savings, and environment's impact mitigation. This project intends to satisfy these imperatives by building and implementing an enhanced Fuel Management System (FMS) that not only collects real-time fuel usage data but also offers in-depth analysis for data-driven decision-making.

In terms of managing energy resources, the creation of a robust Fuel Management System (FMS) marks a significant advancement. The FMS helps enterprises and organizations to make wise decisions that result in optimized resource usage, lower costs, and a smaller environmental impose by effortlessly collecting, processing, and analyzing data on fuel consumption. This research highlights the significance of cutting-edge technology in overcoming the difficulties presented by limited energy supplies and highlights its role in promoting a sustainable and effective future.

1.1 Background

The transportation and industrial sectors are cornerstones of global economic activity, driving progress and development. Yet, their heavy reliance on fuel for day-to-day operations poses significant challenges, particularly in the context of sustainability and resource management. The quest for efficient fuel utilization has become paramount, as it not only affects operational costs but also carries profound environmental implications. Unfortunately, the absence of real-time data collection and advanced analysis tools has impeded progress toward optimal fuel management.
And with that, we'll introduce fuel management systems, which are utilized in every industry that relies on transportation, including rail, road, water, and air, to maintain, control, and monitor fuel usage and stock. Systems for managing the consumption of fuel in the transportation and construction sectors are created to measure and control it effectively.(Wikipedia, 2023)

In many instances, outdated manual record-keeping methods have been the norm, resulting in a time lag between fuel consumption events and data availability. This delay hampers the ability to promptly identify inefficiencies or deviations from established benchmarks. Consequently, sub-optimal fuel usage persists, leading to avoidable financial expenditures and a heightened ecological footprint through increased carbon emissions.

In a time of rapid technological advancement and rising emphasis on sustainable practices, the shortcomings of conventional fuel management strategies have become startlingly obvious. Due to the reactive nature of review and the lack of access to real-time information, industries are unable to make well-informed decisions that could increase efficiency, save costs, and have a smaller environmental effect.

A major change in fuel management strategies is required to address these issues. The creation and use of sophisticated real-time data collection and processing systems is extremely promising. These systems can offer a previously unheard-of level of transparency into fuel use trends, empowering organizations to take wise decisions immediately. Organizations can quickly spot patterns, deviations, or abnormalities with access to precise and current data, enabling quick course corrections and preventative steps.

The achievement of sustainable development goals depends critically on the interaction between technology and resource management. Industries can access previously hidden insights by using cutting-edge sensors, data aggregation systems, and analytical algorithms. Businesses are now better able to optimize fuel allocation, streamline processes, and improve resource management overall thanks to this greater openness.

Additionally, the incorporation of real-time data collection and analysis technologies promotes a continuous improvement culture. Organizations may drastically lower operational expenses and lessen their carbon footprint by tackling inefficiencies as soon as they arise and putting best practices into action. Modern organizations strive to run effectively while fulfilling their ethical and environmental obligations, and this shift from reactive to proactive fuel management fits in perfectly with those goals.

In conclusion, the challenges posed by fuel management in the transportation and industrial sectors are pressing and multifaceted. The shortcomings of conventional approaches highlight the pressing need for adopting cutting-edge technology that permit real-time data collecting and processing. Industry can embark on a transformational journey toward sustainable development by utilizing these instruments, wherein cost reduction, reduced environmental impact, and optimized fuel consumption come together to form

1.2 Problem Statement

The lack of a unified and effective framework for data gathering and analysis characterizes the condition of fuel management practices today, which is a considerable challenge. Traditional approaches, especially manual data input, have shown to be labour-intensive, prone to mistakes, and ultimately ineffective. Due to these inefficiencies, businesses are unable to make timely, informed decisions, which prevents them from realizing their potential for cost savings and emissions reductions. Although it was once the norm, manual data entry has many problems. Human error can occur during the procedure, including errors in transcribing and data omission. Even little inaccuracies like these can cause big disparities in fuel usage records. Additionally, the time spent on manual data entry and collection results in a significant delay when trying to retrieve important data.

The consequences of these inefficiencies are far-reaching. Organizations are unable to promptly identify irregular fuel consumption patterns, detect anomalies, or respond swiftly to deviations from expected norms. As a result, operational costs remain elevated due to sub optimal fuel usage, and carbon emissions continue to escalate due to inefficient resource allocation.

There is an unquestionable and urgent requirement for a strong and cutting-edge Fuel Management System (FMS) to handle these serious problems. By incorporating automation, real-time monitoring, and sophisticated analytical capabilities, such a system would completely transform current fuel management procedures.

Beyond data collection, the Fuel Management System would provide a user-friendly interface that offers advanced analytical tools. These tools would enable comprehensive data analysis, offering deeper insights into consumption patterns, identifying opportunities for optimization, and facilitating predictive modeling for future consumption.

The benefits of such a system are substantial. Organizations would be empowered to make data-driven decisions in real time, optimizing fuel consumption based on accurate and up-to-the-minute information. This would lead to tangible cost savings, as well as a notable reduction in carbon emissions, aligning with sustainability and environmental goals.

In conclusion, the inefficiencies inherent in existing fuel management practices are a pressing concern for organizations across industries. The time has come for a transformational shift toward a streamlined and automated Fuel Management System that empowers organizations with real-time data, advanced analytic, and the capacity to optimize fuel consumption. By embracing such innovation, industries can unlock substantial benefits in terms of cost savings, operational efficiency, and environmental stewardship.

1.3 Aims and objectives

1.3.1 Main objective

Developing a user-friendly online application is our main goal for the Fuel Management System project. The real-time data collection from gas station staff will be greatly enhanced by this application. The ultimate objective is to use this data to greatly increase operational efficiency in the field of fuel management. This involves making sure that decisions are made on time, improving inventory control, and allocating resources as efficiently as possible. By doing this, we want to save the company a significant amount of money, encouraging sustainability and financial responsibility. In order to guarantee user-friendliness for all parties involved and promote broad adoption, the user-friendly interface is essential. Our ultimate goal is to develop a fuel management system that is data-driven and promotes cost- and operational-efficiency.

1.3.2 Specific objective

In the realm of fuel management, efficiency and precision are paramount to the success and sustainability of any operation. This project is dedicated to achieving these crucial goals through a comprehensive Fuel Management System. To guide our efforts and ensure that our project delivers tangible results, we have outlined four specific objectives:

1. **User-Friendly Visualization Interface:** The first objective is to design and implement an intuitive and user-friendly interface. This interface will empower stakeholders to gain deep insights into fuel usage data by offering customization dashboards, graphs, charts, and reports. Real-time monitoring

and historical analysis will become effortless, enabling stakeholders to make informed decisions.

2. **Seamless Inventory Management:** Our second objective is to streamline inventory management. We will implement an inventory system that tracks fuel levels in real-time and triggers reorders as necessary. This approach will significantly reduce the risk of running out of fuel during critical operations, ensuring uninterrupted workflow.

3. **Real-time Attendance Tracking:** The third objective involves the development of a real-time attendance tracking feature. By accurately identifying when employees clock in and out, this feature will provide continuous visibility into staffing levels throughout the day. It will enhance resource allocation and overall workforce management.

4. **Actionable Insights for Decision Making:** Our final objective is to provide actionable insights for decision-making. We will develop a decision-support component that transforms analyzed data into comprehensive reports. These reports will empower decision-makers with the information needed to make well-informed choices regarding fuel allocation and consumption optimization.

By pursuing these specific objectives, the project aims to deliver a comprehensive Fuel Management System that revolutionizes the way organizations approach fuel consumption and resource management. The system will streamline

data collection, empower decision-makers with timely insights, enhance operational efficiency, and contribute to a more sustainable and environmentally responsible future for industries and organizations that heavily rely on fuel for their operations.

1.4 Scope of the project

A data collection-focused fuel management system can cover a wide range of fuel usage, monitoring, and optimization topics. Such a system's main objective is to collect, process, and evaluate information about fuel usage, inventory levels, refilling activities, and other pertinent characteristics. Here are some essential elements and features that make up a fuel management system:

1. **Fuel Consumption Monitoring:** Tracking fuel consumption at fuel forecourts according to the terminals available for serving fuel to customers. This involves collecting data on fuel usage over a period of time to help generate data for decision making.

2. **Inventory Management:** Managing fuel inventory levels in tanks and storage facilities. This includes monitoring fuel levels, predicting when refueling is needed, and optimizing inventory to prevent shortages or excess.

3. **Data Collection:** Gathering data from various sources such as fuel station managers, fuel attendant and fuel transporters. Data could include fuel quantities, refueling dates, locations and more.

4. **Data Aggregation and Storage:** Storing collected data in a centralized database for easy access, analysis, and reporting. Data aggregation allows for a comprehensive overview of fuel consumption patterns and trends.

5. **Reporting and Analytics:** Generating detailed reports, dashboards, and analytics to gain insights into fuel consumption patterns, efficiency improvements, cost analysis, and other key performance indicators.

6. **Fuel Efficiency Analysis:** Analyzing data to identify opportunities for improving fuel efficiency, optimizing routes, reducing idle time, and minimizing fuel wastage.

7. **Security and Fraud Prevention:** Implementing security measures to prevent unauthorized access to fuel data, detecting unusual fuel consumption patterns that could indicate fraud or theft.

8. **Integration:** Integrating with other business systems such as fleet management software, accounting software, and ERP systems for seamless data exchange and work flow automation.

9. **Remote Monitoring:** Enabling remote monitoring and control of fuel-related activities, particularly useful for effective decision making.

10. User Access Control: Implementing role-based access controls to restrict access to sensitive fuel data and functionalities.

11. Future Enhancements: Allowing for scalability and the potential to integrate with emerging technologies like IoT sensors, predictive analytics, and machine learning for even more advanced insights and optimizations.

By focusing on these components, this project aims to provide a comprehensive solution that addresses the limitations of current fuel management practices in industrial and transportation sectors. This tailored Fuel Management System has the potential to revolutionize how organizations manage their fuel resources, leading to optimized consumption, reduced costs, and a minimized environmental footprint.

1.5 Constrains

The outlined constraints below plays a critical role in shaping the project's scope, design, and implementation. By acknowledging and working within these constraints, the project can achieve its objectives effectively while ensuring a practical and feasible outcome. Let's further elaborate on how each constraint can be managed:

1. Technical Constraints:

Conduct a comprehensive assessment of the existing technological landscape in the target industries. Identify integration points and ensure

compatibility with legacy systems. Prioritize modularity and flexibility in the system's design to accommodate potential variations in technology across different stakeholders.

2. Budgetary Constraints:

Develop a detailed budget plan that allocates resources efficiently across hardware, software, testing, and validation stages. Consider open-source and cost-effective software solutions that align with budget limitations without compromising functionality.

3. Time Constraints:

Implement effective project management methodologies, such as agile or iterative approaches, to ensure tasks are prioritized and completed within the given time frame. Break down the project into manageable milestones and allocate time for thorough testing and validation.

4. Data Privacy and Security Constraints:

Collaborate with legal and compliance experts to ensure that the system adheres to data privacy regulations (e.g., GDPR, HIPAA) and industry-specific standards. Implement robust data encryption, access controls, and authentication mechanisms to safeguard sensitive fuel consumption data.

5. Resource Constraints:

Leverage the expertise of available skilled personnel within your project team. Consider collaboration with industry partners, academic advisers,

or external experts to supplement technical expertise where needed. Prioritize essential features and functionalities that align with available resources and expertise.

By embracing these constraints as essential parameters, this project plans navigate challenges effectively and develop a Fuel Management System that aligns with the real-world realities of industrial and transportation sectors. A well-executed approach that balances innovation, pragmatism, and careful resource management will contribute to the successful design and implementation of the system while ensuring its relevance and impact in the field.

1.6 Project Overview

In culmination, this project stands as a transformational force, reshaping the landscape of fuel management by delivering real-time insights into consumption, distribution, and usage trends. The FMS propels operational efficiency, cost-effectiveness, and environmental responsibility, arming decision-makers with the knowledge to optimize fuel usage, ensure compliance, and drive informed strategies.

In the subsequent chapters, we will delve deeper into the core aspects of the project development. This aspects are as follows:

Chapter Two (Literature Review): a review of relevant work by other authors and the relationship between them and the project.

Chapter Three (Requirement analysis): the objectives of the project in terms of requirement and the analysis.

Chapter Four (Design implementation and testing) : explain the design technique chosen and develop a design

Chapter Five (Result and discussion) : contains findings, goals achieved and further work.

Chapter Six (Conclusion): summarizes the project's journey, outcomes, lessons learned, and future implications.

By delving into these chapters, we would portray comprehensive understanding of how Fuel Management System optimizes data collection processes, ensuring that accurate and timely information drives informed decision-making, operational efficiency, and overall success in fuel management.

Chapter Two

Literature Review

This chapter forms the fundamental cornerstone of the discussion that follows. It seeks to create a thorough grasp of the fundamental ideas, procedures, and frameworks that specify the parameters of this project. This literature review's main goal is to give readers a concise summary of the project's goals, advantages, and possible disadvantages. Furthermore, we will examine current systems that are pertinent to our area of study, describing their benefits, drawbacks, and influence on our endeavor.

2.1 Review of Theories

Rapid Application Development (RAD) is a cycle that is important to the development of the proposed system. Given the time constraints of our project, Rapid Application Development (RAD) is a great option because it stresses speed and flexibility in the software development process. Our goals are well-aligned with this development methodology, which is distinguished by its iterative nature and constant feedback loops. The speed with which RAD operates will make it possible for us to deliver our system on schedule, which is essential for any project to be successful.

Rooted in the Rapid Application creation (RAD) cycle is the creation of the proposed system. RAD is a great option for our project because of its emphasis on speed and flexibility in the software development process. This development strategy fits our goals very well because of its iterative nature and ongoing feedback loops.

The quickness with which RAD operates will guarantee the timely delivery of our intended system, which is essential to the accomplishment of any project.

2.2 Review of Existing Systems

2.2.1 Banlaw Fuel Management System

The Banlaw Fuel Management System is a comprehensive solution designed to cater to companies heavily dependent on fuel for their operational needs. It seeks to streamline the management of fuel usage, distribution, and monitoring, with a core accountability. Additionally, the system is aimed at delivering reliable reporting and data analysis capabilities.

Common applications for the Banlaw Fuel Management System include industries such as manufacturing, mining, transportation, and construction. Its benefits are substantial, as it efficiently manages fuel usage, prevents theft, optimizes fuel consumption, and provides valuable data for insightful analysis. However, like any system, it is not without its challenges. These challenges include issues related to data accuracy stemming from sensor and connectivity problems, barriers to user adoption and training, concerns about data security and privacy, difficulties in system integration, and maintenance requirements for hardware components. Furthermore, data validation and auditing, user interface usability, change management, and user knowledge all present potential stumbling blocks.

One noteworthy limitation of the Banlaw system is its focus on collecting and analyzing fuel data primarily from tanks and dispensers, neglecting data collection

from fuel station employees. This absence creates a significant gap, which our project aims to address. Our system will empower fuel station employees, such as managers, fuel transporters, and attendants, to input their data, fostering real-time data analysis and improving the quality of reports generated for more effective decision-making.

2.2.2 NaftaPOS Fuel Management Software

NaftaPOS is a comprehensive software designed to optimize the operations of petrol stations efficiently. It strives to maximize profits for petrol station owners, reduce management costs, ensure transparency in cashier and fuel attendant activities, simplify reporting, and provide control over fuel dispensers and Automatic Tank Gauge (ATG) systems. The software is versatile, catering to various types of petrol stations dispensing petroleum products, diesel fuel, LPG, and CNG. It offers compatibility with a broad range of fuel dispenser and ATG system brands, capable of controlling up to 32 fuel dispenser sides.

The NaftaPOS Fuel Management Software presents several advantages, including profit maximization, cost reduction in management, transparency in activities, simplified reporting, compatibility with various equipment, payment flexibility, and optional remote monitoring capabilities. However, it comes with certain challenges as well. These include initial implementation complexity, hardware compatibility issues, a learning curve for personnel, requirements for maintenance and support, associated costs, and potential data security concerns when employing remote monitoring.

One key distinction between the NaftaPOS system and our project is the collection of data. While NaftaPOS offers a display of fuel usage on forecourt equipment and fuel tanks, it lacks the ability to collect data from fuel station employees. Our proposed system bridges this gap by enabling data collection from employees during their daily activities. This feature provides real-time monitoring and ensures accurate data for analysis and effective decision management.

This chapter represents a pivotal segment of our project, establishing the foundation for the subsequent chapters. It elaborates on our approach to system development and the selection of the most appropriate life cycle for this project. Additionally, it highlights the significance of existing systems within the project's field of research, underscoring their benefits, drawbacks, advantages, and limitations. By exploring these systems in detail, this chapter paves the way for a deeper understanding of our project's context and objectives.

2.3 Methodology

The development process for this project is based on Rapid Application Development (RAD). Rapid Application Development (RAD) is an approach that puts speed and flexibility first when developing software. It is based on a sequence of iterative procedures that guarantee quick system evolution and ongoing user involvement. Users actively participate in the system's development and provide valuable input. This flexible and dynamic approach is especially useful when dealing with the possibility of changing project needs.

2.3.1 Key Principles of RAD

1. Frequent Iterations

RAD breaks the project up into more manageable, smaller iterations. Every version concentrates on particular features or functionalities. This iterative process keeps a constant feedback loop open with users while speeding up development.

2. User-Centric Approach

Users are central to the development process using RAD. By actively participating, they guarantee that the system's evolution is driven by their wants and expectations. It takes frequent user feedback and involvement to modify the system to meet users' needs.

3. Flexibility and Adaptability

The versatility of RAD is one of its main advantages. It can accept alterations and revisions even after the development process has started. This adaptability is essential in a setting where specifications are subject to change.

2.3.2 Advantages of RAD

1. Rapid Development

Iterative processes like RAD guarantee quick progress, which is very helpful for tasks with limited time.

2. User Satisfaction

Users that actively participate in the system ensure that it closely matches their expectations, which in turn leads to increased user satisfaction.

3. Quick Issue Resolution

Rapid issue and change tracking, or RAD, reduces errors and guarantees a more accurate depiction of the work in progress.

2.3.3 Challenges and Considerations

1. Continuous User Engagement

Although it can require a lot of resources, keeping lines of communication open with users throughout the project is crucial.

2. Change Management

Effectively managing modifications and iterations can be difficult, especially in larger projects, but it is essential for project success.

3. Risk Management

While RAD enables prompt reactions to new hazards, it is imperative to recognize and address any problems as soon as feasible.

Our goal in adopting RAD as our project's guiding methodology is to capitalize on its strengths rapid development, adaptability, and user-centricity. By using this method, it is ensured that the Fuel Management System will be designed effectively and adjusted to match the changing requirements of businesses that depend on fuel resources.

2.4 Lessons from Existing Systems

In the journey to develop our innovative Fuel Management System, we draw significant inspiration and insights from existing systems, specifically the Banlaw Fuel Management System and NaftaPOS. These systems, while serving as valuable benchmarks, also offer crucial lessons that guide our project's development and set the stage for a more efficient, secure, and user-friendly system with comprehensive data capture capabilities.

The Banlaw Fuel Management System

The Banlaw Fuel Management System has played a pivotal role in shaping our project's development. It has brought to the forefront several key lessons:

1. Data Security

Banlaw's system has underscored the paramount importance of data security. The sensitivity of the data collected, especially in fuel management, necessitates robust security measures. Understanding this lesson, we are committed to incorporating state-of-the-art security protocols into our system, encompassing encryption, user authentication, and access control. This guarantees data integrity and protects user privacy.

2. User-Friendliness

The success of any system heavily depends on how easily users can navigate it. Banlaw's system highlights the significance of a user-friendly interface. We have taken this lesson to heart, ensuring that our system's interface caters to all stakeholders, from fuel station employees to management. This user-centric design minimizes the learning curve and promotes widespread adoption, ultimately making our system more accessible and valuable.

3. Comprehensive Data Collection

Banlaw's focus on fuel tanks and dispensers serves as a valuable lesson. We recognize the need for comprehensive data collection, including insights from fuel station employees. By collecting data from employees such as managers, transporters, and attendants, our system bridges an existing gap and empowers them to contribute real-time data. This integration significantly refines the quality of reports generated for effective decision-making, ensuring that our system is a powerful asset in fuel management.

NaftaPOS

Similarly, NaftaPOS has contributed its own set of lessons, enriching our project's development:

1. Connectivity Challenges

NaftaPOS has highlighted the connectivity issues that can arise, particularly in remote locations. This lesson has prompted us to address these challenges by incorporating offline data collection capabilities into our system. This feature ensures that data can be collected regardless of network availability and can be synchronized when connectivity is restored. This approach enhances the reliability and robustness of our system.

2. Profit Maximization and Transparency

The NaftaPOS system's emphasis on profit maximization and transparency has reinforced our commitment to these principles. We are dedicated to creating a system that not only optimizes fuel management but also aligns with our users' goals, whether it's reducing costs or increasing profits. The insights from NaftaPOS guide us in providing maximum value to our users.

In summary, the lessons drawn from both the Banlaw Fuel Management System and NaftaPOS serve as valuable touch points in our project's development. They emphasize the critical elements of data security, user-friendliness, comprehensive data collection, addressing connectivity challenges, and aligning with user objectives. By incorporating these lessons into our project, we aim to deliver a

system that not only addresses the challenges of fuel management but also exceeds user expectations, making it an efficient, secure, and user-friendly solution with comprehensive data capture capabilities.

2.5 Chapter Conclusion

In conclusion, this thorough literature study chapter has given our project's future chapters the essential starting point. It has provided a comprehensive understanding of the theories, methodologies, and systems that form the foundation of our innovative Fuel Management System. By thoroughly reviewing existing systems and acknowledging their respective advantages and limitations, we have gained valuable insights that will guide the development of our project.

The following chapters will delve deeper into the practical implementation, challenges, and outcomes of our Fuel Management System project. We will navigate the intricacies of bringing this innovative system to life, address the hurdles that may arise, and present the results and impact of our efforts. This review has equipped us with the knowledge and perspective necessary to make informed decisions and ensure the success of our project.

CHAPTER THREE

SYSTEM ANALYSIS AND REQUIREMENTS

3.1 ANALYSIS

3.1.1 Introduction to Requirements Analysis

Fuel management systems available were designed to enhance operational efficiency, reduce fuel waste, improve fuel accountability, and provide reliable reporting and analysis by utilizing systems that check fuel tanks and equipment for fuel data. This data is collected whenever fuel level is checked by the tank monitoring system, when an authorized user requests fuel, when a transaction occurs, and when inventories are taken. The focus is primarily on the fuel data obtained from the fuel tanks and devices installed to acquire fuel data, which is often prone to errors due to system failure, possibly resulting from delays in system maintenance and other contributing factors.

The proposed system collects fuel data from fuel station employees at the beginning of their shifts and throughout the execution of their various activities at their respective workplaces. These data were initially recorded manually in books, a practice that the proposed system will eliminate. The collected data will be used to validate and ensure the currency of all recorded information, facilitating data analysis to guarantee the provision of accurate data for effective decision-making.

3.1.2 Characterization of the current system

Current fuel management systems utilize data from fuel tanks and various tools available for data collection and report generation. This approach is effective when the system is operational. In contrast, the proposed system acquires data directly from fuel station personnel. The data collected from both sources is combined to create a comprehensive report and analysis, enhancing the effectiveness of decision-making.

During the process of requirement gathering, an evaluation of existing systems was conducted to identify their functionalities and assess areas where they lacked in terms of system usability and health. Additionally, visits to multiple fuel stations were made to analyze the systems in use and ascertain whether our proposed system was already implemented at any of those locations.

3.1.2.1 Functional Requirements

After analyzing existing systems, we incorporated a list of functionalities into the proposed system, defining it as a fuel management system. These functionalities include:

- Registration and user login
- Fuel monitoring
- Sales record keeping
- Inventory management
- Security and access control

- Data analysis and reporting
- Integration capabilities
- Remote monitoring
- Alert and notification features
- Historical data storage

These functionalities collectively contribute to the successful development of the application. Users of this system should be capable of registering and logging into the system, inputting the required data, receiving notifications regarding system events, and generating reports to facilitate effective decision-making.

3.1.2.2 Non-functional Requirements

- Performance
- Realiability
- Security
- Usability
- Scalability
- Mentainabilit
- Compatibility
- Performance under stress
- Logging and monitoring
- Interoperability
- Data mangement

The non-functional requirements play a crucial role in guaranteeing the overall success and quality of the software system. They shape the user experience, reliability, security, and performance characteristics that both users and stakeholders anticipate from the system.

3.1.3 Model of the existing system

Use Case Diagram

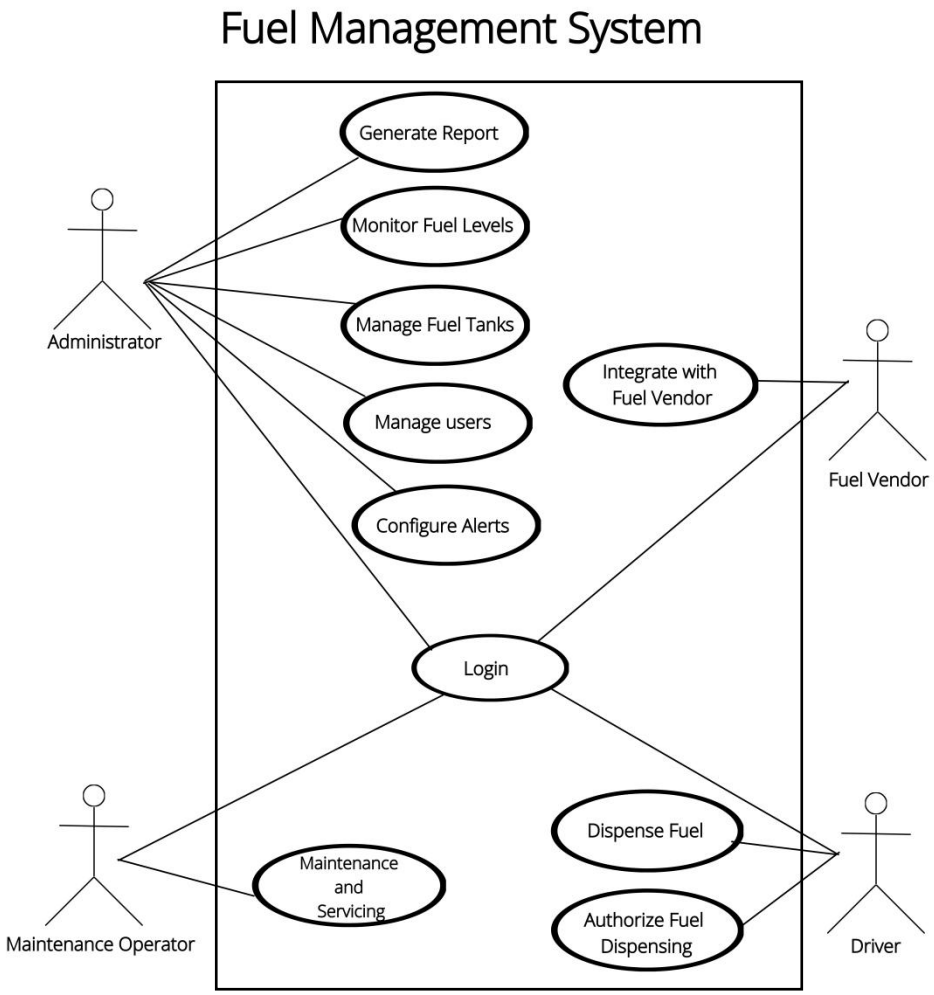


Figure 3.1: Use case diagram of the existing system.

At the start of this section, we discussed the functional and non-functional requirements of the proposed system and how it differs from other fuel management systems. The functional requirements encompass the primary system functionality and are crucial for determining the project's integration and lifespan success. The non-functional requirements focus on usability and application performance. The subsequent chapters will delve into the design aspect.

3.2 DESIGN

3.2.1 Introduction to design

At the beginning of this chapter, the functional and non-functional requirements of the proposed system were discussed, highlighting the distinctions from other fuel management systems. The functional requirements cover the core system functionality, essential for determining project integration and lifespan success. The non-functional requirements center on usability and application performance. The following chapters will explore the design aspect.

3.2.2 Model for the proposed system

Use Case Diagrams

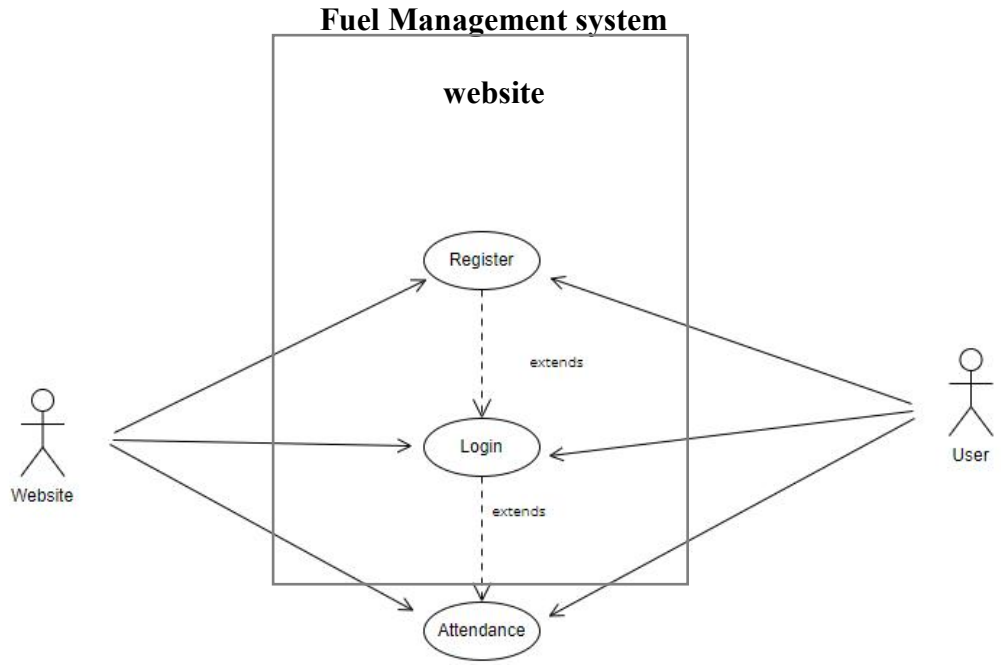


Figure 3.2: Use case diagram of the users

Use Case 1: Create an account

- Name: Registration
- Goal: Create a user account
- Actor: User
- Pre-conditions:
  - The user has an active connection to the website

• Steps:

– The user types in the username, password ,email, contact and position

– The system keeps all the details received in a database

• Post-conditions:

– The user's account is created

• Exception: N/A

Use Case 2: Login

• Name: Login

• Goal: Allows the user to get access to the system

• Actor: User

• Pre-conditions:

– The user has an active connection to the website

– The user has previously registered

– The user must know his or her username and password

• Steps:

–The user inputs their username and password for verification.

– the user is given a token after the verification process and its used as the users identity

• Post-conditions:

–The system has verified the user's username and password.

• Exception: The user enters an invalid username or password

Use Case 3: Attendance

• Name: Attendance

• Goal: To keep track of the users hours on the application

• Actor: User

• Pre-conditions:

  – User's username and password has been verified

• Steps:

  – The system pops out a dialogue box with a button

  – The user clicks the button

• Post-conditions:

  – User records and time stamps have been recorded

• Exception: User tries to cancel the dialogue box

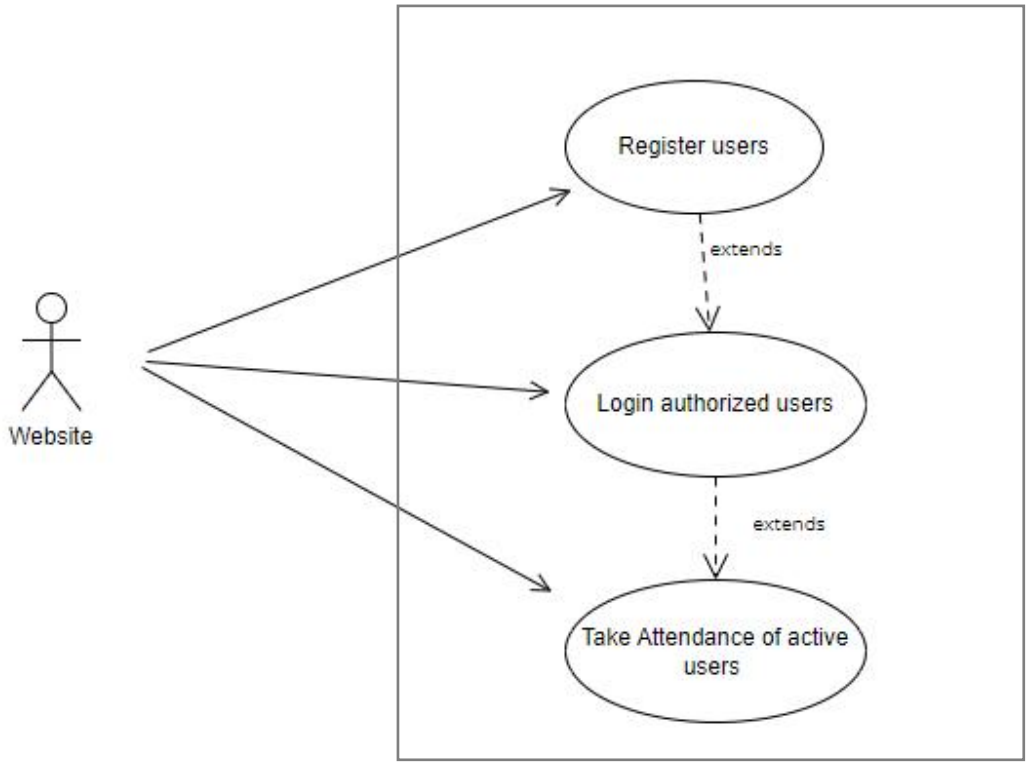


Figure 3.3: Use case diagram of the website

Website

Use Case 1: Register users

- Name: Register users
- Goal: Provides a user registration page to register new users
- Actor: Website
- Pre-conditions:
  - The user has an active connection to the website
  - The browser is compatible with the user's computer
- Steps:
  - Read HTML, CSS, and JavaScript files
  - Run python django files to make server operations
- Post-conditions:

– The page is successfully displayed and user request is saved

• Exception: 404 Not Found

Use Case 2: Login Authorized users

• Name: Login Authorized users

• Goal: Confirm the user trying to access the main system is an authorized user.

• Actor: Website

• Pre-conditions:

– User clicks the login page

– The page is successfully display

– User enters the username and password

• Steps:

– The login page is displayed after user clicks the login button

– User enters the username and password

– the website displays a success page when the login process is successful

• Post-conditions:

– The success page is displayed

• Exception: A try again notification is displayed when user enters wrong credentials.

Use Case 3: Take attendance of active users

• Name: Take attendance of active users

• Goal: Get users active hours on the system

• Actor: Website

• Pre-conditions:

– The user is successfully logged in

• Steps:

– The Attendance dialogue box is displayed

– The user clicks the button on the dialogue box

– The website records user details and timestamps in systems database

• Post-conditions:

– User's information is successfully saved in the database

• Exception:

– User tries to cancel the attendance dialogue box without clicking the attendance button

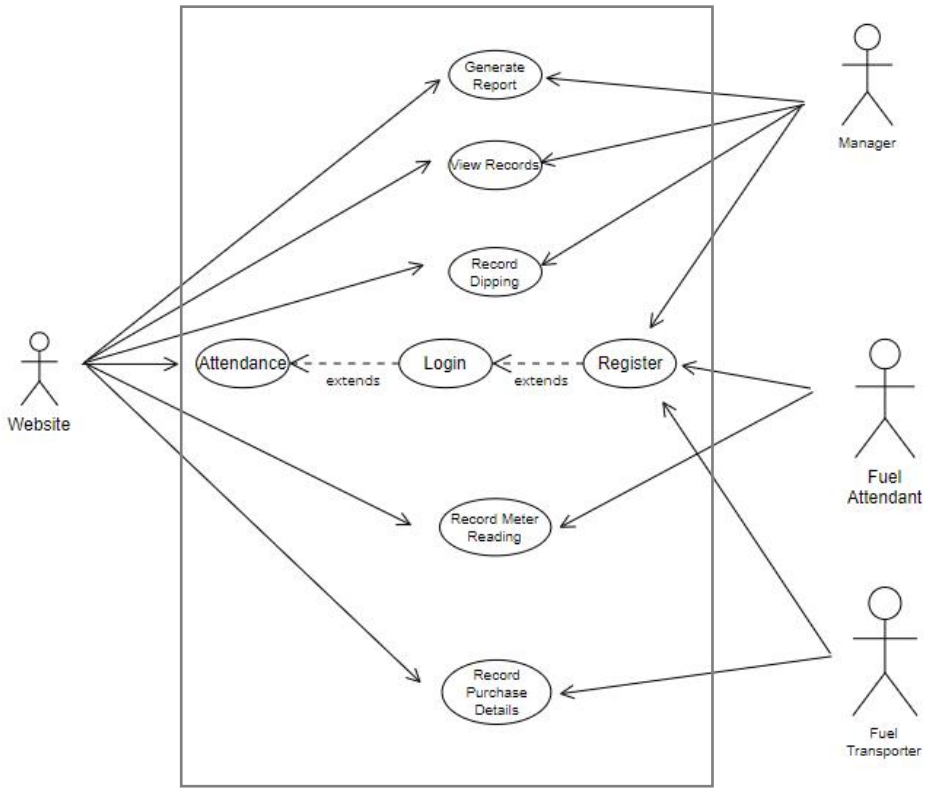


Figure 3.4: Use a case diagram of the whole system on a timely basis

3.3 Methodology

Continuing from the exploration of the Rapid Application Development (RAD) model in the previous chapter, this chapter now takes a closer look at the key components and processes that make RAD an effective approach for the project. The aim is to provide a clear understanding of how RAD's principles and methodologies are being leveraged to ensure the success and timely delivery of the proposed system.

❖ **Iterative Development**

An essential component of Rapid Application Development (RAD) is the iterative development methodology. This methodology breaks the project into smaller, more manageable components rather than attempting to complete the entire system at once. These parts are created in brief iterations or cycles, with each iteration building upon the previous one. This approach allows for continuous assessment and improvement of the system's functionality and features, ensuring they meet the evolving demands and expectations of end users.

❖ **Collaborative Work Environment**

Due to the small development team consisting of only two individuals, the collaborative aspect of RAD proves to be highly beneficial. Team members work closely together, facilitating a faster and more effective review process. With a smaller team, task organization, prompt feedback, and necessary modifications become simpler. The project progresses with precision and speed thanks to this cooperative synergy that enhances productivity.

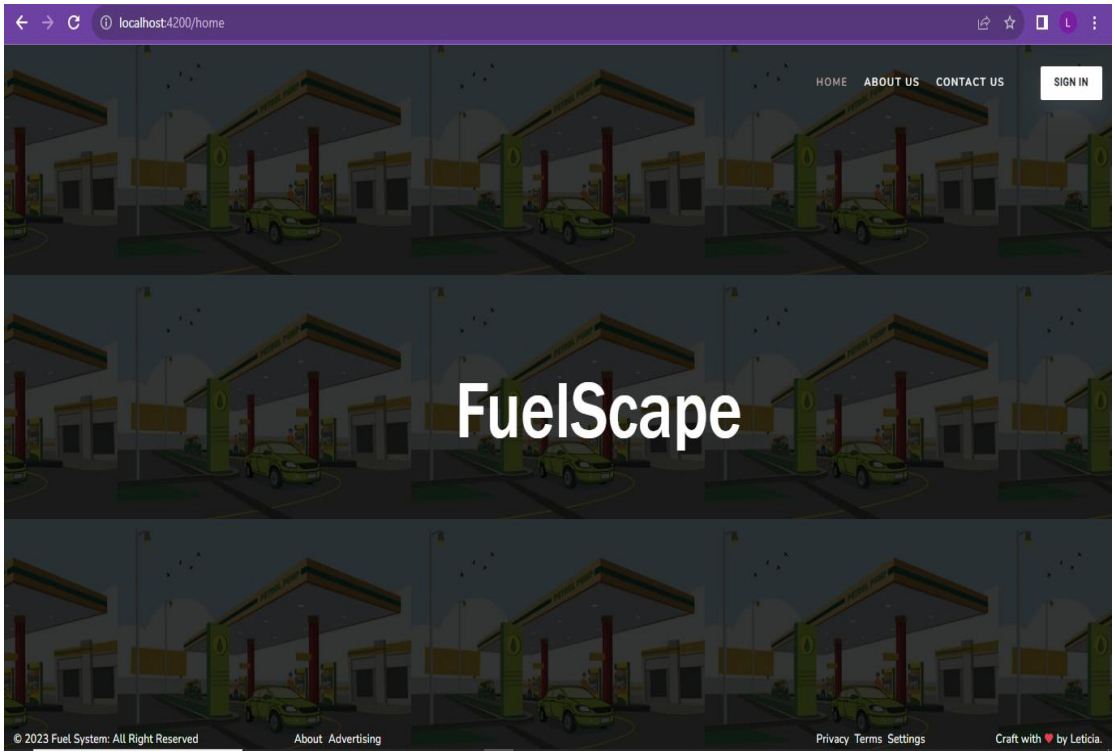
❖ **Embracing Change**

In the field of software development, project requirements often change over time. RAD fully acknowledges this fact, allowing changes in project requirements at any stage of the development process. This flexibility is crucial in keeping the system aligned with the project's dynamic requirements. The RAD methodology enables the incorporation of changes, such as new insights or stakeholder feedback, into the timeline without significant disruptions.

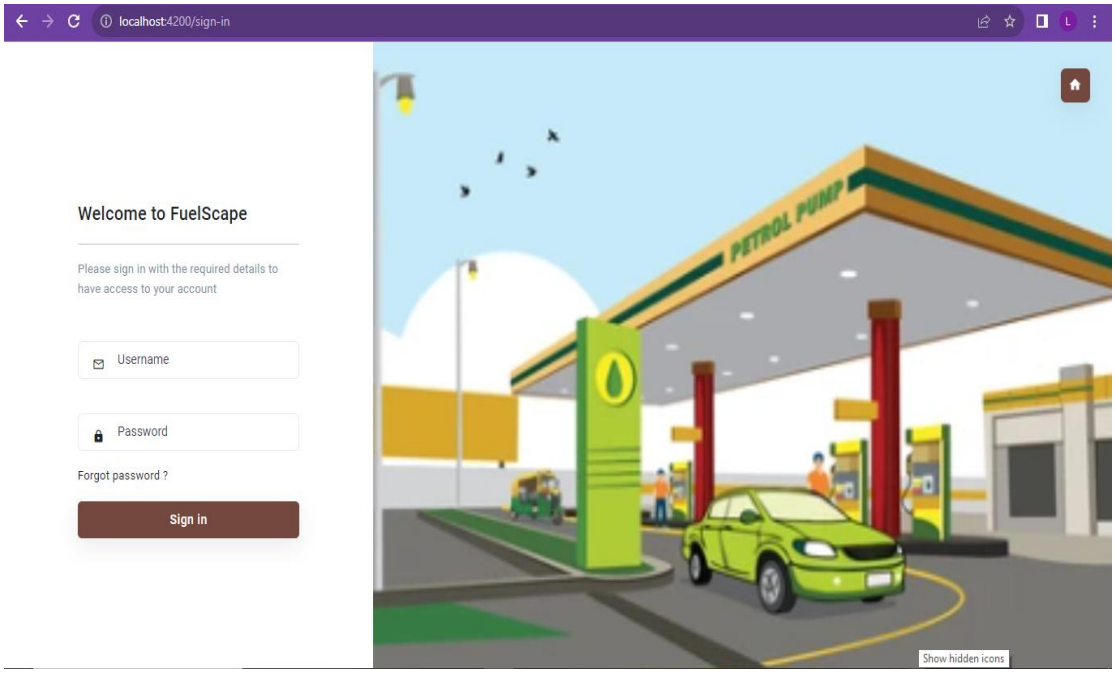
This chapter has covered the fundamental principles of RAD and how they are applied to the project's needs. By using an iterative approach, fostering a collaborative work environment within the small team, and embracing changes in project specifications, the project is set up for success in delivering the planned system on time and with high quality. The adaptability and effectiveness of the RAD model prove to be valuable assets as the complexities of the development process are navigated.

3.3.1 Prototypes

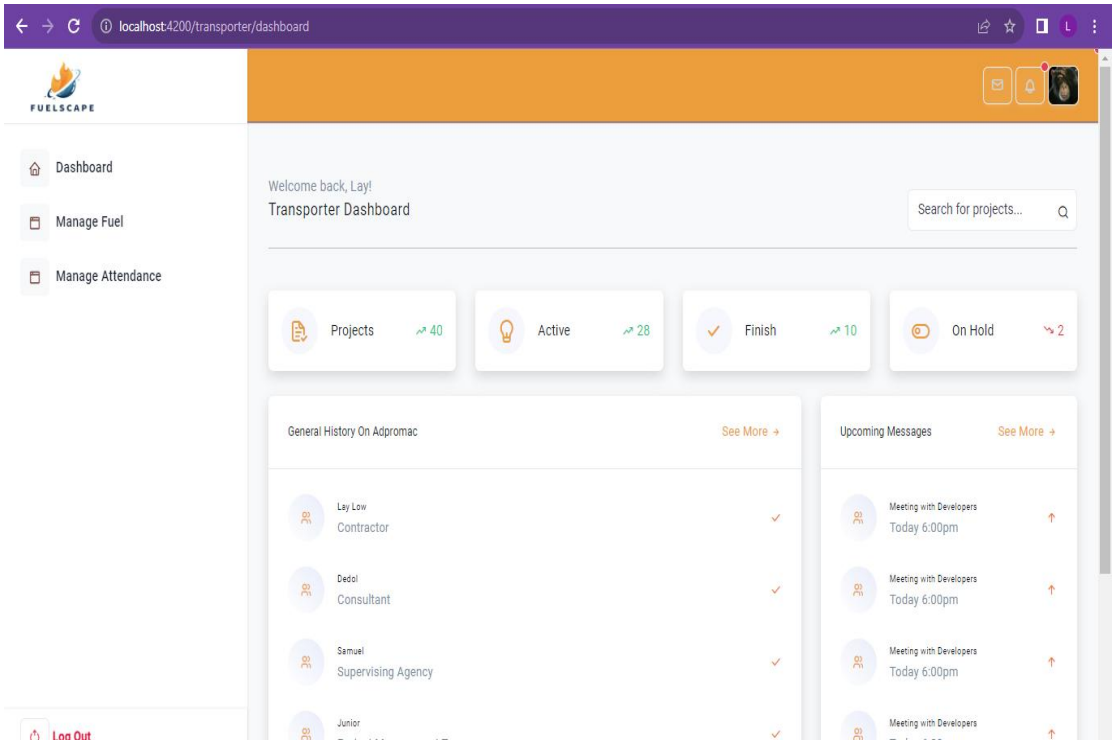
Landing page interface



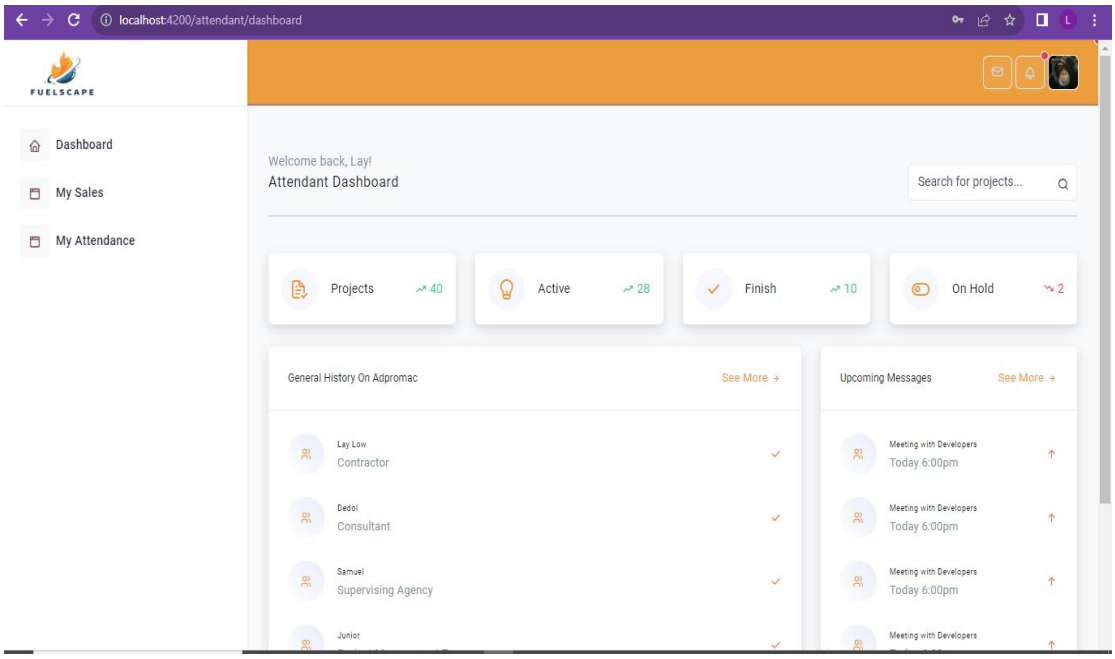
Sign in page



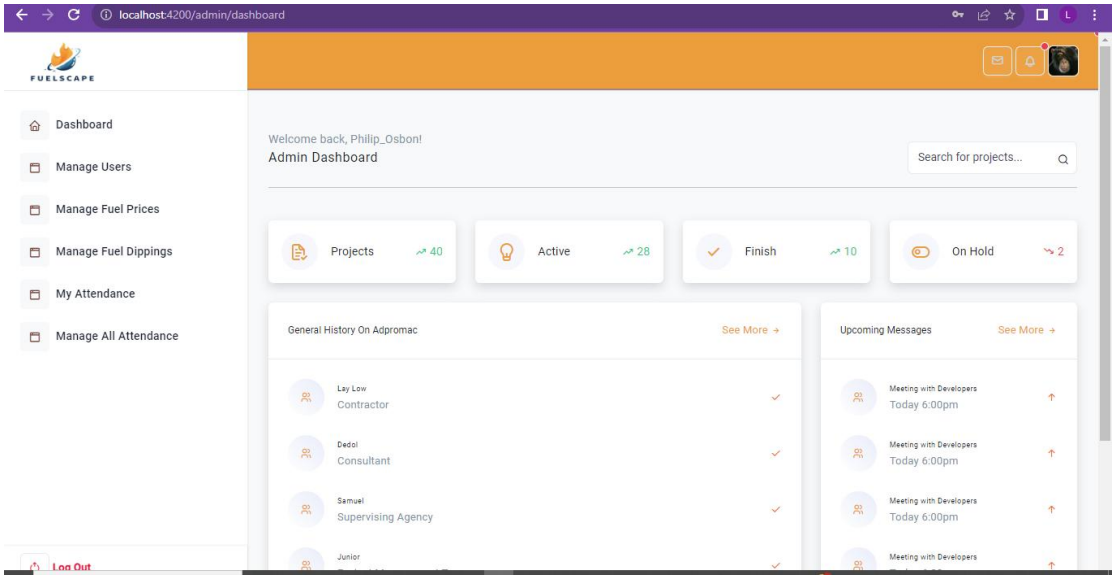
Fuel Transporter Dashboard



Fuel Attendant Dashboard



Manager Dashboard



3.4 Chapter summary

In this chapter, we have explored both the functional and non-functional requirements, delved into the models for both the current system and the proposed system, and outlined some of the prototype design for the proposed system.

Chapter 4

Design, Implementation, and Testing

In this chapter, the focus will be on the design, implementation, and testing phases of the project. These phases play a critical role in ensuring that the proposed solution is not only technically sound but also aligns with the project's objectives and success criteria.

4.1Design

4.1.2 Design Technique

The design technique used in project development is the Model-View-Controller (MVC) architectural pattern. The choice of this pattern is based on several reasons, which will be discussed below. A Model-View-Controller (MVC) architectural UML diagram of the proposed solution will be presented in the design development section, explaining the flow of project design development.

4.1.2.1 Justification of Design Technique

The selected design technique for this project is the Model-View-Controller (MVC) architectural pattern. The suitability of this choice is grounded in several reasons:

1. **Separation of Concerns**

MVC divides the application into three interconnected components, each with a distinct responsibility. This division fosters modularity, ease of maintenance, and promotes code re-usability.

2 Scalability

MVC permits scalability by enabling the addition or modification of components without affecting others. This feature is vital for accommodating future enhancements or alterations to the project.

3 User Interface (UI) and Business Logic Separation

MVC cleanly separates the UI from the underlying business logic. This separation simplifies development by allowing designers to focus on UI elements independently of developers working on the logic.

4 Testing

MVC facilitates effective testing, as each component can undergo separate testing. This ensures the early identification and resolution of issues during the development process.

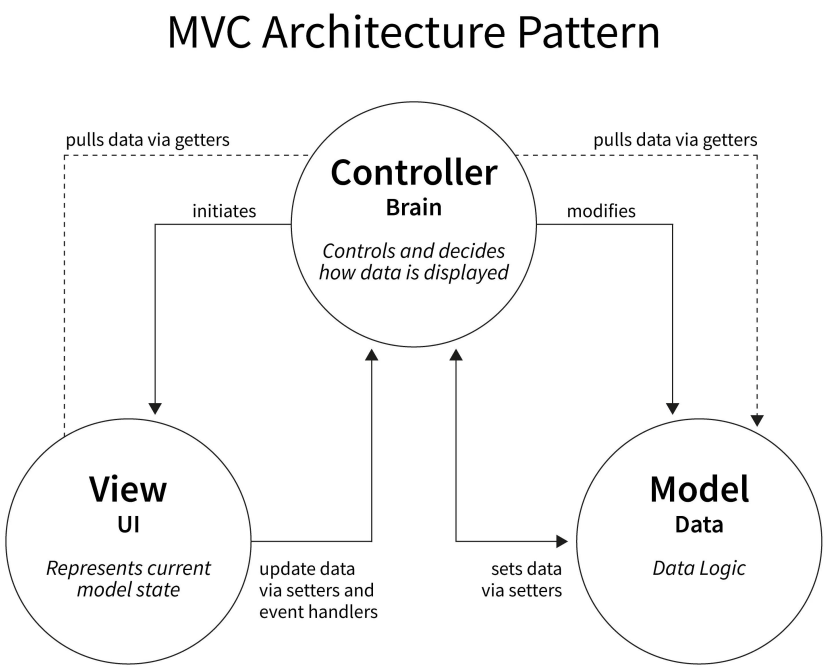


Figure 4.1: MVC Architectural Pattern

4.1.3 Design Development

In the design development process, detailed diagrams have been created using the Unified Modeling Language (UML) to represent the various components of the MVC pattern. These diagrams include state diagrams to model the behavior of specific components.

UML of proposed system MVC pattern

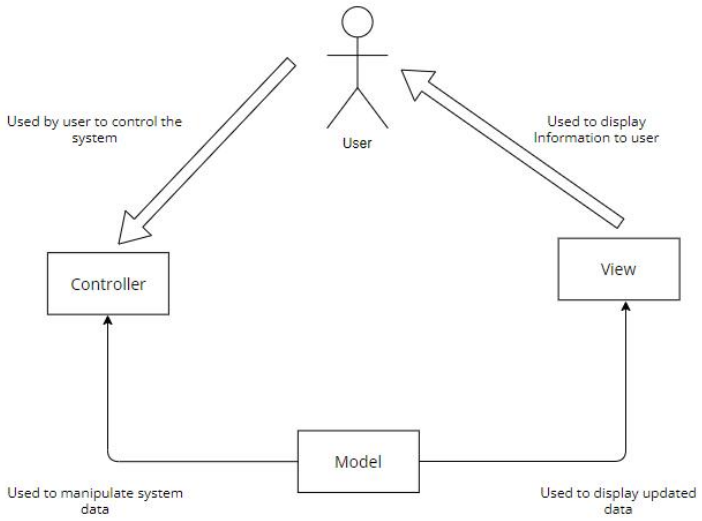


Figure 4.2: UML of the MVC architecture of the proposed system.

The diagram above depicts a user employing the controller component to manage and execute actions within the system. It also features a view component responsible for receiving information from the model component and presenting it to users. Simultaneously, the model component receives updated data from the controller and transmits it to the views for user display. Lastly, a controller component functions as an intermediary between the user and the database.

State diagram of proposed system

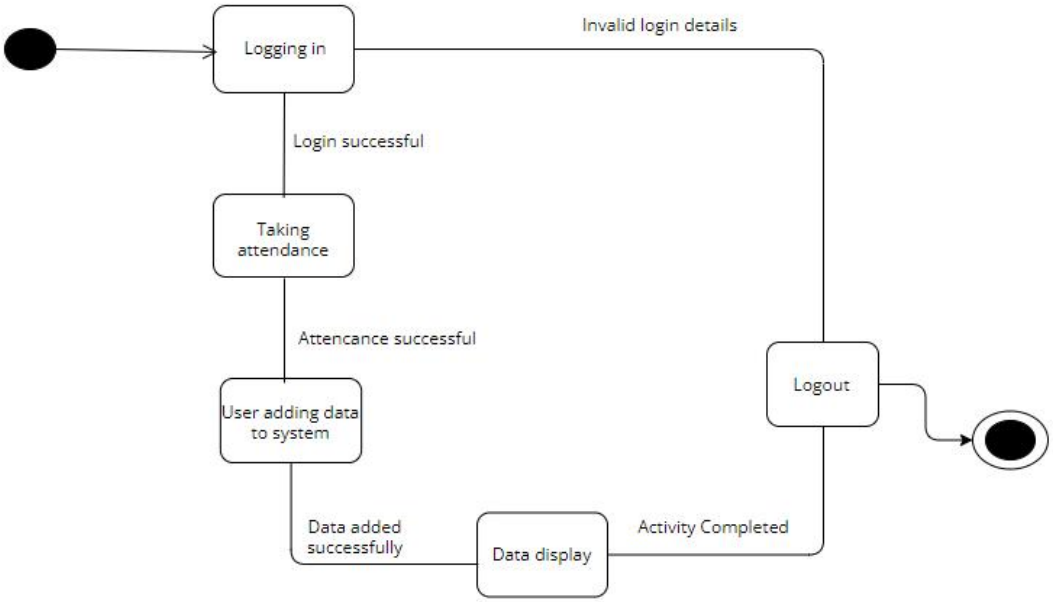


Figure 4.3: State diagram of the proposed system.

The state diagram describes the processes and behavior of general components in the proposed system. It consists of the login component, which directs users to the attendance component upon successful login and logs them out in case of invalid credentials. The attendance component, in turn, redirects users to their respective dashboards, contingent on their login details. Within the user dashboards, users can choose between viewing data or entering data, depending on their privileges. After users complete their role-associated activities, the display component showcases the required information on the diverse dashboards. Finally, users initiate logout through the logout component.

4.2 Implementation

The proposed solution was implemented using a variety of programming tools and languages discussed in the subsequent paragraphs, each with an associated reason for their selection. These tools are open-source and can be employed by anyone to implement any solution. The tools served as the mechanism of the proposed system and significantly contributed to the success of the project development process.

4.2.1 Technology Stack

The chosen technology stack for implementation comprises the following elements:

- **Programming Language - Python**

  Python stands out as a popular, versatile, and readable programming language that is renowned for its ease of use. Python's extensive library and framework support make it suitable for a wide array of applications. Python's simplicity and strong community backing render it an excellent choice for web development.

- **Web Framework - Django**

  Django, a high-level Python web framework, simplifies and expedites web development tasks. Django offers a robust toolkit and libraries for building web applications, encompassing features like user authentication, database management, and URL routing. Django's "batteries-included"

philosophy makes it a potent choice for swift and efficient web application creation.

- **Database Management System - PostgreSQL**

PostgreSQL serves as a robust and open-source relational database management system (RDBMS). PostgreSQL's reputation for reliability, scalability, and support for advanced data types and indexing makes it a stellar choice. PostgreSQL shines in applications demanding data integrity, intricate queries, and high-performance databases.

- **Front-end Technologies - HTML, CSS, JavaScript, and Angular**

Web page content gets organized using HTML (HyperText Markup Language), while CSS (Cascading Style Sheets) takes care of styling and layout to enhance the aesthetic appeal of the user interface. JavaScript, a programming language, introduces interactivity and dynamic behavior, handling user interactions, input validation, and content updates without necessitating full page reloads. Angular is an integral part of the front-end technology stack, contributing to the overall development.

- **Version Control - Git**

Git, a distributed version control system, is widely employed for tracking source code changes and fostering collaborative development. Git empowers developers to work on distinct branches, merge alterations, and maintain an accurate history of code revisions. Git facilitates collaboration and safeguards code integrity.

❖ **Development Environment - Visual Studio Code**

Visual Studio Code (VS Code), a favored and free code editor from Microsoft, offers a plethora of extensions and integration that augment the development experience. Supporting multiple programming languages, featuring built-in Git integration, and furnishing debugging tools, VS Code proves itself as a stellar choice for web development.

By selecting this technology stack, a comprehensive set of tools and technologies is adopted. This amalgamation of tools is meticulously chosen to harmonize with each other, streamlining the development process, reinforcing robustness, and ensuring scalability for the web application. The stack leverages the strengths of Python, Django, PostgreSQL, and a multitude of other technologies, culminating in the creation of a potent and efficient web development environment.

4.2.2 Converting Design to Working System

The process of converting the design into a functional system followed a structured sequence. The implementation adhered to the MVC pattern, albeit incorporating certain adjustments to meet specific requirements and enhance performance. The following stages outline the project's progress during implementation:

1. **Design Transformation**

The design, comprising wire frames, UI mock ups, and system architecture plans, constituted the basis for system development.

2. **MVC Pattern Compliance**

The Model-View-Controller (MVC) pattern was consistently maintained throughout development:

- **Model:** Responsible for managing data and business logic.
- **View:** Focused on presentation and the user interface.
- **Controller:** Orchestrated user input and facilitated communication between Model and View.

3. **Tailored Modifications**

Minor adaptations were introduced to accommodate evolving project requirements. These adjustments addressed changing needs, potentially influencing data models, UI elements, or feature functionality.

4. **Performance Enhancement**

Implementation efforts prioritized system performance optimization. This encompassed refining database queries, implementing caching strategies, and enhancing code to bolster speed and efficiency.

4.2.3 Successful Implementation and Challenges

During the implementation of the project, several key features were effectively realized:

4.2.3.1 Successful Implementation of Key Features

- **User Registration and Authentication**

  This feature enables account creation and secure user authentication to access the system. It involves managing user data and encrypting passwords for data privacy.

- **Data Storage and Retrieval from the Database**

  Efficient data management includes storing fuel monitoring, sales, inventory, and user information in a database. Retrieval ensures swift access to this data for various functionalities.

- **Integration with External APIs**

  Integrating external APIs enhances the application's capabilities by enabling interactions with third-party services, such as fetching fuel price data, weather information, or interfacing with payment gateways.

- **Real-time Notifications using WebSockets**

  Implementing real-time notifications elevates the user experience. Through WebSockets, the application can provide instant

updates to users, such as low fuel alerts, new sales, or system status changes.

4.2.3.2 Challenges Faced During Implementation

1. **Optimizing Database Queries for Improved Performance**

   Database query optimization involves identifying and fine-tuning slow-performing queries to enhance efficiency.

2. **Ensuring Data Security and Privacy Compliance**

   Challenges in this aspect include implementing encryption, access controls, and adhering to data protection regulations to safeguard sensitive user and transaction data.

3. **Managing Concurrent User Sessions for Scalability**

   Scalability is crucial, especially as the user base expands. Managing concurrent user sessions effectively, particularly during peak usage, presents challenges that may require load balancing, session management, and infrastructure scaling.

4.2.3.3 Strategies for Overcoming Challenges

1. **Collaboration**

   Effective collaboration among team members with diverse expertise facilitates problem-solving. Regular communication and brainstorming sessions contribute to finding innovative solutions.

2. **Code Reviews**

   Thorough code reviews ensure optimized, secure, and best-practice-compliant code. Code reviews identify potential performance bottlenecks and security vulnerabilities early in the development process.

3. **Iterative Development**

   Adopting an iterative development approach enables continuous improvement. Developers can revisit and optimize code and database queries based on real-world usage patterns and challenges encountered.

4. **Expertise and Research**

   Leveraging team members' expertise and conducting research on best practices in database optimization, security, and scalability provide valuable insights for overcoming challenges.

In summary, achieving the successful implementation of these key features requires technical competence, collaborative effort, and adaptability. The challenges encountered during development can be surmounted through teamwork, meticulous planning, and a commitment to continual enhancement. This approach ensures that

the fuel management and monitoring application delivers a robust, secure, and efficient user experience.

4.3 System Documentation

4.3.1 User Manual

The User Manual serves as a comprehensive document designed to assist users in understanding and effectively using the system. It was meticulously created as part of the project's documentation efforts, reflecting a dedication to user-friendliness and support.

4.3.1.1 Content and Features of the User Manual

1. **Step-by-Step Instructions**

   The User Manual comprises detailed, step-by-step instructions on performing various tasks within the system. These instructions are presented in a clear and concise manner, ensuring accessibility for users with varying levels of technical expertise.

2. **Online Accessibility**

   Being available online ensures easy access to the manual from any location with an internet connection. This accessibility proves particularly valuable for remote or distributed user bases.

3. **Annotated Screenshots**

Annotated screenshots are interspersed throughout the manual, providing users with visual references to facilitate comprehension of instructions and navigation within the application.

4. **User Assistance**

The User Manual acts as a valuable resource for users encountering challenges or questions while using the system. It offers a self-help option that can reduce the necessity for direct support or help desk inquiries.

5. **Improved User Experience**

By offering a comprehensive User Manual, the project team demonstrates its commitment to enhancing the overall user experience. Users are more likely to feel confident and comfortable using the system when they have access to clear, well-structured documentation.

6. **Reduced Training Time**

For organizations or teams implementing the system, the User Manual can significantly diminish the time and resources required for training. New users can independently refer to the manual as they learn to use the application.

7. **Continuous Updates**

To ensure the User Manual remains pertinent, it is crucial to maintain and update it as the system evolves. Incorporating new features, reflecting changes in the user interface, and providing updates keeps users well-informed.

In summary, the User Manual constitutes an indispensable element of system documentation. It empowers users by furnishing the guidance necessary to navigate and utilize the system effectively. The inclusion of annotated screenshots and online accessibility elevates its usability, contributing to a positive user experience and alleviating the burden on support channels.

4.3.2 Administrator Information

This section presents administrators with detailed information regarding their role and responsibilities within the system. Administrators play a pivotal role in managing and maintaining the system to guarantee its seamless operation.

roles

- **User Account Management**

Responsibilities encompass managing user accounts within the system. Tasks entail creating new accounts, modifying user permissions, resetting passwords, and deactivating or deleting accounts when necessary. The documentation elucidates how to execute these actions through the system's administrative interface.

- **System Performance Monitoring**

To uphold optimal system performance, regular system monitoring is imperative. The documentation delineates procedures for accessing and comprehending performance metrics and logs. It also furnishes guidance on recognizing and rectifying performance issues or bottlenecks.

❖ **Data Security and Privacy**

Upholding data security and privacy remains of utmost importance. This section proffers best practices for securing data, implementing access controls, and conducting security audits. Additionally, it imparts information regarding the management of data breaches or security incidents.

❖ **Routine Maintenance Tasks**

Reliability hinges on systematic system maintenance. This documentation expounds upon routine maintenance responsibilities, such as software updates, database backups, and server maintenance. It may encompass scripts or procedures for automating these tasks.

❖ **Troubleshooting and Issue Resolution**

Administrators must be equipped to address issues and troubleshoot problems adeptly. This section furnishes guidance for diagnosing and resolving common issues, encompassing error messages, system crashes, or performance hitches.

❖ **User Support and Communication**

Administrators frequently serve as the primary point of contact for user support. The documentation incorporates directives for effective user communication, management of support requests, and maintenance of a support ticketing system.

- **Disaster Recovery and Backup Procedures**

In the event of system failures or data loss, administrators must be primed with disaster recovery and backup procedures. This section delineates backup strategies, data recovery processes, and disaster recovery blueprints.

- **Software Updates and Upgrades**

Sustaining system currency is vital for security and functionality. The documentation imparts guidance on performing software updates and upgrading the system to newer iterations.

- **Documentation Updates and Resources**

To ensure administrators have access to the most current information, the documentation encompasses instructions on updating the documentation as the system evolves. Furthermore, it may proffer links to supplementary resources, forums, or support channels.

This administrator documentation stands as an indispensable resource, equipping administrators to proficiently oversee and sustain the system, assuring its unwavering reliability and functionality. Consistent documentation updates are requisite to adapt to system modifications and tackle emerging challenges.

4.3.3 Technical Support Information

This section addresses the inclusion of technical support information in the documentation of the proposed system. It stands as a fundamental element of user guidance, facilitating self-sufficiency in troubleshooting typical issues. Simultaneously, it offers accessible support resources for instances requiring intervention. This methodology reduces dependency on direct support channels, thereby streamlining issue resolution. Furthermore, it institutes a feedback loop through support interactions. This, in turn, informs ongoing system enhancements, culminating in an elevated user experience and heightened satisfaction.

4.4 Testing Strategy

The testing approach followed a multifaceted strategy, incorporating various methods. Functional testing, employing both black-box and white-box techniques, thoroughly scrutinized individual components and the comprehensive functionality of the system. Furthermore, usability testing, represented as user acceptance testing (UAT), assessed the system's user-friendliness, with user feedback assuming a pivotal role in the refinement of the ultimate release.

4.4.1 Testing Methodology

This section delineates the comprehensive testing methodology, encompassing various approaches to ensure the system's robustness and usability. Multiple testing methods were employed to thoroughly assess different aspects of performance and user-friendliness.

- **Functional Testing**

Functional testing is an integral part of the testing strategy, focusing on evaluating the functionality of both individual system components and the system as a whole.

Approaches

Black-box Testing

In black-box testing, the internal structure of the components remains unknown to the tester. This method enabled evaluation of the system's functionality from an end-user perspective, ensuring it performs as expected and meets functional requirements.

White-box Testing

White-box testing delves into the internal logic and structure of the system. It aids in identifying issues related to code quality, data flow, and algorithms, ensuring the correctness of the underlying system architecture.

- **Usability Testing**

Usability testing, particularly in the form of User Acceptance Testing (UAT), plays a pivotal role in assessing the system's ease of use and overall user experience.

Approach

User Acceptance Testing (UAT)

UAT involves real end-users interacting with the system to validate its usability. During this phase, valuable user feedback was collected, issues were addressed, and improvements were made based on their input. This iterative process ensured that the system met user expectations and provided an optimal user experience.

The testing methodology is designed to comprehensively evaluate the functionality and usability of the system. Through a combination of black-box and white-box functional testing, it verifies that the system meets functional requirements while also ensuring the robustness of the underlying architecture. User Acceptance Testing (UAT) assumes a pivotal role in assessing the system's user-friendliness, with user feedback guiding the refinement process to deliver an exceptional final release.

4.4.2 Evaluation Against Objectives

This section underscores the pivotal role of the testing phase in evaluating the system's alignment with its predefined objectives and success criteria. Throughout this phase, defects and issues identified during testing underwent systematic resolution, guaranteeing the development of a final product of superior quality. The evaluation process centered on the project's objectives, encompassing facets like functionality, performance, security, and usability.

Emphasis was notably placed on defect resolution, with defects and issues classified, prioritized, and subjected to a structured resolution process. This process entailed comprehensive analysis, debugging, rectification, and subsequent retesting. The testing phase followed an iterative approach, permitting continuous refinement and enhancement of the system. Swift defect resolution and the integration of user feedback were pivotal in achieving the desired quality and functionality.

In conclusion, the precise execution of the design, implementation, and testing phases culminated in the development of a robust and user-friendly system that adeptly fulfilled the project's predefined objectives and success criteria. The adoption of the MVC architectural pattern, the selection of suitable technologies, and the comprehensive documentation emerged as fundamental contributors to the overall project success. This section underscores the paramount importance of the testing phase in ensuring the system's alignment with project objectives and the delivery of a high-caliber solution.

Chapter 5

RESULT AND DISCUSSION

This chapter serves as a platform for in-depth discussion and analysis, showcasing the results of the project. The main goal of this chapter is to present the discoveries that have come to light throughout the course of the project. To determine how well the original project objectives have been achieved, we will examine the accomplishments and goals made during this endeavor.

Difficulties or shortfalls experienced will be freely admitted since the belief is that the journey is just as important as the final result. Transparency is a fundamental element in this chapter. Additionally, this chapter will discuss the aspects of the work that haven't been finished yet because of limitations or unanticipated complexities. It will also cover new research and development directions inspired by the project's results. By doing this, the groundwork is created for a thorough analysis of the importance of the initiative and its implications for the fuel management industry.

5.1 Findings

In this section, we explore the essence of our project, offering a comprehensive account of the findings arising from our work on the Fuel Management System. This segment encompasses not only the anticipated results, such as system performance metrics and fuel consumption data but also unexpected discoveries that emerged as serendipitous byproducts of our exploration. The primary objective here is to present a detailed overview of these findings, shedding light on

the achievements, surprises, and vital insights acquired throughout the project's life cycle. These findings form the basis of our project, providing a deeper understanding of our accomplishments and paving the way for potential future developments in the field of fuel management and related domains.

Some of the findings are as follows

1. System Performance Insights

Delving into the Fuel Management System's performance, we conduct a detailed analysis covering response times, illustrating the system's speed in responding to user input, efficiency improvements indicating enhanced operations and fuel management, and error rates highlighting the system's reliability and robustness.

2. User Feedback

Users and stakeholders provide invaluable feedback, which is not only cataloged but also deeply considered as it shapes the trajectory of system enhancements and informs the ongoing commitment to user-centered design.

3. Data Accuracy Assurance

The integrity of fuel-related data is paramount. Within this segment, measures and protocols employed to ensure data accuracy are delineated, building a foundation of trust and reliability for the entire system.

4. Unexpected Bonuses

Beyond core objectives, the system has brought forth unanticipated advantages. These serendipitous benefits, such as process streamlining and fleet management enhancements, serve as pleasant surprises and enrich the project's narrative.

5. Learning from Challenges

Commitment to transparency compels candid sharing of challenges encountered during the project, providing valuable insights into areas necessitating further attention and improvement, contributing to a culture of continuous enhancement.

6. User Adaptation Insights

The adaptation of users to the system is a focal point of this section. Observations and insights gained in this regard hold the key to the development of tailored user training programs and support initiatives, ensuring a seamless user experience.

This thorough investigation of findings has exposed a complex tapestry of facts defining the journey in building the Fuel Management System. These findings encompass user feedback, surprise bonuses, and system performance insights, creating a mosaic of successes, difficulties, and unexpected advantages. The results confirm the project's value and set the stage for future development. As we conclude this section, the focus shifts to the conclusions and lessons derived from these

discoveries, laying the groundwork for a lively debate that will guide progress in the field of fuel management and beyond.

5.2 Goals Achieved

In this section, a comprehensive analysis of the project's performance in relation to the initial goals is presented. The extent to which the planned objectives were achieved is evaluated, emphasizing that the success of a project is determined by how well it fulfills its intended aims, rather than simply reaching completion. The assessment methodically breaks down each goal, highlighting the progress made and demonstrating the effectiveness of procedures and efforts. Both areas where initial expectations were exceeded and those where objectives were partially or fully met are explored transparently. Commitment to transparency is evident as any shortcomings in goal achievement are openly acknowledged, showcasing dedication to continuous improvement.

5.2.1 Main Goals Achieved

1. User-Centric Design and Enhancement

Commitment to user-centric design has been realized through the integration of user feedback into the system design. This accomplishment has also led to substantial enhancements in the user experience, with users directly contributing to the system's evolution.

2. Data Accuracy Assurance

Ensuring data accuracy was a non-negotiable objective, diligently assured through rigorous protocols. This goal underpins the trustworthiness and reliability of the system, standing as a dependable tool for data management and analysis.

3. Unanticipated Benefits and Value Addition

The project yielded unexpected benefits, surpassing primary objectives. It streamlined operational processes and enhanced fleet management, adding significant value beyond the project's core purpose. These unanticipated advantages have become integral aspects of the project's success.

4. Transparency and Open Reporting of Challenges

The project's transparency is evident in openly reporting encountered challenges. Acknowledging these challenges underscores dedication to a culture of learning and continuous improvement. This openness ensures a realistic perspective, emphasizing the importance of both acknowledging challenges and celebrating achievements.

5. User Adaptation Insights and Tailored User Experience

Observations and insights gained from user adaptation have been thoroughly documented. These insights directly influence the development of tailored user training programs and support initiatives, ensuring a seamless and user-friendly experience for all users.

6. Identification of Future Research and Development Areas

Inspired by the project's findings, potential areas for future research and development have been identified. These areas promise further exploration and innovation, indicating the project's forward-thinking approach and its potential for future advancements in the field.

7. Valuable Directions for Future Attention and Improvement

Specific aspects of the project that remain unfinished have been pinpointed, providing valuable directions for future attention and improvement. This proactive identification of areas for enhancement ensures that the project continues to evolve and meet evolving needs.

In summary, the achievements outlined here go beyond mere accomplishments; they represent the markers of a journey that combines objectives met and objectives set, affirming not only the project's success but also the importance of the defined processes that have shaped the project's trajectory.

5.2.2 Further work

After concluding our assessment of the project's successes, we turn our attention to the future prospects. The voyage thus far has shown not only our successes but also brand-new avenues that beg to be explored. In this section, we focus on "Further Work," addressing two crucial aspects: the new lines of inquiry motivated by the project's advancements and the areas of our work that are still

unexplored because of time limits and unforeseen difficulties. This section's main idea serves as a beginning rather than a conclusion since it expresses our attention to the developing field of fuel management and our commitment to constant advancement, innovation, and improvement.

One of the most exciting possibilities as we look to the future is the integration of the proposed system with our current IoT fuel management infrastructure. A robust and harmonious environment that actively pursues real-time data collecting, predictive analytic, and improved user experiences is what this strategic integration intends to build.

1. Real-Time Data Fusion

The proposed system will be seamlessly integrated with our current IoT fuel management system during the integration phase. This, along with the improved data capture capabilities of the proposed system, will allow for the synchronous collecting of data from IoT sensors on our vehicles and other pertinent sources. A complete and real-time data stream that gives us a better understanding of fuel usage and vehicle health will be the end result.

2. Predictive Analytics Unleashed

We'll realize the full potential of predictive analytics by combining the capabilities of the two systems. The integrated data pool will make it easier to build complex models that assess past trends as well as predict future ones.

This will fundamentally alter our capacity to anticipate and avert possible problems, maximize fuel efficiency, and support proactive decision-making.

3. User Experience Transformation

User engagement will be redefined by our integrated system. A streamlined interface that provides a comprehensive picture of fuel management insights will be available to users. Intuitive dashboards, immediate alerts, and predictive recommendations will further improve the user experience by simplifying data interpretation and decision-making.

4. Efficiency and Cost Savings

More than merely a technological development, the synergy between these systems is a workable plan for increased effectiveness and cost savings. We can reduce fuel use and streamline maintenance procedures using real-time data and predictive capabilities, leading to considerable cost savings and operational efficiencies.

5. Future-Ready Architecture

Future-focused design is evident in this integration. The architecture is flexible, expandable, and prepared to support new developments and developing fuel management technologies. It places us at the vanguard of innovation, prepared to take on new opportunities and challenges.

In essence, the proposed system's integration with our current IoT fuel management infrastructure represents a huge step forward in our pursuit of excellence. It involves more than just integrating technology; it involves promoting an innovative, efficient, and user-centric culture. This next study exemplifies our steadfast dedication to maintaining our leadership position in the field of fuel management, where the fusion of systems opens the way for a more intelligent, economical, and sustainable future.

Chapter 6

CONCLUSION

Throughout this study, our exploration of fuel management systems has led to invaluable insights and a deepened understanding of this critical field. Through extensive research, data collection, analysis, and discussions, we have delved into the complex world of fuel management. This final chapter serves as a succinct summary of the most crucial lessons learned, bringing together all the pieces of our research and offering a comprehensive overview of the outcomes.

6.1 Recap of Key Findings

Our project commenced with a comprehensive investigation into system performance, user feedback, data accuracy assurance, and unanticipated benefits. We meticulously examined response times, efficiency improvements, and error rates in the context of system performance, leading to modifications that enhanced the user experience and fuel management. User feedback and stakeholder involvement shaped the project's direction, with user insights playing a pivotal role. Ensuring data accuracy was a non-negotiable goal, and rigorous procedures were implemented to guarantee dependable data. Beyond our primary objectives, the project yielded delightful surprises, such as improved fleet management and process optimization. Openly acknowledging the challenges we faced underscored our commitment to continuous improvement. Insights from user adaptation informed the creation of tailored user training initiatives, contributing to a user-centric approach and highlighting the project's challenges, successes, and unexpected benefits.

6.2 Goals Achieved

When we compare the project's outcome to its initial goals, notable successes become evident. Our unwavering commitment to user-centered design and enhancement significantly improved the user experience, thanks to the seamless integration of user feedback. This addition of user insights enhanced the usability of our Fuel Management System and reaffirmed our user-centric approach. Our stringent data quality standards laid the foundation for data reliability and system dependability, providing users with confidence in the accuracy of their information.

Additionally, the project surpassed its initial objectives by delivering unexpected benefits, including optimized fleet management and streamlined operating procedures. By openly acknowledging the challenges we encountered, we fostered a culture of continuous learning and development. Insights from user adaptation have shaped tailored user training programs, ensuring a smooth and intuitive user experience. The project's findings have also sparked the discovery of potential areas for further research and development, reflecting our commitment to innovation and ongoing progress. These achievements underscore the initiative's success and its potential for further advancement in the field of fuel management.

6.3 Further Work

Looking to the future, a world of exciting possibilities unfolds before us. Among the most intriguing prospects is the seamless integration of our proposed system with our existing IoT fuel management infrastructure. This strategic fusion of technologies holds the key to real-time data integration, predictive analytics, and a

transformative user experience. With significant cost savings and unprecedented efficiency levels, it establishes an architecture ready for the future's demands.

Essentially, the integration of the proposed system with our existing IoT fuel management infrastructure represents a substantial leap forward, exemplifying our commitment to efficiency, innovation, and, above all, a user-centric approach. This innovative project signifies our dedication to maintaining our position as industry experts in fuel management. The convergence of these systems not only marks a technological breakthrough but also opens the door to a smarter, more resourceful, and sustainable future. It is evidence of our unwavering commitment to pushing boundaries and shaping a more promising and prosperous future for the fuel management industry.

In conclusion, this final chapter pays tribute to our significant achievements while emphasizing the ongoing need for research and development in the ever-changing field of fuel management. It underscores our unwavering commitment to advancement and innovation as we strive to meet the ever-evolving demands and challenges of this fast-paced industry. As we conclude this chapter, we eagerly anticipate the opportunities, innovations, and inevitable challenges that lie ahead. We recognize that our journey is far from over, and we stand ready to continue making significant contributions to the field of fuel management with a sense of purpose and enthusiasm.

Reference

1. Fuel-management systems. (2023, March 17). Wikipedia. https://en.wikipedia.org/wiki/Fuel-management_systems

2 Chien, C. (2020, February 4). What is Rapid Application Development (RAD)? Codebots. https://codebots.com/app-development/what-is-rapid-application-development-rad

3 J. (2023, August 7). Fuel Management System | Real-Time Tank Gauging & Security. Banlaw. https://www.banlaw.com/fuel-management-systems/

4 J. (2022, June 21). Fuel Inventory Management & Tracking Software - ResTrack | Banlaw. Banlaw. https://www.banlaw.com/fuel-management-systems/fuel-management-system-software/

5 Llc, T. (n.d.). NaftaPOS software for petrol pump - Technotrade LLC. https://www.technotrade.ua/petrol-station-software.html?gclid=CjwKCAjwrranBhAEEiwAzbhNtXDv0Onuzd2QAcx2fiYuREj0g5U4NZnHbsctldzXKcL_JeEkyFnCThoC7bEQAvD_BwE

6."Rapid Application Development." Creatio. https://www.creatio.com/rapid-application-development

Appendices

Front end codes

Home componet

```
<app-website-navbar></app-website-navbar>

<!-- Hero Start -->
<section class="bg-half-260 bg-primary d-table w-100"
   style="background:
url('../../../../assets/images/fuelProject/station.png') center center; background-
size:auto;">

   <div class="bg-overlay"></div>
   <div class="container">
      <h1 style="color: white; position: relative; left: 50px; text-align:
center; font-size: 5rem; top: 5rem;font-family:'Franklin Gothic Medium',
'Arial Narrow', Arial, sans-serif">FuelScape</h1>
      <div class="row">
        <div class="col-12">
          <div class="title-heading text-center mt-4">
            <img src="../../../../assets/images/logo/logo"
              height="50"  class="small" alt>
            <div
```

```
                    class="text-center subcribe-form mt-4 pt-2 animated
fadeInUpBig animated.delay-5s">

                    </div>
                    <div class="mt-4 pt-2">
                    </div>
                  </div>
                </div>
              </div>
            </section>
            <div class="footer-py-30 footer-bar">

              <div class="container-fluid text-center">
                <div class="row align-items-center">
                  <div class="col-sm-3">
                    <div class="text-sm-start">
                      <p class="mb-0 text-white footerText">© 2023
                        Fuel System: All Right Reserved</p>
                    </div>
                  </div>
                  <div class="col-sm-4">
                    <div class="text-sm-start">
                      <p class="mb-0 text-white footerText">
                        <a [routerLink]="(['/#!'])"
```

```
            class="text-white ml-2 privacy">About </a>
          <a [routerLink]="(['/#!'])"
            class="text-white ml-2 terms"
            data-bs-toggle="modal"
            data-bs-target="#LoginForm">Advertising
          </a>
        </p>

    </div>
</div>
<div class="col-sm-3">
    <div class="text-sm-end">
        <p class="mb-0 text-white footerText">
          <a [routerLink]="(['/#!'])"
            class="text-white ml-2 privacy">Privacy </a>
          <a [routerLink]="(['/#!'])"
            class="text-white ml-2 terms">Terms </a>
          <a [routerLink]="(['/#!'])"
            class="text-white ml-2 settings">Settings
          </a>
        </p>
    </div>
</div>
<div class="col-sm-2">
    <div class="text-sm-end">
```

```
                    <p class="mb-0 text-white footerText">Craft with <i
                        class="mdi mdi-heart text-danger"></i> by <a
                        href="https://doltech.io/" target="_blank"
                        class="text-reset"> Leticia</a>.</p>
                </div>
            </div>
        </div>
    </div>
</div>



<!-- Hero End -->


<!-- Footer Start -->


<!-- Footer End -->
```

Attendance componet

```
<div class="page-wrapperr toggled">
    <!-- sidebar-wrapper -->
    <nav id="sidebar" class="sidebar-wrapper">
        <div class="sidebar-content" data-simplebar style="height:
calc(100% - 60px);">
```

```
<div class="sidebar-brand">
  <a href="#!">
    <img src="../../../../assets/images/fuelProject/logo.jpeg"
height="60" width="100" class="logo-light-mode" alt="">
    <img src="../../../../assets/images/fuelProject/logo.jpeg"
height="60" width="100" class="logo-dark-mode" alt="">
    <span class="sidebar-colored">
      <img src=".../../../../assets/images/fuelProject/logo.jpeg"
height="60" width="100" alt="">
    </span>
  </a>
</div>

<ul class="sidebar-menu">
  <li class="sidebar-dropdown">
    <a [routerLink]="(['/attendant/dashboard'])"
class="dashboardText active"><i class="ti ti-home me-2
dashboardHomeIcon"></i>Dashboard</a>
  </li>
  <li class="sidebar-dropdown">
    <a [routerLink]="(['/attendant/my-sales'])"
class="dashboardText"><i class="ti ti-browser me-2
dashboardHomeIcon"></i>My Sales</a>
  </li>
  <li class="sidebar-dropdown">
```

```
                    <a [routerLink]="(['/attendant/my-attendace'])"
class="dashboardText"><i class="ti ti-browser me-2
dashboardHomeIcon"></i>My Attendance</a>
                    </li>
                  </ul>
                  <!-- sidebar-menu  -->
                </div>

                <!-- Sidebar Footer -->
              </nav>
              <!-- sidebar-wrapper  -->
            </div>
```

Login componet

```
<div class="back-to-home">
  <a [routerLink]="(['/home'])" class="back-button btn btn-icon
backHomeBtn"><span
      class="mdi mdi-home icon"></span></a>
</div>

<!-- Hero Start -->
<section class="cover-user">
```

```
<div class="container-fluid px-0">
  <div class="row g-0 position-relative">

    <!-- Form Div -->
    <div class="col-lg-4 cover-my-30 order-2">
      <div class="cover-user-img d-flex align-items-center">
        <div class="row">
          <div class="col-12">
            <div class="card login-page border-0" style="z-index: 1">
              <div class="card-body p-0">
                <div class="card-title text-left">
                  <h2 class="mb-0">Welcome to FuelScape</h2>
                  <hr>
                  <p class="text-muted">Please sign in with the required details to have access to
                    your account</p>
                </div>
                <form class="login-form mt-4">
                  <div class="row">
                    <div class="col-lg-12">
                      <div class="mb-3">
                        <div class="form-icon position-relative">
                          <span class="mdi mdi-email-outline fea icon-sm icons"
```

```
                                        style="position: relative; top: -
15px;"></span>
                                    <input type="email" class="form-control
ps-5"
                                        placeholder="Username"
[(ngModel)]="username" name="username"
                                        required>
                                </div>
                            </div>
                        </div><!--end col-->

                        <div class="col-lg-12">
                            <div class="mb-3">
                                <div class="form-icon position-relative">
                                    <span class="mdi mdi-lock fea icon-sm
icons"
                                        style="position: relative; top: -
15px;"></span>
                                    <input type="password" class="form-
control ps-5"
                                        placeholder="Password"
[(ngModel)]="password"
                                        name="password" required>
                                </div>
                            </div>
```

```
                                    </div><!--end col-->
                                    <!-- <app-hcaptcha [siteKey]="'20000000-ffff-ffff-
ffff-000000000002'" (captchaToken)="onCaptchaToken($event)"></app-hcaptcha> --
>
                                    <div class="col-lg-12">
                                        <div class="d-flex justify-content-between">
                                            <div class="mb-3">
                                                <p class="forgot-pass mb-0
forgottenPassword"><a
                                                        [routerLink]="(['/recovery-
password'])"
                                                        class="text-dark fw-600 ">Forgot
                                                        password ?</a></p>
                                            </div>
                                        </div>
                                    </div><!--end col-->

                                    <div class="col-lg-12 mb-0">
                                        <div class="d-grid">
                                            <button class="btn" id="signInBtn"
(click)="login()">Sign
                                                in</button>
                                        </div>
                                    </div><!--end col-->
                                    <!-- <div class="col-12 text-center">
```

```
                                    <p class="mb-0 mt-3"><small
                                        class="text-dark me-2">Don't
                                        have an account ?</small>
                                    <a
                                        [routerLink]="(['/sign-up'])"
                                        class="text-dark fw-bold
signUpText">Sign
                                        Up</a></p>
                                </div>end col -->
                            </div><!--end row-->
                        </form>
                    </div>
                </div>
            </div><!--end col-->
        </div><!--end row-->
    </div> <!-- end about detail -->
</div> <!-- end col -->

<!-- Image Div -->
<div class="col-lg-8 offset-lg-4 padding-less img order-1 jarallax"
data-jarallax data-speed="0.5"
    style="background-
image:url('../../../../assets/images/fuelProject/station.png') "></div>
<!-- end col -->
</div><!--end row-->
```

```
        </div><!--end container fluid-->
      </section><!--end section-->
      <!-- Hero End -->
```

Transporter component

```
<div class="page-wrapperr toggled">
  <!-- sidebar-wrapper -->
  <nav id="sidebar" class="sidebar-wrapper">
    <div class="sidebar-content" data-simplebar style="height: calc(100% - 60px);">
      <div class="sidebar-brand">
        <a href="#!">
          <img src="../../../../assets/images/fuelProject/logo.jpeg" height="60"
width="100" class="logo-light-mode" alt="">
          <img src="../../../../assets/images/fuelProject/logo.jpeg" height="60"
width="100" class="logo-dark-mode" alt="">
          <span class="sidebar-colored">
            <img src=".../../../../assets/images/fuelProject/logo.jpeg" height="60"
width="100" alt="">
          </span>
        </a>
      </div>
```

```
<ul class="sidebar-menu">
  <li class="sidebar-dropdown">
    <a [routerLink]="(['/admin/dashboard'])" class="dashboardText active"><i class="ti ti-home me-2 dashboardHomeIcon"></i>Dashboard</a>
  </li>
  <li class="sidebar-dropdown">
    <a [routerLink]="(['/admin/manage-users'])" class="dashboardText"><i class="ti ti-browser me-2 dashboardHomeIcon"></i>Manage Fuel</a>
  </li>
  <li class="sidebar-dropdown">
    <a [routerLink]="(['/#!'])" class="dashboardText"><i class="ti ti-browser me-2 dashboardHomeIcon"></i>Manage Attendance</a>
  </li>
</ul>
<!-- sidebar-menu  -->
</div>
<!-- Sidebar Footer -->
<ul class="sidebar-footer list-unstyled mb-0">
  <li class="list-inline-item mb-0">
    <a [routerLink]="(['/#!'])" target="_blank" class="btn btn-icon btn-soft-light"><i class="uil uil-power text-danger"></i></a> <small class="text-muted fw-medium ms-1 logoutText">Log Out</small>
  </li>
</ul>
<!-- Sidebar Footer -->
```

```
    </nav>
    <!-- sidebar-wrapper  -->
</div>
```

Back end codes

Authentication views

```
from django.shortcuts import render
from .serializers import *
from rest_framework.views import APIView
from rest_framework.response import Response
from django.contrib.auth.models import User
from rest_framework import status
# from django.contrib.auth import authenticate
from django.contrib.auth import authenticate, login
from rest_framework.authtoken.models import Token
from .models import *
from fuel_transporter.permissions import *
from django.http import JsonResponse
# Create your views here.

class Registeration_view(APIView):
```

```
    def post(self, request):
        serializer = RegisterationSerializer(data = request.data)
        serializer.is_valid(raise_exception = True)
        serializer.save()

        return Response(serializer.data)



class UserView(APIView):

    def get(self, request):
        users = Registeration.objects.all()
        serializer = RegisterationSerializer(users, many=True)

        return Response(serializer.data)



class Login_View(APIView):



    def post(self, request):
        username = request.data.get('username')
        password = request.data.get('password')

        user = authenticate(request, username=username, password=password)
```

```
        if user is not None:
            login(request, user)
            token = Token.objects.get(user=user)
            user_details = {
                'id': user.id,
                'username': user.username,
                'email': user.email,
                'role': user.role,
                # Add other user details you want to include
            }



        # Generate JWT token or session logic here
            return Response({'token': token.key, 'user': user_details },status=status.HTTP_200_OK)
        else:
            return Response({'error': 'Invalid credentials'}, status=status.HTTP_401_UNAUTHORIZED)


class Attendance_View(APIView):

    def post(self, request):
```

```
    permission_classes = (User_priviledge,)
    data = {
        'user': request.user.pk,
        'action': action,
        'clockin': request.data.get('clockin'),
        'clockout': request.data.get('clockout'),
    }

    serializer = AttendanceSerializer(data = data)
    serializer.is_valid(raise_exception = True)
    serializer.save()

    return Response(serializer.data)



def get(self,request):
    permission_classes = (Admin_priviledge,)
    queryset = Attendance.objects.all()
    serializer = AttendanceSerializer(queryset, many = True)

    return Response(serializer.data)
```

```
def get_attendance(request):
    queryset = Attendance.objects.all()
    serializer = AttendanceSerializer(queryset, many=True)
    return JsonResponse(serializer.data, safe=False)



def list_of_attendance_by_id(request, id):
    queryset = Attendance.objects.filter(user=id)
    serializer = AttendanceSerializer(queryset, many=True)

    return JsonResponse(serializer.data, safe=False)









# get request
def clock_in(request, user_id):
    # Check if there's an existing record for the user on the current date
    today = timezone.now().date()
    existing_record = Attendance.objects.filter(user_id=user_id, clockin__date=today)

    if existing_record.exists():
        return JsonResponse({"error": "You are already clocked in today."}, status=400)
    else:
```

```
        new_record = Attendance(user_id=user_id, has_clocked_in=True)
        new_record.save()
        return JsonResponse({"message": "Clock-in successful."}, status=200)

def clock_out(request, user_id):
    # Check if there's an existing record for the user on the current date
    today = timezone.now().date()
    existing_record = Attendance.objects.filter(user_id=user_id, clockin__date=today)

    if existing_record.exists():
        record = existing_record.first()
        if not record.has_clocked_in:
            return JsonResponse({"error": "You must clock in before clocking out."},
status=400)
        else:
            record.clockout = timezone.now()
            record.has_clocked_in = False
            record.save()
            return JsonResponse({"message": "Clock-out successful."}, status=200)
    else:
        return JsonResponse({"error": "You have not clocked in today."})
```

Fuel Attendant view

```
from django.shortcuts import render
from rest_framework.views import APIView
from .serializers import *
from rest_framework.response import Response
from fuel_transporter.permissions import *
from datetime import datetime
from rest_framework import status


# Create your views here.


class Meter_reading_View(APIView):


    def post(self, request):
        permission_classes = (Fuel_Attendant_priviledge)
        serializer = Meter_readingSerializer(data = request.data)
        serializer.is_valid(raise_exception= True)
        serializer.save()


        return Response(serializer.data)


    # update some details of the meter reading form
    def patch(self, request):
        permission_classes = (Fuel_Attendant_priviledge)
        queryset = Meter_reading.objects.get(date = datetime.today().date())
```

```
        serializer = Meter_readingSerializer(instance = queryset, data = request.data,
partial = True)

        serializer.is_valid()
        serializer.save()

        return Response(serializer.data)



# update all details on the meter reading form
    def patch(self, request):
        permission_classes = (Fuel_Attendant_priviledge)
        queryset = Meter_reading.objects.get(date = datetime.today().date())
        serializer = Meter_readingSerializer(instance = queryset, data = request.data,
partial = True)

        serializer.is_valid()
        serializer.save()

        return Response(serializer.data)

    def put(self, request, id):
        permission_classes = (Fuel_Attendant_priviledge)
        try:
            meter_reading = Meter_reading.objects.get(id=id)
```

```
        except Meter_reading.DoesNotExist:
            return Response({"error": "Meter reading not found"},
status=status.HTTP_404_NOT_FOUND)

        serializer = Meter_readingSerializer(meter_reading, data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)

    def get(self, request):
        permission_classes = (Fuel_Attendant_priviledge)
        queryset = Meter_reading.objects.latest('date')
        serializer = Meter_readingSerializer(queryset, many = False)

        return Response(serializer.data)



    def delete(self, request, id):
        permission_classes = (Fuel_Attendant_priviledge)

        try:
            meter_reading = Meter_reading.objects.get(id=id)
            meter_reading.delete()
```

```
        except Meter_reading.DoesNotExist:
            return Response({"error": "Meter reading not found"}, status=status.HTTP_404_NOT_FOUND)

        return Response({"message": "Meter reading deleted successfully"}, status=status.HTTP_204_NO_CONTENT)




class Meter_reading_View_by_id(APIView):

    def get(self, request, id):
        permission_classes = (Fuel_Attendant_priviledge)
        queryset = Meter_reading.objects.get(id = id)
        serializer = Meter_readingSerializer(queryset, many = False)

        return Response(serializer.data)
```

Fuel Transporter view

```
from django.shortcuts import render
from .serializers import *
from rest_framework.views import APIView
from rest_framework.response import Response
from .permissions import *
from rest_framework import status
from datetime import datetime
# Create your views here.

class Fuel_purchased_details_View(APIView):

    def post(self, request):
        permission_classes = (Fuel_transporter_priviledge,)
        serializer = Fuel_purchased_detailsSerializer(data = request.data)
        serializer.is_valid(raise_exception = True)
        serializer.save()

        return Response(serializer.data)

    def put(self, request, id):
        permission_classes = (Fuel_transporter_priviledge,)
        try:
            fuel_purchase = Fuel_purchased_details.objects.get(id=id)
```

```
        except fuel_purchase.DoesNotExist:
            return Response({"error": "Fuel purchase data not found"},
status=status.HTTP_404_NOT_FOUND)

        serializer = Fuel_purchased_detailsSerializer(fuel_purchase, data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)


    def get(self,request):
        permission_classes = (Admin_priviledge,Fuel_transporter_priviledge)
        queryset = Fuel_purchased_details.objects.latest('date', 'time')
        serializer = Fuel_purchased_detailsSerializer(queryset, many = False)

        return Response(serializer.data)



    def delete(self, request, id):
        permission_classes = (Admin_priviledge,Fuel_transporter_priviledge)

        try:
            fuel_purchase = Fuel_purchased_details.objects.get(id=id)
```

```
            fuel_purchase.delete()
        except Fuel_purchased_details.DoesNotExist:
            return Response({"error": "Fuel purchase data not found"},
status=status.HTTP_404_NOT_FOUND)

        return Response({"message": "Fuel purchase data deleted successfully"},
status=status.HTTP_204_NO_CONTENT)



class Fuel_purchased_details_View_by_id(APIView):

    def get(self, request, id):
        permission_classes = (Admin_priviledge,Fuel_transporter_priviledge)
        queryset = Fuel_purchased_details.objects.get(id = id)
        serializer = Fuel_purchased_detailsSerializer(queryset, many = False)

        return Response(serializer.data)
```

Manager view

```
from django.shortcuts import render
from .serializers import *
from rest_framework.views import APIView
from rest_framework.response import Response
from fuel_transporter.permissions import *
from rest_framework import status
from datetime import datetime

# Create your views here.

class Fuel_dipping_view(APIView):

    def post(self, request):
        permission_classes = (Manager_priviledge)
        serializer = Fuel_dippingSerializer(data = request.data)
        serializer.is_valid(raise_exception = True)
        serializer.save()

        return Response(serializer.data)

    def patch(self, request):
        permission_classes = (Manager_priviledge)
```

```
        queryset = Fuel_dipping.objects.get(date_time = datetime.today().date())
        serializer = Fuel_dippingSerializer(instance = queryset, data = request.data,
partial = True)

        serializer.is_valid()
        serializer.save()

        return Response(serializer.data)

    def put(self, request, id):
        permission_classes = (Manager_priviledge)
        try:
            fuel_dipping_value = Fuel_dipping.objects.get(id=id)
        except fuel_dipping_value.DoesNotExist:
            return Response({"error": "Fuel dipping data not found"},
status=status.HTTP_404_NOT_FOUND)

        serializer = Fuel_dippingSerializer(meter_reading, data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)


    def get(self, request):
```

```
        permission_classes = (Manager_priviledge)
        dips = Fuel_dipping.objects.all()
        serializer = Fuel_dippingSerializer(dips, many= True)

        return Response(serializer.data)




    def delete(self, request, id):
        permission_classes = (Manager_priviledge)

        try:
            fuel_dipping_value = Fuel_dipping.objects.get(id=id)
            fuel_dipping_value.delete()
        except Attendance.DoesNotExist:
            return Response({"error": "Fuel dipping data not found"},
status=status.HTTP_404_NOT_FOUND)

        return Response({"message": "Fuel dipping data  deleted successfully"},
status=status.HTTP_204_NO_CONTENT)
```

```
class Fuel_dipping_view_by_id(APIView):

    def get(self, request, id):
        permission_classes = (Manager_priviledge)
        queryset = Fuel_dipping.objects.get(id = id)
        serializer = Fuel_dippingSerializer(queryset, many = False)

        return Response(serializer.data)




class Fuel_prices_view(APIView):

    def post(self, request):
        permission_classes = (Manager_priviledge)
        serializer = Fuel_pricesSerializer(data = request.data)
        serializer.is_valid(raise_exception = True)
        serializer.save()

        return Response(serializer.data)

    def patch(self, request):
        permission_classes = (Manager_priviledge)
        queryset = Fuel_prices.objects.get(timestamp = datetime.today().date())
```

```
        serializer = Fuel_pricesSerializer(instance = queryset, data = request.data, partial
= True)

        serializer.is_valid()
        serializer.save()

        return Response(serializer.data)

    def put(self, request, id):
        permission_classes = (Manager_priviledge)
        try:
            fuel_price = Fuel_prices.objects.get(id=id)
        except Fuel_prices.DoesNotExist:
            return Response({"error": "Fuel price data not found"},
status=status.HTTP_404_NOT_FOUND)

        serializer = Fuel_pricesSerializer(meter_reading, data=request.data)
        if serializer.is_valid():
            serializer.save()
            return Response(serializer.data, status=status.HTTP_200_OK)
        return Response(serializer.errors, status=status.HTTP_400_BAD_REQUEST)

    def get(self, request):
        permission_classes = (Manager_priviledge)
        prices = Fuel_prices.objects.all()
```

```
        serializer = Fuel_pricesSerializer(prices, many= True)

        return Response(serializer.data)



    def delete(self, request, id):
        permission_classes = (Manager_priviledge)

        try:
            fuel_prices = Fuel_prices.objects.get(id=id)
            fuel_prices.delete()
        except Fuel_prices.DoesNotExist:
            return Response({"error": "Fuel price data not found"},
status=status.HTTP_404_NOT_FOUND)

        return Response({"message": "Fuel price data deleted successfully"},
status=status.HTTP_204_NO_CONTENT)



class Fuel_prices_view_by_id(APIView):

    def get(self, request, id):
        permission_classes = (Manager_priviledge)
```

```
        queryset = Fuel_prices.objects.get(id = id)
        serializer = Fuel_pricesSerializer(queryset, many = False)

        return Response(serializer.data)



class Daily_statistics_view(APIView):

    def get(Self, request):
        queryset = Daily_statistics.objects.all()

        serializer = Daily_statisticsSerializer(data = queryset, many = True)

        return Response(serializer.data)
```

turnitin

Digital Receipt

This receipt acknowledges that Turnitin received your paper. Below you will find the receipt information regarding your submission.

The first page of your submissions is displayed below.

Submission author:	Akwetey Emmanuel
Assignment title:	KTU Student Class
Submission title:	Copy of Ephraim_Gyamfi_Dasey-and-Leticia_Kabu
File name:	Ephraim_Gyamfi_Dasey-and-Leticia_Kabu.docx
File size:	1.73M
Page count:	116
Word count:	13,505
Character count:	92,557
Submission date:	03-Nov-2023 09:13AM (UTC+0000)
Submission ID:	2134004981

KOFORIDUA TECHNICAL UNIVERSITY
FACULTY OF APPLIED SCIENCE AND TECHNOLOGY
(DEPARTMENT OF COMPUTER SCIENCE)

FUEL MANAGEMENT SYSTEM

A PROJECT WORK SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, SCHOOL OF APPLIED SCIENCE AND TECHNOLOGY IN PARTIAL FULFILMENT OF THE AWARD OF THE BACHELOR IN TECHNOLOGY (BTECH) IN COMPUTER SCIENCE PROGRAMME

BY

LETICIA MANLEY KABU
EPHRAIM GYAMFI DASEY

CSD 402 PROJECT IN COMPUTER SCIENCE

DATE:

SUPERVISOR: DR PATRICIA GHANA

Dean of Students Affairs
Koforidua Tech. University
Koforidua E/R.

KOFORIDUA TECHNICAL UNIVERSITY
LIBRARY
3/11/23

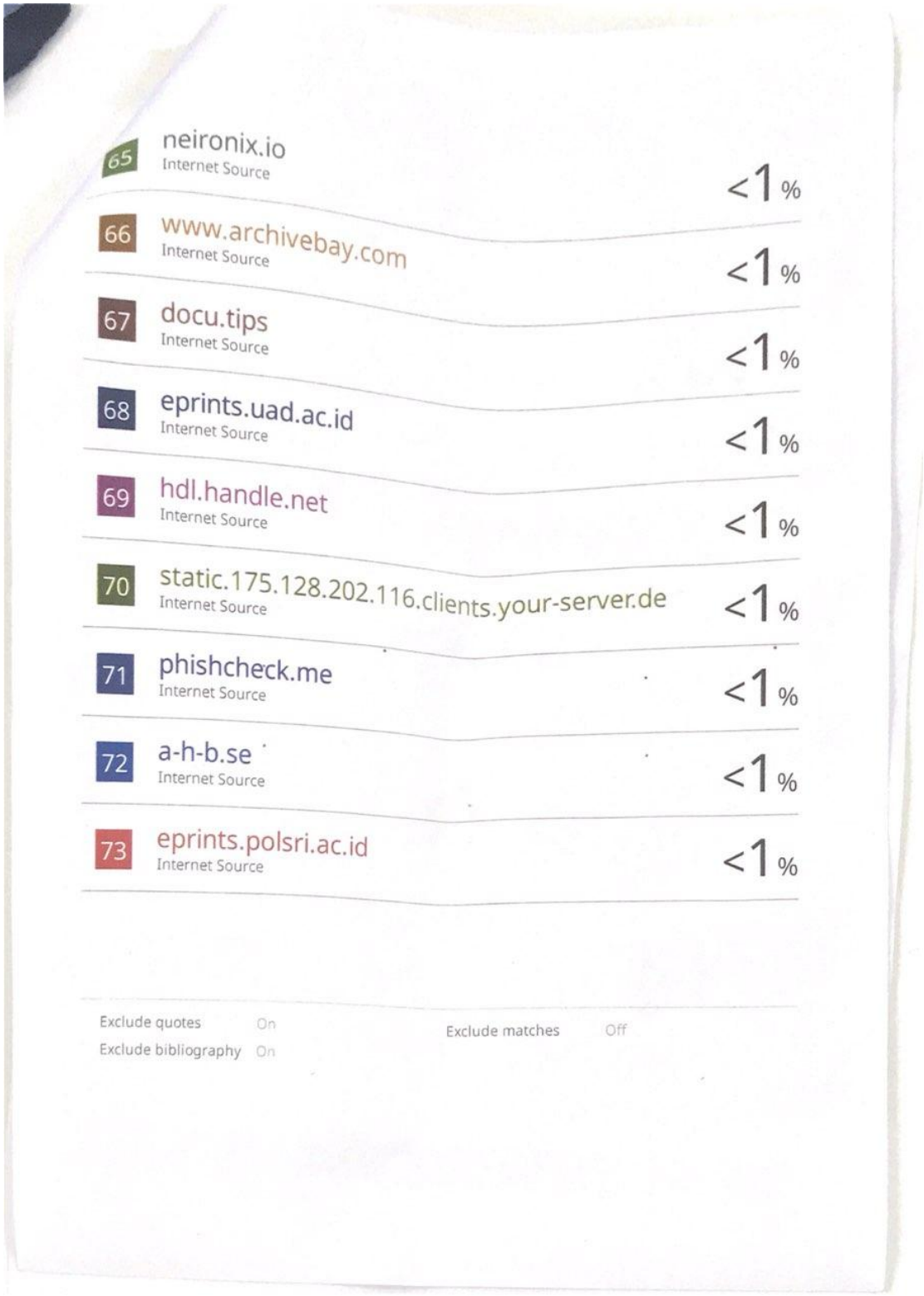
Copy of Ephraim_Gyamfi_Dasey-and-Leticia_Kabu

ORIGINALITY REPORT

11%	10%	2%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	scholarcommons.scu.edu Internet Source	1%
2	morioh.com Internet Source	1%
3	erpsolutions.oodles.io Internet Source	1%
4	www.coursehero.com Internet Source	1%
5	Submitted to King's College Student Paper	1%
6	www.javatpoint.com Internet Source	<1%
7	Submitted to Chester College of Higher Education Student Paper	<1%
8	repository.its.ac.id Internet Source	<1%
9	dev.to Internet Source	<1%


65 neironix.io
Internet Source
<1%
66 www.archivebay.com
Internet Source
<1%
67 docu.tips
Internet Source
<1%
68 eprints.uad.ac.id
Internet Source
<1%
69 hdl.handle.net
Internet Source
<1%
70 static.175.128.202.116.clients.your-server.de
Internet Source
<1%
71 phishcheck.me
Internet Source
<1%
72 a-h-b.se
Internet Source
<1%
73 eprints.polsri.ac.id
Internet Source
<1%
Exclude quotes On
Exclude bibliography On
Exclude matches Off