Online fuel ordering system for dealers
(Fuel Manager)

H.A.K.S Hettiarachchi
2024

Online fuel ordering system for dealers

(Fuel Manager)

A thesis submitted for the Degree of Master of Information Technology

H.A.K.S Hettiarachchi
University of Colombo School of Computing
2024

Declaration

The thesis is my original work and has not been submitted previously for a degree at this orany other university/institute.

To the best of my knowledge, it does not contain any material published or written by another person, except as acknowledged in the text.

Student Name: H.A.K.S Hettiarachchi

Registration Number: 2020/MIT/040

Index Number: 20550405

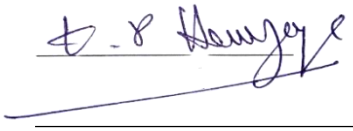
Signature: Date:2024/03/02

This is to certify that this thesis is based on the work of

Mr. Keerthi Sanjaya Hettiarachchi

under my supervision. The thesis has been prepared according to the format stipulated and isof acceptable standard.

Certified by: Supervisor Name: Senior Professor K.P Hewagamage



Signature: Date:2024/03/02

Abstract

In response to the fuel crisis experienced in Sri Lanka, this project aims to address one reason for delay, caused by manual order processing for fuel dealers. The system addresses the inability to track order statuses efficiently. By allowing dealers to register their fuel stations online, get approval, and place orders through the system, it streamlines the process. The system allows users to view invoices, generate reports, receive SMS notifications for order approval to until gate exit, and facilitate online payments. Dealers can also monitor their remaining tank capacity by inputting daily tank dips.

To develop the software, use programming languages and web development frameworks. Requirements analysis, system design, implementation, and testing as the core phases. When developing the system collaborates closely with end users to get some feedback iteratively to verify the system meets the correct output.

The developed online fuel ordering system for dealers addresses the inefficiencies of manual order processing. Dealers can now register their fuel stations, place orders, monitor tank capacities in the shed, and receive real-time updates on their orders.

This system helps to solve one major reason among the reasons that was affected to fuel crisis in Sri Lanka. It uses technology to make things easier and clear when ordering fuel. Dealers can now track their orders better and manage them more efficiently. This means they can get fuel faster and more reliably. Overall, using technology like this can help to solve big problems in society more smartly.

Acknowledgment

I would like to express my sincere gratitude to my supervisor, Senior Professor. K. P. Hewagamage, for his invaluable guidance, encouragement, and support throughout the entire process of completing this project. His expertise and insightful feedback have been instrumental in shaping this research.

I am also thankful to the University of Colombo School of Computing (UCSC) for providing the necessary resources and environment for conducting this study.

Then, my appreciation to all the individuals who contributed to this project, directly or indirectly, including the dealers and stakeholders involved in the online fuel ordering system.

Lastly, I am deeply grateful to my family and friends for their unwavering support and encouragement during this academic journey.

This thesis would not have been possible without the support of these individuals and institutions.

List of Acronyms

UML - Unified Modeling Language

UI - User Interface

CSS - Cascading Style Sheets

MVC - Model–view–controller

ER - Entity Relationship

API - Application Programming Interface

RFC - Remote Function Call

NIC - National Identity Card

OTP - One-Time Password

Table of Contents

Contents

List of Figures

List of Tables

Chapter 1 - Introduction

1.1 Project Overview

Ceylon Petroleum Storage Terminals Ltd (CPSTL) is a leading government organization in Sri Lanka that provides petroleum product storage services and Island-wide distribution of petroleum products via several robust channels. These fuel products come to CPSTL via the Indian Oil Company (IOC) and Ceylon Petroleum Corporation (CPC). There are eleven depots under the CPSTL to distribute fuel in a fast and efficient way to their registered fuel stations (Petrol shed). When Sri Lanka's fuel crisis hit in June, we had to wait in vehicle queues for hours. With the recent high demand and lack of imports which happened due to the dollar issue, Then CPSTL has to face some problems in handling the ordering and distribution process quickly and efficiently as below.

• It is very difficult to track all the dealer-side-related orders and company-side-related orders, because of the current manual process all dealers are making their orders through SMS, Email, and coming to the CPSTL premises order section.

• The existing process is very time-consuming because the dealer comes to the office premises, he/she has to do more paper works such as filling out the order form, and waiting in the queue and most of the time it may be the main issue for getting a delay of the order.

• Sometimes manually entered data may be incorrect and the wrong order details, and misunderstanding may occur most of the time. When entering the wrong order data by staff, it may have a main effect on the material stock and other decisions related to operations.

• There is no proper way for the dealer to track the order status within the premises. Because once the dealer places the order, cannot get the idea until arriving at the location. In premises, it should go through several stages like gantry filling, invoice, and gate exit.

• There is no proper way to manage bowser truck details. Therefore, it is difficult to manage the bowser-related calibration and license update for each vehicle.

• At present distribution function creates reports to monitor and analyze the current situation of the distribution process and day-to-day working process for company decision-making purposes by using the manual process.

• Dealers also have to maintain their order details in the manual process and cannot create accurate reports and cannot capture the previous order-related details well manner.

1.2 Motivation

The motivation behind developing an online fuel ordering system for dealers to provide fuel with streamlined, efficient, convenient service to their customers. This will be helpful to reduce manual process time wastage process as well as reduce cost and open new business opportunities.

When maintaining their fuel supply, shed owners frequently struggle, having to rely on manual procedures to order fuel requirements to the gas station. The goal is to simplify the fuel purchasing process by creating an online fuel ordering system that is specifically designed for shed owners. Online ordering makes it simple for shed owners to place orders, doing away with the necessity for in-person visits and saving time and effort.

Fuel is needed for shed owners who need to run generators, machinery, and other equipment for their businesses. Shed owners may precisely track their fuel usage, real-time data, and analytics, and make knowledgeable decisions about their fuel requirements with the help of the online fuel ordering system. As a result, fuel management is improved, wasting is minimized, and overall operational efficiency is increased.

1.3 Objectives

We are all aware that, during the last experienced pandemic, Web-based systems were more essential than ever. During that, only people realized this was a much-needed service option when compared with the other demanded services. Fuel ordering and distribution have a major impact on the economy as well as society, actually, this is a needed requirement for all fuel station dealers in Sri Lanka, CPSTL management as well and the public. This project will elaborate on the web-based fuel ordering system for dealers.

- Shed owners/Dealers can be able to register with the web-based system with different verification options. Once the user going to log in to the system to place an order mobile verification or email verification is sent for verification purposes.
- Identify the accuracy of the dealer side ordering process and the company side ordering process.
- Reduce time consumption and waiting in a queue to place an order, Dealer can just place an order by using the system.
- Reduce human-related errors, security, accuracy, and processing time by using the system.
- Identify the proper way to manage the order status in every stage by SMS for the dealer when placing the order.
- Omit the existing staff preferences bias ordering system by using the system.
- Implement a proper way to manage bowser truck details and keep details up to date.
- Identify manager or administrative wise decision making activities by generating reliable reports.
- Identify the dealer side decision-making or relevant activities by generating reliable reports.

1.3 Background of the study

Mainly the background study found some existing systems related to the proposed fuel ordering system.

- Purbis- Purbis is a cloud-based fuel ordering system that helps users to manage, order and get fuel-related information as well as reports for decisions making. The main purpose of this system is to make everything online and automated as well as earn more profit for the business. This system has the following features and mobile applications. When compared with the proposed system, gantry filling notification, invoice notification, gate exit notification, bowser management, and generated invoices are not done in this system (Online fuel ordering and delivery system, software - Fuelfill (purbis.com)).
- Peerbits- Peerbits is also a fuel ordering system that is used to streamline the fuel ordering system with a high-cost GPS tracking system. Also, the purpose of this system is to unlock fuel business opportunities for both fuel delivery aggregators and fuel station owners. This system also does not fulfill the requirement of gantry filling notification, invoice notification, gate exit notification and generated invoice. This system consists of the mobile app of the customer app as well as the driver app.(On-demand Fuel App Delivery Services, Company - Peerbits)
- FuelMe- This is also a cloud-based fuel ordering system that will automatically re-order fuel options. The main purpose of this is to streamline the fuel purchasing process with fast, reliable, and cost-effective. This major runs on the mobile application and this also has the following drawbacks gantry filling notification, invoice notification, gate exit notification, and GPS tracking. Though there is no GPS, they are developed with a live order tracking process integrated for mobile (Diesel Fuel Delivery | Order Fuel Online - Fuel Me).

1.4 Scope of the study

This web based fuel ordering system consists of the following modules.

Shed Management

Mainly this module will allow registering their shed with the system and do the ordering operation in a well-organized manner.

- Maintain shed registration details with the system.
- Manage approval of the shed registration in order to check the shed details are original.
- Manage rejection of the shed registration when not giving accurate information.
- Manage received an order confirmation.
- Manage the shed license cancelation option.

Admin User Management

The main task of this module is to create a user account and do the operation as well as handle the administrative operations well-organized manner.

- Once the user login to the system, it will be sent an OTP to the shed user for security purposes of the system.
- Maintain user related operations.
- Manage order related operations.
- Maintain administrative-related operations.

Order Management

The main task of this module is to manage the existing ordering process in a well-organized manner.

- Manage order related information.
- Maintain the creation of the order information.
- Maintain gantry filling operation.
- Manage and track the order process status to the shed owner by SMS
- Create and generate an invoice for the relevant order.

Bowser Management

The main task of this module is to manage bowser truck details in a well-organized manner.

- Maintain bowser related details.
- Maintain bowser license details and status

Reports

The main task of this module is to generate reports for the shed owner and management to make decisions and identify the improvements that should be made related to streamlining the order process.

- Maintain order related operation reports.
- Maintain bowser management reports.
- Maintain shed owner related reports.

1.5 Structure of the Dissertation

The structure of the dissertation refers to the chapters within the document. This will help users to navigate through the document and understand the flow of the document. This is following the development life cycle further.

- **Introduction**

  This chapter describes to the user the problem and solution with the subject matter and the background of the project. Also, readers can get an idea about motivational facts that are used to develop the system.

- **Background**

  Utilizing the proper UML diagrams, describe the functionality of the current system. Describe the system's functional and non-functional requirements. Additionally, contrast and compare related systems.

- **Design**

  The system architecture and design are planned during this stage. Both high-level and low-level design specifications that describe the system's architecture, elements, and interfaces are included in the software design. Additionally, it takes performance, security, and scalability into the system.

- **Implementation**

  In this stage, the software is created using the design guidelines. Code is written by best practices and coding standards. In the implementation phase, the design is converted into executable code, various components are combined, and any necessary databases or data structures are created. As well as describe the special code segment in this section.

- **Testing and Evaluation**

  The testing stage examines the software system's performance, quality, and functionality. It includes various types of testing, such as unit testing, integration testing, system testing, and acceptance testing.

- **Conclusion**

  In this stage provide a summary of the system from the problem stage to solution steps and further future enhancements.

Chapter 2 – Background

2.1 Introduction

Managing the ordering and delivery of fuel traditionally has been done by petrol shed owners or gas station managers using a manual fuel ordering system. In this method, placing orders is often done by hand without the aid of cutting-edge technology or automated processes. The history and main features of a manual gasoline ordering system are described as follows:

Ordering procedure in a manual fuel ordering system, it is up to the owner of the gas station to keep an eye on the fuel levels and choose when to submit an order for a refill. Using data from things like current inventory levels, sales volume, and anticipated demand, they manually determine the amount of fuel needed. After determining the requirement for gasoline replenishment, the owner of the gas station contacts the distributor or supplier of fuel to start a conversation. You can accomplish this in a number of ways, including via phone, fax, or by physically visiting the supplier's office. Order placement by the owner of the gas station gives the supplier essential information, such as the fuel type required, the desired quantity, and the preferred method of delivery. Usually, this information is disseminated orally or in written forms. Delivery planning is based on their own logistics and availability, the fuel supplier processes the order and plans the delivery. During the ordering procedure, the owner of the gas station may bargain certain delivery dates or windows of time. In manual record-keeping in a manual system, the owner of the fuel shed uses manual logs or registers to keep track of fuel orders, deliveries, and inventory levels. After each delivery, they manually update the inventory count and determine how much product is still available. Inventory control is the proprietor of the petrol shed manually keeps track of the fuel inventory levels, stock levels, consumption rates, and reorder points. In order to assure proper inventory management, this necessitates routine physical inspections of the storage tanks. Limitations and challenges in manual fuel ordering systems are susceptible to mistakes, misunderstandings, and delays. Human error is more likely to occur while calculating fuel quantities, documenting data, or conveying orders. Additionally, manual systems could not have real-time access to inventory information, making it difficult to plan orders and avoid stockouts or overstocking.

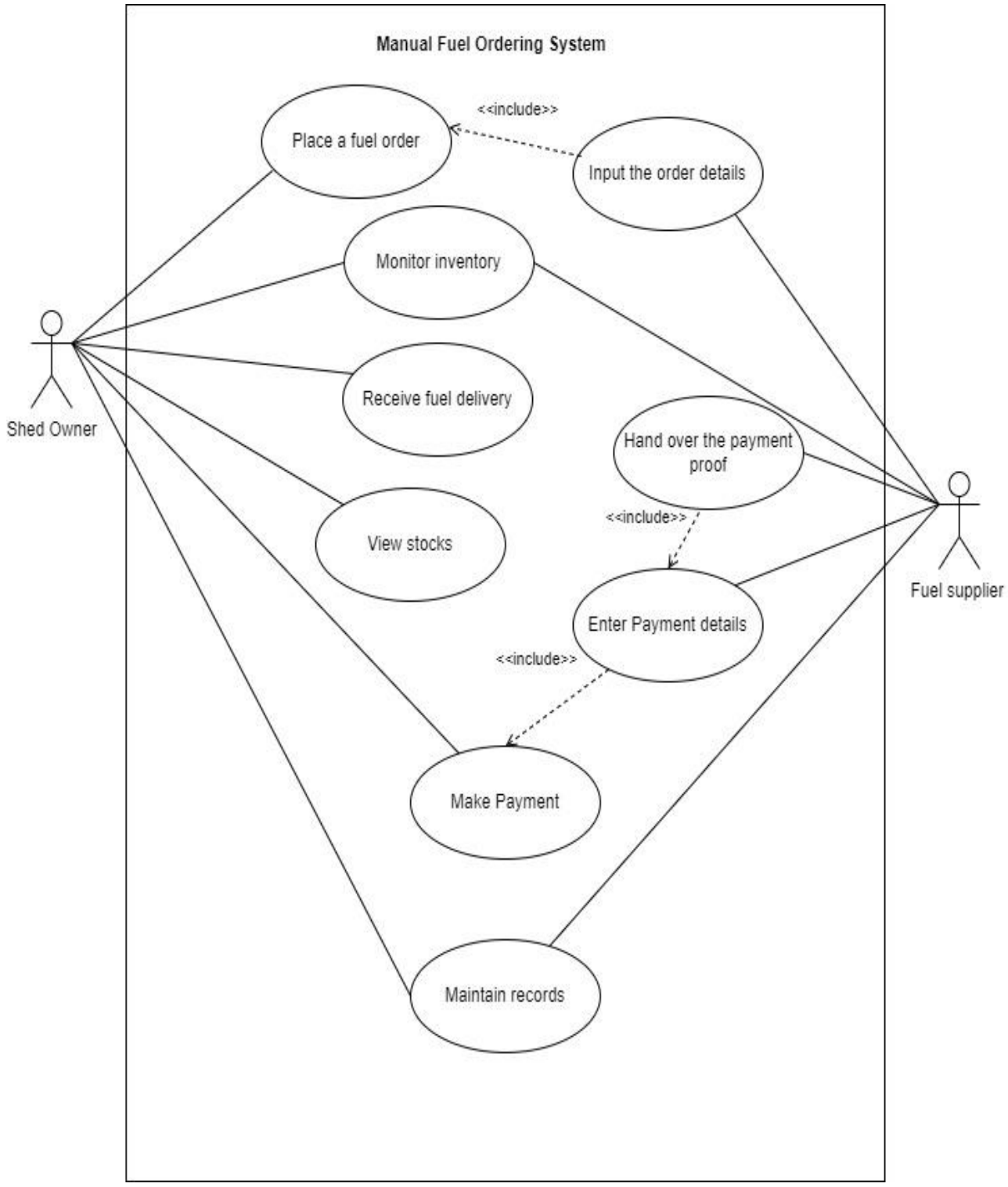


Figure 1: Existing Manual Fuel Ordering Process

2.2 Requirement Analysis

Requirement analysis is a major part of the software development life cycle because when we try to automate the manual system, we should capture the genuine and needed can be fulfilled to solve the problem.

2.2.1 Functional Requirement

- **Shed Management Module**
  - ❖ Dealer registration: The dealer can register with the Online fuel ordering system
  - ❖ Change dealer profile: Update the dealer profile
  - ❖ Change password: Dealer can change their login password
  - ❖ Block dealer: Admin can identify dealer details and block the dealer

- **Admin User Management Module**
  - ❖ Create an admin user: Admin can create the user-level admin privileges
  - ❖ Change admin profile: Admin can change their system profile
  - ❖ Change admin password: Admin can change their password

- **Order Management Module**
  - ❖ Order Create: The dealer can create an order from the online fuel ordering system
  - ❖ Order Accept: The requested order can be approved by the admin user
  - ❖ Order not Accept: Admin users can review the order and if payment is not done he can stay that as remain without approve the order from the system
  - ❖ Track the Order: The dealer can view the order status until the order is completed within the premises
  - ❖ Gantry filling: Admin users can fill the relevant material to bowser once order is approved ,whether the created order will be filled or not he can decide.
  - ❖ Order View: Admin user can view the order’s progress through the system

- **Bowser Management Module**
  - ❖ Bowser creates: Admin user can create the bowser details
  - ❖ Bowser Profile change: Admin users can update and change the bowser-related details

- **Reports Module**
  - ❖ Generate order-related operation reports
  - ❖ Generate bowser management reports
  - ❖ Generate shed owner-related reports

2.2.2 Non-functional Requirement

- **Reliability:** A system, product, or service's reliability refers to its expected function or produce its desired results in a consistent and dependable manner.
- **Performance:** The system should be able, admin or dealer access at any time without any delay. Because it is very useful to create an order in the busy schedule of the user.
- **Availability:** The system should be available when a user requests a system.
- **Portability:** It should be supported for multiple operating systems as well as should be a platform independence
- **User friendly:** The system should be easy to operate as well and it should contain a user-attractive interface, documentation, and manuals.
- **Effectiveness:** By using the system user can be able to overcome the existing manual process drawbacks.

2.3 Review of Similar System

A software program called Purbis offers a fuel ordering system. The Purbis fuel ordering system is intended to simplify the ordering process, making it more practical and effective for both customers and gasoline suppliers. The ordering procedure for Purbis gasoline includes the following specifics:

User registration and authentication: Users can register and set up accounts on the system. Users can then access the system's functionalities by securely logging in with their credentials.

Fuel Ordering: Customers can use the system to place orders for fuel. They can choose the required fuel type and quantity, delivery site, and preferred delivery time.

Fuel Supplier Management: The system allows fuel providers to keep track of their inventory, schedules for deliveries, and products. Suppliers may track orders, adjust fuel availability, and update prices.

Delivery Management: The system enables effective management of gasoline deliveries. For prompt and economical gasoline deliveries, suppliers can allocate delivery staff, monitor delivery progress, and optimize routes. Payment processing is supported by the system, enabling users to pay for fuel orders with a variety of payment options, including credit/debit cards, online wallets, and cash on delivery.

Notifications and Alerts: To keep customers and suppliers informed of changes to order status, delivery confirmations, and other pertinent information, the system sends notifications and alerts to both parties. Detailed reporting and analytics options are offered by the Purbis fuel ordering system. It produces data on fuel usage, sales, delivery performance, and other important metrics, assisting suppliers and clients in making knowledgeable choices. Mobile-Friendly Interface: The system is made to be mobile-friendly, making it simple to place orders or manage delivery while on the go. Customers and suppliers may access the platform from their smartphones or tablets.

FuelMe is a cloud-based fuel ordering system that also has the ability to automatically reorder fuel. The major goal of this is to make fuel purchases faster, more dependable, and more affordable. It also has the following drawbacks: gantry filling notification, invoice notification, gate exit notification, and GPS tracking. This major runs on a mobile application. Although there is no GPS, a live order tracking method has been built for mobile. The following figure shows how the Fuel Me system achieved their solution system

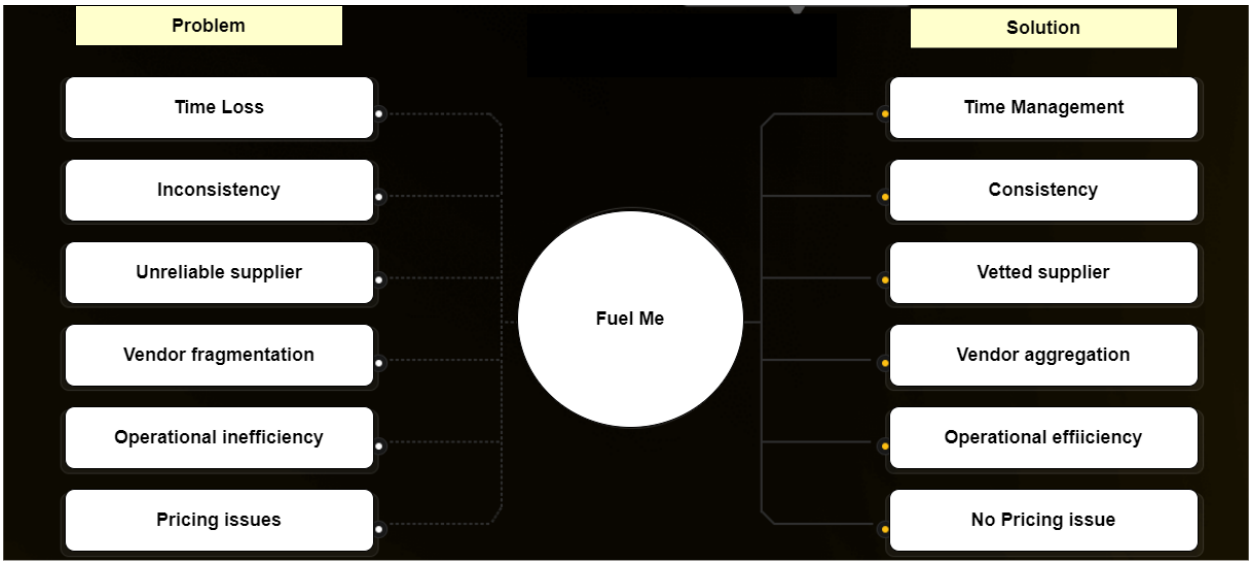


Figure 2: Fuel Me Problems and Solutions

Peerbits is a fuel ordering platform that integrates a high-cost GPS tracking technology with a fuel ordering platform. Additionally, this approach aims to open up new economic prospects for owners of gas stations and gasoline delivery aggregators. Additionally, this system does not create invoices or provide gantry filling, invoicing, gate exit, or invoice notification notifications. Both the driver app and the customer app are available as mobile apps under this system.

Comparison between the above existing system and proposed system details are shown in the below table to get a summary of the idea above.

Table 1: Features of the existing system and proposed system

Features	Purbis	Peerbits	FuelMe	Web-Based Fuel Ordering System for Dealers (Fuel manager
Register shed account	√	√	√	√
Make fuel order	√	√	√	√
Approve order	√	√	√	√
Reject order	√	√	√	
Manage order	√	√	√	√
Gantry filling notification				√
Invoice notification				√
Generate invoice			√	√
Gate exit notification				√
GPS tracking	√	√		
Create user accounts	√	√	√	√
Online payment	√	√	√	√
Choose location	√	√	√	
Bowser management		√	√	√
Generate reports	√	√	√	√

2.4 Related Technologies

The Prototype model is one of the major software development life cycle methods, when the user exactly doesn't know the project requirement beforehand then this is well suited. In this model, a prototype of the end system is first developed, tested, and refine according to the feedback. Then there after developing the system and testing it after getting the acceptable prototype. This is an iterative process between the developer and the end user.This is a web-based system and set of technologies used to design, develop, and run the software such as MVC architecture (Framework), database, Programming language (PHP), and web server as the server side (backend)of the system. JavaScript, CSS, HTML, and applications use as client-side (frontend) technology. Hence, there is a lot of user interaction with the web-based systems most dealers are preferred to use the web-based system according to the background study.

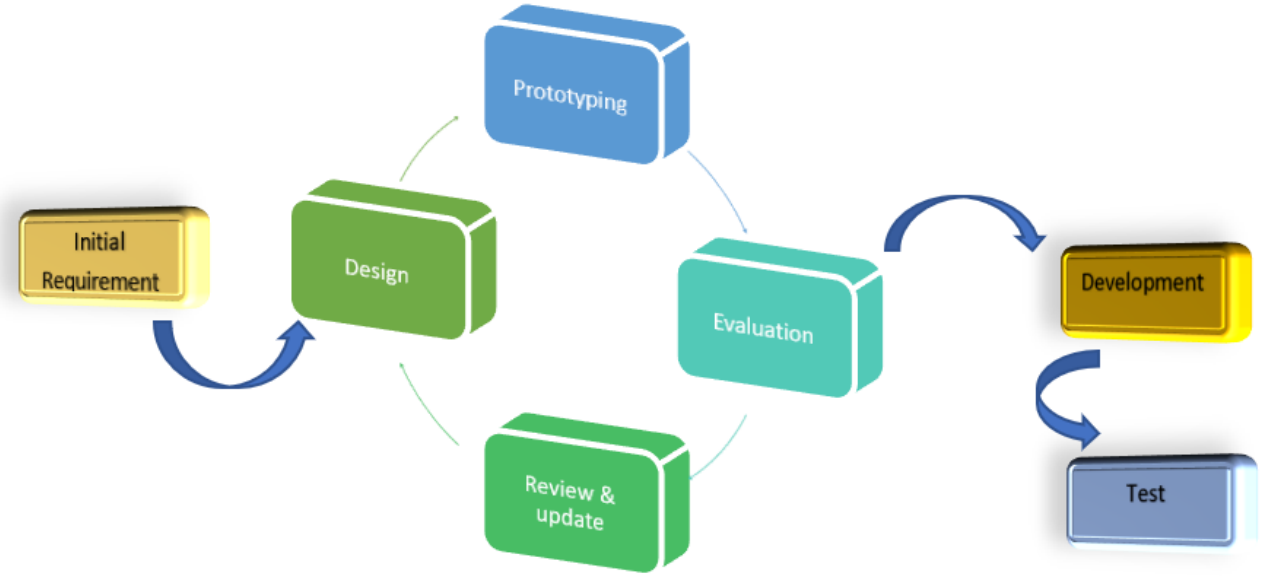


Figure 3: System development method

Chapter 3 – Design

3.1 Introduction

Between requirements analysis and implementation, software design is a crucial stage in the software development lifecycle. Planning how a software system will work is really important. It needs to do what users want and be easy to use, but also work well and grow as needed. This overview explores the basic ideas and ways of thinking that help make software development go smoothly.

The foundation of successful software development is good software design. It affects the ultimate product's quality, the development process, and the project's success. An effective software system reduces the possibility of errors and ensures long-term viability since it is simpler to comprehend, adapt, and maintain.

Making a number of crucial judgments is necessary for software design. Which programming language is ideal for this project? How should data be efficiently stored and retrieved? Should the system be modular or a single piece? Trade-offs are frequently included in these choices. While putting security first may slow down processing performance, choosing a user-friendly design may result in more complexity. When making a system, it's important to find a good balance between different goals like making sure it works well, runs fast, stays safe, can grow, and is easy to use.

The prototyping model is a method for developing systems that involves creating a model, testing it, and then making any necessary revisions until, at long last, a reliable model is created from which the complete system or item may now be created. This methodology performs effectively in situations where not all of the project needs are initially fully known. There is an iterative, experimental process between the designers and the clients. This method has a few main steps including defining the core needs, creating a working prototype, testing the working prototype, and modifying or expanding the requirements.

3.2 Related Design Strategies

Because the internet is spread out and user interfaces need to work on different devices, and it's tricky to manage how data moves around online, making web systems requires special thinking. The following software design tactics were specifically chosen for web-based systems because the advantages are very high.

- Divide the system into a client-side (the browser) and a server-side (the backend) component using a client-server architecture. Scalability is made possible by this division since you may separately scale and optimize each component.
- In order to provide a consistent user experience on computers, tablets, and smartphones, responsive web design involves creating user interfaces that can adjust to various screen sizes and devices.
- Maintenance and installation are easy.
- Development is cost-effective.

3.2.1 Object Oriented analysis and design (OOAD)

A process called object-oriented analysis and design (OOAD) is used to create software systems that adhere to object-oriented concepts. It entails decomposing a challenging issue into smaller, more manageable pieces, and then designing and arranging these pieces with object-oriented principles. An overview of the main procedures and ideas in object-oriented analysis and design is given below

- Recognize and collect from stakeholders the functional and non-functional requirements of the software system.
- To depict the interactions between system actors and the system itself, create use case diagrams.
- Determine the relationships between the main system components.
- Dividing the system into more compact, coherent, and loosely connected components will help.

Modular, maintainable, and adaptive software systems are the goals of object-oriented analysis and design. In order to model and represent real-world items and their interactions in the software domain, it places a strong emphasis on the use of objects, encapsulation, inheritance, and

polymorphism. Developers can build strong and adaptable software systems that closely match the problem domain and effectively address user needs by adhering to these guidelines.

3.2.2 MVC Architecture

A design pattern called Model-View-Controller (MVC) architecture is used in software development to divide an application's responsibilities into three primary parts: Model, View, and Controller. This division aids in managing complexity, enhancing maintainability, and fostering reuse in software systems. Here is a brief description of each MVC component:

The separation of functions is the primary principle of MVC. It is possible to modify one aspect of the system without impacting the others by keeping the Model, View, and Controller distinct. The codebase becomes more modular and is simpler to maintain as a result. For instance, you can modify the View (data presentation) without affecting the Model (data processing logic) or the Controller (user input handling logic).

The Model maintains data management (for example, database queries), the View manages UI presentation (for example, HTML templates), and the Controller coordinates the interaction between the Model and the View depending on user requests (for example, URL routing). MVC is frequently used in web development.

3.3 System Architecture

Any software project’s foundation is its system architecture, which offers a tactical plan for how all the parts fit together. A well-architected software system necessitates careful thinking to accomplish usefulness, performance, and maintainability, much as a well-designed structure necessitates meticulous planning to assure stability, usability, and beauty. The fundamental components, interactions, and technologies that make up the software solution are defined by the system architecture. It is comparable to generating a roadmap to direct developers in building a stable and unified digital environment. System architecture lays the foundation for the successful design, implementation, and evolution of a software system by defining the links between distinct modules, defining communication protocols, choosing appropriate technologies, and handling scalability, security, and other issues. It’s an important first step that establishes the framework.

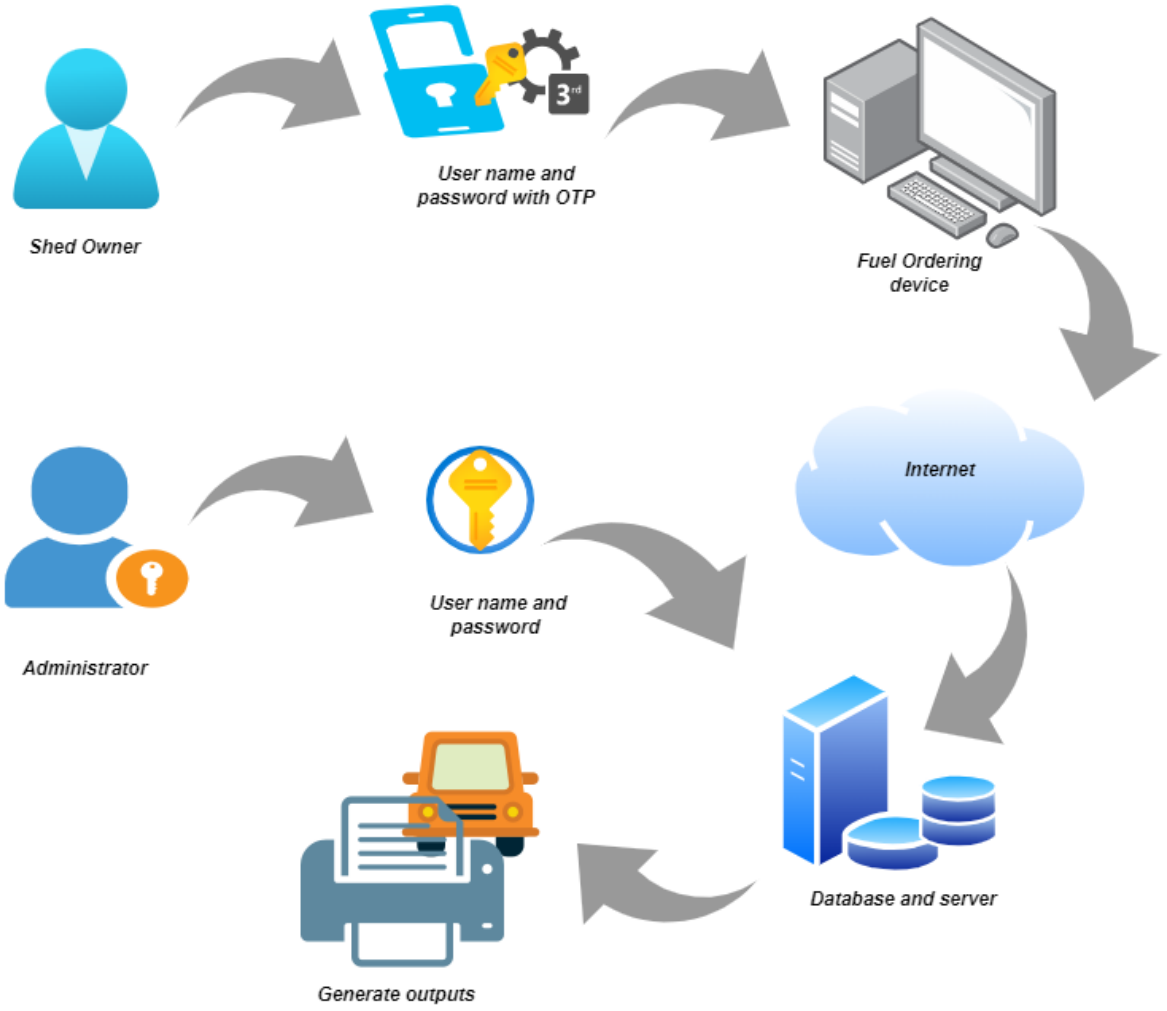


Figure 4: System Architecture Diagram

3.4 UML Diagrams

A graphical representation that is widely used in the fields of software engineering and system design is the Unified Modeling Language (UML) diagram. It acts as a uniform notation system, providing a visual framework to express the complex structure, dynamic behavior, and relationships present in various systems and processes. Classes, objects, interactions, and component relationships can all be represented using a variety of symbols, shapes, and relationships in UML diagrams. Both technical and non-technical stakeholders can benefit from this visual language's ability to clarify complicated ideas and help them understand the basic construction and operation of a system. Throughout the whole software development lifecycle, UML diagrams are essential because they help with requirements analysis, design structuring,

code implementation, and documentation. UML diagrams essentially go above the limitations.

3.4.1 Use Case Diagrams

An illustration that provides a high-level view of the functional requirements of a system from the perspective of its users or external entities is a use case diagram, a key tool in the Unified Modeling Language (UML). It functions as an effective communication tool, allowing stakeholders to understand and see the interactions between different actors (users, other systems, or entities), as well as the features offered by the system.

Actors and use cases are shown as stick figures and ovals, respectively, in a use case diagram. The interactions or functionalities that each actor can start or take part in are represented by the lines connecting actors to use cases. The diagram's clarity and simplicity make it especially helpful for requirement analysis since it assists in defining and identifying the essential features that the system must provide to meet user needs.

By aiding in the identification of system boundaries, the assessment of critical user interactions, and the discovery of viable use case situations, use case diagrams serve as a foundation for the development process. This kind of model makes it easier to match user expectations with development efforts, ensuring that the finished system effectively satisfies those demands.

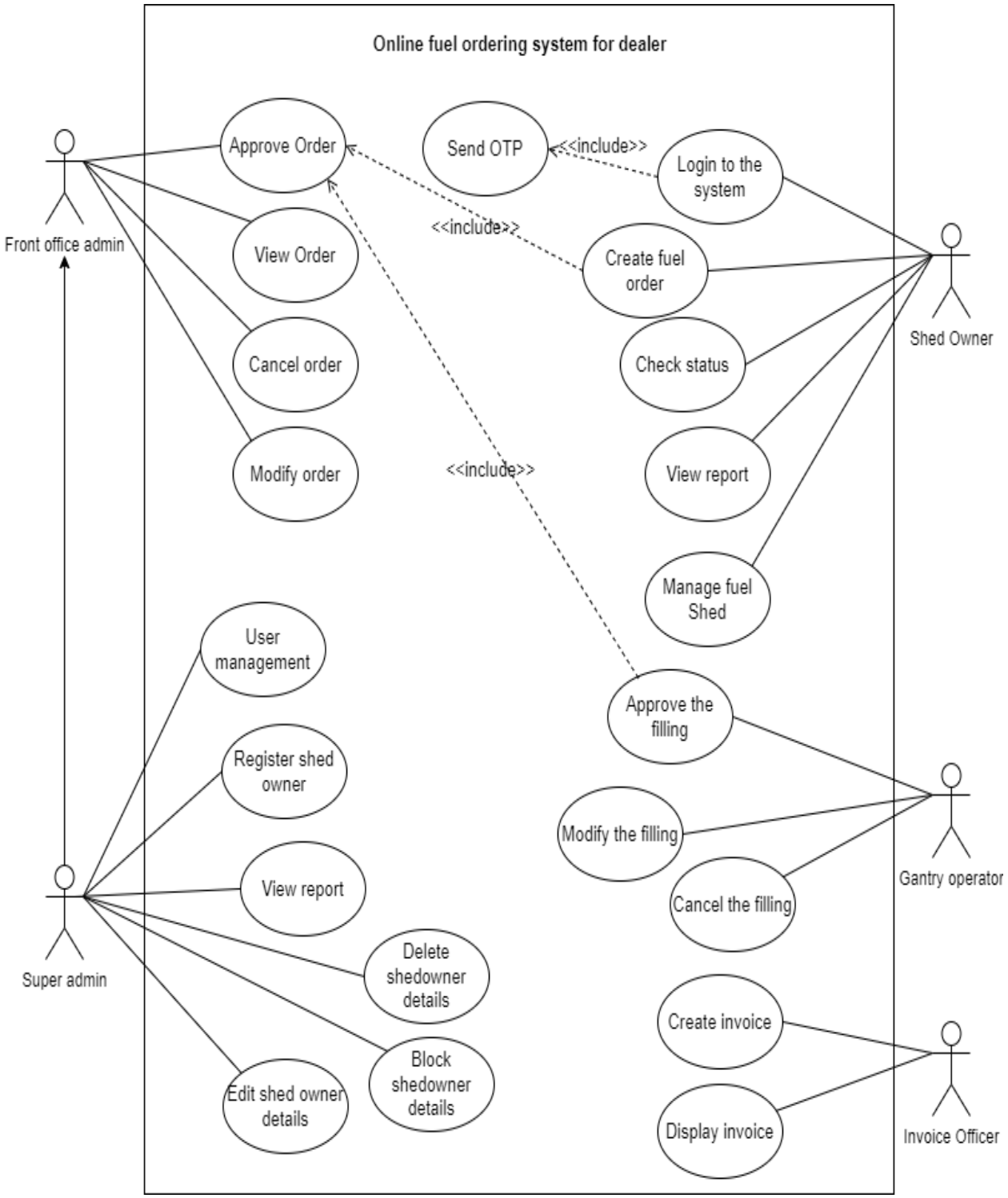


Figure 5: Use Case Diagram

3.4.1.1 Use Case Scenarios

Table 2: Use case scenario 1

Use case	**Description**
Name	Place a fuel order
Description	Shed owner place an order
Actors	Shed Owner
Scenario	1. The Shed owner logs into the system 2. The user selects material type and quantity 3. The User selects create order button.
Alternative flow	1. User can add another fuel order by click on add another button

Table 3: Use case scenario 2

Use case	**Description**
Name	Approve fuel order
Description	Admin can approve the fuel order which is requested by shed owner.
Actors	Super admin, front office admin
Scenario	1. Admin log into the system 2. Admin view the new order list 3. Select the order from the list 4. Admin approve the order for proceed
Alternative flow	1. Admin can reject the order as well as cancel the order

3.4.2 Activity Diagram

An essential component of the Unified Modeling Language (UML), an activity diagram is a graphic representation of how decisions, actions, and systems flow inside a system or process. This diagram type offers a simple and understandable approach to representing the order of events and the reasoning underlying them, making it particularly useful for representing dynamic parts of software systems, business processes, and workflows.

Activities are shown as rounded rectangles in an activity diagram, and arrows linking these rectangles show the direction of control moving between activities. Diamond-shaped decision points symbolize decision points that use conditional logic to direct the flow leading to various branches of activities. As parallel or concurrent execution paths are indicated by forks and joins, complicated scenarios involving numerous concurrent operations can be modeled.

Activity diagrams are frequently used for system comprehension, process design, and requirement analysis. They assist in determining the series of actions necessary to achieve a particular objective, recording the order of execution and any decision points that might affect the result. This level of specificity makes it easier for stakeholders to work together since it gives them a shared visual language to talk about and improve the process logic.

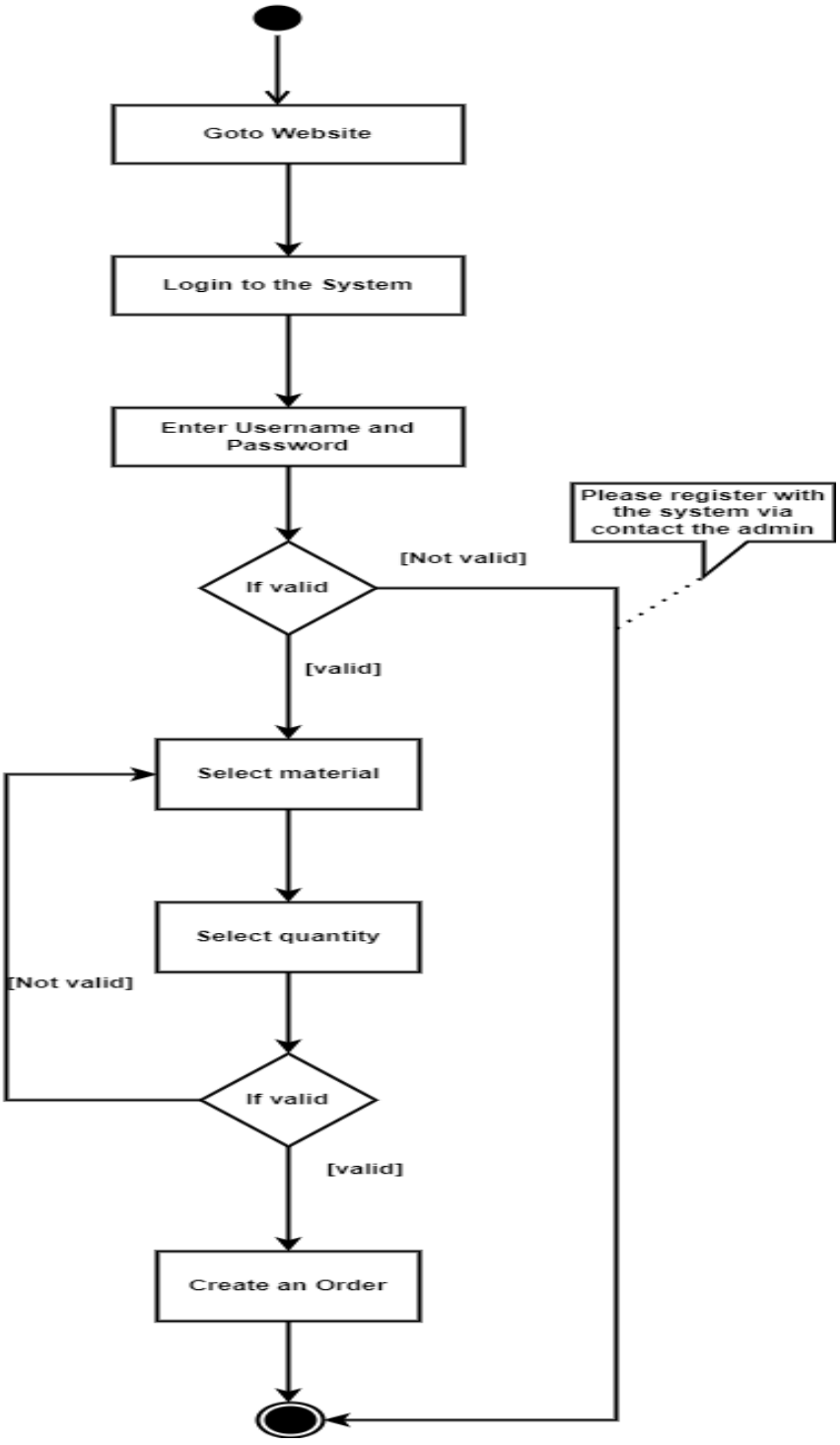


Figure 6: Place order by Shed Owner activity diagram

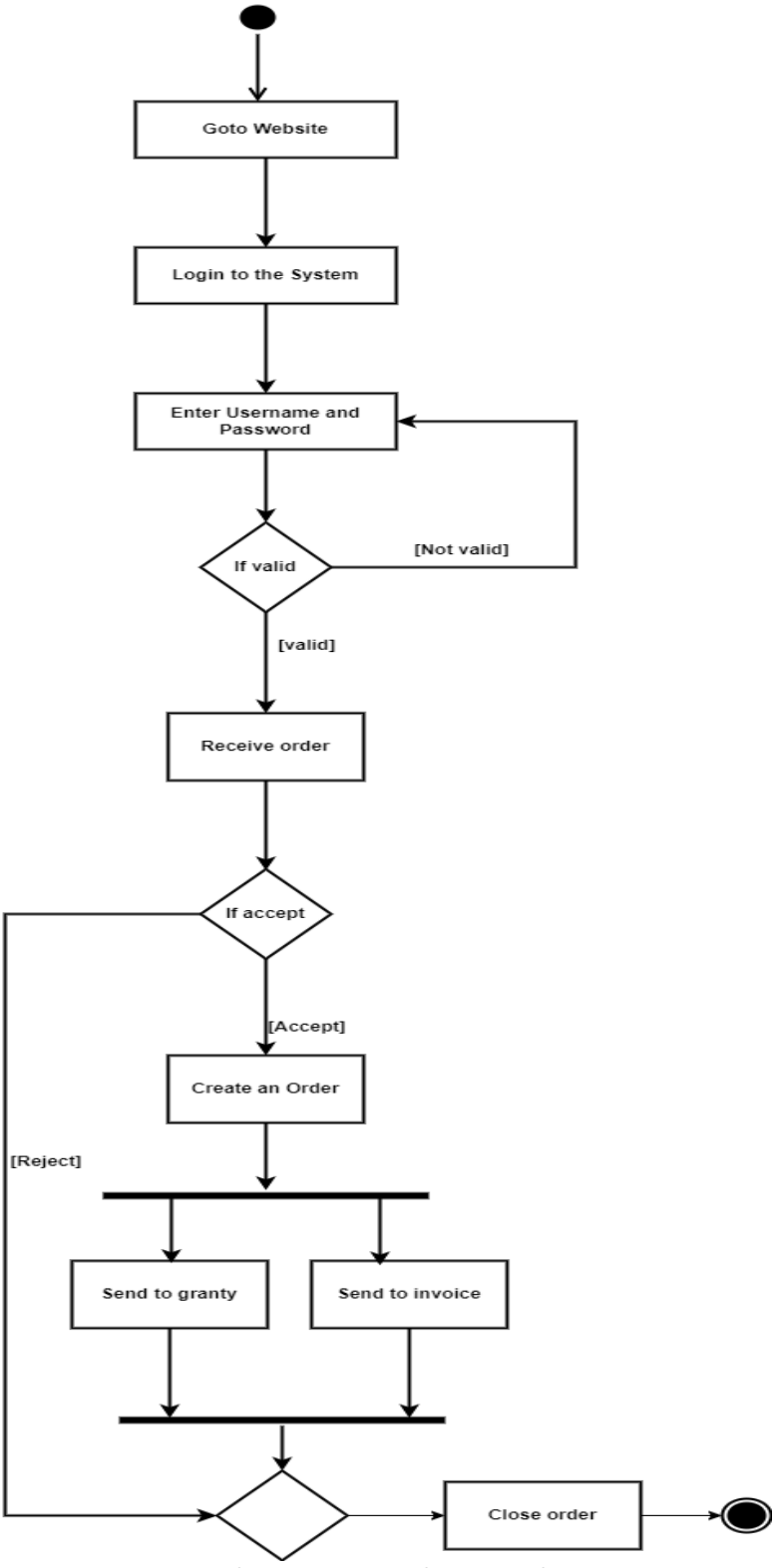


Figure 7: Employee Create an order activity diagram

3.4.3 Sequence Diagram

A visual representation that captures the interactions and chronological order of messages sent between distinct objects or components inside a system is a sequence diagram, which is a key component of the Unified Modeling Language (UML). This particular form of diagram is particularly adept at capturing the dynamic behavior of software systems and processes, offering a thorough explanation of how objects interact to carry out particular functionalities.

In a sequence diagram, messages are represented by arrows that go horizontally between lifelines, which stand in for objects or actors as lifelines. Stakeholders can comprehend how objects communicate, what activities they take, and how they react to one another's demands thanks to the messages' orderly transmission.

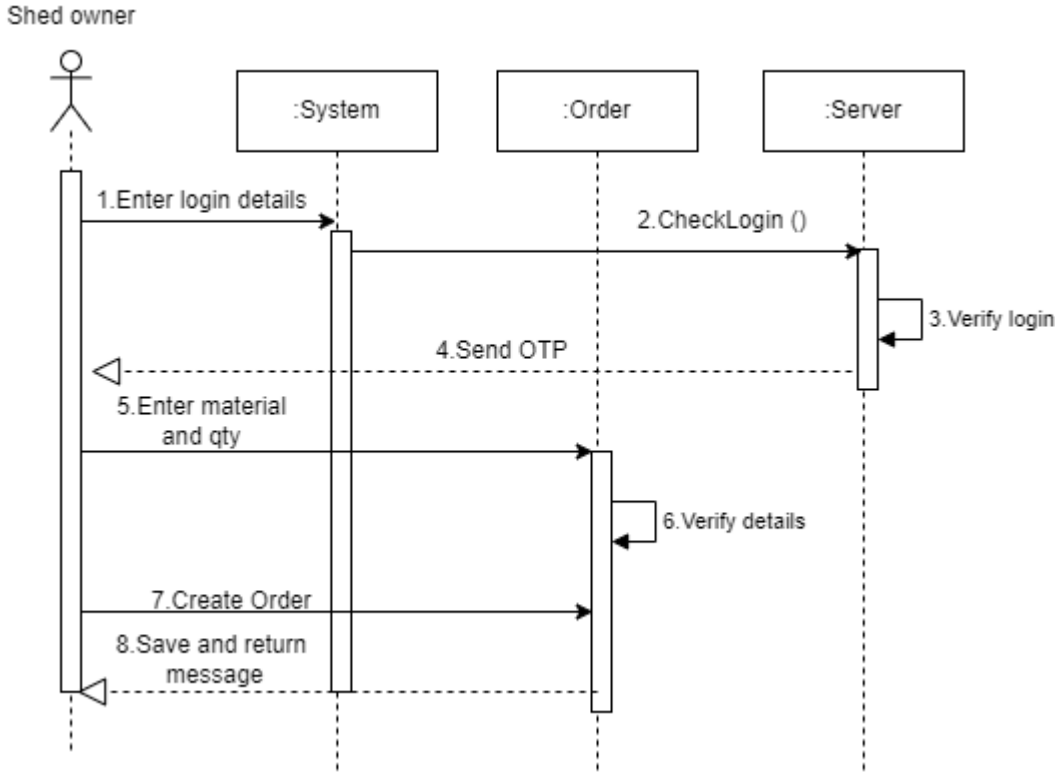


Figure 8: Place order by Shed Owner sequence diagram

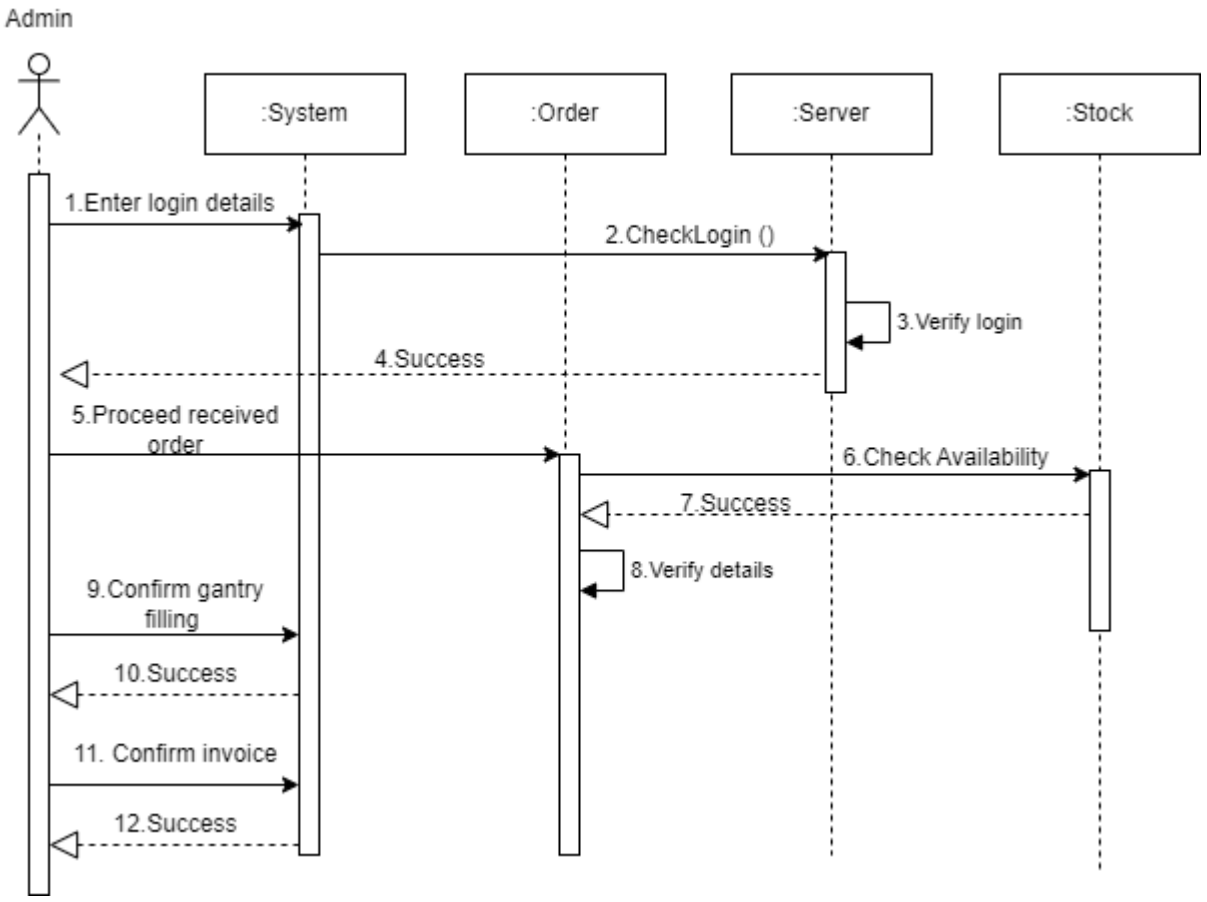


Figure 9: Employee process an order sequence diagram

3.4.5 Database Design

For the online fuel ordering system designed for dealers, the database must be structured one to ensure efficient management of orders, dealers' information, and transactions. At the core of the database design, there should be tables representing essential entities such as dealers, fuel products, orders, transactions.

The "fillingstation" table and "users" table would store detailed information about each dealer, including their unique identifier, contact details, and any relevant business information. This table managing dealer-related operations and interactions within the system.

The "Material" table would contain comprehensive information about the available types of fuel, including attributes like product ID, name, description, price, and fuel type (e.g., petrol, diesel).

The "Orders" and "order Items" table would track all orders placed by dealers, with each order linked to a specific dealer and containing details such as order ID, dealer ID, date/time of order, delivery address, and status (e.g., pending, processing, completed).

Relationships between these tables would be established through foreign key constraints to maintain data integrity. For instance, each order would be linked to a specific filling station through a foreign key referencing the user ID in the "users" table.

Finally, designing the database schema with these considerations in mind, the online fuel ordering system can effectively support the operational needs of dealers while facilitating order processing, and transaction tracking.

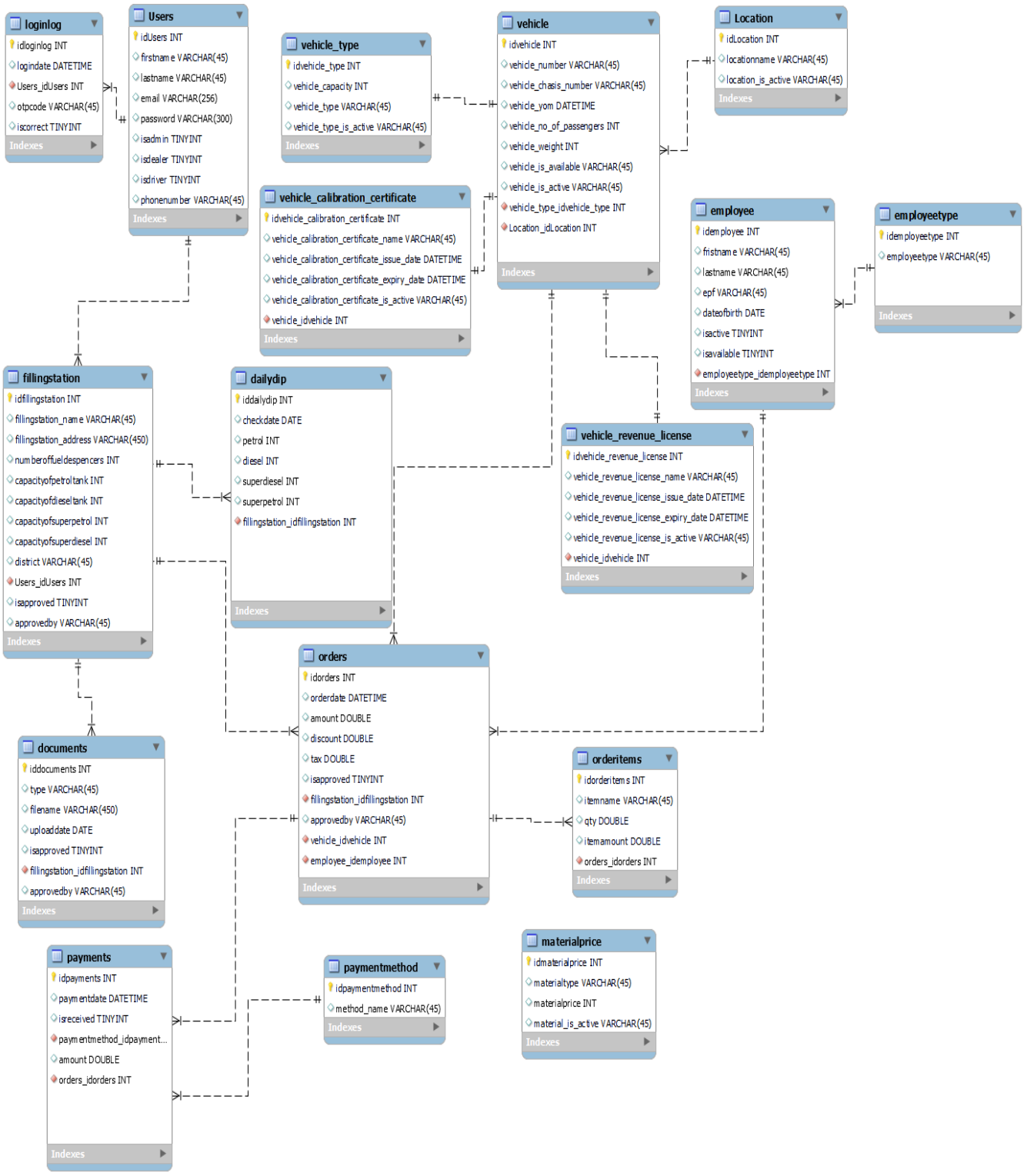


Figure 10: ER Diagram

3.4.6 User Interface Design

The goal of user interface design is to make things simple for users to use on computers, mobile devices, and websites. When done correctly, it assists consumers in learning and using things without feeling confused or frustrated. Additionally, it improves the appearance of objects and promotes user confidence. By avoiding issues and ensuring user satisfaction, good user interface design helps users save time and money. Additionally, it is aware of new technologies and consumer trends, so it's always attempting to improve. In order to have things go smoothly and seem good for everyone. Here, I am using draw.io to create some major interfaces for the project.

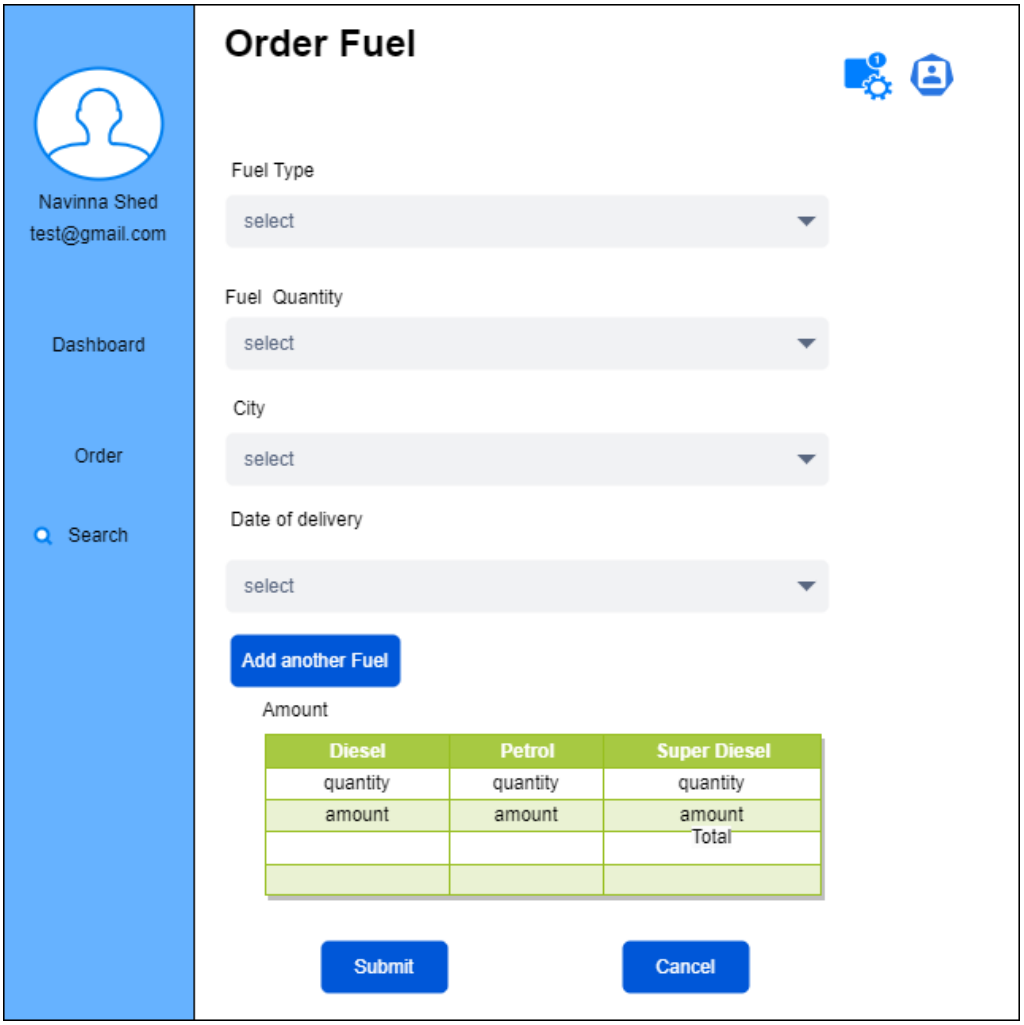


Figure 11: Place an Order by Shed owner

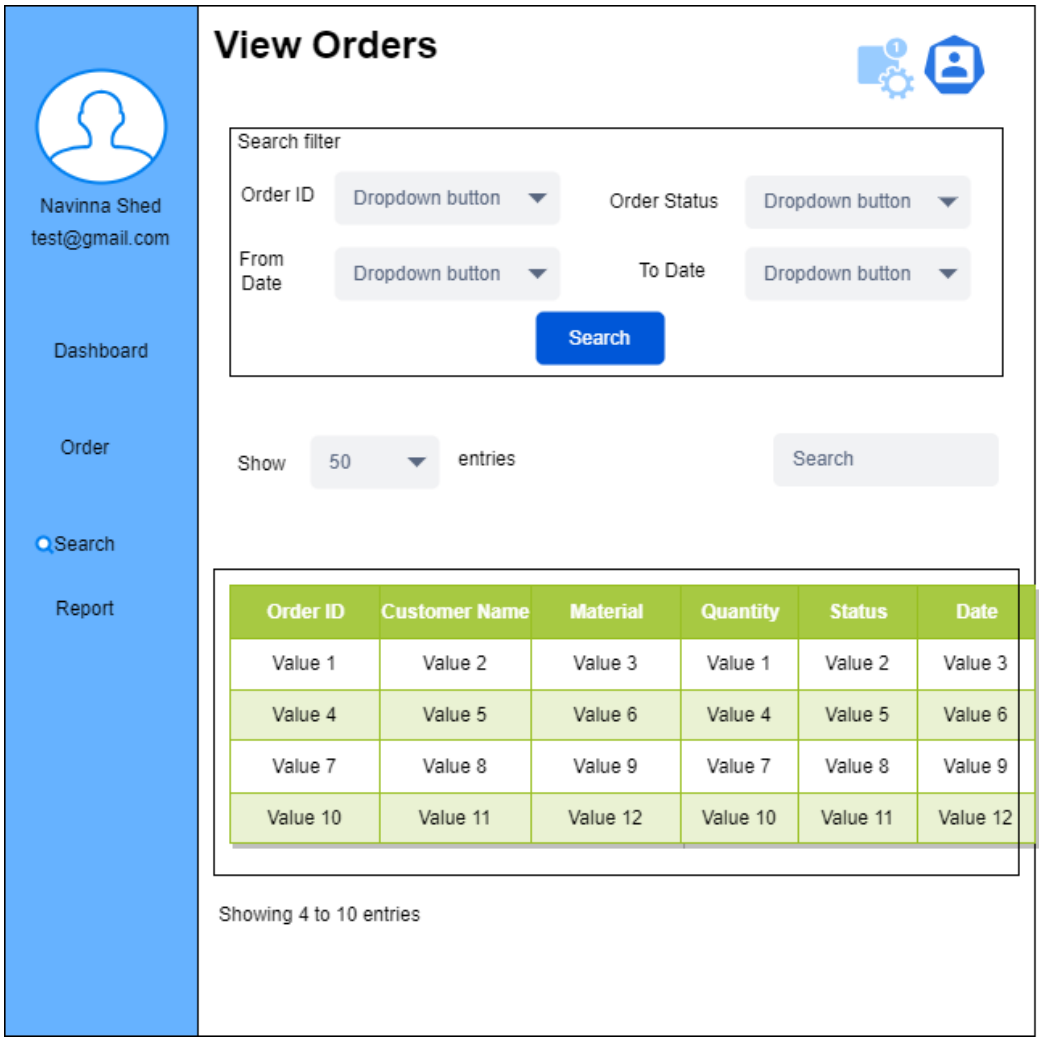


Figure 12: Create an order by Employee

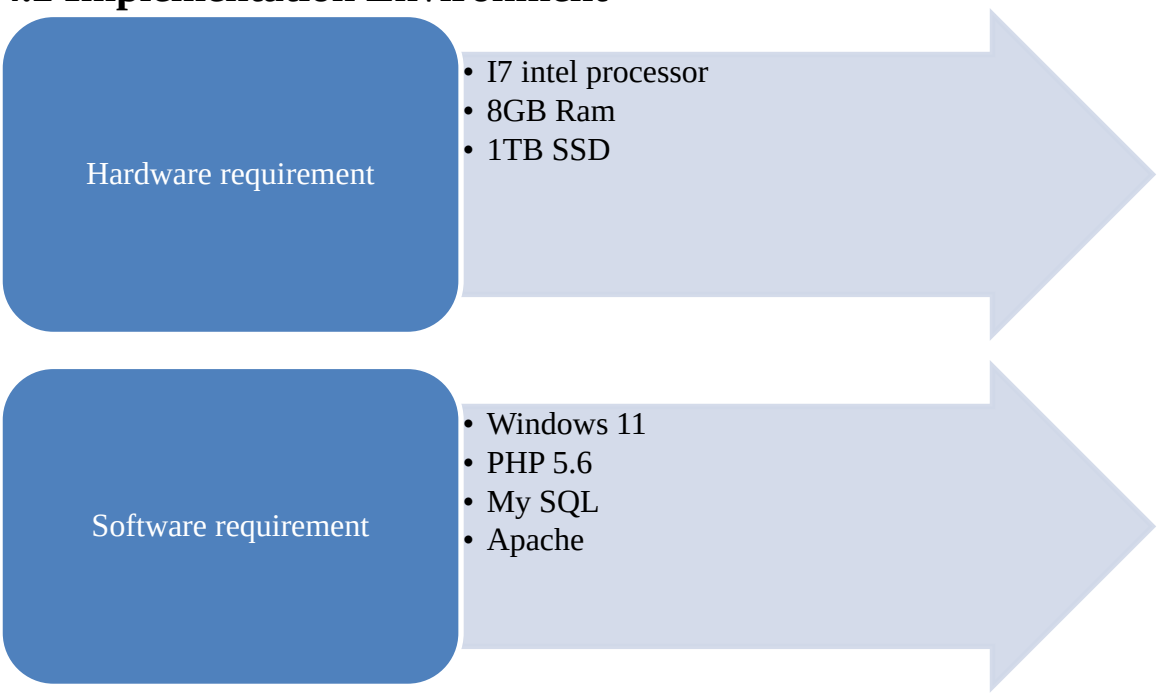
Chapter 4 – Implementation

4.1 Introduction

The implementation phase is carried out in accordance with what was planned during the design stage utilizing the proper tools and procedures once the design phase has been successfully completed. A software development life cycle includes this stage, which is crucial. This stage results in a system that is ready for execution. The system was initially broken down into functionally distinct modules, and then all of the produced modules were finally combined to form the system as a whole. In this chapter, the primary tools, techniques, and code segments that are utilized to implement the system in the implementation environment are the main points of emphasis.

Following diagram shows the using hardware requirement and software requirement for the system.

4.2 Implementation Environment



4.3 System Development Tools and Technologies

The following tools and techniques are used while developing the system. A brief description is also given in addition.

- **Code Ignitor 3**

The system is coded using this program. Popular web development framework CodeIgniter supports the Model-View-Controller architectural style. Because it offers superior security and supports effective search engine optimization, this framework was chosen.

- **PHP**

The back end of the system is developed using this server-side scripting language. This was chosen because it offers quick loading times, versatile use, and learning-friendly documents.

- **My SQL**

This database management program is used to manage the system's database. This is the greatest database management solution when security and data protection are taken into consideration. The best database for this poll management system is MySQL because it demands data security.

- **HTML**

Hypertext Markup Language, or HTML, is a unique kind of coding that is used to construct web pages on the internet. It resembles the foundational elements of websites. To put it simply, HTML employs tags to instruct web browsers how to display material (such as p> for paragraphs or img> for images). Consider it a collection of guidelines that a browser follows to determine where to place text, graphics, and links on a webpage. In order to display a web32 page the way it was intended when you visit a website, your browser must first understand the HTML code. The fundamental language that powers the internet is HTML.

- **Java Script**

JavaScript is a programming language that gives websites functionality and interactivity. It serves as the equivalent of glue for web pages. Simply put, JavaScript enables web designers to build dynamic features like responsive components, interactive forms, and pop-up notifications. It's common for JavaScript code to be running in your web browser when you click a button that modifies the content of a page

- **Bootstrap/CSS**

Bootstrap is a common framework for building aesthetically pleasing and responsive webpages

and online applications. It functions something like a ready-made toolkit that makes it easier for developers to create web pages. To help you organize and create your website, Bootstrap offers a collection of pre-made, reusable elements like buttons, navigation bars, and grids. Additionally, because of its adaptable design elements, it makes sure that your website appears excellent and works properly on a variety of gadgets and screen sizes, including desktop computers and smartphones.

- **Draw.io**

Draw.io is a tool used for creating diagrams, charts, and flowcharts. This tool is very helpful for creating a prototype interface, users can add shapes, text, and images to create professional-looking diagrams. This tool is needed when creating a project.

- **Visual studio code (VS code)**

The Visual Studio Code (VS code) is used for writing and editing the code for this project. It has a user-friendly interface with features like syntax highlighting and other programming features.

- **Data Tables**

The Data Tables are the tool used to create interactive tables for displaying data on websites. It provides to organize and present data in a structured format. The Bootstrap integration, the tables can be styled easily to match the overall design of the website.

- **MS Word**

Microsoft Word is a widely used word processing program that helps users create, edit, and format documents easily. Anyone can write letters, reports, essays, and other documents efficiently. Its various features, such as spell check, grammar check, and formatting tools. In this project. I used this tool for write the documents.

- **Google Chrome**

When developing a web-based fuel ordering system, I try to utilize Google Chrome as my primary web browser for testing and debugging. In this process open my project in Google Chrome and navigate through the system's various features to ensure compatibility and functionality.

4.4 Major Code Segment

Modules that can be reused are used to arrange the system's code. The main code modules are listed here, along with a succinct description of each function. Comments are added to help identify the functionality of a particular code.

4.4.1 Database Connection

In every project there is main part, database connection is the most valuable part when we start the project. It will not check whether it is web based, cloud base or standalone, in every data base connection are used to operate insert values, update values, retrieve values and delete values. This is a configuration snippet for connecting to a database in CodeIgniter 3. It will define the settings needed to connect to a MySQL database named "fueldb2" running on the localhost server. The username and password for the database are set as "root" and blank, respectively. The configuration specifies the UTF-8-character set and collation for database operations.

Figure 13 : Database connection code snip

4.4.2 Login With OTP

The checklogin() function is responsible for handling the login process. It first checks if the request method is POST. If not, it sets a flash message and loads the login view. If the request method is POST, it validates the email and password fields using CodeIgniter's form validation library. If form validation passes, it retrieves the email and password from the input fields and hashes the password. It then checks the credentials against the database using the checkcredential() method of the UsersModel. Depending on the user's role (admin, dealer, or employee), it sets the user role variable. If the user exists in the database, it creates session data for the user and generates an OTP. The OTP is then inserted into the database and sent to the user via SMS. Finally, the user is redirected to the OTP verification page. If the user does not exist in the database or if form validation fails, it sets an appropriate error message and reloads the login view. The following code segment snip shows the described output.

- **Login Verified code Segment**

Below figure 14 shows the login verification code.

```php
public function checklogin()
{
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {

        $this->form_validation->set_rules('email', 'email', 'required|valid_email');
        $this->form_validation->set_rules('password', 'password', 'required|min_length[0]|max_length[15]');

        if ($this->form_validation->run()) {

            $email = $this->input->post('email');
            $password = $this->input->post('password');

            $password_hash = hash_hmac('sha256',$password, $this->config->item('systemkey'));

            $result = $this->UsersModel->checkcredential($email, $password_hash);

            $userrole = "";

            if($result != false){
                $isdealer = $result[0]->isdealer;
                if($isdealer == 1){
                    $userrole = 666;
                }

                $isadmin = $result[0]->isadmin;
                if($isadmin == 1){
                    $userrole = 777;
                }

                if($isadmin == 0 && $isdealer == 0){
                    $resultusr = $this->LoginlogModel->usertype($result[0]->idUsers);
                    $userrole = $resultusr[0]->employeetype_idemployeetype;
                }

            }
        if(sizeof($result)>0){
            $data = array(
                'user_id' => $result[0]->idUsers,
                'username' => $result[0]->firstname." ".$result[0]->lastname,
                'email' => $result[0]->email,
                'isactive' => 0,
                'phone' => $result[0]->phonenumber,
                'userrole' => $userrole,
                'is_otp_verify' => false
            );

            $this->session->set_userdata($data);

            $otp = random_int(100000, 999999);

            $data = array(
                'logindate' => date("Y-m-d H:i:s"),
                'Users_idUsers' => $result[0]->idUsers,
                'otpcode' => $otp,
                'iscorrect' => 0,
                'isdelete' => 0,
            );

            $resultid = $this->LoginlogModel->insert_loginlog($data);

            sendSms($result[0]->phonenumber, "Your Test OTP code is :".$otp);

            redirect("otp");

        }else{
            $this->session->set_flashdata('error', 'Error in Credentials');
            $this->load->view('login/login');
            }


        }else{
            $this->session->set_flashdata('error', validation_errors());
            $this->load->view('login/login');
        }
    }else{
        $this->session->set_flashdata('error', 'Bad Request');
        $this->load->view('login/login');
    }
}
```

Figure 14: Login verify with OTP snip

- **OTP verification code segment.**

This following figure shows how the otp is verified once enter the otp number.

```
public function checkotp()
{
    if($this->session->userdata('email') == true && $this->session->userdata('is_otp_verify') == false){
        $userid = $this->session->userdata('user_id');
        $userotp = $this->input->post("otpcode");

        $result = $this->LoginlogModel->checkotp($userid);

        if(strcmp($result[0]->otpcode,$userotp)==0)
        {
            $data = array(
                'is_otp_verify' => true,
                'isactive' => 1
            );

            $this->session->set_userdata($data);

            redirect('dashboard');
        }else{
            redirect('otp');
        }
    }
    else if($this->session->userdata('email') == true && $this->session->userdata('is_otp_verify') == true){
        redirect('dashboard');
    }
    else if($this->session->userdata('email') == false && $this->session->userdata('is_otp_verify') == true){
        redirect('login/auth');
    }
    else if($this->session->userdata('email') == false && $this->session->userdata('is_otp_verify') == false){
        redirect('login/auth');
    }

}
```

Figure 15: OTP verified code segment snip

- **Register user code segment.**

To register a user with the online fuel ordering system, the provided code segment manages the registration process. Upon receiving a POST request, the system validates user input fields such as first name, last name, NIC, email, password, and phone number using predefined rules. If validation is successful, it checks if the NIC and email are unique within the system. If unique, the system formats the phone number appropriately and prepares user data for insertion into the database. After successfully registering the user, the system redirects them to the login page with a success message. If any errors occur during the registration process, appropriate error messages are set, and the user is redirected back to the registration page for corrections. If the request method is not POST, the system sets a "Bad Request" error message.

```php
public function registeruser()
{
    if ($_SERVER['REQUEST_METHOD'] === 'POST') {
        $this->form_validation->set_rules('firstname', 'First Name', 'required|min_length[1]|max_length[45]');
        $this->form_validation->set_rules('lastname', 'Last Name', 'required|min_length[1]|max_length[45]');
        $this->form_validation->set_rules('nic', 'NIC', 'required|min_length[1]|max_length[12]');
        $this->form_validation->set_rules('email', 'Email', 'required|valid_email|min_length[1]|max_length[256]');
        $this->form_validation->set_rules('password', 'Password', 'required|min_length[1]|max_length[300]');
        $this->form_validation->set_rules('phonenumber', 'Phone Number', 'required|min_length[1]|max_length[11]');

        if ($this->form_validation->run()) {
            // Check if NIC is unique
            $nic = $this->input->post('nic');
            if ($this->UsersModel->is_nic_unique($nic)) {
                // Check if email is unique
                $email = $this->input->post('email');
                if ($this->UsersModel->is_email_unique($email)) {
                    $this->load->helper('string');

                    $phonenumber = $this->input->post('phonenumber');
                // Format phone number
                if (strpos($phonenumber, '+94') !== false) {
                    // Remove '+' if present
                    $phonenumber = str_replace('+94', '94', $phonenumber);
                } elseif (strpos($phonenumber, '0') === 0) {
                    // Add '94' if it starts with '0'
                    $phonenumber = '94' . substr($phonenumber, 1);
                }

                    $data = array(
                        'firstname' => $this->input->post('firstname'),
                        'lastname' => $this->input->post('lastname'),
                        'nic' => $nic,
                        'email' => $email,
                        'password' => hash_hmac('sha256', $this->input->post('password'), $this->config->item('systemkey')),
                        'phonenumber' => $phonenumber,
                        'isadmin' => 0,
                        'isdealer' => $this->input->post('dealer'),
                        'isdriver' => 0,
                        'isdelete' => 0,
                    );

                    $resultid = $this->UsersModel->insert_users($data);

                    if($resultid > 0){
                        $this->session->set_flashdata('message', 'Successfully Registered');
                        redirect("login");
                        return;
                    } else {
                        $this->session->set_flashdata('error', 'Error in Register');
                        redirect("register");
                    }
                } else {
                    $this->session->set_flashdata('error', 'Email is already registered');
                    redirect("register");
                }
            } else {
                $this->session->set_flashdata('error', 'NIC is already registered');
                redirect("register");
            }
        } else {
            $this->session->set_flashdata('error', validation_errors());
        }
    } else {
        $this->session->set_flashdata('error', "Bad Request");
    }

    $this->load->view('login/login');
}
```

Figure 16: Registration code segment snip

Chapter 5 – Testing and Evaluation

5.1 Introduction

Every success of the software application development, the main part of the software development life cycle is software testing and evaluation. Testing increase the trust of the software system by checking the every part of software, finding bugs, and meet the user expected system behavior. It may help to improve user satisfaction about the completed software system. The important part of quality assurance is developers can minimize costs and effort by identifying and correcting problems early in the software development lifecycle through complete the testing. Software testing makes competitive in the market. It helps users feel trust about it. When software is tested effectively, it works fine and has no issues. This makes users satisfaction and they keep using it. Also, testing finds and fixes security issues, so the software meets the rules.

As well as the processes of software testing and evaluation are based on the verification and validation process. Verification is majorly done by the software development team and validation is mostly done after the development of the system and checks whether the running system meets its end user requirement.

5.1.1 Unit Testing

This testing method was used while developing the project. It was very helpful to verify the functionality of the components and modules. Here, I tested every component and functionality of the system just after finishing the programming. This was very helpful before the development took in the hands of the end user. As an example, I was able to check the user login function and register function without any disturbance.

5.1.2 Regression Testing

Regression testing is like checking to make sure that after you fix one thing in your software, you didn't accidentally break something else that was already working fine. It's a way to ensure that changes or updates to the software haven't caused any unexpected problems in the existing functionality. When you retest the parts of the software that were working before, make sure they

still work as expected after making changes. It helps to maintain the overall quality and reliability of the software.

5.1.3 Integrated Testing

In integrated testing which is used small different units, components and modules are tested as a combine system. This is doing once the all-unit testing is completed. Actually, this is focus on the correctness of the interface as well as this technique verify data exchange between different components and modules in completed system.

5.1.4 User Acceptance Testing

User Acceptance Testing is the last stage of testing before a new website, or piece of software goes into production (live). It is the phase in which the end users test the product to ensure that it fulfills their needs and functions as intended. They test everything to make sure everything works as it should, that there are no bugs or issues, and that it is simple to use. It's generally ready for general use if everything appears to be in working order.

5.2 Test Cases

To test the system with perfect manner we have to design the test cases in order to accomplish the task. When we are going to design the test cases, we should identify what need to be tested and what are the actual outputs. To reduce the complexity this system tested by module wise and some of the major module test cases are elaborate in following tables.

Table 4: Important test cases

Test ID	Prerequisite	Test Description	Test Steps	Expected Result	Actual Result
TC01	1. User should be in login page. 2. User should have an internet facility.	User logging to the system by entering the correct input credentials	1. User enter valid registered email. 2. Enter valid password.	User should be redirected to the OTP page	Redirected to the OTP page

Test ID	Prerequisite	Test Description	Test Steps	Expected Result	Actual Result
TC02	1. User should be in login page. 2. User should have an internet facility.	User logging to the system by entering the incorrect email and valid password credentials	1. User enter invalid registered email. 2. Enter valid password.	System should display error message.	Displayed error message
TC03	1. User should be in login page. 2. User should have an internet facility.	User logging to the system by entering the correct email and invalid password credentials	1. User enter valid registered email. 2. Enter invalid password.	System should 5display error message.	Displayed error message
TC04	1. User should be in login page. 2. User should have an internet facility.	User logging to the system by entering the incorrect email and invalid password credentials	1. User enter invalid registered email. 2. Enter invalid password.	System should display error message.	Displayed error message
TC05	1. User should be in login page. 2. User should have an internet facility.	User logging to the system by entering the correct OTP number.	1. User enter valid registered email. 2. Enter a valid password. 3. The user enters the	The system should be redirected to the dashboard.	Redirected to the dashboard.

	3. User should receive an OTP number		received OTP number		
Test ID	**Prerequisite**	**Test Description**	**Test Steps**	**Expected Result**	**Actual Result**
TC06	1. The user should be on the login page. 2. The user should have an internet facility. 3. User should receive an OTP number	The user logs into the system by entering the incorrect OTP number.	1. The user enters an invalid registered email. 2. Enter a valid password. 3. The user enters invalid OTP number	The system should be redirected to the same OTP enter page.	Stay on the OTP number and stay on the same page.
TC07	1. User should be in login page. 2. User should have an internet facility.	User register with the system as a dealer by entering the correct details	1. User clicks on “create an account” options in login view. 2. User enter valid details to register with the system 3. User selects the “Register me as a dealer” check box	System should allow to user enter the correct details and redirected to the login page.	Registered with the system successfully.

			4. User clicks on “sign up” button.		
TC08	1. The user should be on the login page. 2. The user should have an internet facility.	Users register with the system as a dealer by entering the incorrect details	1. The user clicks on the “create an account” option. 2.Enters invalid details and selecting “Register me as a dealer” check box 3. Clicks on Sign up button.	System should not allow to user enter the incorrect details	Does not registered with the system successfully.

TC09	1. User should be in dashboard.	The user registers a filling station by entering and uploading the valid details.	1. The user select the register fuel station option from the dashboard and select the enter button. 2. The user enter valid details. 3. The user enter valid documents. 4. user select the filling station name from the drop-down. 5. clicks on “save” button	In dash board display the newly created filling station name in approve pending state.	Displayed the created filling station name on dashboard with pending approval.
TC10	1. User should be in dashboard.	The user places an order by selecting the relevant filling station he wants.	1. The user selects the relevant filling station from the drop-down. 2. Then click on the “order” button. 3. Then enter the quantity	System should allow to order the fuel and redirect to the checkout page	Redirected to the checkout page.

			that he wants and clicks on "order" button.		
TC11	1. User should be in dashboard.	The admin user approves the pending filling station.	1.The admin user go to the approval fuel station option. 2. Then clicks on Enter button. 3. Then he verifies the details by viewing the request. 4. Then clicks on update button.	System should allow to approve filling station successfully.	Approve filling station successfully and change the dealer login status as well.
TC12	1. User should be in dashboard.	The admin user approves the fuel order received.	1.Admin select the view order option. 2. Then he verifies the order by selecting the order and approve it.	System should allow to approve order and send SMS to dealer "your order is approved"	Get a SMS your order is approved.

5.3 User Evaluation

The User evaluation for the online fuel ordering system was presented to assess its usability and effectiveness. Users were asked to navigate the system and perform various tasks such as ordering fuel, checking prices, and tracking delivery status. Most users found the system easy to use, with clear instructions and intuitive design. They appreciated the convenience of ordering fuel online and the ability to track their orders in real time. Anyhow some users encountered issues with slow loading times and occasional glitches when placing orders. Then the majority of the users, the online fuel ordering system received positive feedback for its user-friendly interface and efficient functionality, but improvements are needed to address technical issues and optimize performance. Following table shows the twenty-five users' evaluation about the system who are selecting randomly.

Table 5 : User evaluation summary

No	Description	Excellent	Good	Average	Fair	Weak
01	This system fulfills the online fuel ordering system for dealers.	32%	48%	20%		
02	This system responds quickly.	36%	60%	4%		
03	This system can be understood easily.	36%	52%	12%		
04	User can navigate with the system menu easily.	40%	48%	12%		
05	System reports helps to make decisions quickly.	12%	72%	16%		
06	This system increases the productivity and the efficiency.	32%	64%		4%	
07	This system shows attractive interfaces/UIs	40%	40%	20%		
08	System display error messages for understanding.	32%	56%	12%		

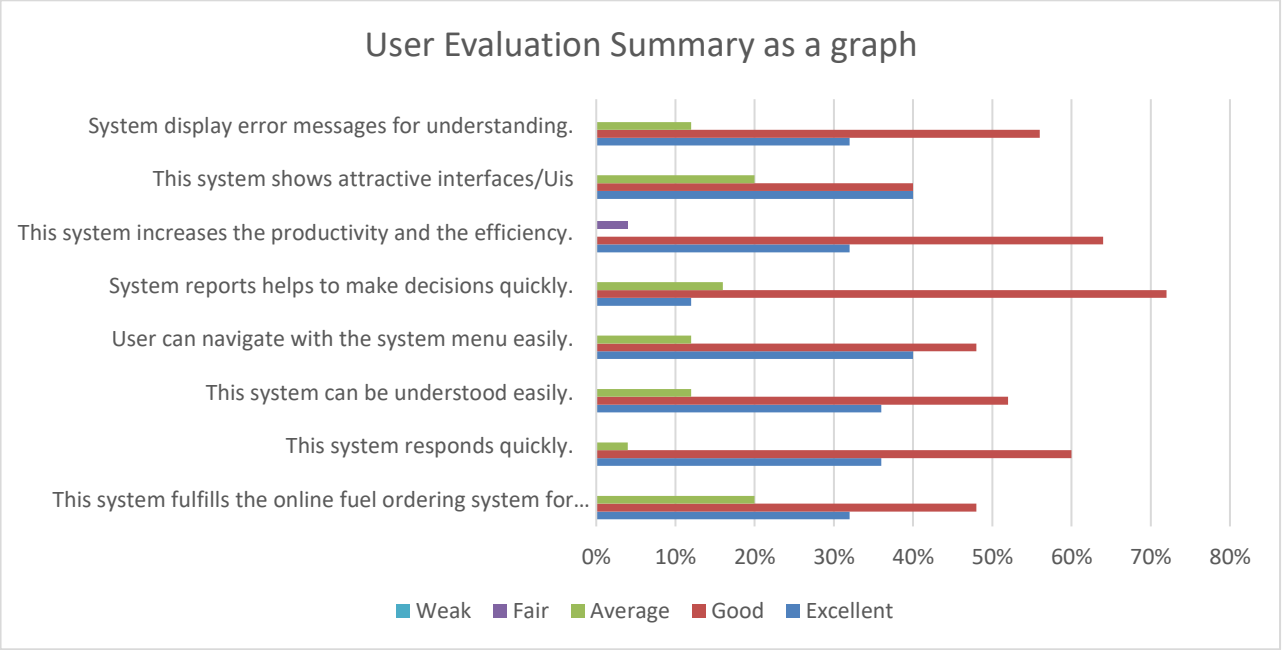


Figure 17 : User Evaluation summary as a graph

5.4 Results of the Testing

All test cases have been successfully passed in our online fuel ordering system for dealers. These tests ensure that our system functions correctly and meets the needs of our users. Here's a summary of the test results: All the test cases were passed.

User Login Tests:

- All login scenarios were tested, including valid and invalid email/password combinations.
- The system correctly redirected users to the OTP page when credentials were valid.
- Error messages were displayed when credentials were invalid.
- The system behaved as expected in all login scenarios.

OTP Verification Tests:

- Tests were conducted for both correct and incorrect OTP entries.
- The system redirected users to the dashboard upon successful OTP verification.
- For incorrect OTP entries, the system remained on the OTP entry page as expected.

User Registration Tests:

- Successful registration tests were performed for dealer accounts.
- Users were able to register successfully with valid details.
- Error messages were displayed for invalid registration attempts.

Chapter 6 – Conclusion

In this chapter includes all the summary of the project as an author's perspective, this will elaborate all the achievements, constraint as well as drawbacks of the system, while system was developing.

6.1 Introduction

In conclusion, this online fuel ordering system for dealers prioritizes security and ease of use. Upon user login, both email and password are verified for authentication. To verify the extended security measures, I have to integrate the Notify.lk SMS API for two-factor authentication, where users receive a unique OTP number. Only after entering the successful OTP number and verification, users can access the system dashboard. This streamlined process not only enhances the security of the system but also provides a straightforward user experience. This will be helpful to user in user-friendly manner.

The online fuel ordering system streamlines the fuel procurement process for dealers. Instead of relying on traditional methods that may be time-consuming and prone to errors, dealers can easily place orders through a user-friendly online platform. This is not only saving time but also ensures accuracy in order placement, reducing the risk of mistakes and delays.

As further the integration of SMS notifications increases communication and transparency throughout the fuel distribution process. Dealers receive instant confirmation of their orders via SMS, providing them with reassurance that their requests have been received and processed. Then, SMS notifications from the order approval to bowser gate exit also the new concept in here to process enable dealers to track the movement of fuel shipments in real time, allowing for better coordination and planning.

In conclusion, it's satisfying to see that the efforts put into developing this system, after thorough analysis of requirements, have resulted in the successful creation of this newly designed software. As the world continues to embrace the concept of a global village through the internet's evolution,

systems like this for online fuel ordering for dealers are becoming increasingly valuable and practical. The internet is now widely accessible, and people's proficiency in using technology is growing rapidly. Consequently, implementing such a system for fuel ordering can significantly enhance the efficiency of the process. Just this systems have revolutionized the way we conduct fuel orders, online fuel ordering systems can similarly streamline and improve the fuel procurement process for dealers, contributing to better management and distribution of fuel resources in the country.

6.2 Lesson Learnt

Developing the online fuel ordering system using CodeIgniter 3 and PHP as a Master's degree student has been a valuable learning experience. Throughout the process, I've gained knowledge of the complex of web development and database management, my skills in programming and problem-solving. One key lesson learned is the importance of thorough planning and requirement analysis before developing. Understanding the user needs and the functionalities required for the system is crucial for creating a robust and user-friendly application. Additionally, I've learned and adhered to coding standards and best practices, ensuring the system's scalability, maintainability, and security. Overall, this experience has not only enhanced my technical abilities but also equipped me with valuable project management and teamwork skills that will benefit my academic and professional journey.

6.3 Critical Assessment of the Project

A critical assessment of the online fuel ordering system for dealers involves based on the effectiveness and efficiency of the system. When went through the development with integrating with the Notify.lk SMS gateway. When developing the project, alternative SMS gateways were considered, but they were found to be not suitable for various reasons.

In this critical assessment, I will examine how well the online fuel ordering system performs when connected to the Notify.lk SMS gateway. This includes some various factors such as reliability, speed of message delivery, cost-effectiveness, and ease of integration. When integrate payment gateway (payhere sandbox) it was not working on local host, there fore need to host the system in live server to get the payment gateway as working function.

6.4 Future Works

Actually this is online base fuel ordering system for dealers. Then there are few suggestion mention below to improve this efficiency and effectiveness.

- **Connecting with SAP RFC API**

The online fuel ordering system for dealers connects with the SAP API (Application Programming Interface) system using a Remote Function Call (RFC). This integration fulfills the communication between the fuel ordering system and SAP, then the online system can directly interact with SAP, and access important data and functionalities such as pricing information, employee information, and tank capacity. This integration streamlines the ordering process for dealers, as well as eliminating the need for manual data entry to get the real-time data.

- **Google authenticate login.**

The online fuel ordering system for dealers connects with Google authorization login to enhance security and streamline user access. By integrating with Google authorization, dealers can log in to the system using their Google credentials, making the login process simpler and more convenient. This eliminates the need for dealers to create and remember separate login details for the fuel ordering system.

- **Connecting with storage Depot**

The online fuel ordering system for dealers connects with depots, allowing dealers to place orders based on their specific depot requirements. This means that dealers can conveniently order fuel from the depot closest to them, optimizing logistics and ensuring timely deliveries. By connecting depot-wise, the system streamlines the ordering process, making it easier for dealers to manage their fuel inventory and meet customer demand efficiently.

List of References

- Looking for fuel order management and Delivery Software? (no date) purbis.com. Available at: https://www.purbis.com/online-fuel-ordering-software.html (Accessed: February 18, 2023).

- Gundaniya, N. (no date) Fuel Delivery & ordering software, Best Fuel Management Systems,Fuel Delivery & Ordering Software, Best Fuel Management Systems. Available at: https://www.peerbits.com/fuel-delivery-software.html (Accessed: February 18, 2023).

- How it works (2022) Fuel Me. Available at: https://fuel.me/how-it-works/ (Accessed: February 18, 2023).

- UML 1.4 versus UML 2.0 as languages to describe software ... - researchgate (no date). Available at: https://www.researchgate.net/publication/221105847_UML_14_versus_UML_20_as_Languages_to_Describe_Software_Architectures (Accessed: November 2, 2023).

- Add advanced interaction CONTROLSTO YOUR HTML tables the Free & Easy Way (no date) DataTables. Available at: https://datatables.net/ (Accessed: November 18, 2023).

- Hendrix, N. (2022) Using .htaccess files for pretty urls, Code Envato Tuts+. Envato Tuts. Available at: https://code.tutsplus.com/tutorials/using-htaccess-files-for-pretty-urls--net-6049 (Accessed: November 18, 2023).

- Mark Otto, J.T. (no date) Introduction, · Bootstrap v5.0. Available at: https://getbootstrap.com/docs/5.0/getting-started/introduction/ (Accessed: November 2, 2023).

APPENDIX A - User Manual

Register

- Go to the https://fuelorderdashboard.xyz/ URL, You can view the following user interface.

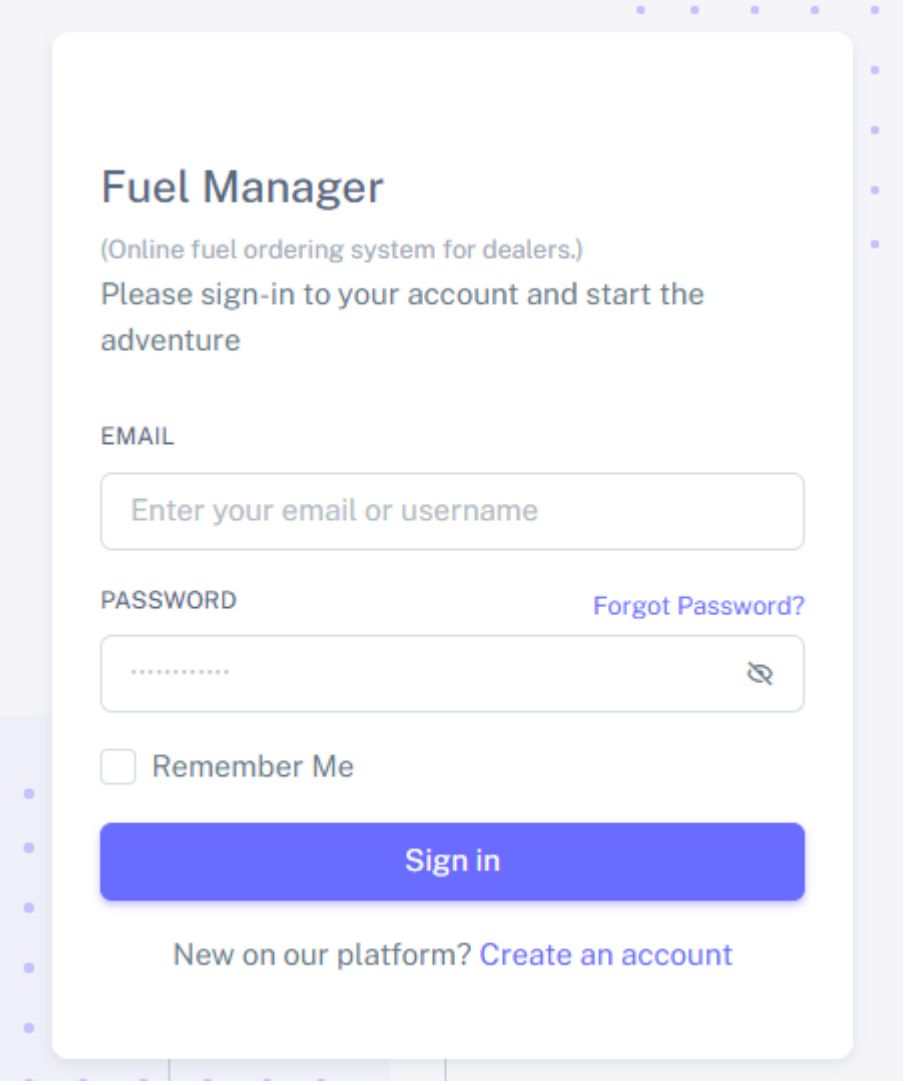


Figure 18:Login Interface

- Click on "Create an account" option and you will be redirected to the following user interface.

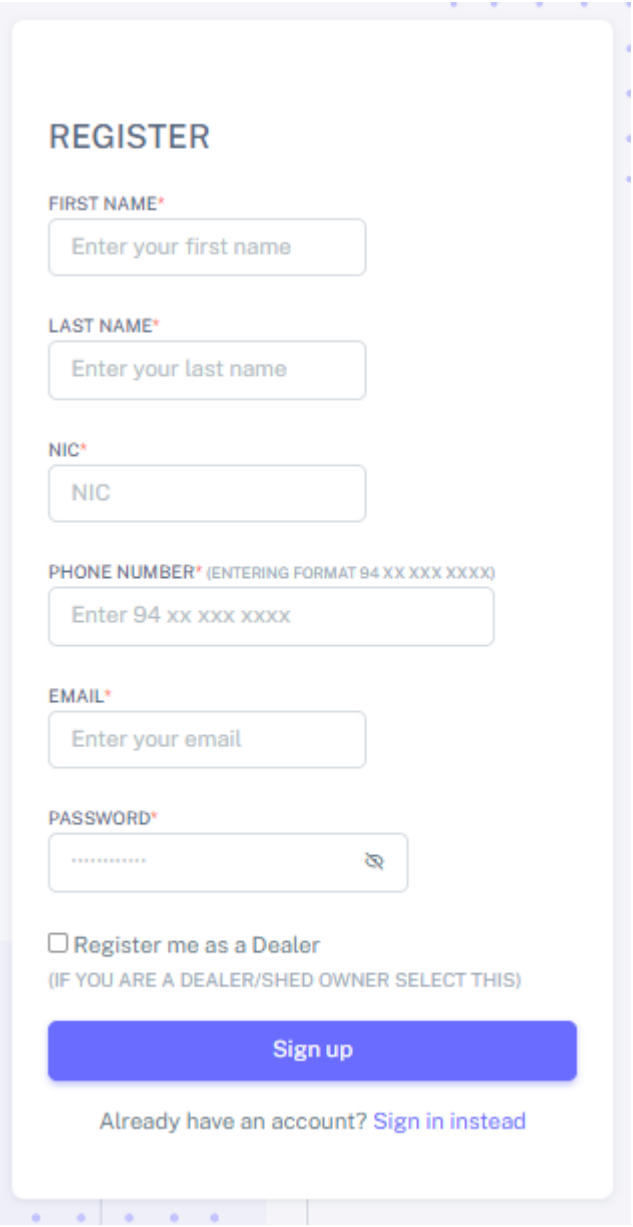


Figure 19: Register user interface

- Enter valid first name, last name, NIC, phone number, Email, and password, if you need to register as a fuel station owner click on "Register me as a Dealer" tick option and click on "Sign up" button.

Forgot Password.

- In figure 19 there is an option call "forgot Password" Click this and you will be redirected to the following user interface.

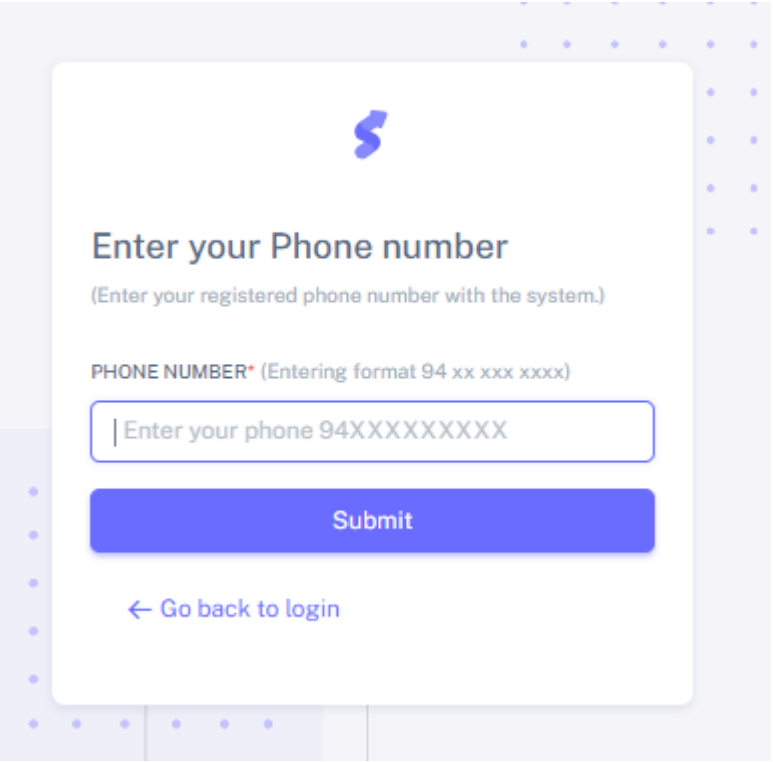


Figure 19: Forgot password phone number interface

- Enter your phone number and click on "Submit" button. Then you will be redirected to the following user interface.

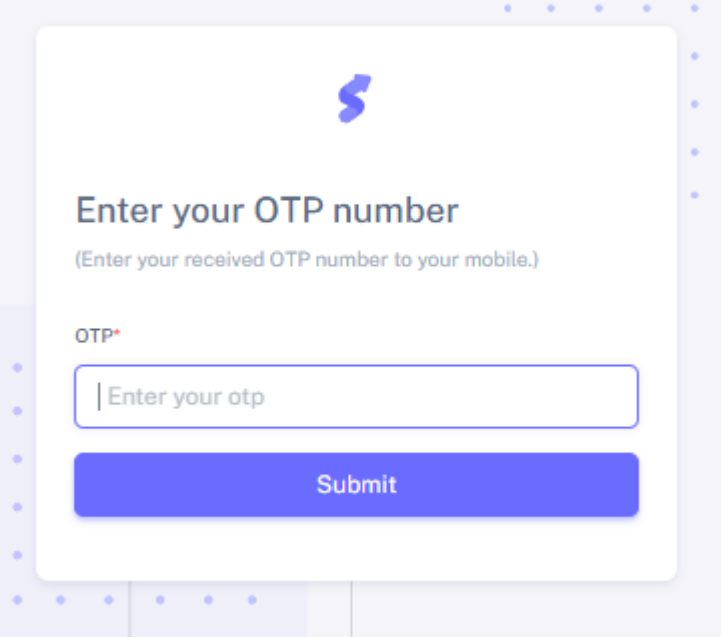


Figure 20: Forgot password OTP interface

- Enter the valid OTP number and click on "Submit" button. Then you will be redirected to the following interface.

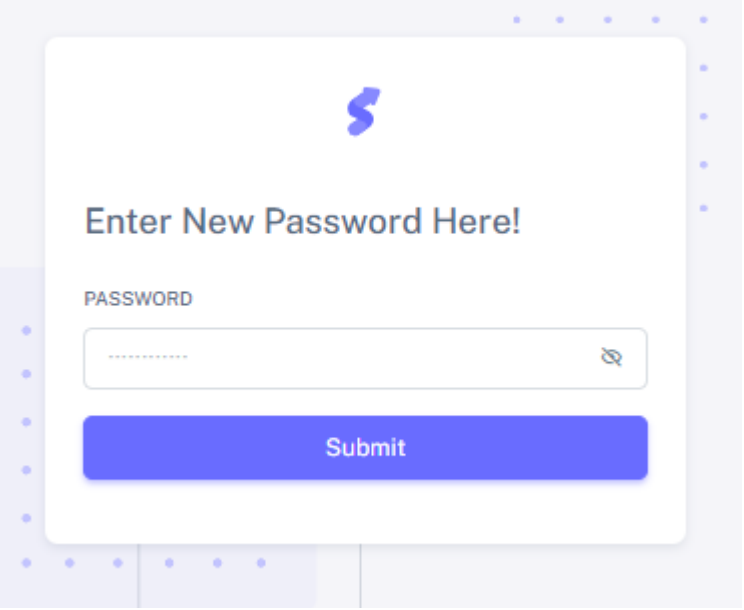


Figure 21: New Password interface

- Then Enter valid password and click on "Submit" button. Then redirect to figure 19.

Login

- In figure 19 interface above, Enter valid email address and password in the given area and click on "Sign in" button.
- Then you will received an OTP number, just enter the valid OTP number in below user interface and click on "Submit" button.

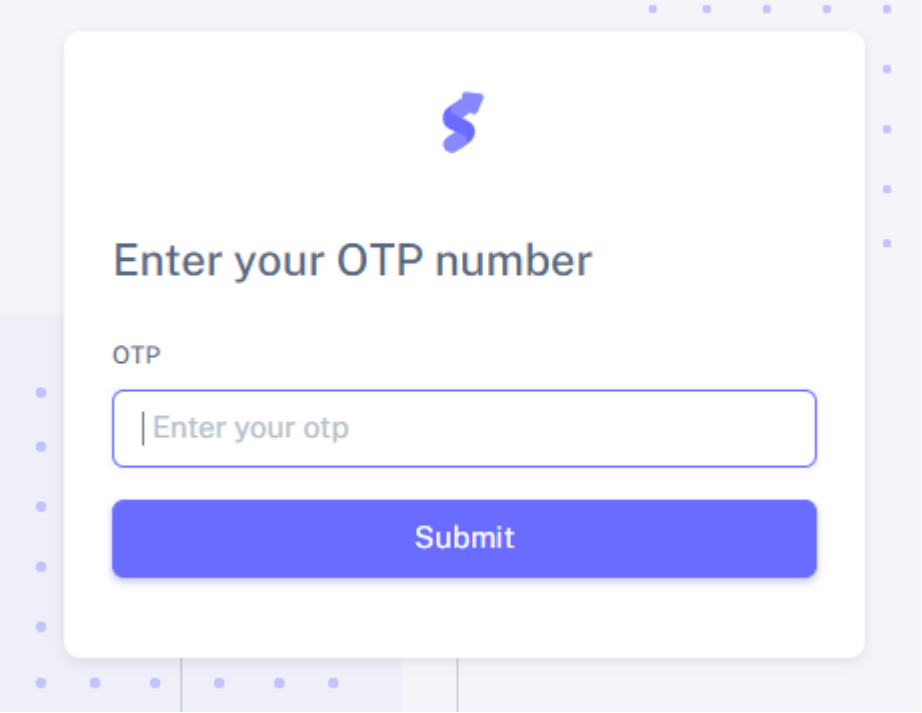


Figure 22: OTP enter Interface

- Then you will be redirected to the following interface (Dashboard).
- You can scroll down to view the details in dashboard.

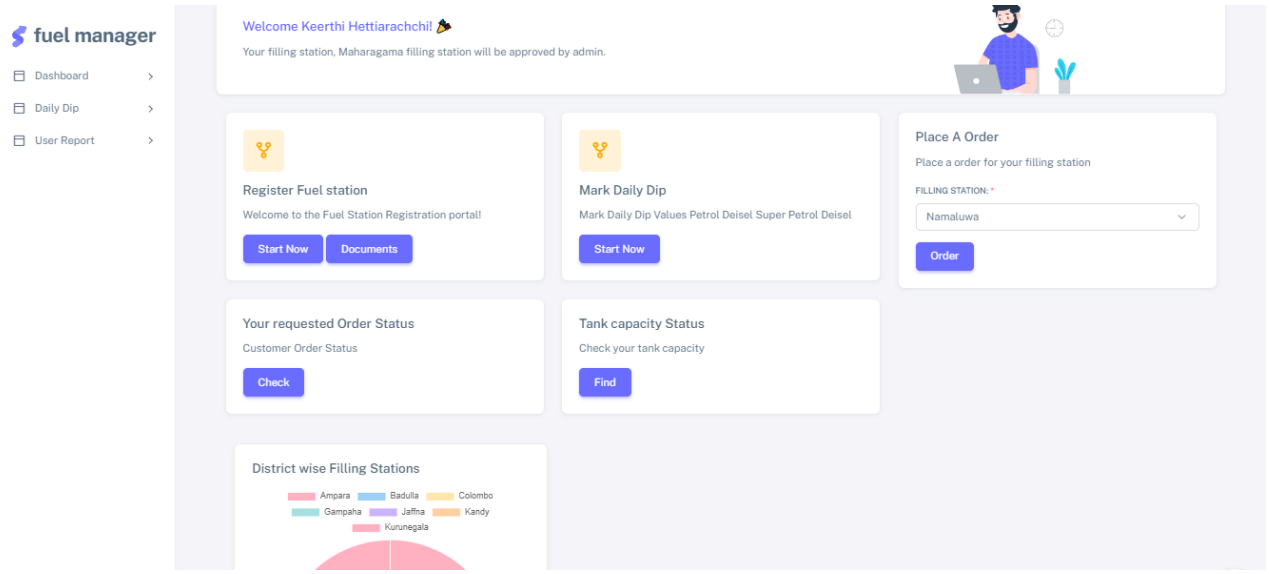


Figure 23: Dealer Dashboard

Register Filling station

Hence you are registered with the system you cannot access the dashboard options.

- First select the Register fuel station option in dashboard and clicks on "Start Now", Then you will be redirected to the following interface.

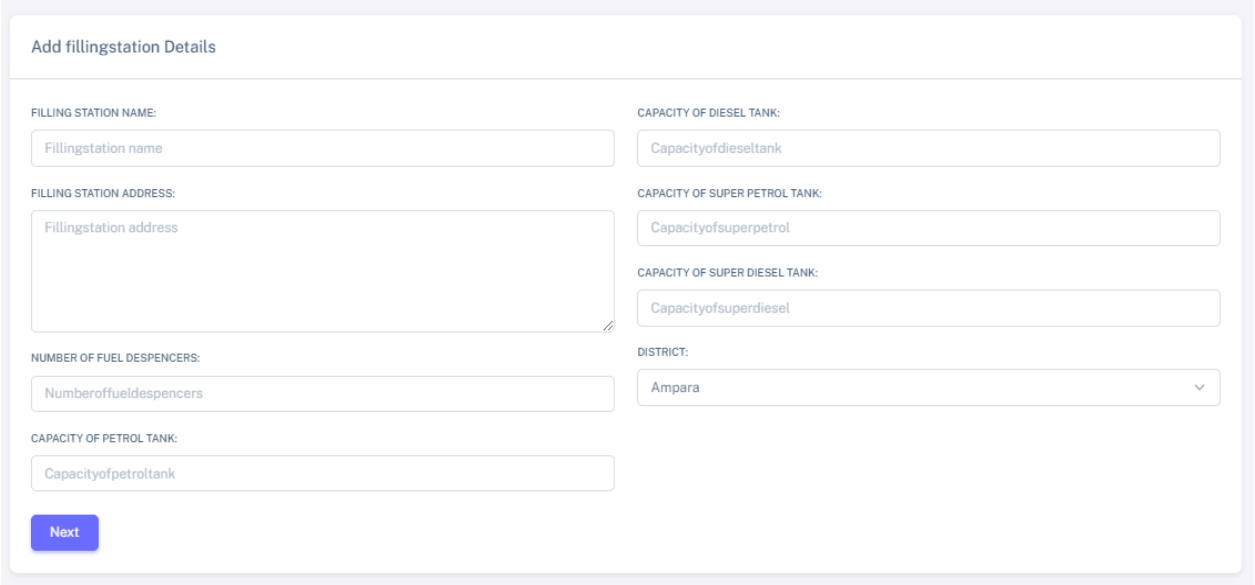


Figure 24: Register filling station interface

- Enter filling station name, address, number of fuel dispensers, capacity of petrol, diesel, super diesel, super petrol tank, and select your district of fuel station located and clicks on "Next" button. Then you redirected to the add documents details. (In here you should add the document you received from the organization when get the approval by ministry.)

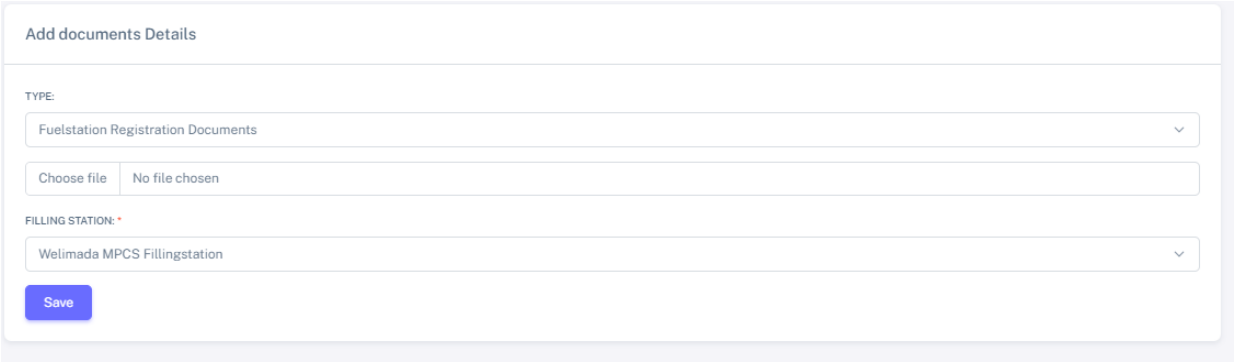


Figure 25: Add document interface

- You should browse the document from your device and select the filling station name you created figure 26. If you need to reupload the document you can click on the “document” button from the figure 25 fuel station register option. Select the filling station name and re upload the document.

Place an Order

- When after admin approves the filling station details user can make an order, for that user select a place an order option from the dashboard and select the filling station name and clicks on the “Order” button.
- Then user redirects to following interface.

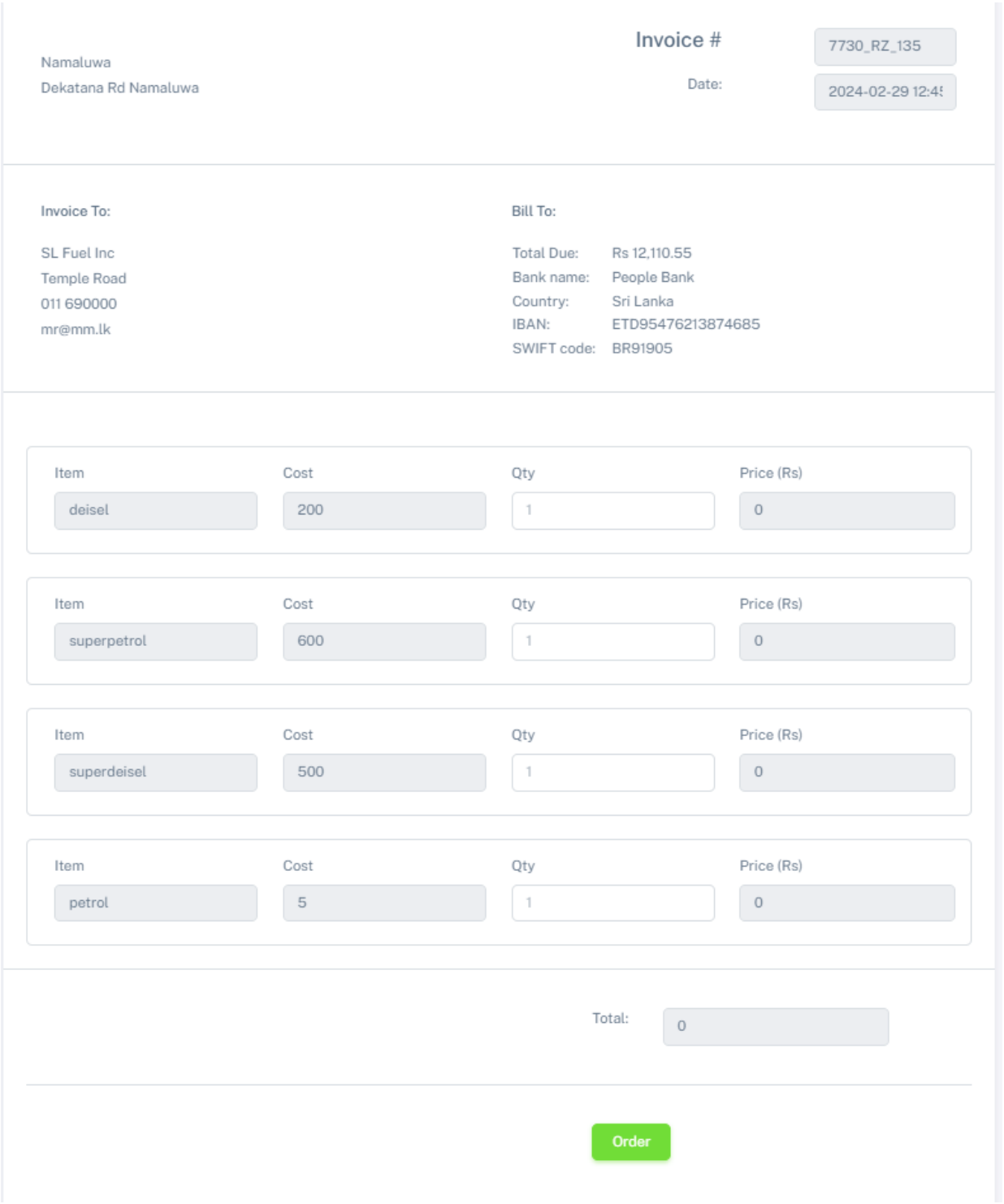


Figure 26: Place an order interface

- User enter quantity that he wants and clicks on “Order” button.
- Then SMS sends to user whether his order is pending. With relevant product details

Check out and Payment

- When user clicks on “Order” button from above figure 28, He can be redirected to the check-out page as below.

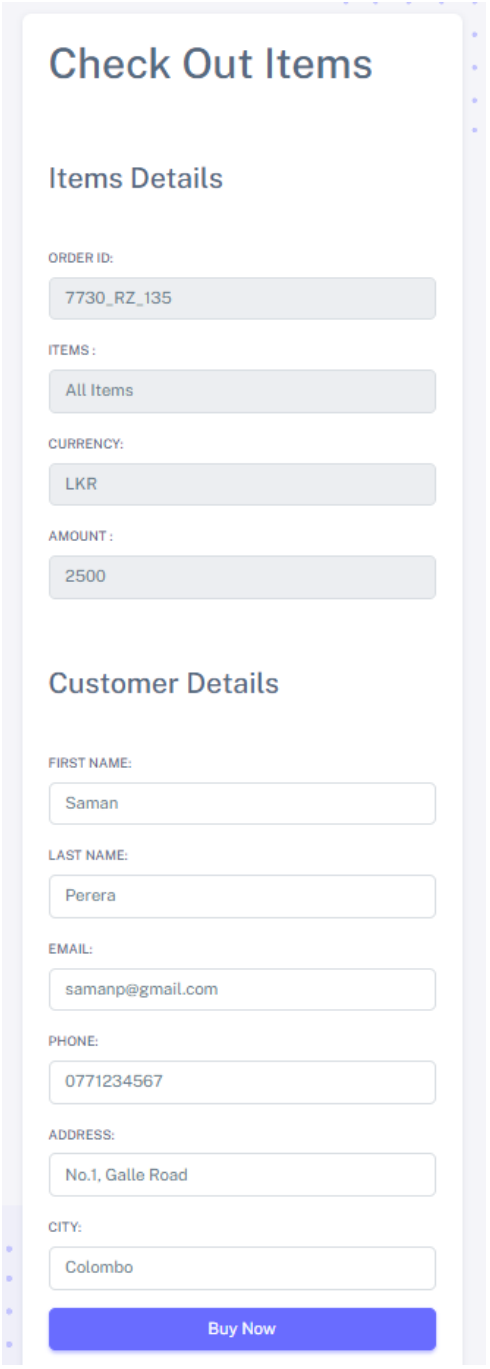


Figure 27: Check out page

- Then user enter relevant customer details and above order related amount and invoice number displays.

- Then after enter the valid details. User clicks on the “Buy Now” button. Then user will be redirected to the online payment gateway(sandbox payhere).

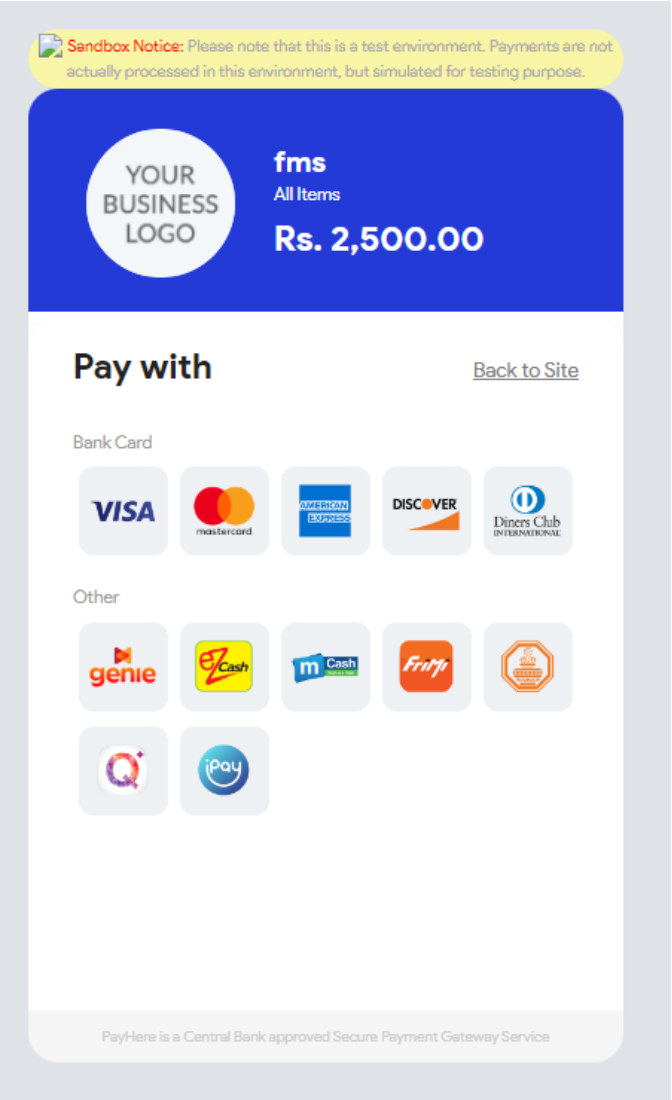


Figure 28: Payment gateway interface

- Then user can select the relevant payment method and enter the valid card details, then after he will be redirected to the thankyou page and after few seconds move to the dashboard.

- Then customer can view the created order from the customer dashboard table with relevant order details.

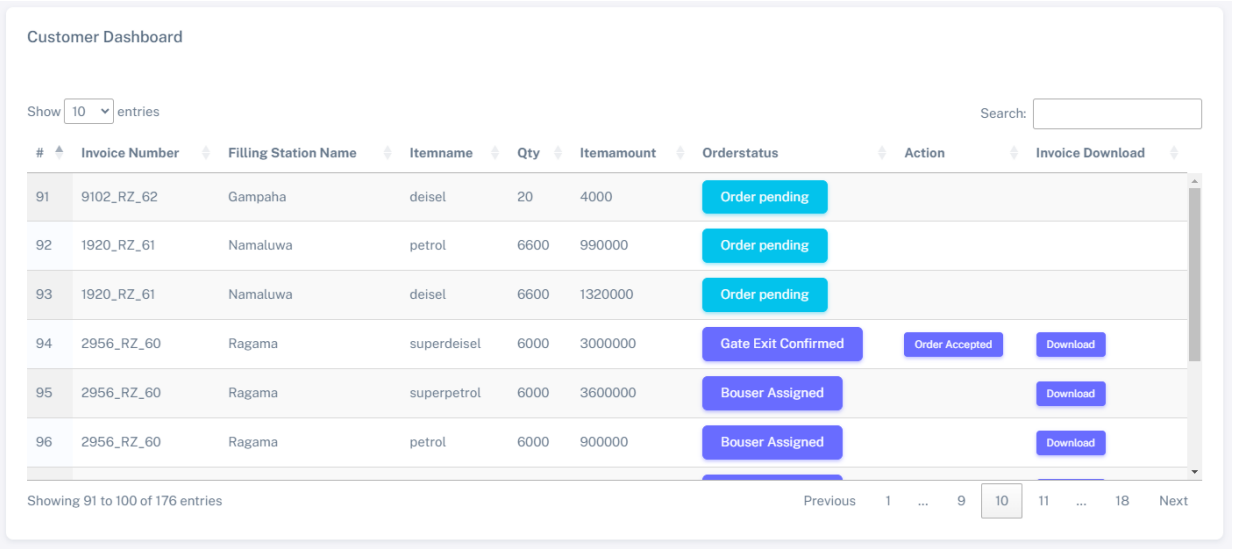
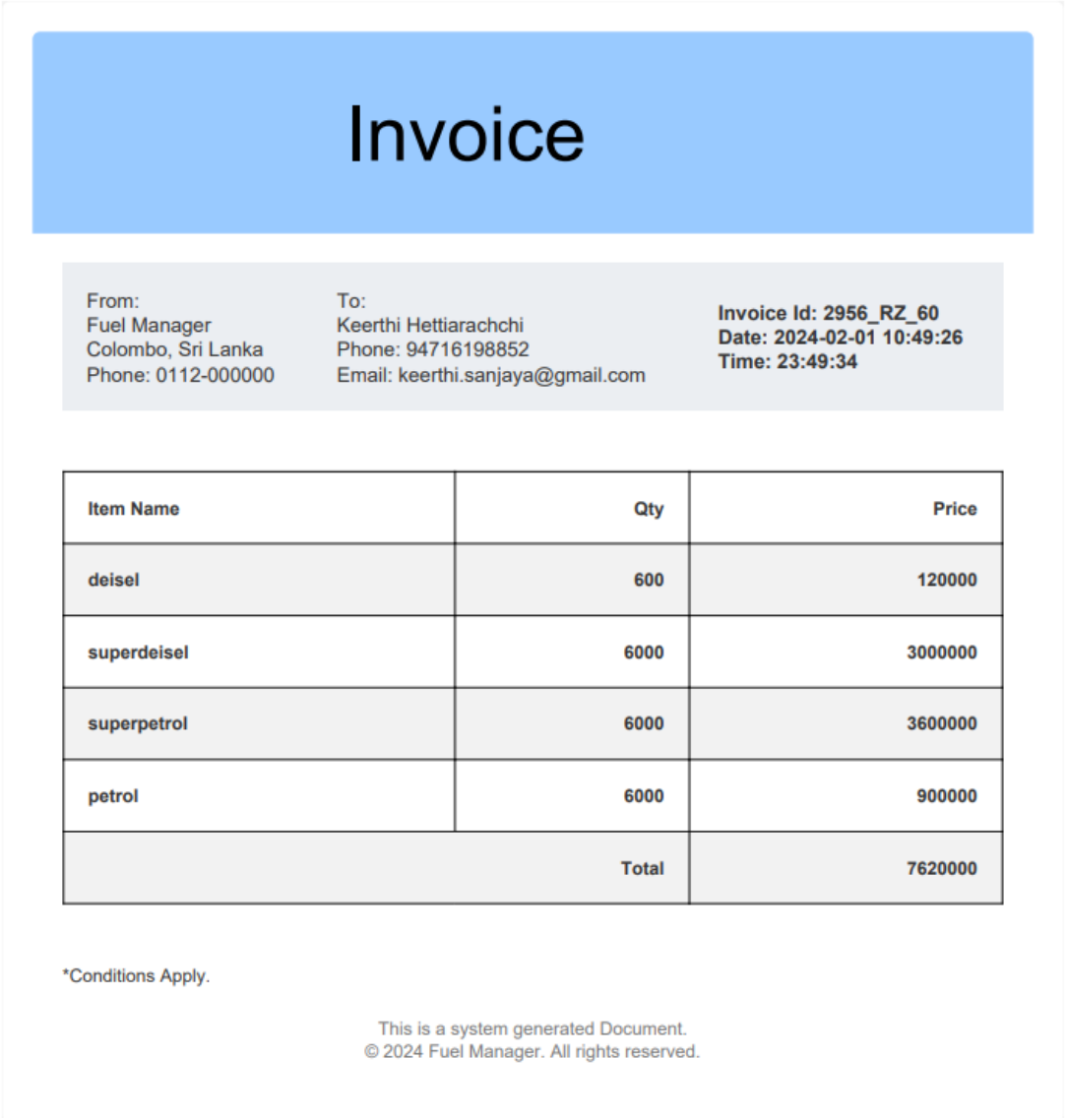


Figure 29:Customer dashboard for order details

- It will display the order status respectively while admin doing the things for order.

- When gate exit process complete user can accept the order by click on the button appeared on action field as “order Accepted” button. Before he accepts the order, he can view the order related SMS whether the details are correct.

- When after order is accepted by the admin user, he can view “Download” button in the Invoice Download field and once clicks on the “Download” button the invoice will be downloaded to the created order as below.



Invoice

From:
Fuel Manager
Colombo, Sri Lanka
Phone: 0112-000000

To:
Keerthi Hettiarachchi
Phone: 94716198852
Email: keerthi.sanjaya@gmail.com

Invoice Id: 2956_RZ_60
Date: 2024-02-01 10:49:26
Time: 23:49:34

Item Name	Qty	Price
deisel	600	120000
superdeisel	6000	3000000
superpetrol	6000	3600000
petrol	6000	900000
	Total	7620000

*Conditions Apply.

This is a system generated Document.
© 2024 Fuel Manager. All rights reserved.

Figure 30:Order Invoice interface

Mark Daily Dip

- User select the mark daily dip option from the dashboard (figure 25) and clicks on “Start Now” button.
- Then he will be redirected to the following interface.

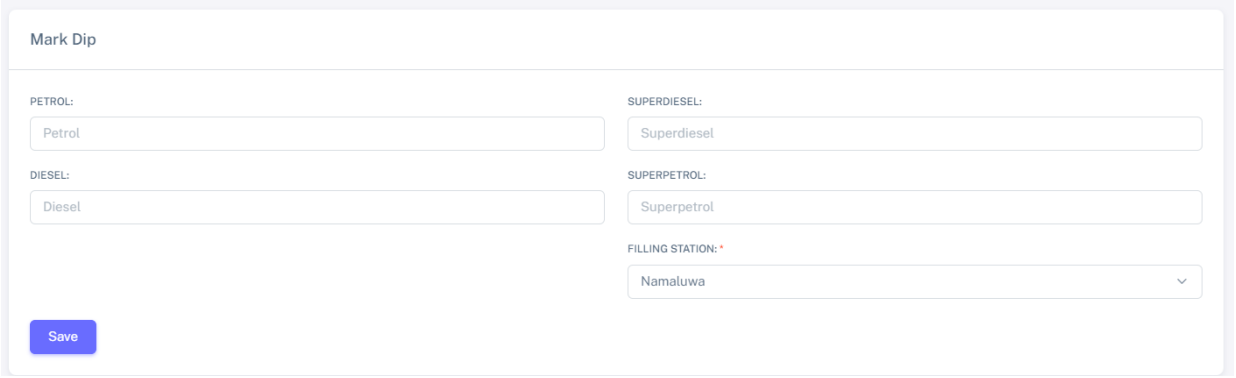


Figure 31 :Mark daily dip interface

- Then user can measure the relevant dip level for the specific product and check the correspondence dip level quantity by the given chart.
- After select the relevant filling station and click on “Save” button.
- Then only user can view the tank capacity status of the relevant filling station as below. Red color shows the reorder level, and green color shows it is in safe level.

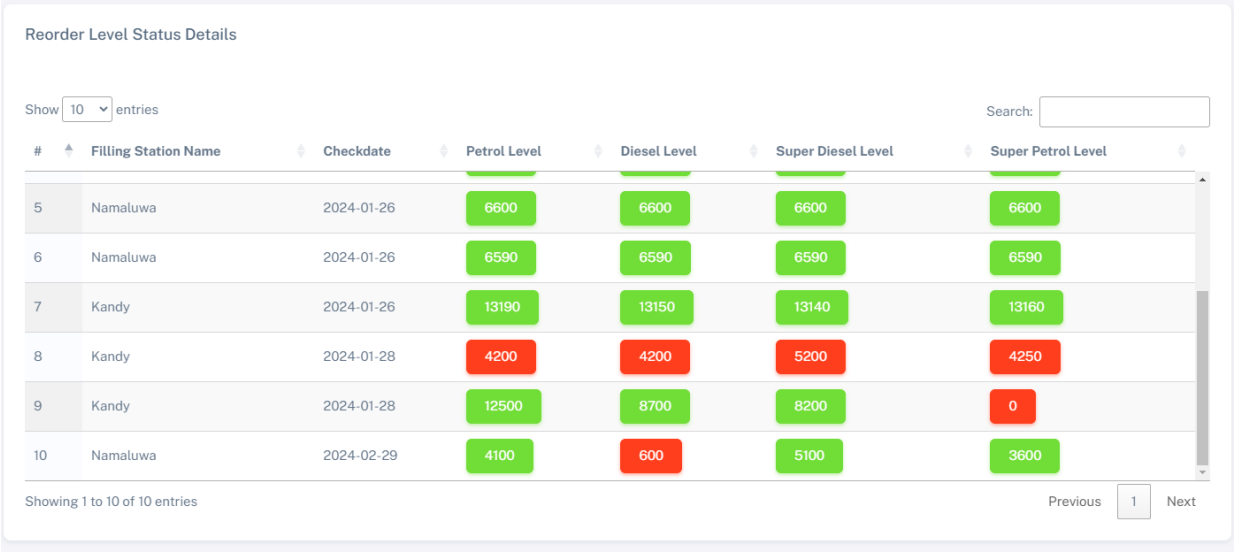


Figure 32 : View the Reorder and safe level of the tank

- If the user is **new** then he can select the daily dip from the left side menu and it will be redirected to the following interface.

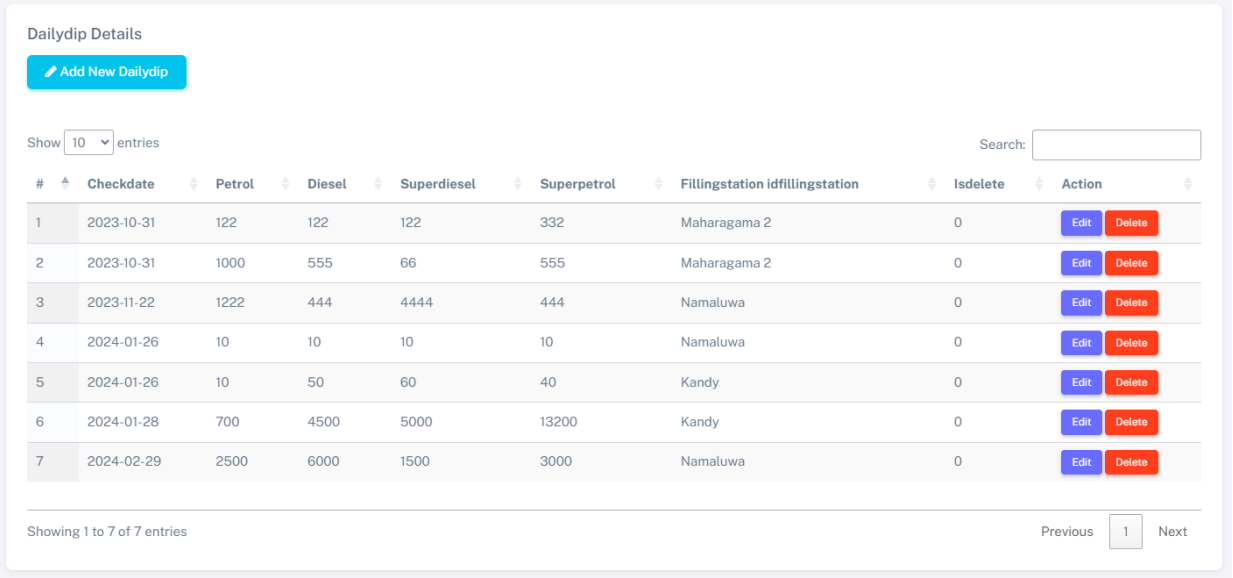


Figure 33:Daily dip details interface

- Then user selects the “Add New Daily dip” button to add the first record.
- If user needs to edit the record he can use and clicks on “Edit” button.
- If user needs to delete the record, can clicks on the “Delete” button.

Profile Edit

If user need to edit his profile data he can select the “My profile” option from the right side top corner and display the edit interface as below.

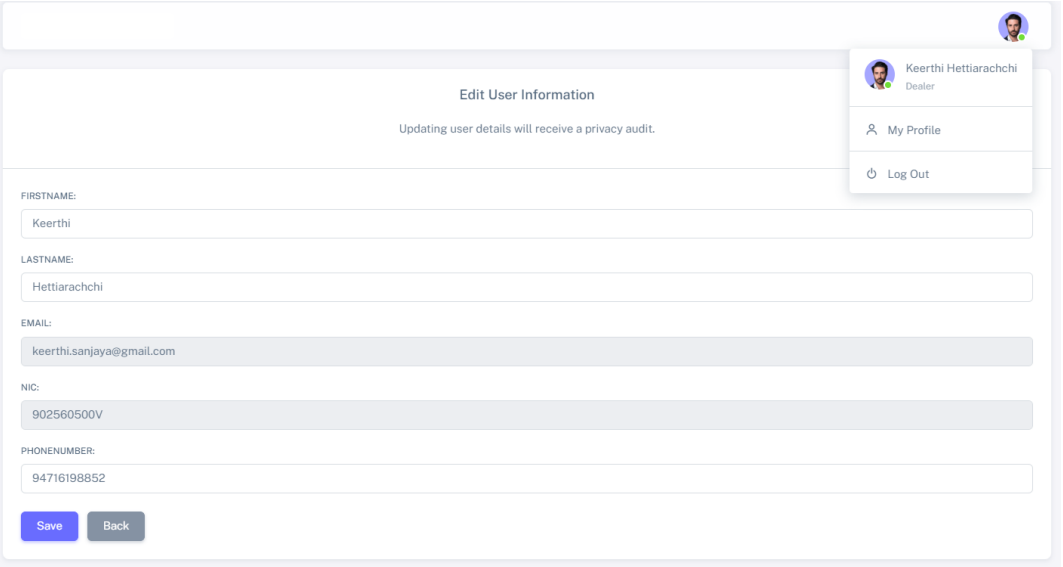


Figure 34: Edit user information

- Then user can edit first name, last name, phone number.

Admin User Manual

- When admin login to the system by entering valid email, password and valid OTP number he will be redirected to the following dashboard.
- User can scroll down to view the dashboard options.

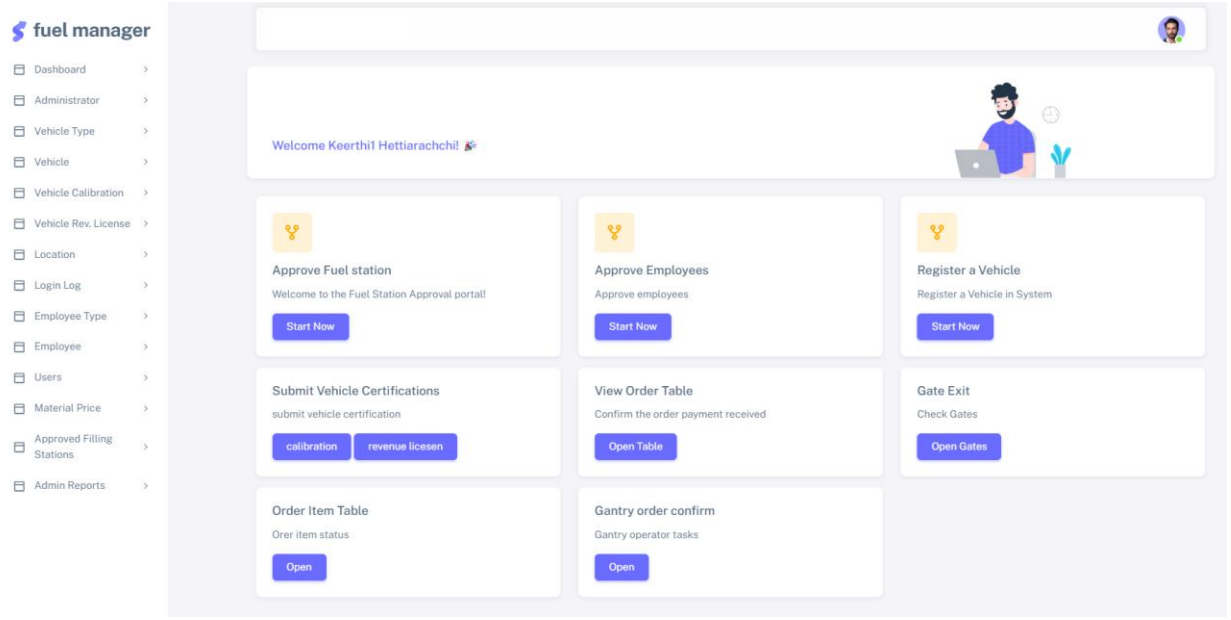


Figure 35:Admin dashboard

Approve Filling Station

- Admin user select the "Approve Fuel Station" option from admin dashboard (figure 36) and clicks on "Start Now" button.
- Then it will be redirected to the following interface.

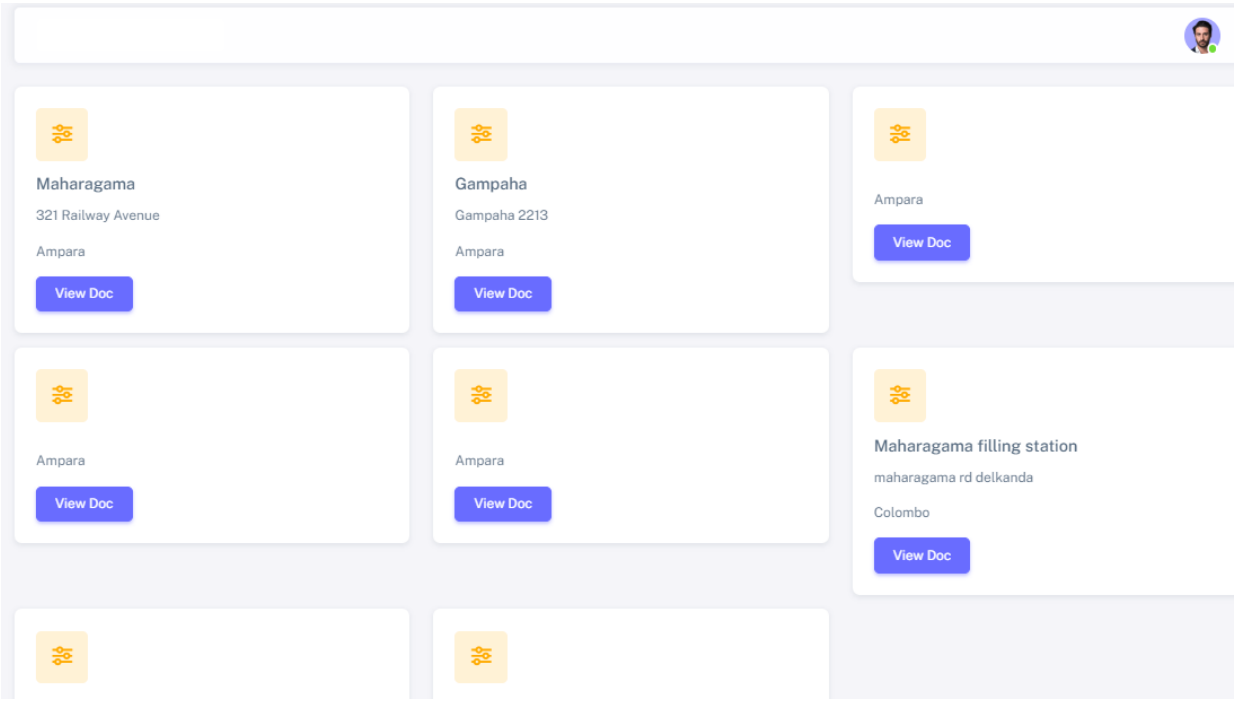


Figure 36 :Filling station registration pending interface

- Then clicks on "view Doc" button and display following interface.

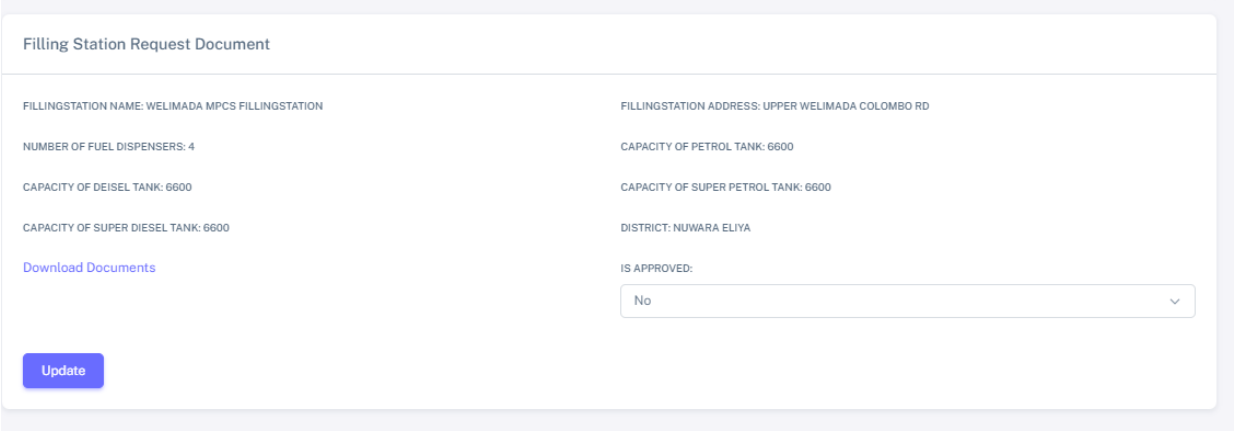


Figure 37 : Filling station request document interface

- Then admin user view the document by click on the blue color download document option and check the user filled documents and if it is correct then select approve "Yes" from the drop down and clicks on "update" button.
- Then shed owner can view the filling station approve on his dashboard.

Order Approval

- When the admin user selects the view orders table option from the dashboard to view the incoming orders.
- Then clicks on "Open Table" button to view the order list.

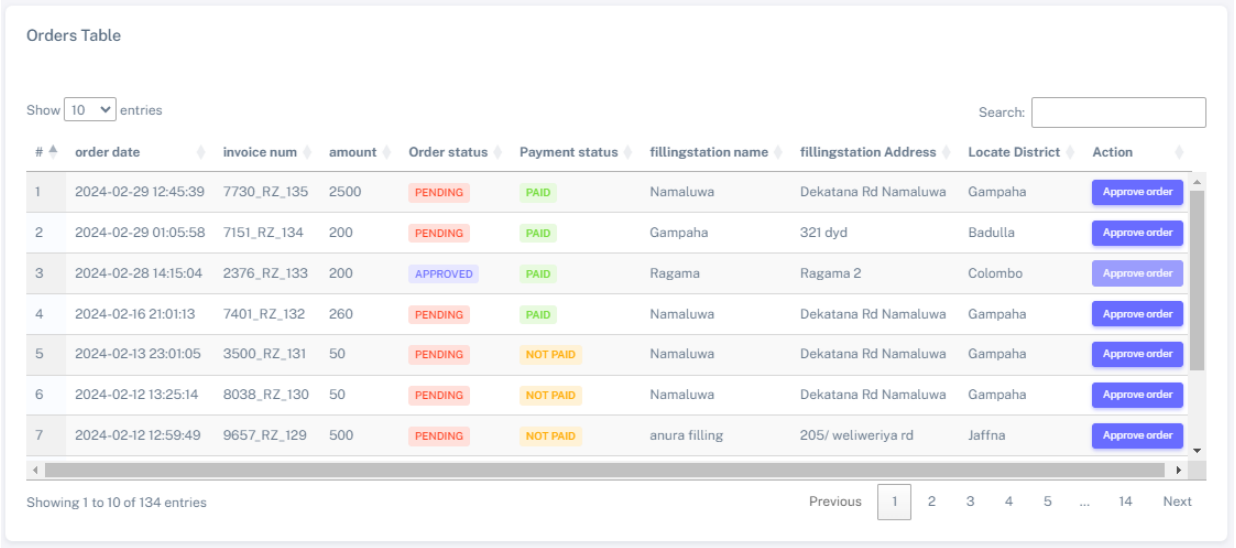


Figure 38 : Incoming order approve interface

- Then user clicks on "Approve Order" button when payment status "PAID".
- Then shed owner received a message for order approval.

Assign Driver and Bowser to Order Items

- Admin user selects the "order item Table" option from dashboard and clicks on "open" button.
- Then view the following interface.

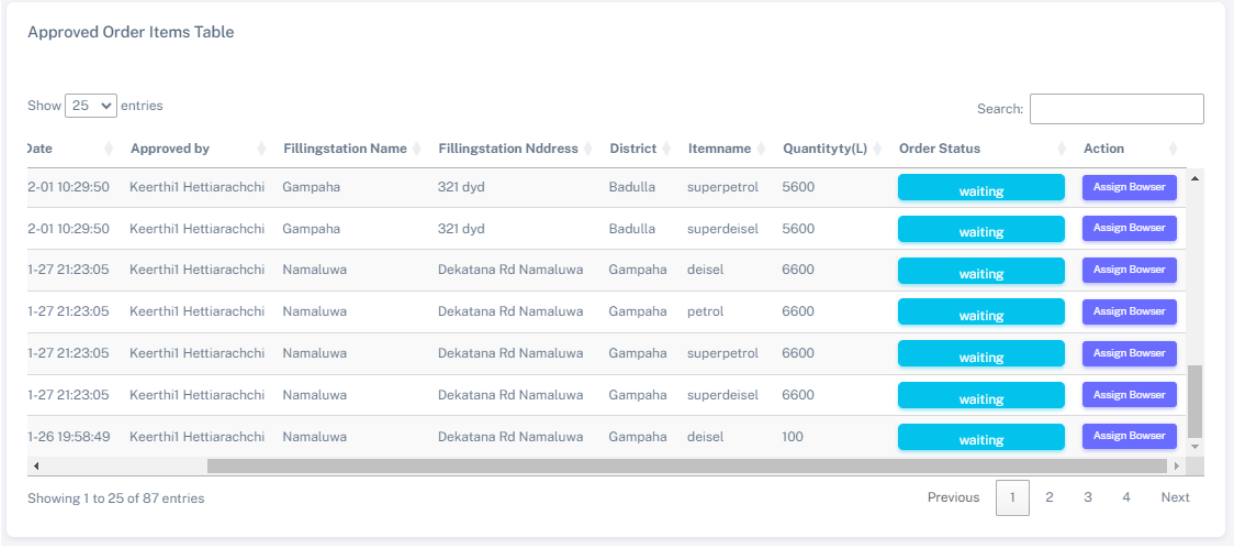


Figure 39: Approved order items table interface

- Then user clicks on "Assign Bowser" button and redirected to following view.

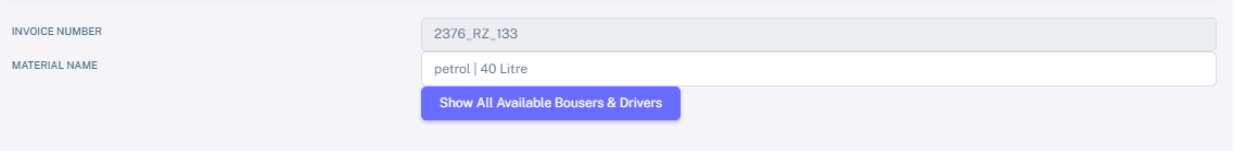


Figure 40: Bowser & Driver Assign interface 1

- Then clicks on "Show All Available Bowsers & Drivers" button. then display the interface as below.

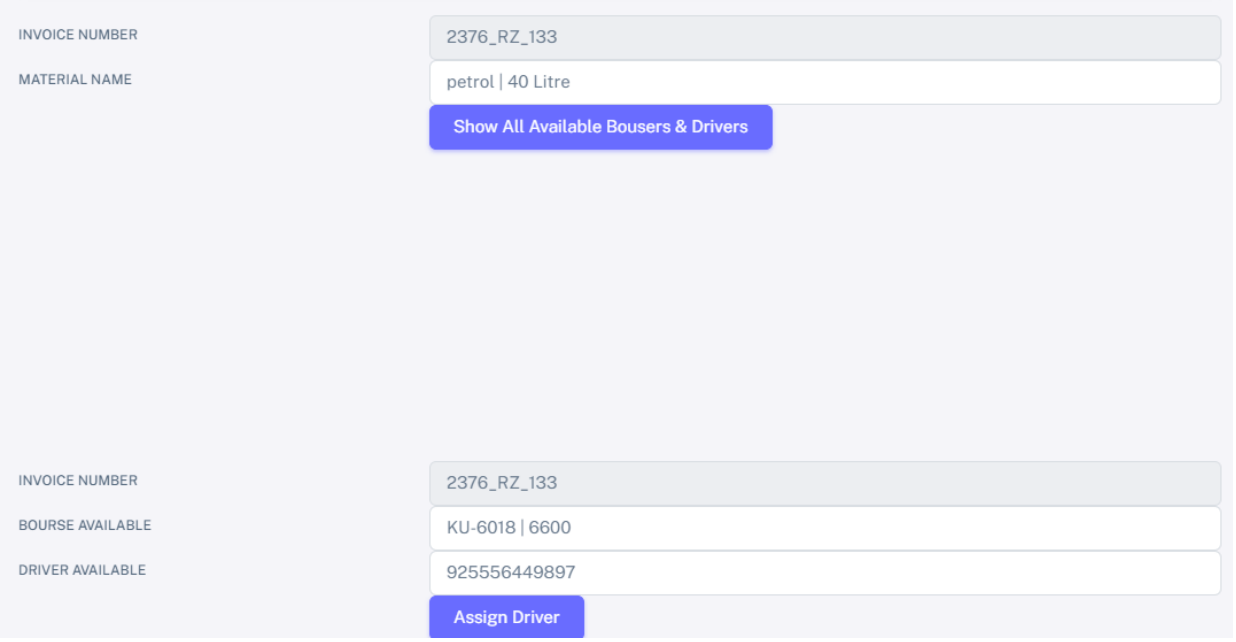


Figure 41:Bowser & Driver Assign interface 2

- Then click on "Assign Driver" button and send SMS to relavent shed owner who created the order.

Gantry Filling

- Admin user selects the "Gantry order confirm" option from dashboard. Then clicks on "Open" button.
- Then display the following interface

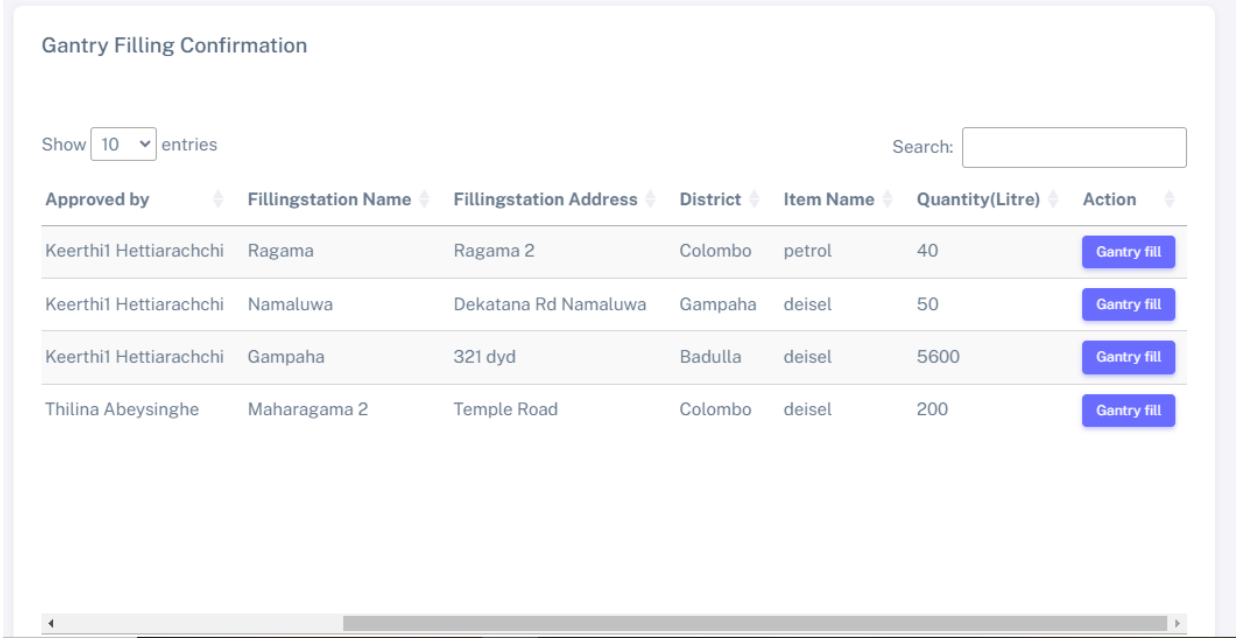


Figure 42:Gantry filling confirmation interface

- Then display the following interface once click on the "Gantry Fill" button.

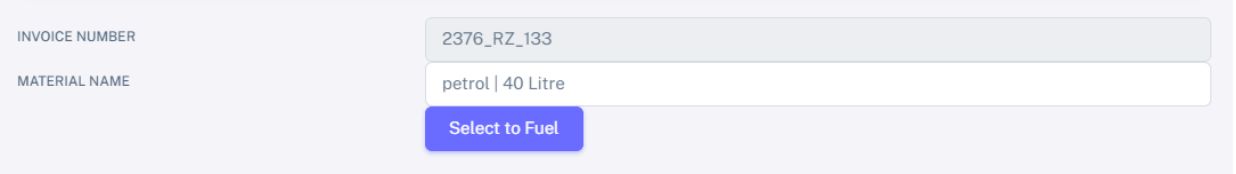


Figure 43: Gantry order confirm interface 1

- When clicks on the "Select to Fuel" button then display the following interface.

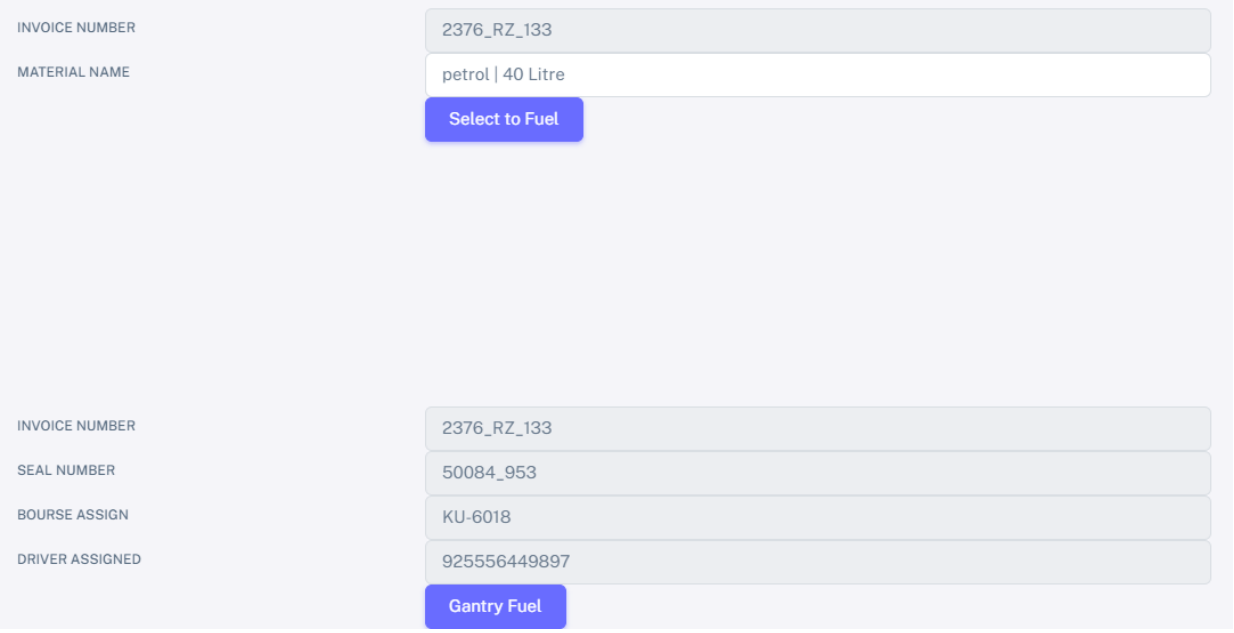


Figure 44: Gantry order confirm interface 2

Gate Exit of the Order

- When admin user selects the "Gate Exit" option from the dashboard and clicks on the "Open Gates" button.
- Then display the following interface.

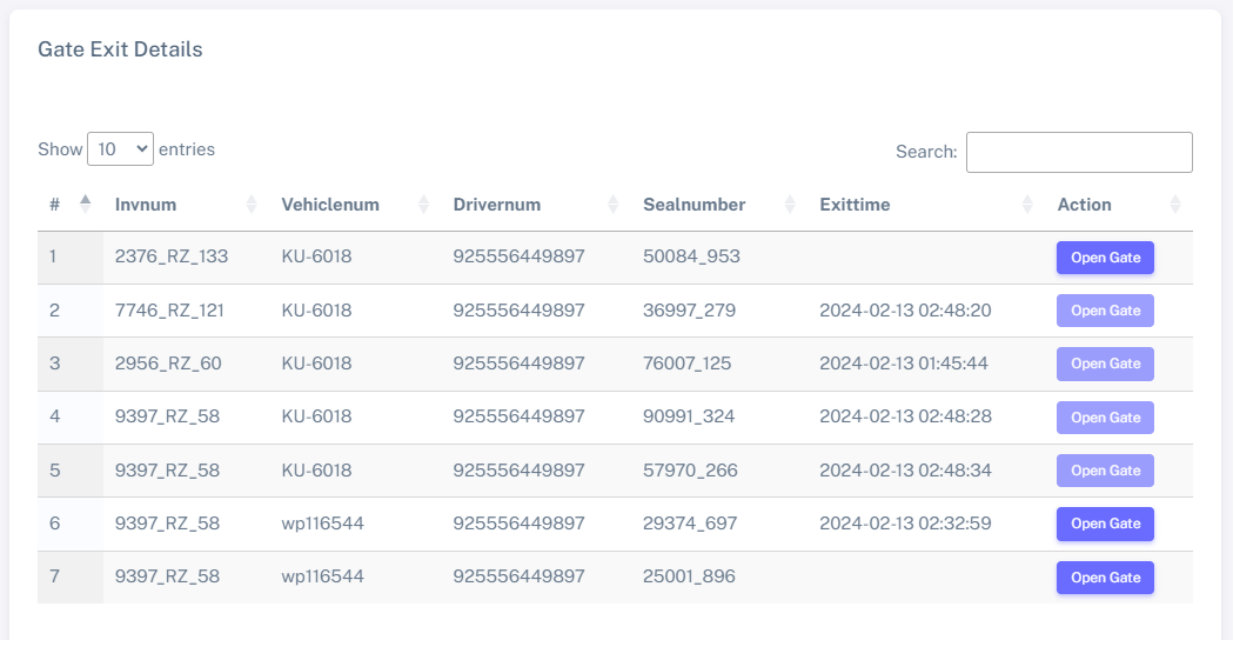


Figure 45: Gate exit confirmation interface

Vehicle Management Type

- Admin user selects the "vehicle type" from the menu bar.
- Then click on the "View Details" option.

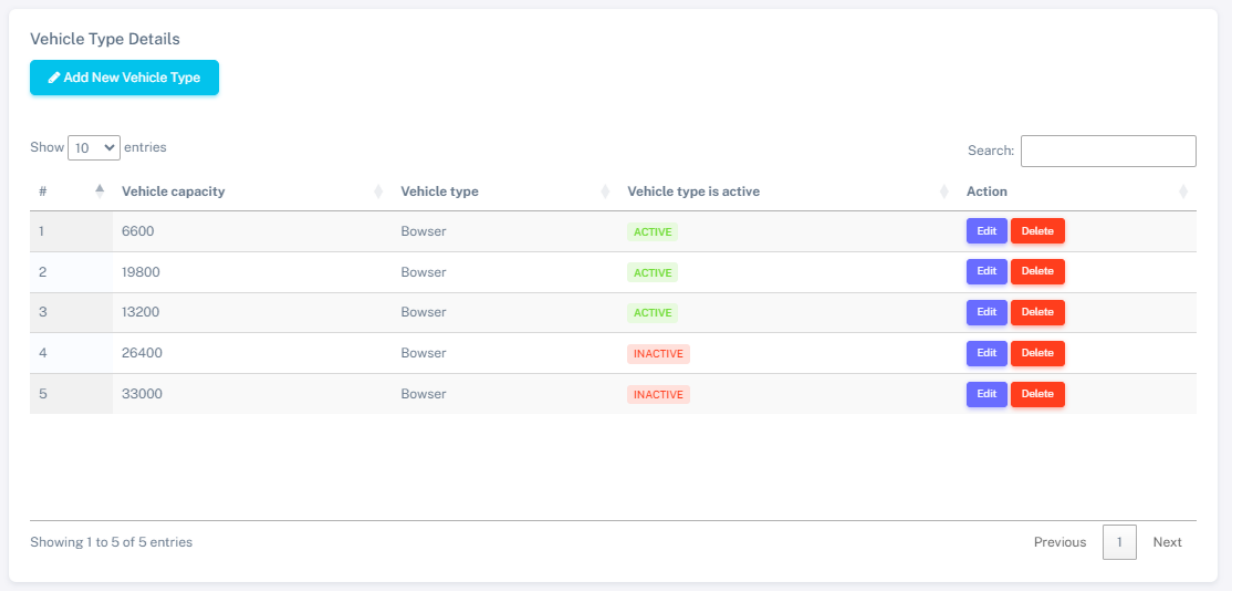


Figure 46: Vehicle type interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Vehicle Type" button.

Vehicle Management Details

- Admin user selects the "Vehicle" from the menu bar.
- Then click on the "View Details" option.

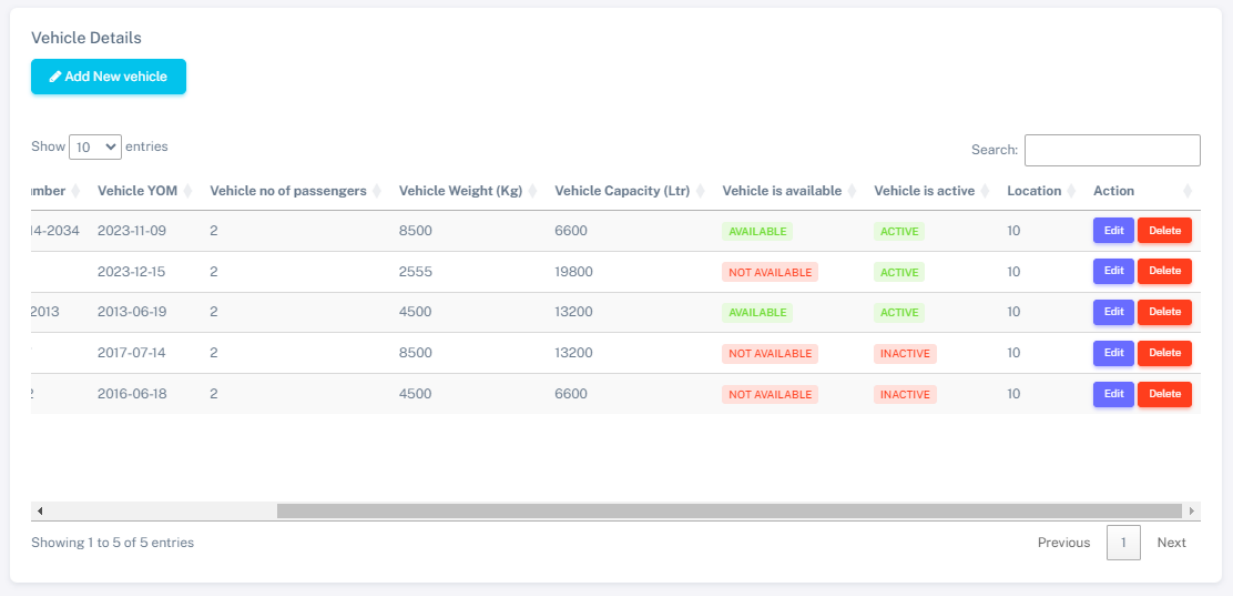


Figure 47: Vehicle details manage interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Vehicle" button.

Vehicle Calibration Certificate Details

- Admin user selects the "Vehicle Calibration" from the menu bar.
- Then click on the "View Details" option.

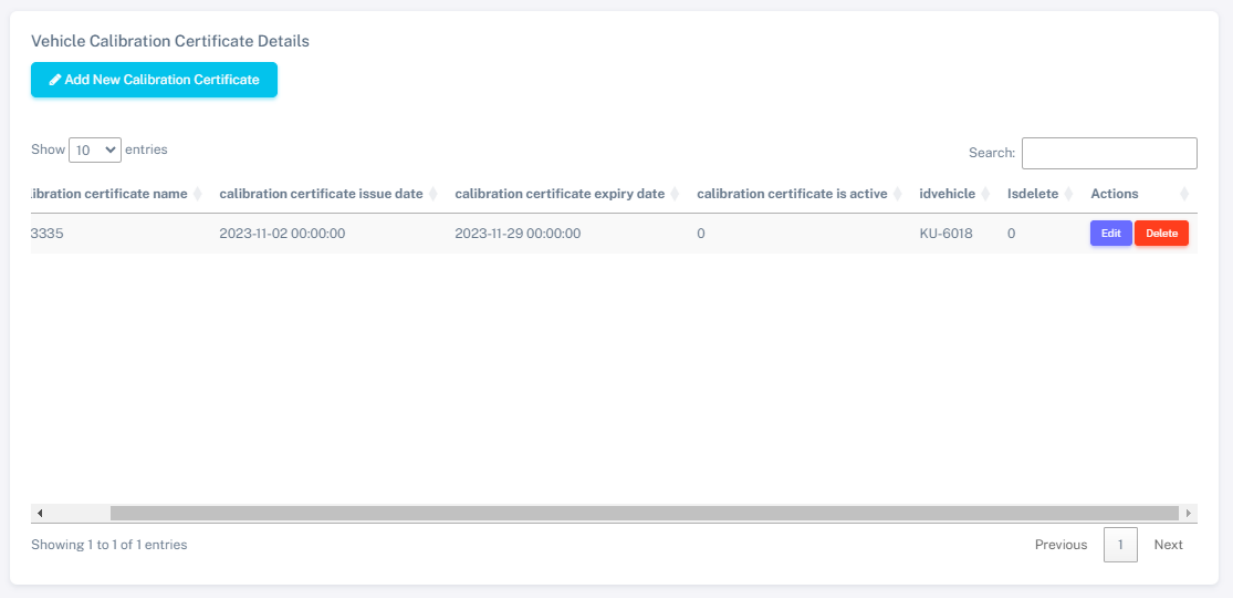


Figure 48: Vehicle Calibration certificate interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Calibration Certificate" button.

Vehicle Revenue License Details

- Admin user selects the "Vehicle Rev. License" from the menu bar.
- Then click on the "View Details" option.

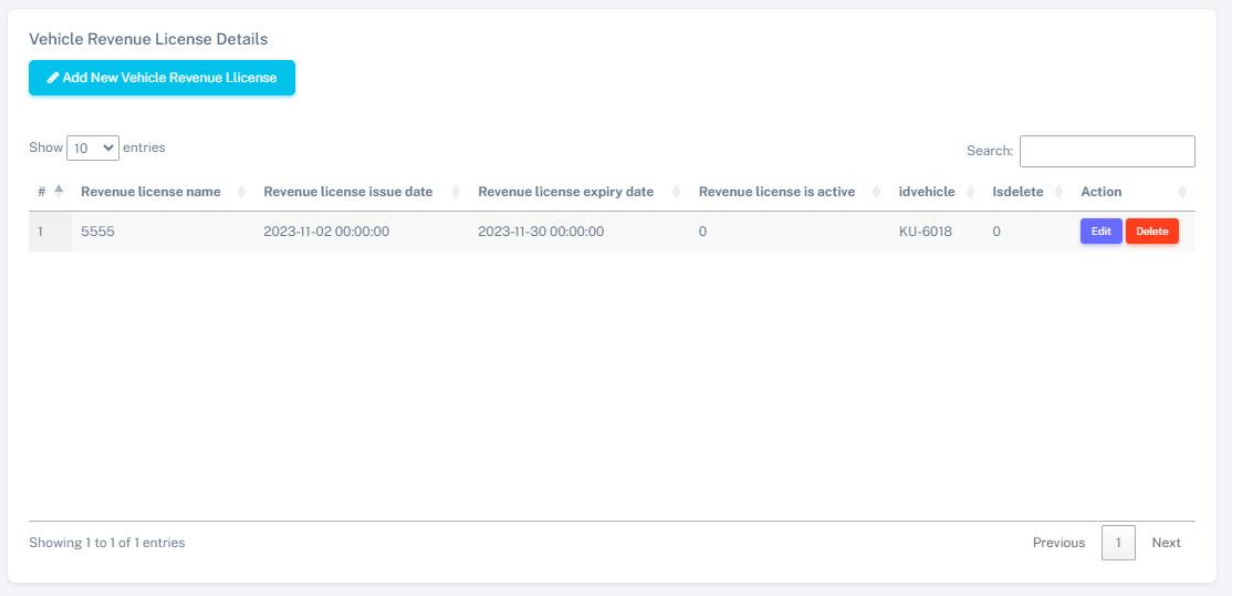


Figure 49: Vehicle revenue license interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Vehicle Revenue License" button.

Location Management Details

- Admin user selects the "Location" from the menu bar.
- Then click on the "View Details" option.

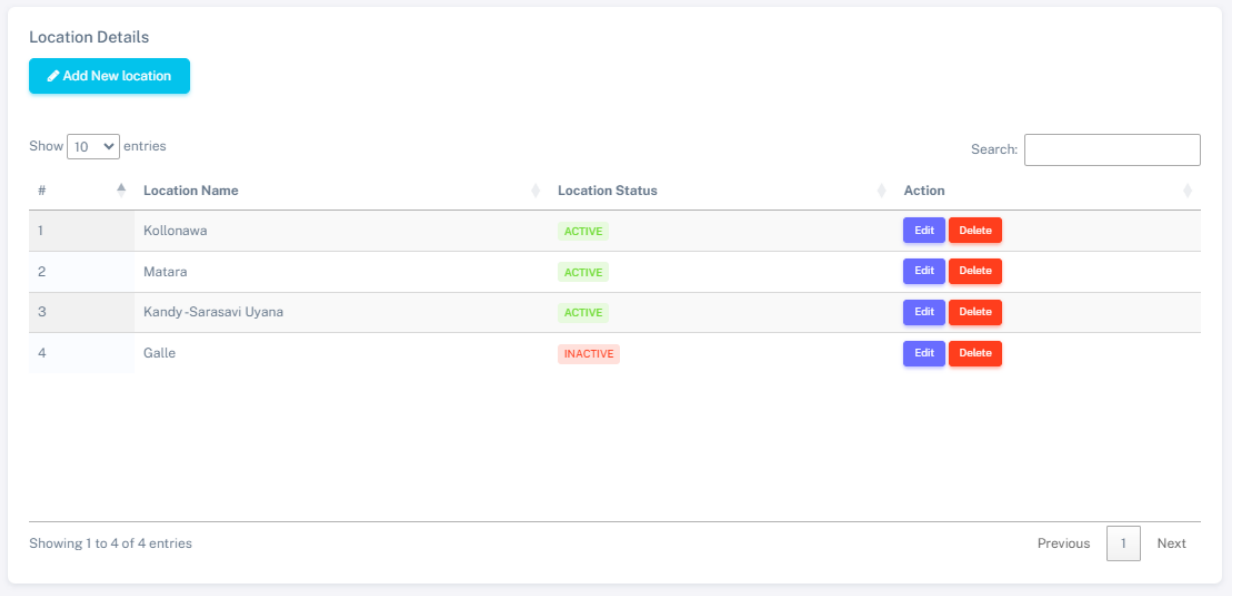


Figure 50: Location interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Location" button.

Suspend Filling Station Details.

- Admin user selects the "Approved Filling Station" from the menu bar.
- Then click on the "View Details" option.

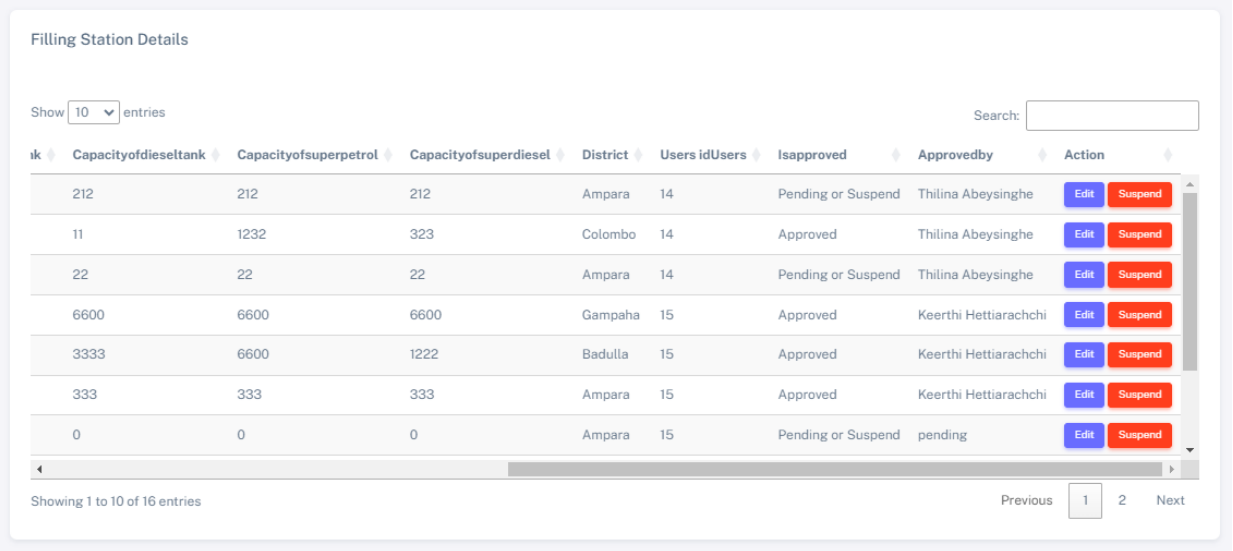


Figure 51: Suspend fuel station interface

- If the admin needs to suspend the relevant fuel station, clicks on the "Suspend" button. Then the filling station appear on approval pending registration interface until approve.

Material Price Management Details.

- Admin user selects the "Material Price" from the menu bar.
- Then click on the "View Details" option.

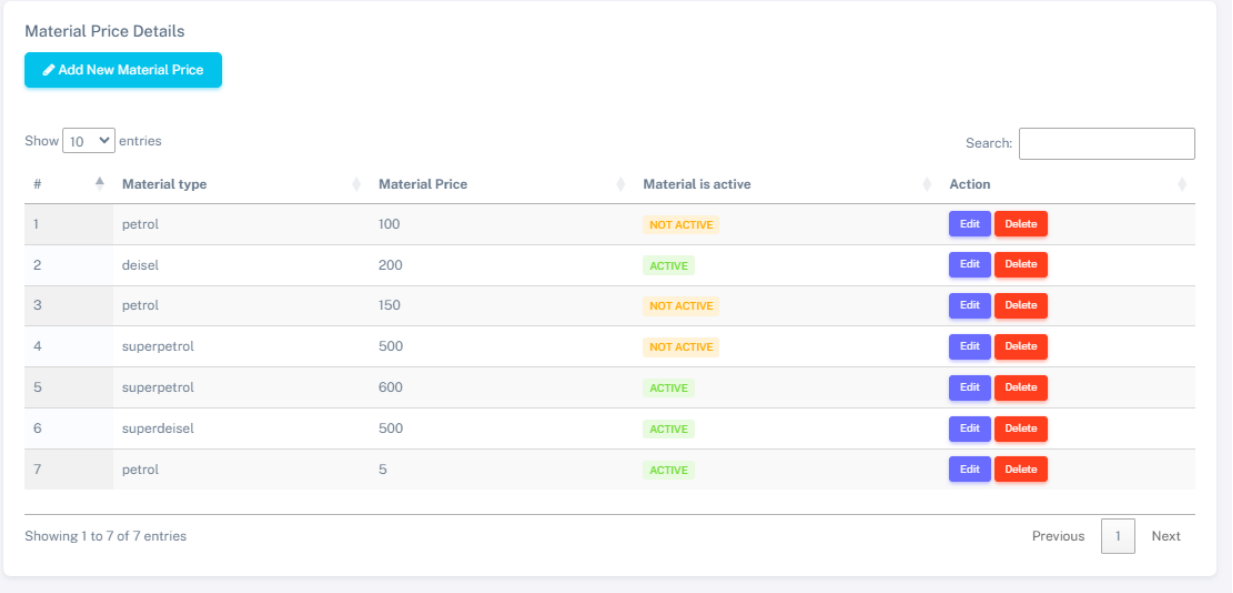


Figure 52: Material price interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Material Price" button.

User management Details.

- Admin user selects the "Users" from the menu bar.
- Then click on the "View Details" option.

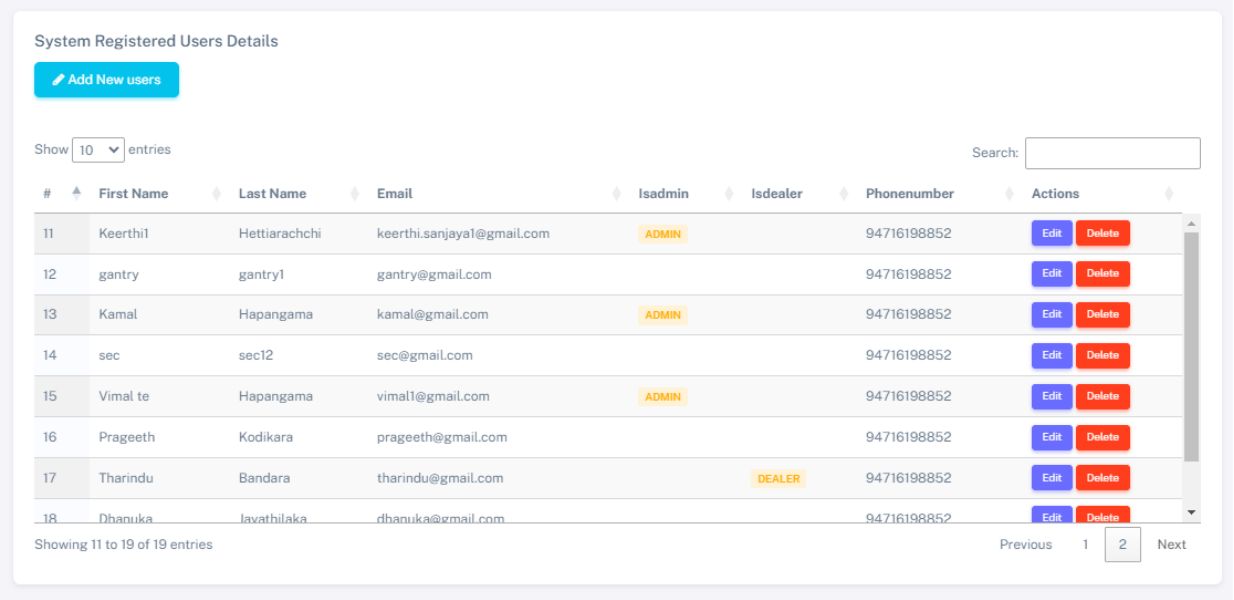


Figure 53: Users Interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Users" button.

Employee Management Details.

- Admin user selects the "Employee" from the menu bar.
- Then click on the "View Details" option.

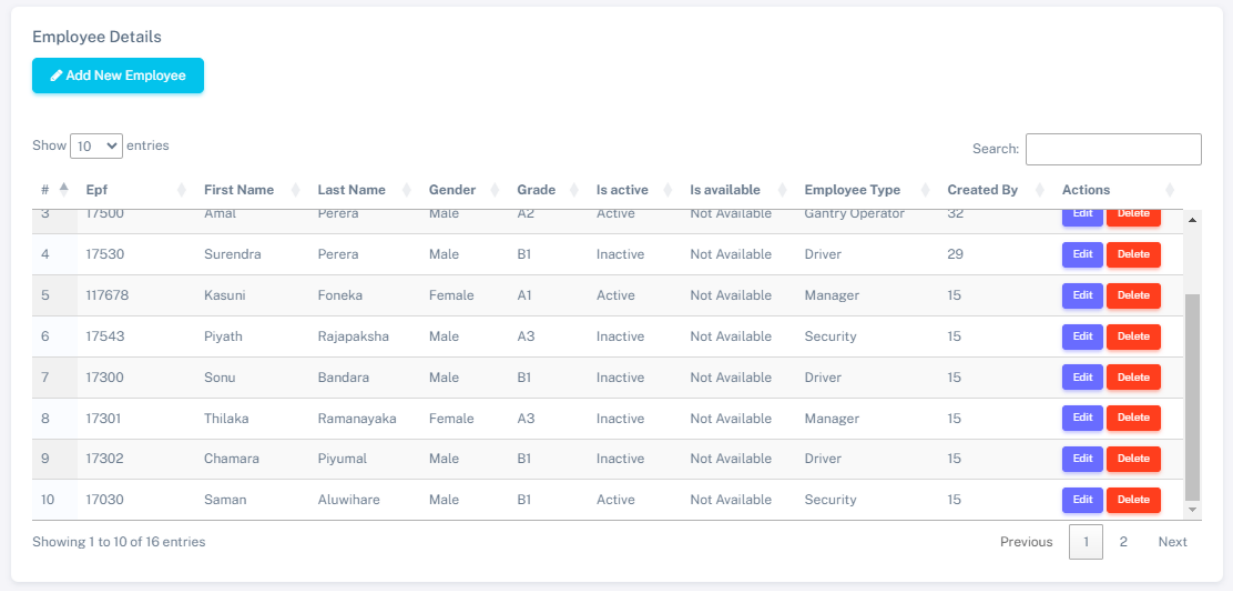


Figure 54: Employee management interface

- Admin user can edit the record by using "Edit" button.
- Admin user can delete the record by using "Delete" button.
- Admin user can add the new record by using "Add New Employee" button.

Apply as an Employee

- Admin user selects "Apply as a Employee button like below interface
- Then fill the appeared form with valid details.
- Then clicks on the "Save" button.

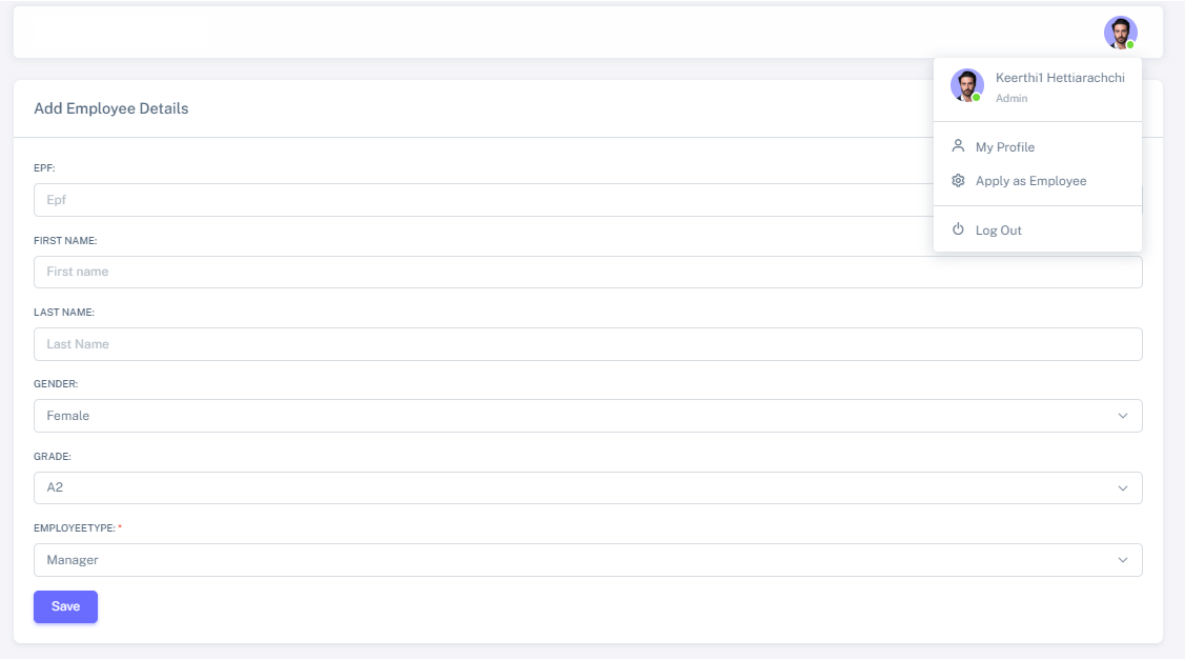


Figure 55: Apply as employee interface

Approve Employee Request

- Admin selects "Approve Employees" option from the dashboard and clicks on "Start Now" button.

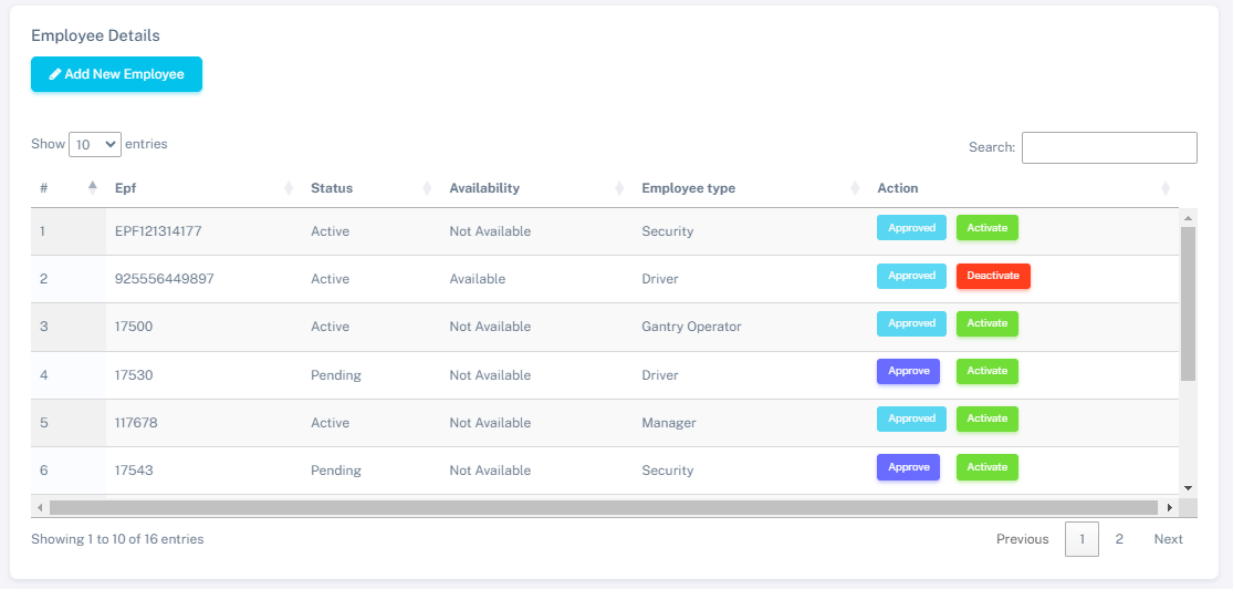


Figure 56: Approve employee request

Admin user can approve de active the employee by click on the "Activate" and "Deactivate"

button.

APPENDIX B - MIS Reports

Introduction

The MIS (Management Information System) report is a simple summary of important information collected by the system. It includes data such as the total amount of fuel ordered by dealers, the types of fuel ordered, and the frequency of orders. These reports help managers and administrators understand their filling station details and organization details for admin, which can be used to make informed decisions about management and business strategy.

Decision Making

This report consists of information in a clear manner, helping dealers to understand their orders, and stock levels. With analysis, dealers can quickly view the current status of their operations and identify areas for improvement. This report helps dealers to make strategic decisions that enhance efficiency

Record Tracking

In our online fuel ordering system for dealers, management information systems are like helpful assistants that keep track of all important records. They give us a clear summary of every transaction we make, acting as a sort of guide for future activities. This helps us keep an eye on how our business is growing and performing.

Evidence of the Report

In here display some report of the system as below.

Employee Details Report

Admin can view their employee details in comprehensive manner to get idea and separately display pie charts to get as summary to the dashboard about employee grade wise distribution and employee gender wise distribution.

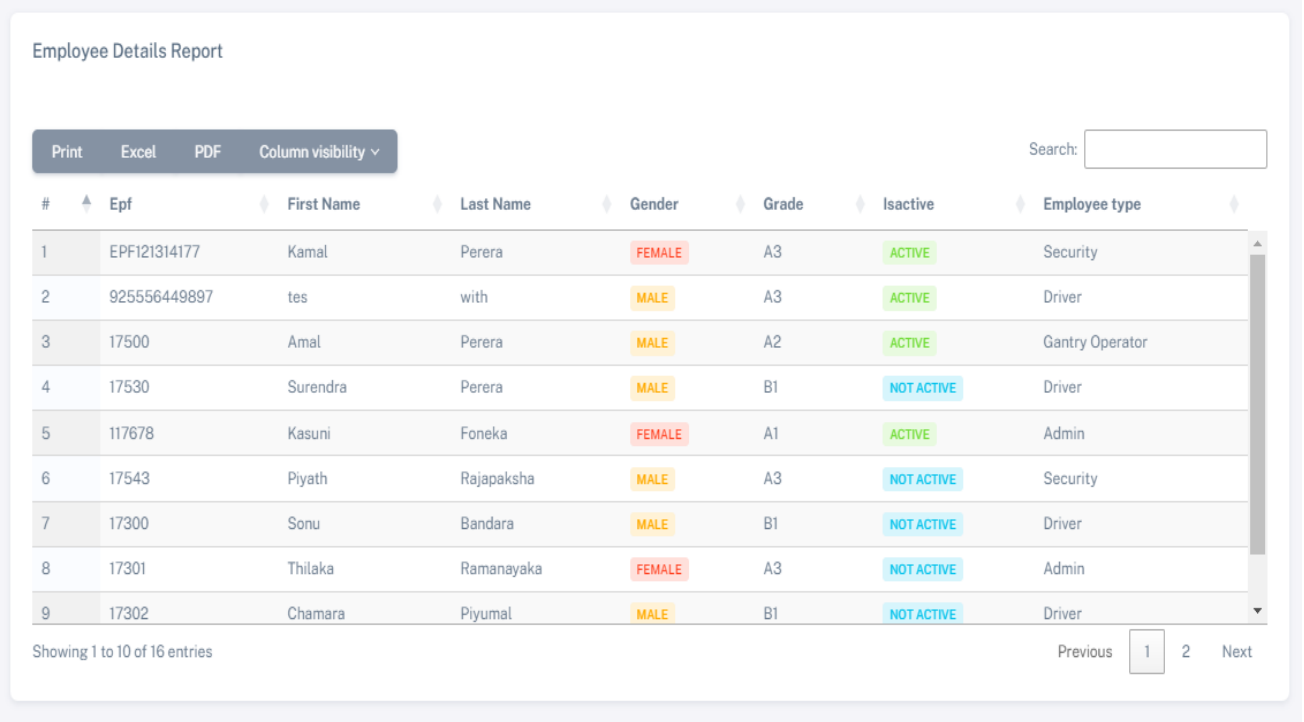


Figure 57: Employee details report

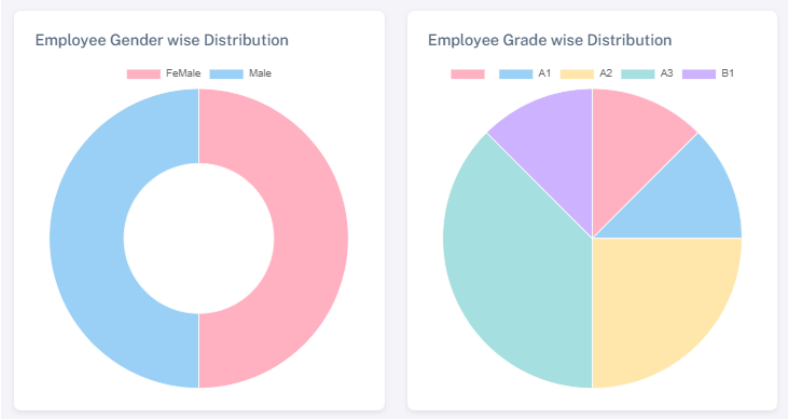


Figure 58: Dashboard pie chart

Order Items Report

This report include with filter option how the admin can make a decision about total quantity issue for a specific date period or date product wise. They can use “print”, “Excel”, and ”Pdf” option to download report and make decsions.

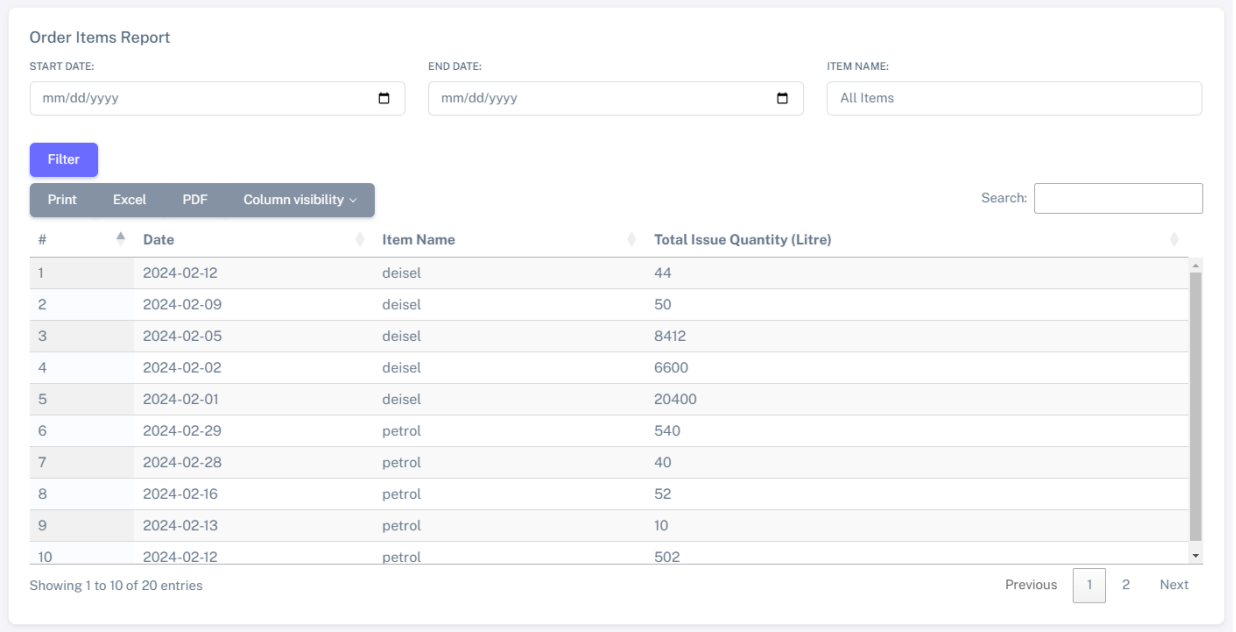


Figure 59: Order item wise product issue

Login Log Report

Login log report is really important because it tells us who has been accessing our system and when they did it. It's like a diary that keeps track of every time someone logs in. This information helps us keep our system safe.

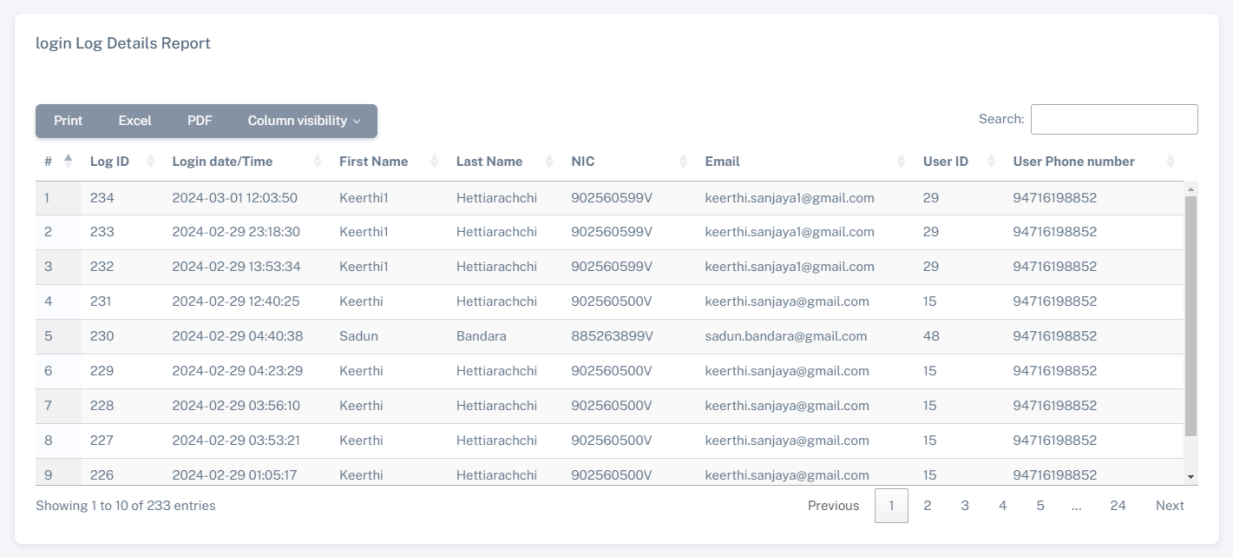


Figure 60: Login log report

Figure 61: Dashboard bar chart for total income

Filling Station Details Report

Admin can get the filling station details with by selecting the download options as well as filter option.

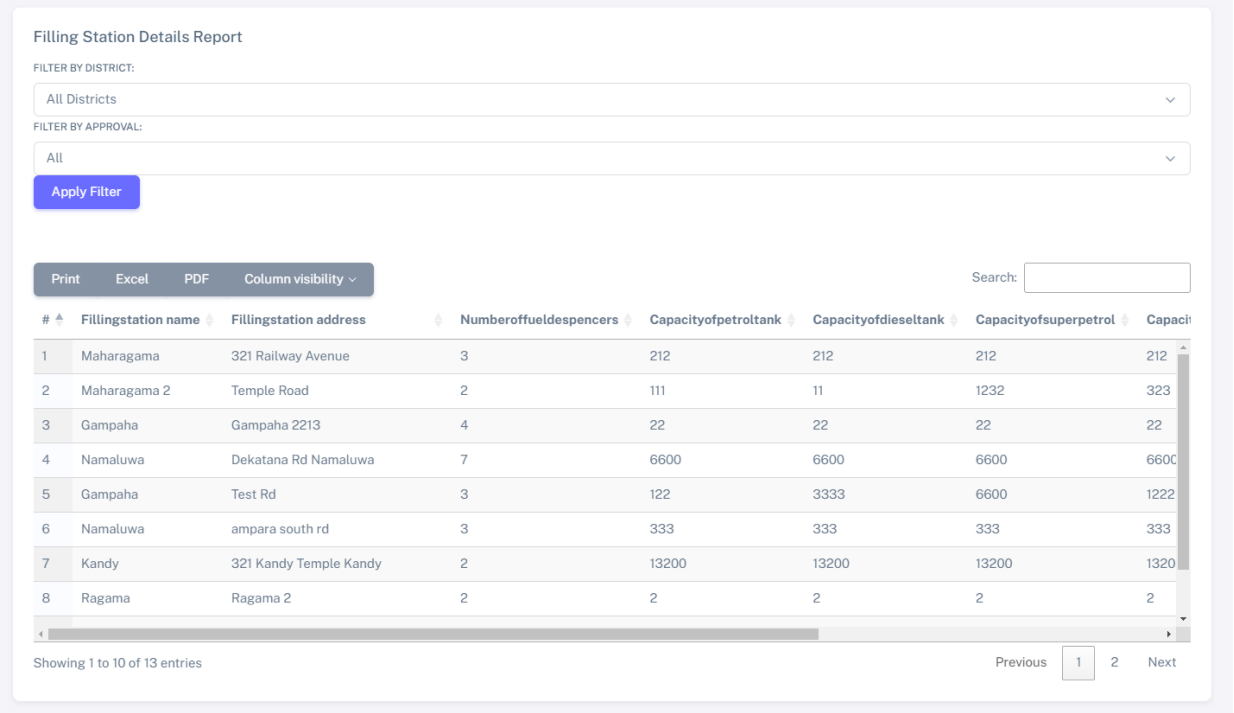


Figure 62: Filling station details report

Filling Station Owner Order Items Report

Shed owner can download and filter their fuel order details with the specific time period with product, quantity and amount.

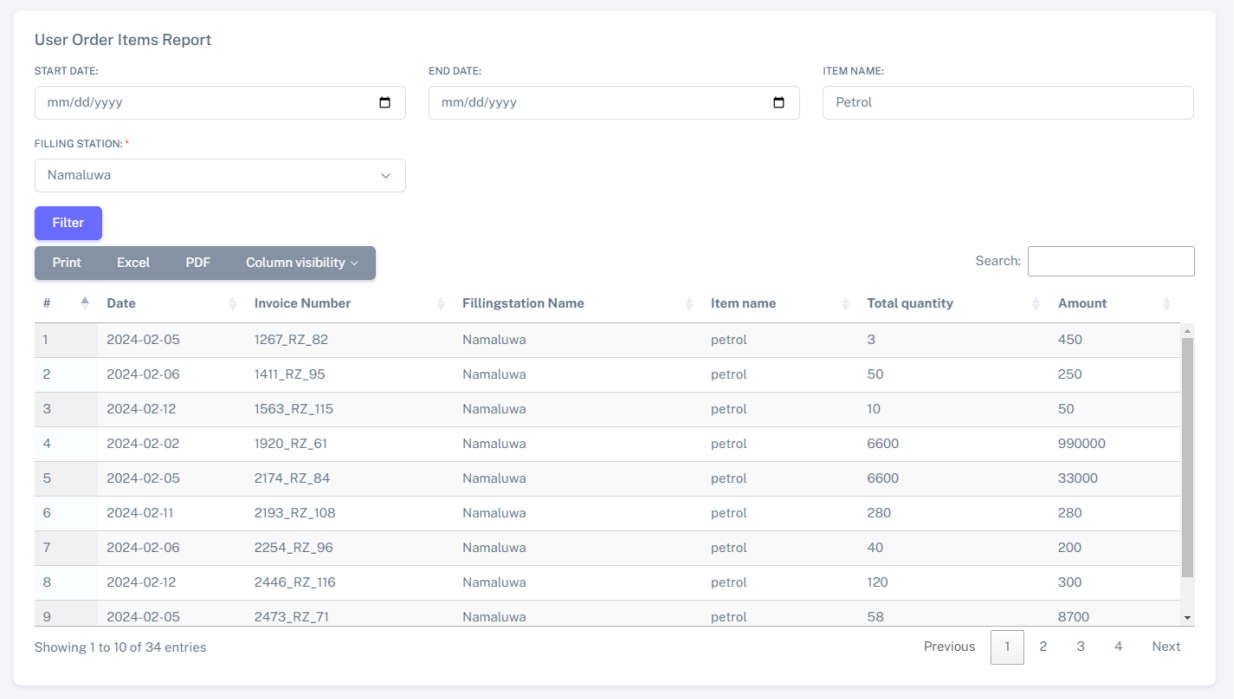


Figure 63: Filling station owner order items report