KSRTC Management System

Mini Project Report

Submitted by

Athira Ramesh

Reg. No.: AJC22MCA-2031

In Partial fulfillment for the Award of the Degree of

MASTER OF COMPUTER APPLICATIONS
(MCA TWO YEAR)
[Accredited by NBA]

APJ ABDUL KALAM TECHNOLOGICAL UNIVERSITY

AMAL JYOTHI COLLEGE OF ENGINEERING
KANJIRAPPALLY

[Affiliated to APJ Abdul Kalam Technological University, Kerala. Approved by AICTE, Accredited by NAAC. Koovappally, Kanjirappally, Kottayam, Kerala – 686518]

2023-2024

DEPARTMENT OF COMPUTER APPLICATIONS

AMAL JYOTHI COLLEGE OF ENGINEERING

KANJIRAPPALLY

CERTIFICATE

This is to certify that the Project report, "**KSRTC Management System"** is the bona fide work of **ATHIRA RAMESH (Regno: AJC22MCA-2031)** in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications under APJ Abdul Kalam Technological University during the year 2023-24.

Ms. Nimmy Francis	**Ms. Meera Rose Mathew**
Internal Guide	**Coordinator**

Rev. Fr. Dr. Rubin Thottupurathu Jose

Head of the Department

DECLARATION

I hereby declare that the project report **“KSRTC Management System”** is a bona fide work done at Amal Jyothi College of Engineering, towards the partial fulfilment of the requirements for the award of degree of the Master of Computer Applications (MCA) from APJ Abdul Kalam Technological University, during the academic year 2023-2024.

Date: **ATHIRA RAMESH**

KANJIRAPPALLY **Reg: AJC22MCA-2031**

ACKNOWLEDGEMENT

First and foremost, I thank God almighty for his eternal love and protection throughout the project. I take this opportunity to express my gratitude to all who helped me in completing this project successfully. It has been said that gratitude is the memory of the heart. I wish to express my sincere gratitude to our Manager **Rev. Fr. Dr. Mathew Paikatt** and Principal **Dr. Lillykutty Jacob** for providing good faculty for guidance.

I owe a great depth of gratitude towards our Head of the Department **Rev.Fr.Dr. Rubin Thottupurathu** Jose for helping us. I extend my whole hearted thanks to the project coordinator **Ms.Meera Rose Mathew** for her valuable suggestions and for overwhelming concern and guidance from the beginning to the end of the project. I would also express sincere gratitude to my guide **Ms.Nimmy Francis** for her inspiration and helping hand.

I thank our beloved teachers for their cooperation and suggestions that helped me throughout the project. I express my thanks to all my friends and classmates for their interest, dedication, and encouragement shown towards the project. I convey my hearty thanks to my family for the moral support, suggestions, and encouragement to make this venture a success.

ATHIRA RAMESH

ABSTRACT

.

The Kerala State Road Transport Corporation (KSRTC) information system is a comprehensive platform designed to streamline and optimize the management of the state's road transport services. At its core, the system consists of three main levels of functionality: Admin, Terminal, and Staff.

The Admin module acts as the central hub for administrative tasks, facilitating secure login for authorized personnel. It oversees the registration of KSRTC terminals, buses, and the allocation of routes and staff to ensure an efficient and well-organized transportation network. Terminals, serving as regional centers, utilize the system to manage staff, buses, and routes effectively. The terminal functionality includes features such as secure login, viewing allocated staff and transportation details, changing passwords for security, and addressing daily financial collections and customer complaints with responsive replies.

Individual staff members, including drivers and terminal personnel, access the system through their dedicated functionality. Staff functionality involves registration, secure login, profile editing, and access to essential information regarding allocated buses, routes, and daily financial collections. Staff members can also report and view responses to customer complaints.

In summary, the KSRTC information system integrates administrative oversight, terminal management, and individual staff responsibilities to create a cohesive and efficient approach to the administration of road transport services in Kerala. This comprehensive system aims to enhance transparency, operational efficiency, and overall customer satisfaction within the Kerala State Road Transport Corporation.

CONTENT

List of Abbreviation

IDE	Integrated Development Environment
HTML	Hyper Text Markup Language.
CSS	Cascading Style Sheet
SQLite	Relational Database Management System
UML	Unified Modeling Language
AJAX	Asynchronous JavaScript and XML
JS	Java Script

CHAPTER 1

INTRODUCTION

1.1 PROJECT OVERVIEW

The Kerala State Road Transport Corporation (KSRTC) information system is a sophisticated and comprehensive platform designed to optimize the management of the state's road transport services. Operating on three key levels—Admin, Terminal, and Staff—the system facilitates secure and efficient administration. The Admin module serves as the central hub for overseeing administrative tasks, including the registration of terminals, buses, and the strategic allocation of routes and staff. Terminals utilize the system to manage personnel, buses, and routes, enhancing security, addressing financial collections, and managing customer complaints. Individual staff members, including drivers and terminal personnel, engage with tailored functionality, enabling seamless communication and efficient issue resolution. In summary, the KSRTC information system aims to enhance transparency, operational efficiency, and overall customer satisfaction within the Kerala State Road Transport Corporation, offering a cohesive and adaptive solution for transportation administration.

1.2 PROJECT SPECIFICATION

Functionalities:

1.Admin Functionality:
Login
Register KSRTC terminal
Register bus
Bus route allocation
Staff allocation to terminal

2.Terminal Functionality
Login
View allocated staffs
View allocated bus and route
Change password
Assign bus to staff
View daily collection
View complaint and post replay

3.staff functionality

Register

Login

Edit profile

View allocated bus and its route

Post Daily collection

Post complaint view replay

CHAPTER 2

SYSTEM STUDY

2. INTRODUCTION

The Kerala State Road Transport Corporation (KSRTC) information system represents a pioneering and sophisticated approach to the management of road transport services within the state. This comprehensive platform operates across three pivotal levels—Admin, Terminal, and Staff—signifying a multifaceted solution tailored for secure and efficient administration. At its core, the Admin module serves as the nerve center, orchestrating key administrative tasks such as the registration of terminals, buses, and the strategic allocation of routes and staff. Terminals, functioning as regional hubs, utilize the system to optimize the management of personnel, buses, and routes, addressing security, financial transactions, and customer feedback. Individual staff members, encompassing drivers and terminal personnel, engage with purpose-built functionality, facilitating seamless communication and prompt issue resolution. In essence, the KSRTC information system is designed to elevate transparency, operational efficiency, and overall customer satisfaction within the Kerala State Road Transport Corporation, presenting a unified and adaptive solution for the dynamic landscape of transportation administration.

2.1 EXISTING SYSTEM

Presently, a notable gap exists in the availability of comprehensive application systems that seamlessly integrate essential features for efficient access to crucial services. Many of these services are intricately tied to government operations, often lacking a unified and user-friendly platform. The absence of such a consolidated system results in individuals facing challenges when seeking necessary support. Inefficiencies may arise as various services are dispersed across different government entities, leading to a fragmented user experience. Recognizing this gap, there is a need for a centralized application system that amalgamates essential features, streamlining access to a range of services. This endeavor aims to simplify and enhance the user experience, providing individuals with a one-stop solution for their diverse needs

2.2.1 NATURAL SYSTEM STUDIED

The Kerala State Road Transport Corporation (KSRTC) management system operates as a dynamic and interconnected network, analogous to a natural system in its complexity and interdependence. In this ecosystem, various components interact seamlessly to facilitate the efficient functioning of road transport services. The buses, terminals, staff, and routes form integral elements, akin to the diverse elements of a natural environment. The buses, acting as the transport vessels, navigate through predefined routes, mirroring the interconnected pathways in a natural ecosystem. Terminals serve as regional hubs, analogous to key ecological centers, coordinating the activities of buses and personnel. Staff allocation to specific terminals reflects the distribution of species in different ecological niches, optimizing the overall balance of the system. Similar to the harmonious interactions in nature, the KSRTC system involves intricate processes for allocating routes strategically. This mirrors the adaptation and allocation of resources in a natural ecosystem, ensuring optimal utilization and functionality. The administrative layer acts as the central authority, overseeing and orchestrating the various components, resembling the role of a governing force in nature that maintains balance and order. Much like the natural world, the KSRTC management system is not static; it adapts to changing conditions. The system's evolution is comparable to the natural progression and adaptation observed in ecosystems over time. In essence, the KSRTC management system embodies the complexity, interdependence, and adaptability inherent in natural systems, as it strives to provide efficient and sustainable road transport services within the state of Kerala.

2.2.2 DESIGNED SYSTEM STUDIED

The designed system for the Kerala State Road Transport Corporation (KSRTC) management reflects a well-orchestrated framework, systematically crafted to enhance efficiency, transparency, and overall operational effectiveness. Within this meticulously planned structure, various elements synergize to create a seamless and responsive network. In this designed system, buses function as optimized transport vessels, adhering to well-defined and strategically allocated routes. The routes are not only planned to connect key locations but are also adaptable to changing conditions, ensuring flexibility and efficiency in response to demand fluctuations. Terminals, strategically positioned as regional hubs, serve as centralized coordination centers. They facilitate the smooth interaction of buses and personnel, akin to well-managed ecological centers in a natural system. The allocation of staff to specific terminals is systematically organized, mirroring the distribution of roles in different ecological niches to optimize overall system balance. The system incorporates advanced processes for route allocation, embracing modern technologies to ensure optimal resource utilization. This administrative layer is equipped with tools and protocols that enhance decision-making, monitor performance, and adapt to changing conditions. Much like the natural world's adaptability

2.2 DRAWBACKS OF EXISTING SYSTEM

The existing KSRTC management system faces several drawbacks that impact its overall efficiency and user experience. One prominent issue is the lack of a centralized and integrated platform, leading to a fragmented user experience. Services and information are dispersed across various channels and platforms, contributing to difficulties in accessing comprehensive assistance.

- Lack of Centralized Platform: The absence of a centralized and integrated platform results in a fragmented user experience. Services and information are dispersed across various channels, leading to difficulties in accessing comprehensive assistance.
- Inefficient Staff Allocation: Traditional methods of staff allocation may not be optimized, leading to suboptimal distribution of personnel across terminals. This can result in uneven workloads and affect overall operational efficiency.
- Absence of Real-Time Data Analytics: The lack of real-time data analytics is a significant drawback, preventing the system from responding dynamically to operational challenges. Real-time insights are crucial for making informed decisions and optimizing resources promptly.

2.3 PROPOSED SYSTEM

The proposed KSRTC Management System envisions a transformative approach to streamline and elevate the management of road transport services. Rooted in efficiency and user-centricity, the system introduces a unified platform with three distinct functionalities: Admin, Terminal, and Staff. For administrators, the system offers secure login capabilities, along with functionalities to register new terminals, buses, allocate routes, and manage staff assignments. This central administrative hub ensures a holistic oversight of the entire transportation network. Terminals, serving as regional centers, gain access to a comprehensive suite of tools. Terminal staff can log in securely, view allocated resources such as buses and routes, manage daily collections, and effectively address customer complaints. The proposed system aims to enhance operational transparency and facilitate seamless coordination at the terminal level. Individual staff members, including drivers and terminal personnel, experience a user-friendly interface with functionalities such as secure login, profile editing, and access to critical information about allocated buses and routes. They can efficiently report daily financial collections and manage customer complaints, fostering improved communication and issue resolution. Crucially, the proposed system introduces a centralized and real-time data analytics framework. This empowers the entire KSRTC network to respond dynamically to operational challenges, providing decision-makers with timely insights for optimized resource allocation and adaptive strategies. The shift towards a more integrated and technology-driven approach positions the proposed KSRTC Management System as a catalyst for enhanced operational efficiency, customer satisfaction, and sustainability in road transport services within the state of Kerala.

2.4 ADVANTAGES OF PROPOSED SYSTEM

The proposed KSRTC Management System offers several advantages that contribute to enhanced efficiency, improved user experience, and sustainable road transport services:

- Streamlined Operations: The unified platform centralizes administrative tasks, simplifying processes related to registering terminals, buses, and allocating routes. This streamlining leads to more efficient operations and decision-making.
- Enhanced User Experience: The user-centric design, including secure logins and intuitive interfaces for administrators, terminal staff, and individual employees, fosters a positive and user-friendly experience. This can result in increased productivity and satisfaction among system users.
- Holistic Oversight: The system provides administrators with a comprehensive view of the entire transportation network. This holistic oversight allows for better planning, resource allocation, and strategic decision-making to optimize the overall system.
- Operational Transparency: Terminals gain access to a suite of tools that improve transparency in operations. This transparency facilitates better coordination, as terminal staff can view allocated resources, manage daily collections, and address customer complaints seamlessly.
- Improved Communication: The system facilitates improved communication among staff members. This is evident in functionalities that allow staff to report daily financial collections and manage customer complaints, fostering a more responsive and communicative environment

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDY

This examination is a fundamental step in determining whether a project will achieve the organization's objectives considering the resources, time invested in it. It assists the developer in assessing the potential benefits and long-term possibilities of the project. To ascertain the feasibility and worthiness of further analysis, a feasibility study must be conducted for the proposed system. The feasibility study evaluates how the proposed system would impact the organization, its ability to meet customer demands, and the efficient use of resources. Consequently, a feasibility study is typically conducted before proceeding with the development of a new application. The assessment carefully considers various factors, including technical, financial, and operational viability, as outlined in the feasibility study document.

3.1.1 Economical Feasibility

The economic feasibility study for the KSRTC Management System indicates a promising and advantageous venture. This system is anticipated to bring about substantial economic benefits by streamlining operations, enhancing efficiency, and fostering sustainability in the management of road transport services within Kerala. The proposed system's economic viability is grounded in its potential to optimize resource allocation, reduce operational costs, and improve overall productivity. By centralizing administrative tasks and introducing a unified platform, the system aims to simplify processes related to registering terminals, buses, and allocating routes. This streamlining of operations is expected to lead to more efficient decision-making, contributing to cost savings and resource optimization. The introduction of real-time data analytics is a key driver of economic feasibility. This feature empowers decision-makers with insights for optimized resource allocation, enabling a dynamic response to operational challenges. The ability to adapt swiftly to changing conditions can result in cost-effective strategies and improved overall financial performance. Improved communication among staff members further enhances economic feasibility. Functionalities that allow for the reporting of daily financial collections and efficient management of customer complaints foster a responsive and communicative environment. This can lead to increased productivity, reduced downtime, and enhanced customer satisfaction, positively impacting the economic aspects of the road transport services

3.1.2 Technical Feasibility

The technical feasibility study for the KSRTC Management System reveals a robust and viable technological foundation. The proposed system demonstrates a high degree of technical viability by leveraging contemporary technologies to streamline and enhance road transport services management in Kerala. The system's architecture, built around a unified platform, showcases its technical prowess. The centralization of administrative tasks, including the registration of terminals, buses, and the allocation of routes, signifies a seamless integration of various components.
This centralized approach enhances system reliability, accessibility, and overall performance. The incorporation of real-time data analytics is a key technical advantage. This feature empowers decision-makers with dynamic insights, enabling them to respond promptly to operational challenges. The system's adaptability to changing conditions showcases its technical sophistication, ensuring the utilization of cutting-edge tools for optimized resource allocation. Furthermore, the secure login functionalities and intuitive interfaces for administrators, terminal staff, and individual employees contribute to the system's technical feasibility. This emphasis on user-friendly design enhances accessibility and usability, making the system adaptable to the technical competencies of diverse users.

3.1.3 Behavioral Feasibility

The behavioral feasibility study for the KSRTC Management System reveals a positive outlook, indicating a high likelihood of user acceptance and successful integration into the existing organizational culture. The proposed system recognizes and addresses the behavioral aspects of the users, ensuring a smooth transition and fostering positive user experiences. One of the key behavioral strengths lies in the user-centric design of the system. The emphasis on secure login functionalities and intuitive interfaces for administrators, terminal staff, and individual employees contributes to the system's behavioral feasibility. This design approach enhances accessibility and usability, making the system adaptable to the diverse behavioral preferences and competencies of users. The centralization of administrative tasks, such as the registration of terminals, buses, and route allocation, aligns with existing behavioral patterns within the organization. By streamlining these processes, the system minimizes resistance to change and encourages user acceptance. The behavioral feasibility is further enhanced by the system's ability to cater to the varying needs and preferences of different user groups.

3.1.4 Feasibility Study Questionnaire

1. Project Overview?

   The KSRTC Management System is a holistic solution designed for the modernization and optimization of road transport services in Kerala. Through a unified platform, it streamlines crucial administrative tasks like terminal registration, bus management, and route allocation. The system's user-centric design accommodates the diverse needs of administrators, terminal staff, and individual employees. Leveraging real-time data analytics empowers decision-makers, encouraging adaptability and fostering a culture of innovation. The project's primary objective is to elevate efficiency, transparency, and sustainability within the Kerala State Road Transport Corporation.

2. To what extend the system is proposed for?

   The proposed KSRTC Management System is designed to comprehensively address and enhance the entire spectrum of road transport services within the Kerala State Road Transport Corporation. From streamlining terminal operations, bus management, and route allocation to providing user-centric functionalities for administrators, terminal staff, and individual employees, the system aims to be an integrated solution. Leveraging real-time data analytics, it is poised to empower decision-makers, fostering adaptability and innovation across the organization. In essence, the system is proposed to revolutionize and optimize the entire administrative landscape of road transport services in Kerala.

3 .Identify the users in your project?

- Admin
- Terminal
- Staff (Conducter/Driver)

4.Are you familier with the concept of KSRTC management system?

yes

5 .Does the system provide a robust mechanism for safeguarding user information, thereby ensuring optimal security?

The system guarantees a high level of security in the storage of user information.

6.Is the required technology available and accessible for the implementation of the system?

Yes,the required technology,including hardware and software,is readily available and accessible for the implementation of the KSRTC management system

7.What specific goal and objectives do you believe this management system shuld achieve for KSRTC?

The system will improve route scheduling and fleet management .reduce operational costs,and enhance passenger services.

8. How do you envision this system improving KSRTC operation and services?

Provide real time data for better decision making ,automate manual processes,and improve resource allocation.

7.What are the understanding of the goals and objectives of this project?

The project aims to modernize KSRTC internal operations and improve efficiency.

9. To what extend the system is proposed for?

The proposed KSRTC Management System is designed to efficiently manage the operations of the Karnataka State Road Transport Corporation, catering to the diverse needs of administrators, terminal staff, and bus operators. For administrators, the system facilitates secure login, terminal registration, and bus registration, along with features for bus route allocation and staff assignment. Terminal functionalities include login access, viewing allocated staff and buses, password management, and the assignment of buses to staff. Staff members can register, log in, edit profiles, view allocated buses and routes, and contribute to daily collections and complaint management. The system aims to streamline tasks, enhance communication, and optimize resource allocation for a more effective and transparent management of the state-owned road transport corporation.

10. Specify the viewers/public which is to be involved in the system in a short paragraph

The KSRTC Management System targets a diverse audience involved in the operations and oversight of the Karnataka State Road Transport Corporation. This includes administrators responsible for strategic decision-making, terminal staff engaged in day-to-day operations, and bus operators contributing to service delivery. Administrators gain insights into system-wide performance, ensuring efficient resource allocation. Terminal staff benefit from streamlined processes and improved communication, enhancing their ability to manage allocated buses and staff. Bus operators have tools for daily collections and a platform for lodging and reviewing complaints. The system is designed to be accessible, providing a transparent and collaborative environment for all stakeholders involved in the successful functioning of the state-owned road transport corporation.

11. Who owns the system

The KSRTC Management System is owned and operated by the Kerala State Road Transport Corporation (KSRTC). As a state-owned entity, the corporation takes full ownership and responsibility for the development, implementation, and maintenance of the system. The system aligns with the corporation's goals of enhancing operational efficiency, optimizing resource allocation, and providing improved services to both administrators and end-users. Ownership by the KSRTC ensures that the system is tailored to the specific needs and objectives of the state's road transport operations, contributing to the overall success and effectiveness of the organization.

3.1 SYSTEM SPECIFICATION

3.2.1 Hardware Specification

Processor	- Intel I3 above
RAM	- 4 GB
Hard disk	- 256 GB

3.2.2 Software Specification

Front End - HTML5, Bootstrap, CSS

Back End - Django Framework, Python

Database - SQLite

Client on PC - Windows 7 and above.

Technologies used - JS, HTML5, AJAX, J Query, Python, CSS, Django, SQLite

3.3 SOFTWARE DESCRIPTION

3.3.1 DJANGO:

The Django Framework is a popular and robust web framework for Python developers. It is revered for its simplicity, clean code, and rapid development capabilities. Django is built on the Model-Template-Views (MTV) architectural pattern, which shares similarities with the Model-View-Controller (MVC) pattern used in other frameworks. One of its standout features is the Object-Relational Mapping (ORM) system, simplifying database interactions by representing database tables as Python objects. This abstraction eliminates the need for writing raw SQL queries, making database operations more straightforward.

Django also offers a built-in administrative interface, making content management a breeze. Its URL routing system allows developers to define clean and user-friendly URLs for their web applications. Furthermore, Django provides comprehensive support for form handling, data validation, and user authentication, reducing the complexity of common web development tasks. Security is a top priority in Django, with built-in protections against common web vulnerabilities like Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF). With a thriving community and an array of reusable packages, Django is a versatile choice for web development projects of varying sizes and complexities.

3.3.2 SQLite:

SQLite is a lightweight, self-contained, and serverless relational database management system. Unlike traditional databases, it does not require a separate server but is embedded directly within the application. One of its standout features is the self-contained nature of SQLite databases; the entire database is stored in a single file, simplifying management, backups, and transfers. Despite its lightweight design, SQLite is ACID-compliant, meaning it ensures data integrity by supporting transactions, making it suitable for multi-user environments.

SQLite is cross-platform and compatible with various operating systems, making it a versatile choice for applications targeting different platforms. Its small code size and minimal resource usage make it suitable for resource-constrained environments, such as mobile devices. SQLite is known for its speed and efficiency, especially in read-heavy workloads. It is commonly used in mobile applications, desktop applications, and embedded systems, where a full-fledged database server might be excessive, and portability is essential.

CHAPTER 4
SYSTEM DESIGN

4.1 INTRODUCTION

Any designed system or product's development starts with the design phase. An efficient system depends on well-executed design, which is a creative process. It entails utilizing a variety of approaches and concepts to define a process or system in enough depth to allow for its actual execution. Regardless of the development model chosen, the design phase is critical in software engineering. It strives to produce the architectural detail needed to build a system or product and serves as the technical backbone of the software engineering process. This program has through a thorough design phase that optimizes every aspect of effectiveness, performance, and accuracy. A user-oriented document is converted into a document for programmers or database employees during the design process.

4.2 UML DIAGRAM

A standardized dialect called Unified Modelling Language (UML) is utilized to conceptualize, characterize, plan, and depict program frameworks. The Question Administration Gather (OMG) was dependable for creating UML, and the primary draft of the UML 1.0 definition was discharged in January 1997. Programming dialects like Java, C++, and COBOL is not the same as UML. It could be a nonexclusive visual demonstrating dialect utilized for computer program frameworks and a pictorial dialect utilized for program outlines. UML may be utilized for non-software frameworks, such as fabricating forms, indeed though it is generally utilized to speak to program frameworks.

- Class diagram
- Object diagram
- Use case diagram
- Sequence diagram
- Collaboration diagram
- Activity diagram
- State chart diagram
- Deployment diagram
- Component diagram

4.2.1 USE CASE DIAGRAM

A use case diagram may be a graphical delineation that appears how clients and other outside onscreen characters associated with a system's inside components. A use case diagram's essential work is to perceive, layout, and orchestrate a system's utilitarian needs as seen through the eyes of its clients. The Unified Modelling Language (UML), a standard language for modelling actual things and systems, is frequently used to construct use case diagrams.

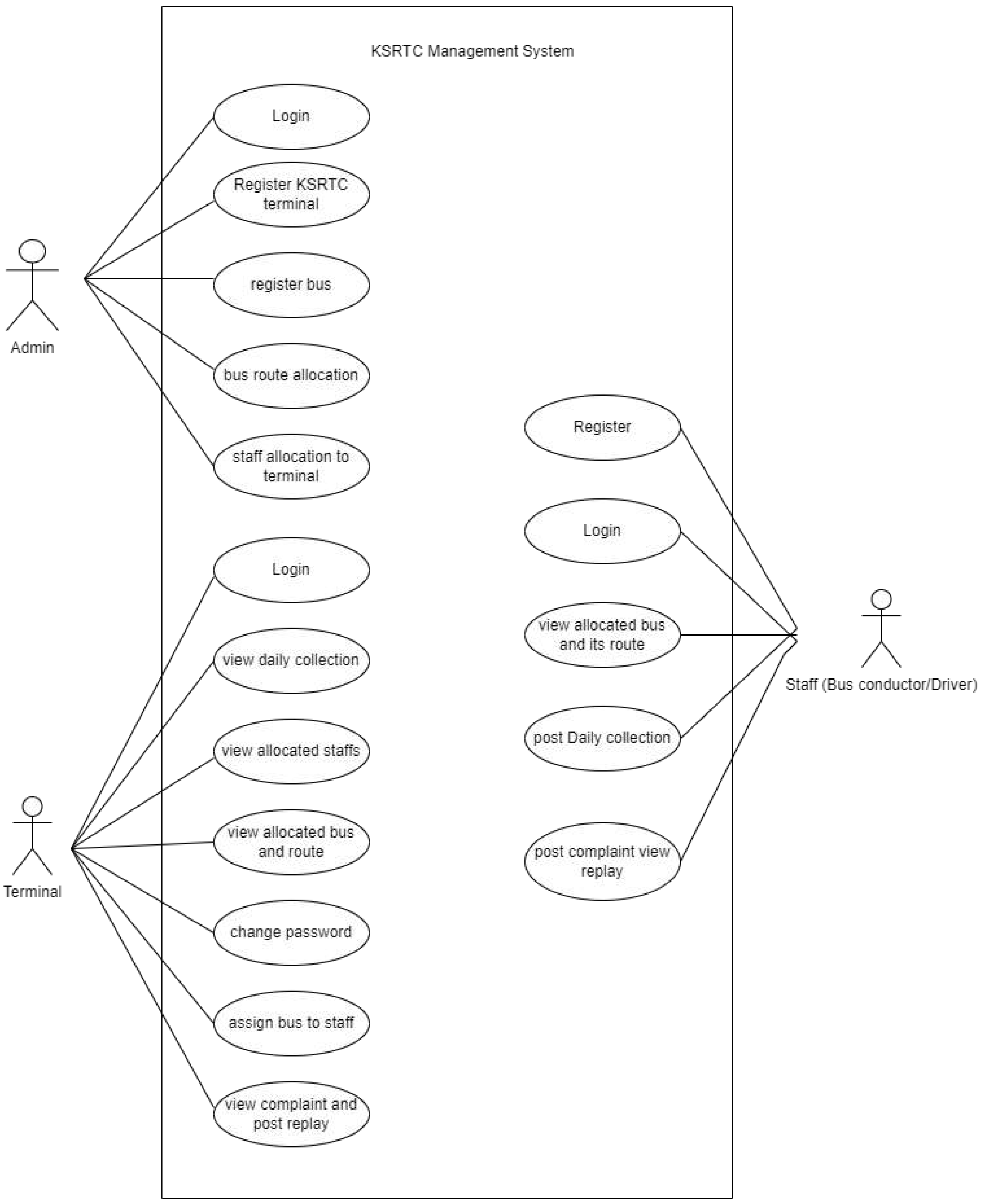
Use cases can be utilized to achieve an assortment of framework objectives, counting setting fundamental prerequisites, confirming equipment plans, testing, and investigating program, creating online offer assistance references, or performing client bolster obligations. Customer support, product obtaining, catalogue overhauling, and payment processing are as it were a couple of illustrations of use cases within the setting of item deals. The system boundaries, actors, use cases, and their connections together make up a use case diagram. The system boundary establishes the system's boundaries in reference to its surroundings. Actors are often defined depending on the roles they play and reflect the people or systems that interact with the system. The precise activities or behaviors that actors carry out within or close to the system are known as use cases. Finally, the graphic shows the connections between actors and use cases as well as the use cases themselves.

Use case diagrams are graphical representations used to capture the functional requirements of a system. When drawing a use case diagram, it is important to follow these guidelines to ensure an efficient and effective diagram:

- Choose descriptive names for use cases that accurately reflect the functionalities they perform.
- Assign appropriate names to actors to help identify their roles in the system.
- Ensure that relationships and dependencies are clearly depicted in the diagram.
- Avoid including every possible relationship, as the main goal is to identify the essential requirements.
- Use notes when necessary to clarify important points.

By following these guidelines, we can create a clear and concise use case diagram that accurately represents the functional requirements of the system.

Use Case Diagram



4.2.1 SEQUENCE DIAGRAM

The chronological order of interactions between various system components is shown in a sequence diagram, a form of interaction diagram. It demonstrates how several things communicate with one another over the course of a series of messages. These images are sometimes referred to as event scenarios or event scenarios diagrams. In software engineering, sequence diagrams are frequently used to describe and comprehend the needs of both new and old systems. They support the visualization of object control relationships and the detection of systemic issues.

Sequence Diagram Notations –

i **Actors** - In UML, a role that interacts with the system and its objects is represented by an actor. Actors frequently exist outside of the system that the UML diagram is intended to portray. Actors can play a variety of roles, including those of external topics or human users. A stick person notation is used in UML diagrams to represent actors. Depending on the situation that is being modelled, a sequence diagram may have more than one actor.

ii **Lifelines** - A lifeline in a sequence diagram is a vertical dashed line that represents the lifespan of an object participating in the interaction. Each lifeline represents an individual participant in the sequence of events and is labeled with the name of the participant. The lifeline shows the timeline of events for the participant and is drawn as a vertical line extending from the participant's activation point to its deactivation point.

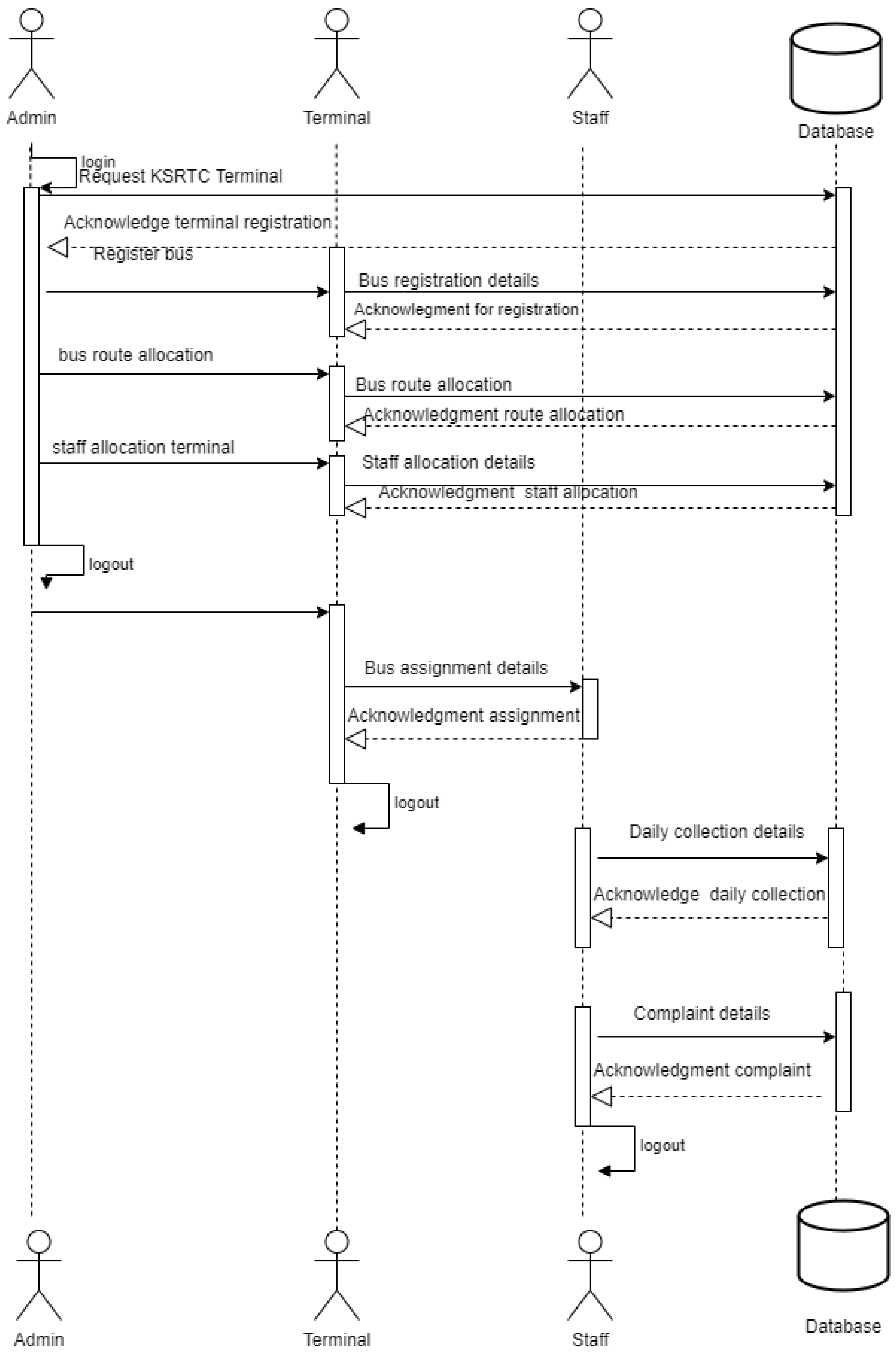
iii **Messages** - Messages are a key component of sequence diagrams, representing the interactions and communication between objects or components in a system. They can be categorized into synchronous and asynchronous messages, create, and delete messages, self-messages, reply messages, found messages, and lost messages. Guards are also used to model conditions and restrictions on message flow.

iv **Guards** - Guards in UML are used to model conditions and are employed to restrict the flow of messages when a certain condition is met. This feature is essential for letting software developers know about any constraints or limitations associated with a system or a particular process.

Uses of sequence diagram –

Modeling and visualizing the logic of complex functions, operations, or procedures. Showing details of UML use case diagrams. Understanding the detailed functionality of current or future systems.

Sequence Diagram


Admin
Terminal
Staff
Database
login
Request KSRTC Terminal
Acknowledge terminal registration
Register bus
Bus registration details
Acknowlegment for registration
bus route allocation
Bus route allocation
Acknowledgment route allocation
staff allocation terminal
Staff allocation details
Acknowledgment staff allocation
logout
Bus assignment details
Acknowledgment assignment
logout
Daily collection details
Acknowledge daily collection
Complaint details
Acknowledgment complaint
logout
Admin
Terminal
Staff
Database

4.2.2 State Chart Diagram

A state diagram is a visual representation, often created using the Unified Modeling Language (UML), that shows the different states that an object can exist in and how it can transition between those states. It is also referred to as a state machine diagram or state chart diagram.

The State Chart Diagram is a behavioral diagram in UML that describes the behavior of a system or object over time. It includes various elements such as:

Initial State - This state represents the starting point of the system or object and is denoted by a solid black circle.

State - This element describes the current state of the system or object at a specific point in time and is represented by a rectangle with rounded corners.

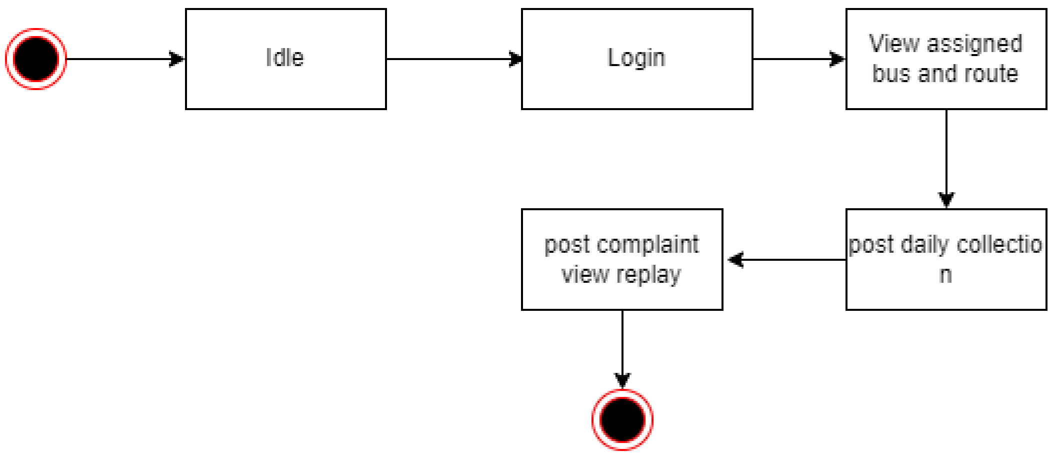
Transition - This element shows the movement of the system or object from one state to another and is represented by an arrow.

Event and Action - An event is a trigger that causes a transition to occur, and an action is the behavior or effect of the transition.

Signal - A message or trigger caused by an event that is sent to a state, causing a transition to occur.

Final State - The State Chart Diagram ends with a Final State element, which is represented by a solid black circle with a dot inside. It indicates that the behavior of the system or object has completed.

State Chart Diagram



4.2.2 Activity Diagram

An activity diagram is a visual representation of a workflow that shows how one activity leads to another. An activity is referred to as a system operation, and one operation leads to another in the control flow. A flow can be parallel, concurrent, or branched, and activity diagrams use various functions such as branching, joining, etc., to manage all types of flow control. Activity diagrams are a type of behavior diagram that shows the behavior of a system. They show the flow of control from the start point to the end point and show the different decision paths that exist during the execution of the activity.

The key components of an activity diagram are:

Initial node - A starting point of the activity diagram, denoted by a black circle.

Activity - A task or action performed by the system or entity, represented by a rectangle with rounded corners.

Control flow - It represents the sequence of activities or actions performed by the system or entity, represented by an arrow.

Decision node - A decision or branching point in the activity flow, denoted by a diamond shape.

Merge node - Used to merge multiple branches of the activity flow into a single flow, represented by a diamond shape with a plus sign inside.

Fork node - Used to split the activity flow into multiple parallel flows, represented by a solid black circle with multiple arrows.

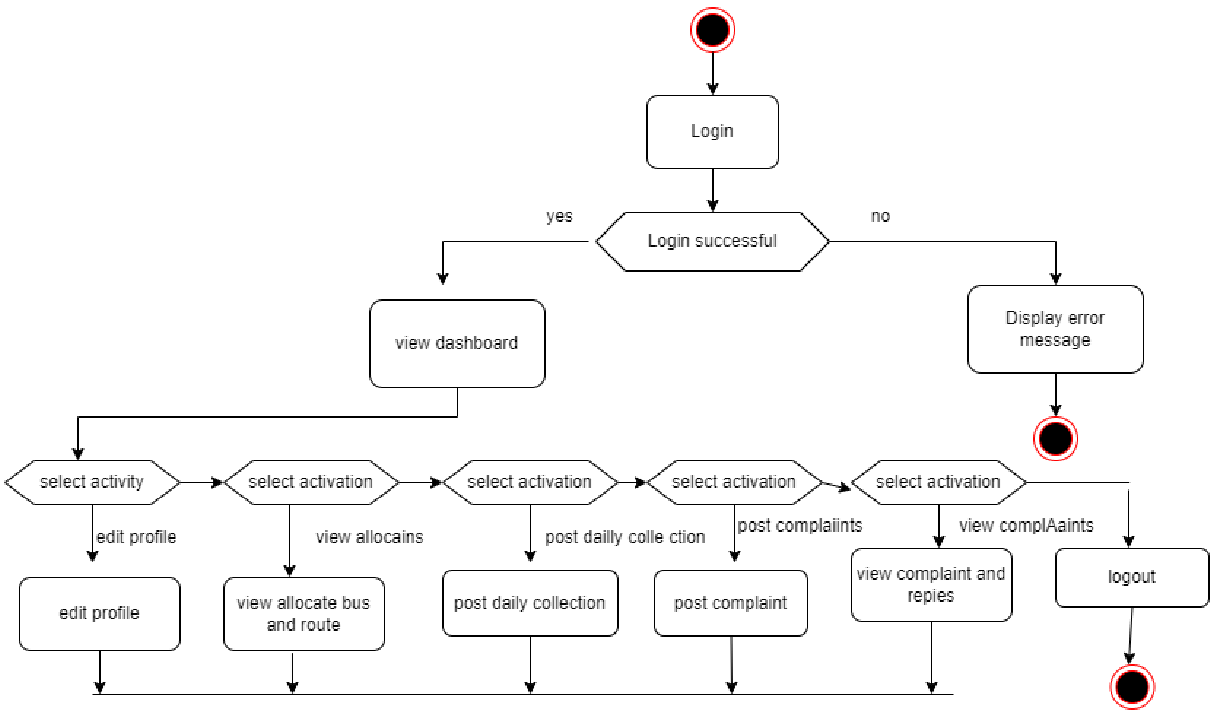
Join node - Used to join multiple parallel flows back into a single flow, represented by a solid black circle with multiple arrows pointing towards it.

Final node - The end point of the activity diagram, denoted by a black circle with a dot inside.

Object flow - Represents the flow of objects or data between activities, represented by a dashed arrow.

Activity diagrams are useful in clarifying complex processes, identifying potential issues, and communicating process flows to stakeholders and project team members.

Activity Diagram



4.2.3 Class Diagram

The class diagram is a fundamental component of object-oriented modeling and serves as the primary means of conceptual modeling for the structure of an application. Additionally, class diagrams can be used for detailed modeling that can be translated into programming code. They can also be employed for data modeling purposes. Class diagrams are a crucial component of UML used to represent classes, objects, interfaces, and their relationships and attributes in a system.
Some important components of a class diagram are:
Class: It is a blueprint or template for creating objects and is represented as a rectangle with the class name, attributes, and methods.

Interface: It is a collection of abstract methods that specify a contract between a class and the outside world. It is represented as a circle with the interface name inside.

Object: It is an instance of a class with state and behavior. It is represented as a rectangle with the object name inside.

Association: It is a relationship between two classes that represents a connection or link and is represented as a line with optional directionality, multiplicity, and role names.

Aggregation: It is a part-whole relationship where the whole (aggregator) is composed of parts (aggregates) and is represented as a diamond shape on the aggregator side.

Composition: It is a stronger form of aggregation where the parts cannot exist without the whole and is represented as a filled diamond shape on the aggregator side.

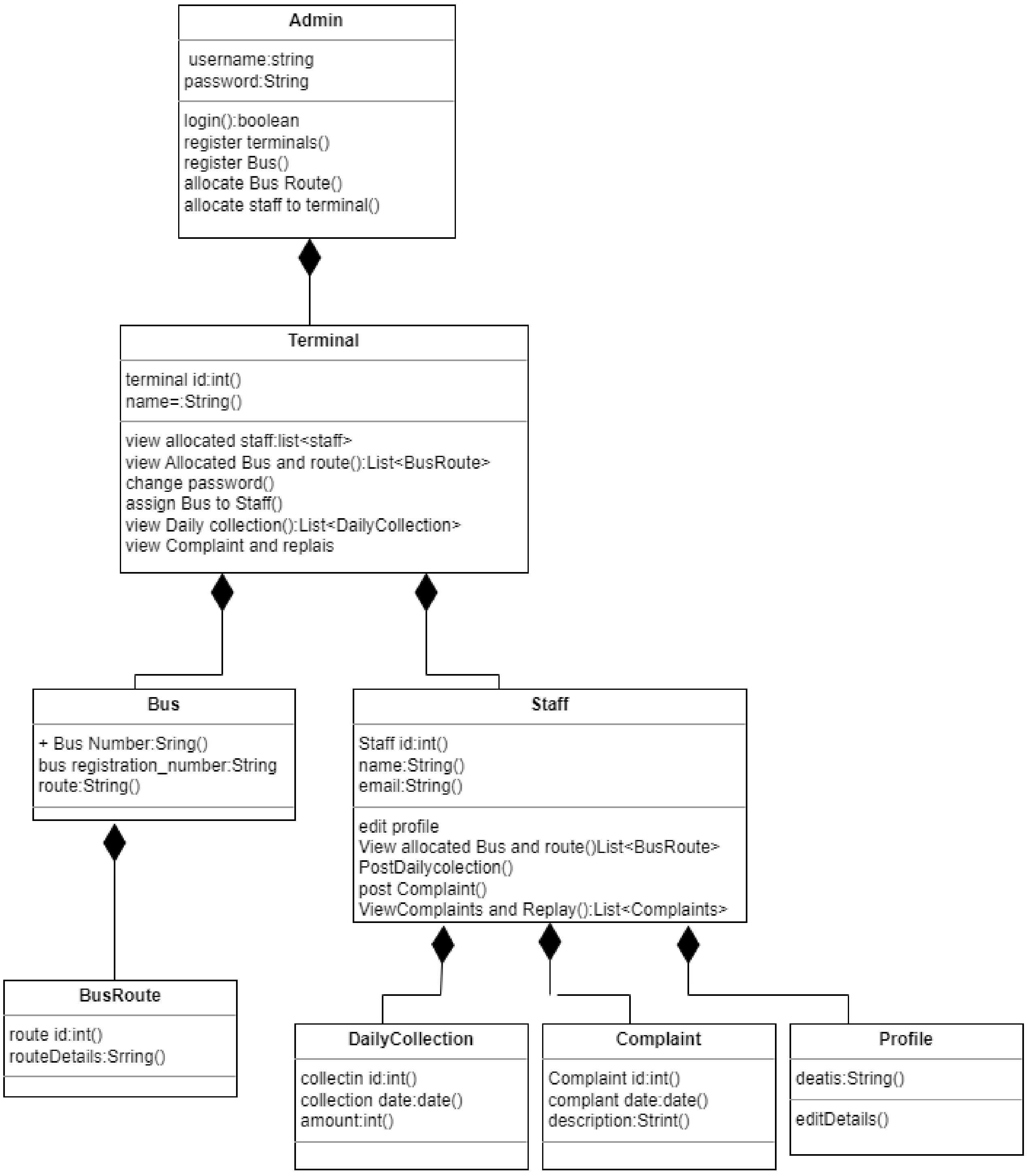
Inheritance: It is a relationship between a superclass and its subclasses that represents an "is-a" relationship and is represented as a line with an open arrowhead pointing from the subclass to the superclass.

Dependency: It is a relationship where a change in one class may affect the other class and is represented as a dashed line with an arrowhead pointing from the dependent class to the independent class.

Multiplicity: It represents the number of instances of a class that can be associated with another class and is represented as a range of values near the association or aggregation line.

Class diagrams are essential in designing and modeling object-oriented software systems as they provide a visual representation of the system's structure, its functionality, and the relationships between its objects. They facilitate software development, maintenance, and improve communication among team members.

Class Diagram



4.2.4 Object Diagram

Class diagrams and object diagrams are closely related in object-oriented modeling. Object diagrams are instances of class diagrams, which represent a snapshot of the system at a given moment in time. Both types of diagrams use the same concepts and notation to represent the structure of a system. While class diagrams are used to model the structure of the system, including its classes, attributes, and methods, object diagrams represent a group of objects and their connections at a specific point in time.

An object diagram is a type of structural diagram in UML that shows instances of classes and their relationships. The main components of an object diagram include:

Object: An object is an instance of a class that represents a specific entity in the system. It is represented as a rectangle with the object name inside.

Class: A class is a blueprint or template for creating objects that defines its attributes and methods. It is represented as a rectangle with three compartments for the class name, attributes, and methods.

Link: A link is a relationship between two objects that represents a connection or association. It is represented as a line connecting two objects with optional labels.

Attribute: An attribute is a property or characteristic of an object that describes its state. It is represented as a name-value pair inside the object rectangle.

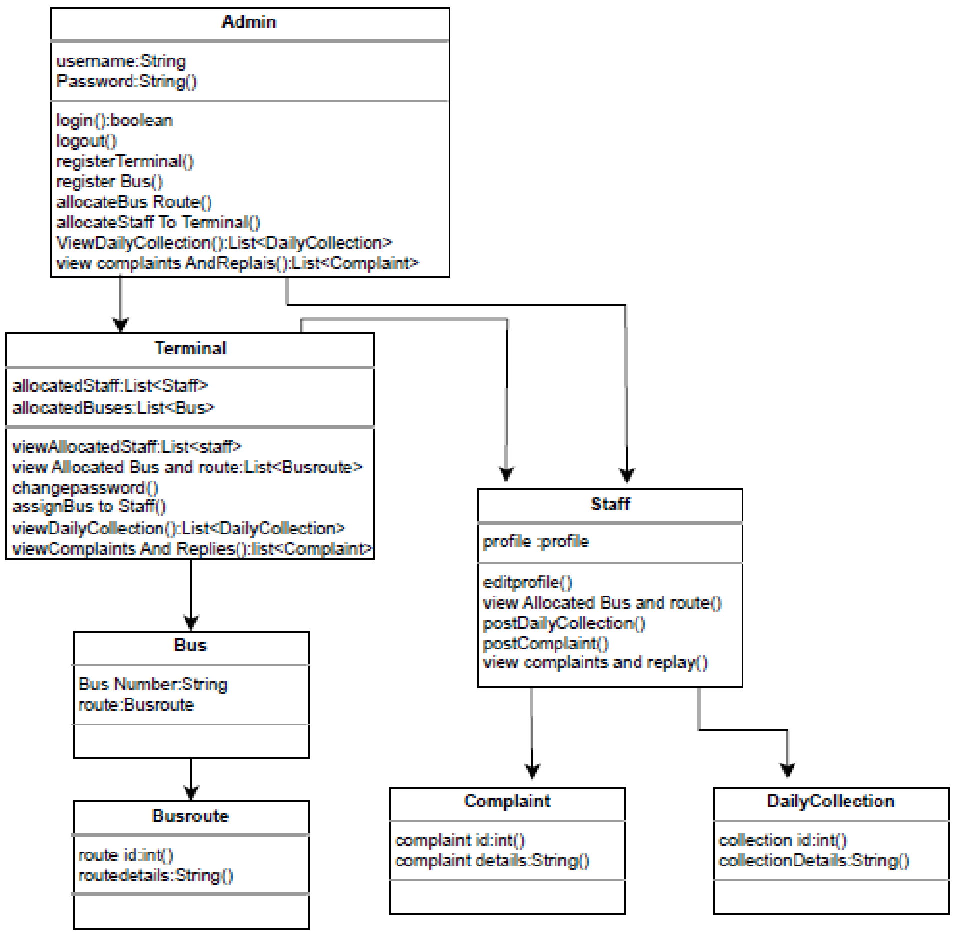
Value: A value is a specific instance or setting of an attribute. It is represented as a value inside the attribute name-value pair.

Operation: An operation is a behavior or action that an object can perform. It is represented as a method name inside the class rectangle.

Multiplicity: Multiplicity represents the number of instances of a class that can be associated with another class.

Object diagrams help to visualize the relationships between objects and their attributes in a system. They are useful for understanding the behavior of a system at a specific point in time and for identifying potential issues or inefficiencies in the system.

Object Diagram


Admin
username:String
Password:String()
login():boolean
logout()
registerTerminal()
register Bus()
allocateBus Route()
allocateStaff To Terminal()
ViewDailyCollection():List<DailyCollection>
view complaints AndReplais():List<Complaint>
Terminal
allocatedStaff:List<Staff>
allocatedBuses:List<Bus>
viewAllocatedStaff:List<staff>
view Allocated Bus and route:List<Busroute>
changepassword()
assignBus to Staff()
viewDailyCollection():List<DailyCollection>
viewComplaints And Replies():list<Complaint>
Staff
profile :profile
editprofile()
view Allocated Bus and route()
postDailyCollection()
postComplaint()
view complaints and replay()
Bus
Bus Number:String
route:Busroute
Busroute
route id:int()
routedetails:String()
Complaint
complaint id:int()
complaint details:String()
DailyCollection
collection id:int()
collectionDetails:String()

4.2.5 Component Diagram

A component diagram in UML illustrates how various components are interconnected to create larger components or software systems. It is an effective tool for representing the structure of complex systems with multiple components. By using component diagrams, developers can easily visualize the internal structure of a software system and understand how different components work together to accomplish a specific task.

Its key components include:

Component: A modular and encapsulated unit of functionality in a system that offers interfaces to interact with other components. It is represented as a rectangle with the component name inside.

Interface: A contract between a component and its environment or other components, specifying a set of methods that can be used by other components. It is represented as a circle with the interface name inside.

Port: A point of interaction between a component and its environment or other components. It is represented as a small square on the boundary of a component.

Connector: A link between two components that enables communication or data exchange. It is represented as a line with optional adornments and labels.

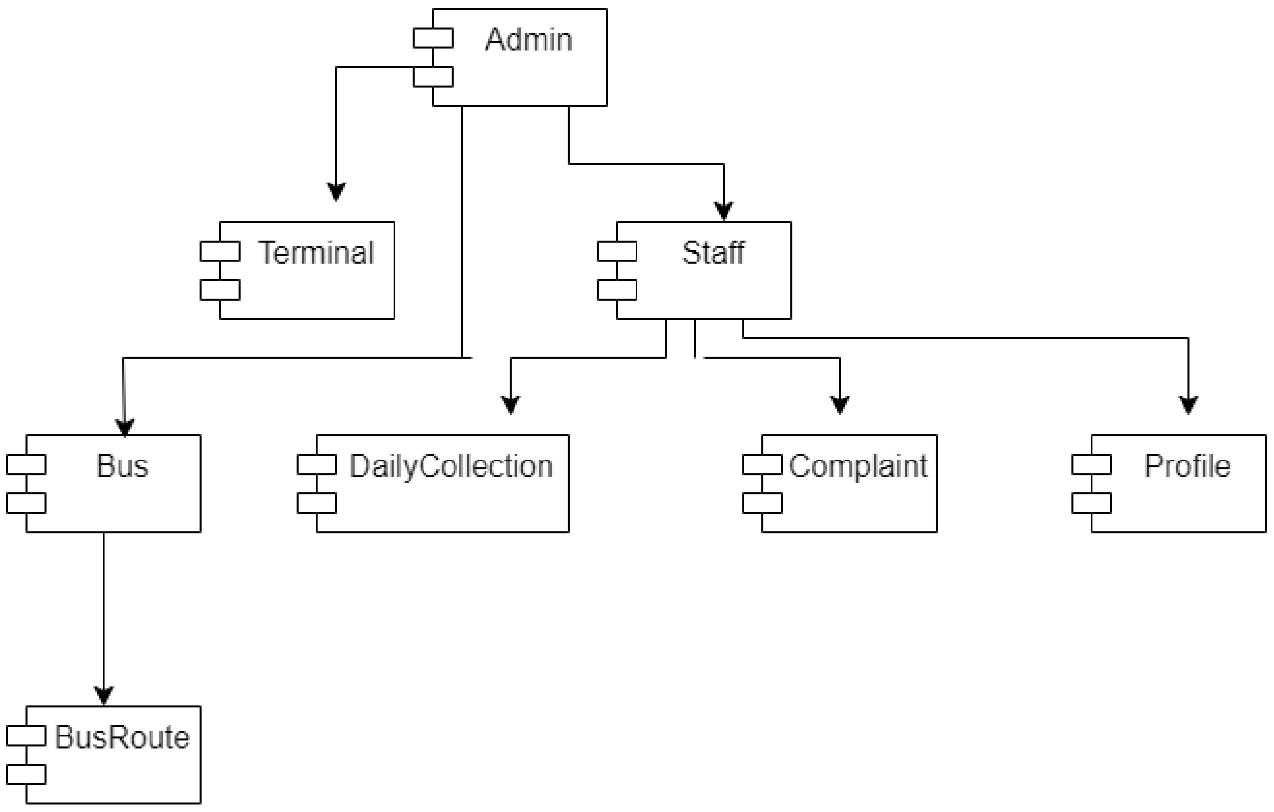
Dependency: A relationship between two components where one component depends on another for its implementation or functionality. It is represented as a dashed line with an arrowhead pointing from the dependent component to the independent component.

Association: A relationship between two components that represents a connection or link. It is represented as a line connecting two components with optional directionality, multiplicity, and role names.

Provided/Required Interface: A provided interface is an interface that a component offers to other components, while a required interface is an interface that a component needs from other components to function properly. These are represented by lollipops and half-circles respectively.

Component diagrams are useful for modeling the architecture of a software system, and can help identify potential issues and improvements in the design. They can also be used to communicate the structure and behavior of a system to stakeholders, such as developers and project managers.

Component Diagram



4.2.8 Deployment Diagram

A deployment diagram is a type of UML diagram that focuses on the physical hardware used to deploy software. It provides a static view of a system's deployment and involves nodes and their relationships. The deployment diagram maps the software architecture to the physical system architecture, showing how the software will be executed on nodes. Communication paths are used to illustrate the relationships between the nodes. Unlike other UML diagram types, which focus on the logical components of a system, the deployment diagram emphasizes the hardware topology.

The key components of a deployment diagram are:

Node - A node is a physical or virtual machine on which a component or artifact is deployed. It is represented by a box with the node's name inside.

Component - A component is a software element that performs a specific function or provides a specific service. It is represented by a rectangle with the component's name inside.

Artifact - An artifact is a physical piece of data that is used or produced by a component. It is represented by a rectangle with the artifact's name inside.

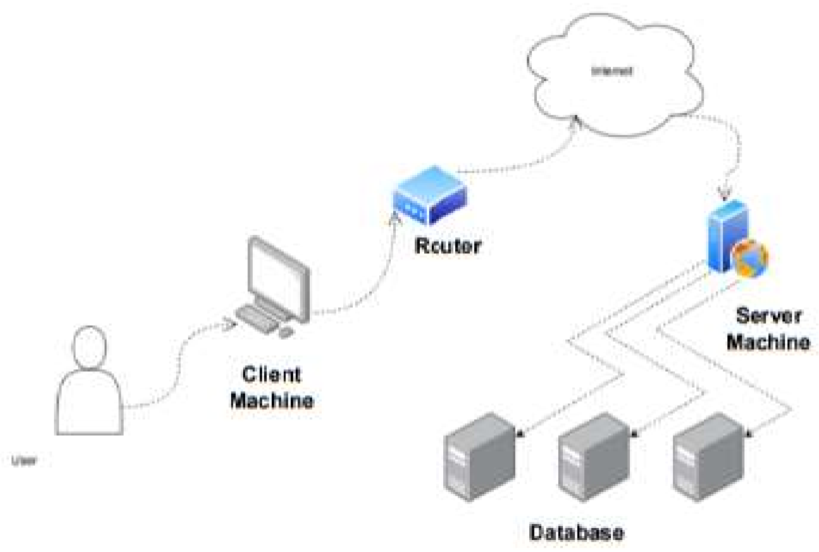
Deployment Specification - A deployment specification describes how a component or artifact is deployed on a node. It includes information about the location, version, and configuration parameters of the component or artifact.

Association - An association is a relationship between a node and a component or artifact that represents a deployment dependency. It is represented by a line connecting the two components with optional directionality, multiplicity, and role names.

Communication Path - A communication path represents the connection between nodes, such as a network connection or communication channel. It is represented by a line with optional labels and adornments.

Deployment diagrams help in visualizing the physical architecture of a system and identifying any potential issues or bottlenecks in the deployment process. They also aid in planning the deployment strategy and optimizing the use of hardware resources.

Deployment Diagram



4.2.9 Collaboration Diagram

A collaboration diagram is a diagram that is used to represent the relationships between objects in a system. It is like a sequence diagram in that it represents the same information, but it does so in a different way. Instead of showing the flow of messages between objects, it depicts the structure of the objects in the system. This is because collaboration diagrams are based on object-oriented programming, where objects have various attributes and are connected to each other. Thus, collaboration diagrams are a visual representation of the object architecture in a system.
A component diagram includes the following components:
Objects: Objects are represented by symbols with their name and class underlined, separated by a colon. In a collaboration diagram, objects are used to represent a class instance and specify its name and class. It is not necessary for every class to have an object representation, and a single class may have multiple objects. Objects are created first, and their class is specified afterwards. Naming objects is important to differentiate them from one another.

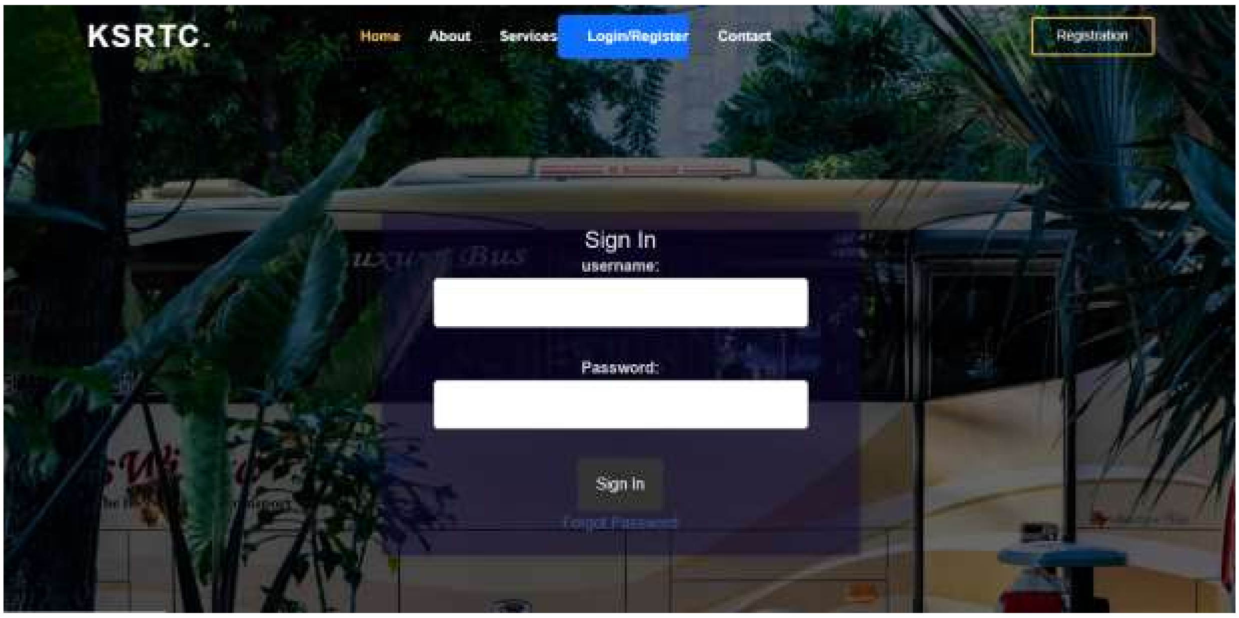
Actors: Actors play a key role in the collaboration diagram as they invoke the interaction. Each actor has its own name and role. In the diagram, one actor initiates the use case.

• Links: Links are instances of association that connect objects and actors. They represent the relationship between objects through which messages are sent. Links are represented by solid lines and help objects to navigate to other objects.

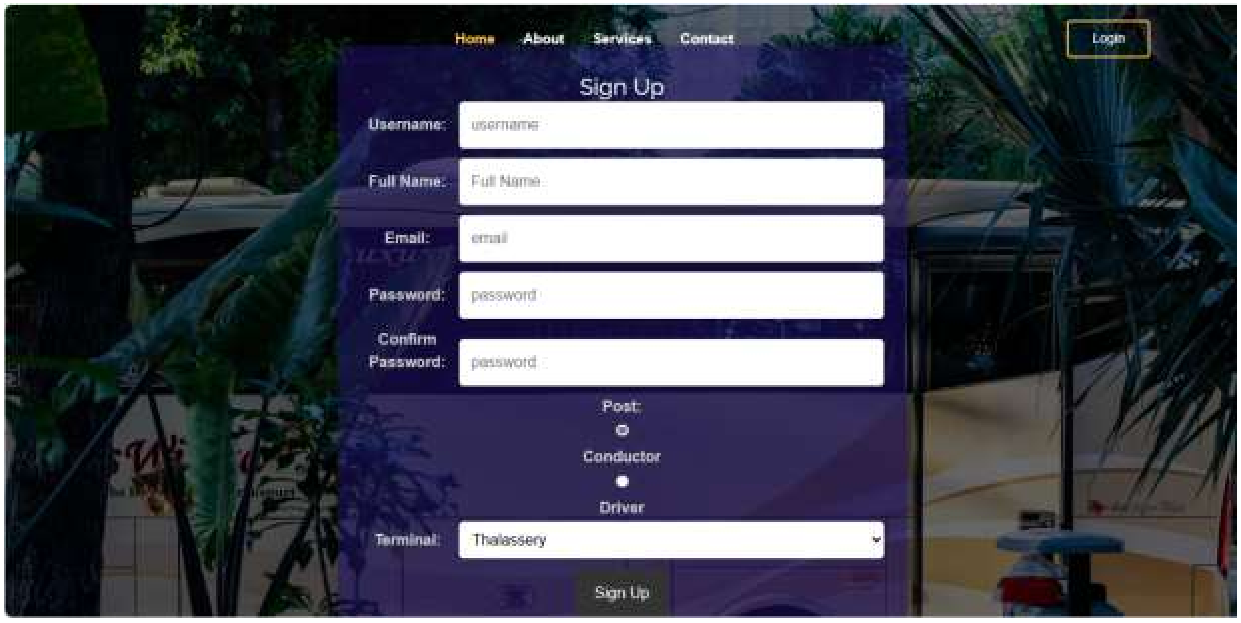
• Messages: Messages represent communication between objects that carry information and are identified by a sequence number. They are represented by labeled arrows placed near the link and sent from the sender to the receiver. The direction must be navigable in that specific direction, and the receiver must understand the message.

4.3 USER INTERFACE DESIGN USING FIGMA

Form Name: Login page



Form Name: Registration Form



4.4 DATABASE DESIGN

A database is an organized collection of information that's organized to enable easy accessibility, administration, and overhauls. The security of information could be a essential objective of any database. The database design process comprises of two stages. In the first stage, user requirements are gathered to create a database that meets those requirements as clearly as possible. This is known as information-level design and is carried out independently of any DBMS. In the second stage, the design is converted from an information-level design to a specific DBMS design that will be used to construct the system. This stage is known as physical-level design, where the characteristics of the specific DBMS are considered. Alongside system design, there is also database design, which aims to achieve two main goals: data integrity and data independence

4.4.1 Relational Database Management System (RDBMS)

A relational database management system (RDBMS) is a popular type of database that organizes data into tables to facilitate relationships with other stored data sets. Tables can contain vast amounts of data, ranging from hundreds to millions of rows, each of which are referred to as records. In formal relational model language, a row is called a tuple, a column heading is an attribute, and the table is a relation. A relational database consists of multiple tables, each with its own name. Each row in a table represents a set of related values.

In a relational database, relationships are already established between tables to ensure the integrity of both referential and entity relationships. A domain D is a group of atomic values, and a common way to define a domain is by choosing a data type from which the domain's data values are derived. It is helpful to give the domain a name to make it easier to understand the values it contains. Each value in a relation is atomic and cannot be further divided.

In a relational database, table relationships are established using keys, with primary key and foreign key being the two most important ones. Entity integrity and referential integrity relationships can be established with these keys. Entity integrity ensures that no primary key can have null values, while referential integrity ensures that each distinct foreign key value must have a matching primary key value in the same domain. Additionally, there are other types of keys such as super keys and candidate keys.

4.4.2 Normalization

The simplest possible grouping of data is used to put them together so that future changes can be made with little influence on the data structures. The formal process of normalizing data structures in a way that reduces duplication and fosters integrity. Using the normalization technique, superfluous fields are removed and a huge table is divided into several smaller ones. Anomalies in insertion, deletion, and updating are also prevented by using it. Keys and relationships are two notions used in the standard form of data modelling. A row in a table is uniquely identified by a key. Primary keys and foreign keys are two different kinds of keys. Primary key is an element, or set of components, in a table that serves as a means of distinguishing between records from the same table. A column in a table known as a foreign key is used to uniquely identify records from other tables. Up to the third normal form, all tables have been normalized.

Normalization is a process in database design that aims to organize data into proper tables and columns, making it easily correlated to the data by the user. This process eliminates data redundancy that can be a burden on computer resources. The main steps involved in normalization include:

- Normalizing the data
- Choosing appropriate names for tables and columns
- Choosing the correct names for the data

By following these steps, a developer can create a more efficient and organized database that is easier to manage and maintain.

First Normal Form-

The First Normal Form (1NF) requires that each attribute in a table must contain only atomic or indivisible values. It prohibits the use of nested relations or relations within relations as attribute values within tuples. To satisfy 1NF, data must be moved into separate tables where the data is of similar type in each table, and each table should have a primary key or foreign key as required by the project. This process eliminates repeating groups of data and creates new relations for each non-atomic attribute or nested relation. A relation is in 1NF only if it satisfies the constraints that contain the primary key only

Second Normal Form-

Second normal form (2NF) is a rule in database normalization that states that non-key attributes should not be functionally dependent on only the part of the primary key in a relation that has a composite primary key. In other words, each non-key attribute should depend on the entire primary key, not just a part of it. To achieve this, we need to decompose the table and create new relationships for each subkey along with their dependent attributes. It is important to maintain the relationship with the original primary key and all attributes that are fully functionally dependent on it. A relation is said to be in 2NF only if it satisfies all the 1NF conditions for the primary key and every non-primary key attribute of the relation is fully dependent only on the primary key

Third Normal Form-

Third normal form (3NF) requires that a relation have no non-key attribute that is functionally determined by another non-key attribute or set of non-key attributes. This means that there should be no transitive dependency on the primary key. To achieve 3NF, we decompose the relation and set up a new relation that includes non-key attributes that functionally determine other non-key attributes. This helps eliminate any dependencies that do not just rely on the primary key. A relation is considered a relation in 3NF if it satisfies the conditions of 2NF and, moreover, the non-key attributes of the relation are not dependent on any other non-key attribute

4.4.3 Sanitization

Data sanitization is the process of removing any illegal characters or values from data. In web applications, sanitizing user input is a common task to prevent security vulnerabilities. PHP provides a built-in filter extension that can be used to sanitize and validate various types of external input such as email addresses, URLs, IP addresses, and more. These filters are designed to make data sanitization easier and faster. For example, the PHP filter extension has a function that can remove all characters except letters, digits, and certain special characters (!#$%&'*+- =?_`{|}~@.[]), as specified by a flag. Web applications often receive external input from various sources, including user input from forms, cookies, web services data, server variables, and database query results. It is important to sanitize all external input to ensure that it is safe and does not contain any malicious code or values.

4.4.4 Indexing

An index is a database structure that enhances the speed of table operations. Indexes can be created on one or more columns to facilitate quick lookups and efficient ordering of records. When creating an index, it is important to consider which columns will be used in SQL queries and to create one or more indexes on those columns. In practice, indexes are a type of table that store a primary key or index field and a pointer to each record in the actual table. Indexes are invisible to users and are only used by the database search engine to quickly locate records. The CREATE INDEX statement is used to create indexes in tables. When tables have indexes, the INSERT and UPDATE statements take longer because the database needs to insert or update the index values as well. However, the SELECT statements become faster on those tables because the index allows the database to locate records more quickly

4.5 TABLE DESIGN

1.tbl_bus

Primary key: **busid**

Foreign key:**terminal id reference table tbl_terminal**

No	Field Name	Datatype	Key Constraints	Description
1	Bus_id	Int(10)	Primary key	Bus id
3	status	Varchar(50)	Not null	Status of bus
4	Registration no	Int(10)	Not null	Registration number
5	Terminal ID	Int(10)	Not null	terminal

2.tbl_staff

Primary key: **staffid**

No	Field Name	Datatype	KeyConstraints	Description
1	Staff_id	Int(10)	Primary key	Id of staff
2	username	Varchar(50)	Not null	Username of staff
3	fullname	Varchar(50)	Not null	Staff full name
4	email	Int(10)	Not null	Email of staff
5	Post(conductor/Driver)	Varchar(50)	Not null	Conductor/driver
7	Terminal	Int(10)	Not null	Terminal name

3.tbl_terminal

Primary key: **terminalid**

No	Field Name	Datatype	Key Constraints	Description
1	Terminal_id	Int(10)	Primary key	Id of terminal
2	Terminal Name	Varchar(50)	Not null	Name of terminal
3	username	Varchar(50)	Not null	Terminal username
4	location	Varchar(50)	Not null	Location of terminal

4.tbl_complaint

Primary key: **complaint_id**

Foreign key**:staff_alocation id references table tbl_staff_allocation, assignment_id references table tbl_assignment**

No	Field Name	Datatype	Key Constraints	Description
1	Complaint_id	Int(10)	Primary key	Id of complaint
2	Description	Varchar(50)	Not null	Complaint description
3	Date	Adate	Not null	Date of complaint
4	staff_allocation_id	Int(10)	Foreign key	Foreign key in staff_allocation
5	Assignment_id	Int(10)	Foreign key	Foreign key in bus assignment

5.tbl_collection id

Primary key: **collection_id**

Foreign key**: staff_alocation id references table tbl_staff_allocation, assignment_id references table tbl_assignment**

No	Field Name	Datatype	Key Constraints	Description
1	Collection_id	Int(10)	Primary key	Id in collection
2	Amount	Int(10)	Not null	Collection amount
3	Date	Adate	Not null	Date of collection
4	staff_allocation_id	Int(10)	Foreign key	Foreign key in staff allocation

5	assignment_id	Int(10)	Foreign key	Foreign key in bus assignment

6.tbl_route

Primary key: **route_id**

Foreign key**:bus_id references table tbl_bus**

No	Field Name	Datatype	Key Constraints	Description
1	Route_id	Int(10)	Primary key	Id of route
2	Source	charfield	Not null	Source place
3	Source time	Time	Not null	Source time
4	Destination	charfield	Not null	Destination place
5	Destination time	Time	Not null	Destination time
6	bus_id	Int(10)	Foreign key	Foreign key of bus

7.tbl_assignment

Primary key: **assignment_id**

Foreign key**:bus_allocation_id refences table tbl_bus_allocation,staff_allocation id references table tbl_staff_allocation**

No	Field Name	Datatype	Key Constraints	Description
1	assignmentid	Int(10)	Primary key	Id of bus assignment
2	bus_allocation id	Int(10)	Foreign Key	Foreign key in bus_allocation
3	staff_allocation id	Int(10)	Foreign Key	Foreign key of staff_allocation

8.tbl_staff_allocation

Primary key: **staff_allocation id**

Foreign key**:staf _id references table tbl_staff ,terminal_id references table tbl_terminal**

No	Field Name	Datatype	Key Constraints	Description
1	Staff_allocation_id	Int(10)	Primary key	Id of allocation
2	Staff ID	Int(10)	Foreign Key	Foreign key of staff
3	Terminal ID	Int(10)	Foreign Key	Foreign key of terminal
4	Allocation_date	date	Not null	Date of allocation

9.tbl_bus_allocation

Primary key: **bus_allocation id**

Foreign key**:busid reference table tbl_bud ,terminal_id reference table tbl_terminal**

No	**Field Name**	**Datatype**	**Key Constraints**	**Description**
1	bus_allocation_id	Int(10)	Primary key	Id of allocation
2	Bus_id	Int(10)	Foreign Key	Foreign key of bus
3	terminalid	Int(10)	Foreign Key	Foreign key of terminal
4	Allocation_date	date	Not null	Allocatiuon date

CHAPTER 5
SYSTEM TESTING

5.1 INTRODUCTION

Software testing involves executing a software program in a controlled manner to determine if it behaves as intended, often using verification and validation methods. Validation involves evaluating a product to ensure it complies with specifications, while verification can involve reviews, analyses, inspections, and walkthroughs. Static analysis examines the software's source code to identify issues, while dynamic analysis examines its behavior during runtime to gather information like execution traces, timing profiles, and test coverage details.

Testing involves a series of planned and systematic activities that start with individual modules and progress to the integration of the entire computer-based system. The objectives of testing include identifying errors and bugs in the software, ensuring that the software functions according to its specifications, and verifying that it meets performance requirements. Testing can be performed to assess correctness, implementation efficiency, and computational complexity.

A successful test is one that detects an undiscovered error, and a good test case has a high probability of uncovering such errors. Testing is crucial to achieving system testing objectives and can involve various techniques such as functional testing, performance testing, and security testing.

5.2 TEST PLAN

A test plan is a document that outlines the required steps to complete various testing methodologies. It provides guidance on the activities that need to be performed during testing. Software developers create computer programs, documentation, and associated data structures. They are responsible for testing each component of the program to ensure it meets the intended purpose. To address issues with self-evaluation, an independent test group (ITG) of it established

Testing objectives should be stated in quantifiable language, such as mean time to failure, co-
st to find and fix defects, remaining defect density or frequency of occurrence, and test work-
hours per regression test.

The different levels of testing include:

- Unit testing
- Integration testing
- Data validation testing
- Output testing

5.2.1 Unit Testing

Unit testing is a software testing technique that focuses on verifying individual components or modules of the software design. The purpose of unit testing is to test the smallest unit of software design and ensure that it performs as intended. Unit testing is typically white-box focused, and multiple components can be tested simultaneously.

During unit testing, the modular interface is tested to ensure that data enters and exits the software unit under test properly. The local data structure is inspected to ensure that data temporarily stored retains its integrity during each step of an algorithm's execution. Boundary conditions are tested to ensure that all statements in a module have been executed at least once, and all error handling paths are tested to ensure that the software can handle errors correctly.

Before any other testing can take place, it is essential to test data flow over a module interface. If data cannot enter and exit the system properly, all other tests are irrelevant. Another crucial duty during unit testing is the selective examination of execution pathways to anticipate potential errors and ensure that error handling paths are set up to reroute or halt work when an error occurs. Finally, boundary testing is conducted to ensure that the software operates correctly at its limits.

In the ShareSphere System, unit testing was carried out by treating each module as a distinct entity and subjecting them to a variety of test inputs. Any issues with the internal logic of the modules were fixed, and each module was tested and run separately after coding. Unused code was eliminated, and it was confirmed that every module was functional and produced the desired outcome.

5.2.2 Integration Testing

Integration testing is a systematic approach that involves creating the program structure while simultaneously conducting tests to identify interface issues. The objective is to construct a program structure based on the design that uses unit-tested components. The entire program is then tested. Correcting errors in integration testing can be challenging due to the size of the overall program, which makes it difficult to isolate the causes of the errors. As soon as one set of errors is fixed, new ones may arise, and the process may continue in an apparently endless cycle.

Once unit testing is complete for all modules in the system, they are integrated to check for any interface inconsistencies. Any discrepancies in program structures are resolved, and a unique program structure is developed.

5.2.3 Validation Testing or System Testing

The final stage of the testing process involves testing the entire software system as a whole, including all forms, code, modules, and class modules. This is commonly referred to as system testing or black box testing. The focus of black box testing is on testing the functional requirements of the software. A software engineer can use this approach to create input conditions that will fully test each program requirement The main types of errors targeted by black box testing include incorrect or missing functions, interface errors, errors in data structure or external data access, performance errors, initialization errors, and termination errors.

5.2.4 Output Testing or User Acceptance Testing

User acceptance testing is performed to ensure that the system meets the business requirements and user needs. It is important to involve the end users during the development process to ensure that the software aligns with their needs and expectations. During user acceptance testing, the input and output screen designs are tested with different types of test data. The preparation of test data is critical to ensure comprehensive testing of the system. Any errors identified during testing are addressed and corrected, and the corrections are not for future reference.

5.2.5 Automation Testing

Automation testing is a software testing approach that employs specialized automated testing software tools to execute a suite of test cases. Its primary purpose is to verify that the software or equipment operates precisely as intended. Automation testing identifies defects, bugs, and other issues that may arise during product development. While some types of testing, such as functional or regression testing, can be performed manually, there are numerous benefits to automating the process. Automation testing can be executed at any time of day and uses scripted sequences to evaluate the software. The results are reported, and this information can be compared to previous test runs. Automation developers typically write code in programming languages such as C#, JavaScript, and Ruby..

5.2.6 Selenium Testing

Selenium is an open-source automated testing framework used to verify web applications across different browsers and platforms. Selenium allows for the creation of test scripts in various programming languages such as Java, C#, and Python. Jason Huggins developed Selenium in 2004 while working on a web application that required frequent testing. He created a JavaScript program called "JavaScriptTestRunner" to automate browser actions and improve testing efficiency. Selenium has since evolved and continues to be developed by a team of contributors. In addition to Selenium, another popular tool used for automated testing is Cucumber. Cucumber is an open-source software testing framework that supports behavior-driven development (BDD). It allows for the creation of executable specifications in a human-readable format called Gherkin. One of the advantages of using Cucumber is its ability to bridge the gap between business stakeholders and technical teams. By using a common language, Cucumber facilitates effective communication and collaboration during the testing process. It promotes a shared understanding of the requirements and helps ensure that the developed software meets the intended business goals.

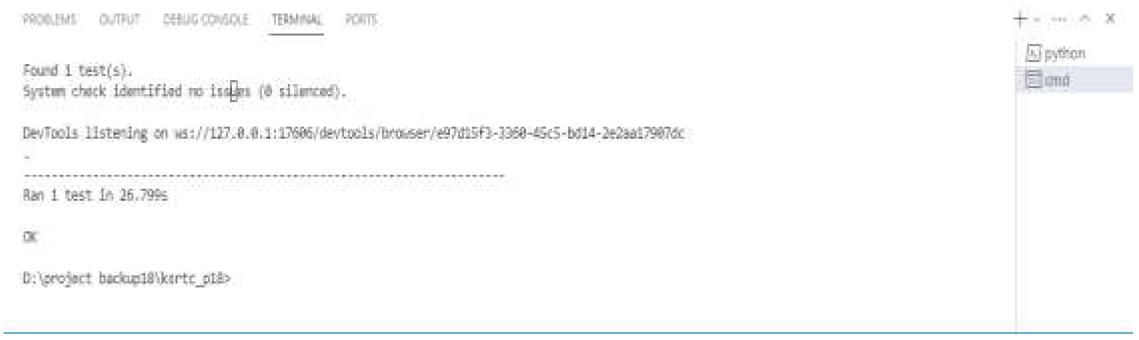
Cucumber can be integrated with Selenium to combine the benefits of both tools. Selenium is used for interacting with web browsers and automating browser actions , while Cucumber provides a structured framework for organizing and executing tests. This combination allows for the creation of end-to-end tests that verify the behavior of web application.

Test Case 1

Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

class LoginTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.get("http://127.0.0.1:8000/login.html")
        time.sleep(10)  # Wait for 10 seconds before accessing the URL

    def test_login_successful(self):
        username_input = self.driver.find_element(By.ID, "username")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.XPATH, "//button[text()='Sign
In']")
        username_input.send_keys("pavi")
        password_input.send_keys("Pavi@123")
        login_button.click()
        time.sleep(5)

        expected_url_after_login = 'http://127.0.0.1:8000/dashboard'
        self.assertEqual(self.driver.current_url, expected_url_after_login, "Login
failed or unexpected page after login")

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main()
```

Screenshot



```
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:17606/devtools/browser/e97d15f3-3360-45c5-bd14-2e2aa17907dc
.
----------------------------------------------------------------------
Ran 1 test in 26.799s

OK

D:\project backup18\ksrtc_p18>
```

Test Report

Test Case 1	
Project Name: Readify	
Login Test Case	
Test Case ID: Test_1	**Test Designed By**: Athira Ramesh
Test Priority (Low/Medium/High): High	**Test Designed Date:** 24-10-2023
Module Name: Login Screen	**Test Executed By :** Ms.Nimmy Francis
Test Title : User Login	**Test Execution Date:** 05-12-2023
Description: Verify login with valid username and password	
Pre-Condition : User has valid username and password	

Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide Valid username	Username: pavi	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: Pavi@123			

4	Click on Login button				
Post-Condition: User is validated with database and successfully login into account. The Account session details are logged in database					

Test Case 2:

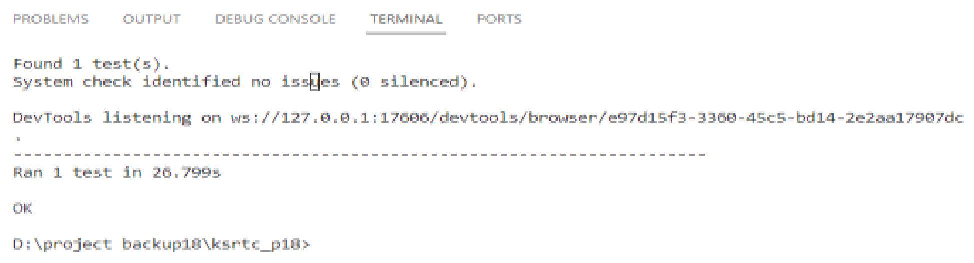
Code

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import TimeoutException
import time

class DailyCollectionTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.get("http://127.0.0.1:8000/login.html")
        time.sleep(10)  # Wait for 10 seconds before accessing the URL

    def test_daily_collection_workflow(self):
        # Login
        self.login()

        # Navigate to the daily collection page
        self.driver.get("http://127.0.0.1:8000/daily_collection_sent/2/")
        time.sleep(2)

        # Wait for the page to load completely
        WebDriverWait(self.driver, 10).until(
            EC.presence_of_element_located((By.XPATH, "//a[contains(@href,
'/daily_collection_sent/2/') and contains(text(), 'Add Daily Collection')]"))
        )

        # Click the "Add Daily Collection" button
        add_collection_button = self.driver.find_element(By.XPATH,
"//a[contains(@href, '/daily_collection_sent/2/') and contains(text(), 'Add Daily
Collection')]")
        add_collection_button.click()

        # Verify if redirected to the correct URL after clicking the "Add Daily
Collection" button
        expected_url_after_adding_collection =
'http://127.0.0.1:8000/daily_collection_sent/2/'
```

```
        self.assertEqual(self.driver.current_url,
expected_url_after_adding_collection,
                         "Redirect to Daily Collection Sent page failed")

        # Add daily collection
        self.add_daily_collection()
        # Wait for the success message or any element on the new page indicating
the submission
        success_message_locator = (By.XPATH, "//div[@class='success-message']")
        WebDriverWait(self.driver, 10).until(
            EC.presence_of_element_located(success_message_locator)
        )

        # Verify if the success message is displayed
        success_message = self.driver.find_element(*success_message_locator)
        self.assertEqual(success_message.text, "Daily collection added
successfully.",
                         "Failed to display success message after adding Daily
Collection")

    def login(self):
        username_input = self.driver.find_element(By.ID, "username")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.XPATH, "//button[text()='Sign
In']")
        username_input.send_keys("pavi")
        password_input.send_keys("Pavi@123")
        login_button.click()
        time.sleep(5)

        expected_url_after_login = 'http://127.0.0.1:8000/dashboard'
        self.assertEqual(self.driver.current_url, expected_url_after_login,
                         "Login failed or unexpected page after login")

    def add_daily_collection(self):
        # Fill in daily collection details
        collection_date_input = self.driver.find_element(By.NAME,
'collection_date')
        collection_date_input.send_keys("01-12-2023")  # Use the correct format,
e.g., "YYYY-MM-DD"

        amount_collected_input = self.driver.find_element(By.NAME,
'amount_collected')
        amount_collected_input.send_keys("8643.99")  # Replace with a valid amount

        staff_allocation_select = self.driver.find_element(By.NAME,
'staff_allocation')
```

```
        staff_allocation_select.send_keys("pavi p")  # Replace with a valid staff
allocation

        bus_assignment_select = self.driver.find_element(By.NAME,
'bus_assignment')
        bus_assignment_select.send_keys("KL-2367")  # Replace with a valid bus
assignment

        # Submit the form
        submit_button = self.driver.find_element(By.XPATH,
"//button[@type='submit']")
        submit_button.click()

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main()
```

Screenshot



```
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:17606/devtools/browser/e97d15f3-3360-45c5-bd14-2e2aa17907dc
.
----------------------------------------------------------------------
Ran 1 test in 26.799s

OK

D:\project backup18\ksrtc_p18>
```

Test report

Test Case 2

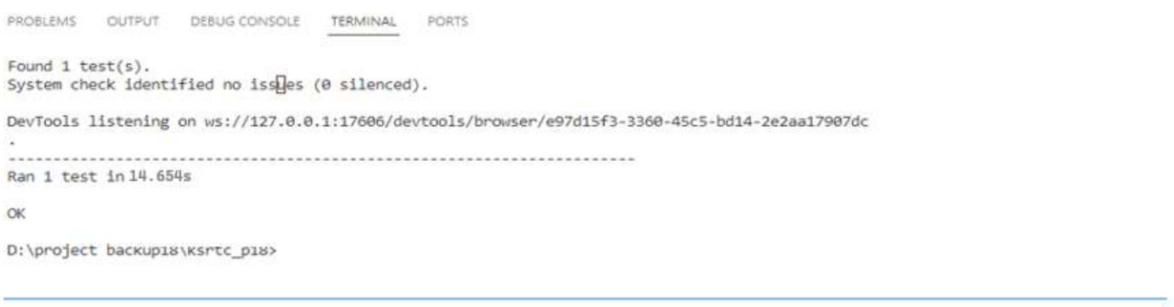
Project Name: KSRTC Management System					
DailyCollection Test Case					
Test Case ID: Test_2			**Test Designed By:** Athira Ramesh		
Test Priority (Low/Medium/High): High			**Test Designed Date:** 05-12-2023		
Module Name: daily collection			**Test Executed By :** Ms. Nimmy Francis		
Test Title : post the daily collection			**Test Execution Date:** 05-12-2023		
Description: staff post the daily collection					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**

1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	username: pavi	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: Pavi@123			
4	Click on Login button				
5	Click on Daily Collection		Daily collection form is displyed	Daily collection form is showed	Pass
6	Fill the form and click on submit	Amount:459 Date:2023-12-07	Post the daily collection	Daily collection post	Pass
Post-Condition: post the daily collection successfully					

Test Case 3:

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.common.exceptions import NoSuchElementException
import time

class PostComplaint(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.get("http://127.0.0.1:8000/login.html")
        self.driver.maximize_window()
        time.sleep(2)

    def test_post_complaint_workflow(self):
        # Login
        self.login()

        # Navigate to the complaint page
        self.driver.get("http://127.0.0.1:8000/complaint/1/")
        time.sleep(2)

        # Wait for the complaint form to be visible
        try:
            WebDriverWait(self.driver, 10).until(
                EC.visibility_of_element_located((By.NAME, "staff_allocation"))
            )
        except NoSuchElementException:
            self.fail("Complaint form not found on the page")

        # Click the "Add Complaint" button
        try:
            add_complaint_button = self.driver.find_element(By.XPATH,
"//a[contains(@href, '/complaint/1/') and contains(text(), 'Add Complaint')]")
            add_complaint_button.click()
        except NoSuchElementException:
            self.fail("Add Complaint button not found on the page")

        # Verify if redirected to the correct URL after clicking the "Add
Complaint" button
        expected_url_after_post_complaint = 'http://127.0.0.1:8000/dashboard'
        self.assertEqual(self.driver.current_url,
expected_url_after_post_complaint,
                         "Redirect to post complaint page failed")

        # Add complaint
        self.add_complaint()

        # Wait for the success message or any element on the new page indicating
the submission
        success_message_locator = (By.XPATH, "//div[@class='success-message']")
        WebDriverWait(self.driver, 10).until(
            EC.presence_of_element_located(success_message_locator)
        )

        # Verify if the success message is displayed
        success_message = self.driver.find_element(*success_message_locator)
        self.assertEqual(success_message.text, "Complaint added successfully.",
                         "Failed to display success message after posting
complaint")

    def login(self):
        username_input = self.driver.find_element(By.ID, "username")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.XPATH, "//button[text()='Sign
In']")
        username_input.send_keys("pavi")
        password_input.send_keys("Pavi@123")
        login_button.click()
        time.sleep(5)
        # Wait for the login process to complete
        try:
            WebDriverWait(self.driver, 10).until(
                EC.url_to_be("http://127.0.0.1:8000/dashboard")
            )
        except:
            self.fail("Login failed or unexpected page after login")

    def add_complaint(self):
        # Fill in complaint details
        terminal_select = self.driver.find_element(By.NAME, 'panoor')
        terminal_select.send_keys("Your Terminal")

        bus_assignment_select = self.driver.find_element(By.NAME, 'KL-2367-MBC-
32')
        bus_assignment_select.send_keys("Your Bus Assignment")

        collection_date_input = self.driver.find_element(By.NAME, '2023-12-04')
        collection_date_input.send_keys("2023-12-01")

        collection_description_input = self.driver.find_element(By.NAME, '')
        collection_description_input.send_keys("no tier available")

        # Submit the form
        submit_button = self.driver.find_element(By.XPATH,
"//button[@type='submit']")
        submit_button.click()

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main()
```

Screenshot



```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

Found 1 test(s).
System check identified no iss▯es (0 silenced).

DevTools listening on ws://127.0.0.1:17606/devtools/browser/e97d15f3-3360-45c5-bd14-2e2aa17907dc
.
----------------------------------------------------------------------
Ran 1 test in 14.654s

OK

D:\project backup18\ksrtc_p18>
```

Test report

Test Case 3

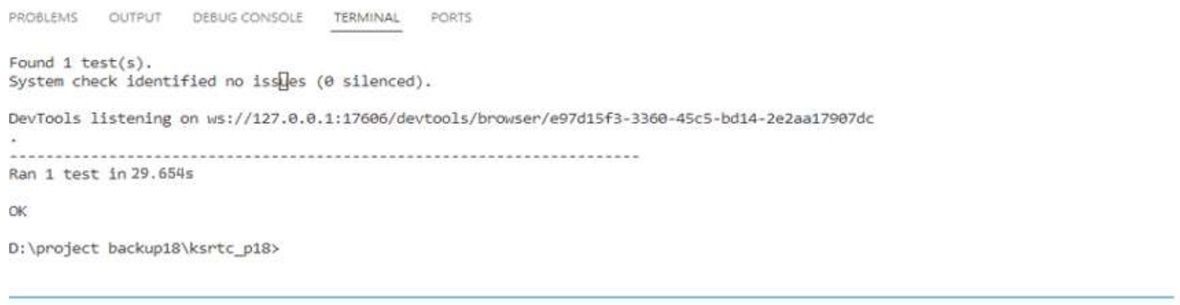
Project Name: KSRTC Management System					
Complaint post Test Case					
Test Case ID: Test_3			**Test Designed By:** Athira Ramesh		
Test Priority (Low/Medium/High): High			**Test Designed Date:** 05-12-2023		
Module Name: Post Complaint			**Test Executed By :** Ms. Nimmy Francis		
Test Title : post the Complaint			**Test Execution Date:** 07-12-2023		
Description: staff post Complaint					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	username: pavi	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: Pavi@123			

4	Click on Login button				
5	Click on post complaint		Complaint form is displayed	Post complaint form should be showed	Pass
6	Fill the form and click on submit	Complaint description:no tyre available Date:2023-12-04	Post the complaint	Post the complaint	Pass
Post-Condition: post the complaint successfully					

Test Case 4:

```
import unittest
from selenium import webdriver
from selenium.webdriver.common.by import By
import time

class TerminalAddTest(unittest.TestCase):
    def setUp(self):
        self.driver = webdriver.Chrome()
        self.driver.get("http://127.0.0.1:8000/login.html")
        time.sleep(10)

        # Login
        username_input = self.driver.find_element(By.ID, "username")
        password_input = self.driver.find_element(By.ID, "password")
        login_button = self.driver.find_element(By.XPATH, "//button[text()='Sign
In']")
        username_input.send_keys("office")
        password_input.send_keys("office@123")
        login_button.click()
        time.sleep(5)

        # Navigate to admin dashboard
        self.driver.get("http://127.0.0.1:8000/admindashboard.html")
        time.sleep(5)
    def test_terminal_add_successful(self):
        # Click the "terminal_add" button in the admin dashboard
        terminal_add_button = self.driver.find_element(By.XPATH,
"//a[contains(@href, 'terminal_add')]")
        terminal_add_button.click()
        time.sleep(5)

        # Fill the terminal_add form
        terminal_name_field = self.driver.find_element(By.ID, "terminal_name")
        username_field_terminal = self.driver.find_element(By.ID, "username")
        location_field = self.driver.find_element(By.ID, "location")
        password_field_terminal = self.driver.find_element(By.ID, "password")
        confirm_password_field = self.driver.find_element(By.ID,
"confirm_password")

        terminal_name_field.send_keys("panoor")
        username_field_terminal.send_keys("panoor")
        location_field.send_keys("kannur")
        password_field_terminal.send_keys("Panoor@123")
        confirm_password_field.send_keys("Panoor@123")

        # Click the "Register Terminal" button
        register_button = self.driver.find_element(By.XPATH,
"//button[text()='Register Terminal']")
        register_button.click()
        time.sleep(5)

        expected_url_after_terminal_add = 'http://127.0.0.1:8000/dashboard'
        self.assertEqual(self.driver.current_url, expected_url_after_terminal_add,
"Terminal add failed or unexpected page after terminal add")

    def tearDown(self):
        self.driver.quit()

if __name__ == "__main__":
    unittest.main()
```

Screenshot



```
Found 1 test(s).
System check identified no issues (0 silenced).

DevTools listening on ws://127.0.0.1:17606/devtools/browser/e97d15f3-3360-45c5-bd14-2e2aa17907dc
.
----------------------------------------------------------------------
Ran 1 test in 29.654s

OK

D:\project backup18\ksrtc_p18>
```

Test report

Test Case 4

Project Name: KSRTC Management System					
Terminal add Test Case					
Test Case ID: Test_4			**Test Designed By:** Athira Ramesh		
Test Priority (Low/Medium/High): High			**Test Designed Date:** 05-12-2023		
Module Name: Add Terminal			**Test Executed By :** Ms. Nimmy Francis		
Test Title : Terminal Add			**Test Execution Date:** 07-12-2023		
Description: staff post Complaint					
Pre-Condition : User has valid username and password					
Step	**Test Step**	**Test Data**	**Expected Result**	**Actual Result**	**Status(Pass/ Fail)**
1	Navigation to Login Page		Login Page should be displayed	Login Page be displayed	Pass
2	Provide valid username	username: office	User should be able to login	User is logged in and navigated to corresponding Home Page	Pass
3	Provide Valid Password	Password: office@123			
4	Click on Login button				
5	Click on Add terminal		Form is displyed	Add terminal form should be showed	Pass
6	Fill the form and click on submit	Terminal name:panoor Location:kannur	Add the terminal details	Terminal add	Pass
Post-Condition: terminal added successfully					

CHAPTER 6
IMPLEMENTATION

6.1INTRODUCTION

The implementation phase of a project is where the design is transformed into a functional system. It is a crucial stage in ensuring the success of the new system, as it requires gaining user confidence that the system will work effectively and accurately. User training and documentation are key concerns during this phase. Conversion may occur concurrently with user training or at a later stage. Implementation involves the conversion of a newly revised system design into an operational system.

During this stage, the user department bears the primary workload, experiences the most significant upheaval, and has the most substantial impact on the existing system. Poorly planned or controlled implementation can cause confusion and chaos. Whether the new system is entirely new, replaces an existing manual or automated system, or modifies an existing system, proper implementation is essential to meet the organization's needs. System implementation involves all activities required to convert from the old to the new system. The system can only be implemented after thorough testing is done and found to be working according to specifications. System personnel evaluate the feasibility of the system. Implementation requires extensive effort in three main areas: education and training, system testing, and changeover. The implementation phase involves careful planning, investigating system and constraints, and designing methods to achieve changeover

6.2 IMPLEMENTATION PROCEDURES

Software implementation is the process of installing the software in its actual environment and ensuring that it satisfies the intended use and operates as expected. In some organizations, the software development project may be commissioned by someone who will not be using the software themselves. During the initial stages, there may be doubts about the software, but it's important to ensure that resistance does not build up.
This can be achieved by:

- Ensuring that active users are aware of the benefits of the new system, building their confidence in the software.

- Providing proper guidance to the users so that they are comfortable using the application. Before viewing the system, users should know that the server program must be running on the server. Without the server object up and running, the intended process will not take place.

6.2.1 User Training

User training is designed to prepare the user for testing and converting the system. To achieve the objective and benefits expected from computer-based system, it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for training is more important. By user training the user comes to know how to enter data, respond to error messages, interrogate the database, and call up routine that will produce reports and perform other necessary functions

6.2.2 Training on the Application Software

After providing the necessary basic training on computer awareness, it is essential to provide training on the new application software to the user. This training should include the underlying philosophy of using the new system, such as the flow of screens, screen design, the type of help available on the screen, the types of errors that may occur while entering data, and the corresponding validation checks for each entry, and ways to correct the data entered. Additionally, the training should cover information specific to the user or group, which is necessary to use the system or part of the system effectively. It is important to note that this training may differ across different user groups and across different user groups and levels of hierarchy.

6.2.3 System Maintenance

The maintenance phase is a crucial aspect of the software development cycle, as it is the time when the software is actually put to use and performs its intended functions. Proper maintenance is essential to ensure that the system remains functional, reliable, and adaptable to changes in the system environment. Maintenance activities go beyond simply identifying and fixing errors or bugs in the system. It may involve updates to the software, modifications to its functionalities, and enhancements to its performance, among other things. In essence, software maintenance is an ongoing process that requires continuous monitoring, evaluation, and improvement of the system to meet changing user needs and requirements.

CHAPTER 7
CONCLUSION AND FUTURE SCOPE

7.1 CONCLUSION

The KSRTC (Kerala State Road Transport Corporation) Management System is a comprehensive software solution designed to streamline and optimize the operations of a state-owned road transport corporation. This system serves various stakeholders, including administrators, terminal staff, and bus operators, facilitating efficient management and improved service delivery.

Here's a description of its key functionalities: The system provides an intuitive user interface for administrators to log in, facilitating secure access to the management platform. Administrators can register new KSRTC terminals, ensuring accurate representation of the corporation's infrastructure within the system. Bus registration functionality allows administrators to add new buses to the system, providing essential details such as registration information, seating capacity, and other relevant data. This ensures a comprehensive and up-to-date database of the entire bus fleet. Route allocation is a critical feature, allowing administrators to assign specific bus routes to optimize transportation efficiency.

This functionality considers factors like distance, demand, and operational logistics to create an effective and balanced schedule. Staff allocation to terminals ensures that the right personnel are assigned to each terminal, optimizing workforce distribution and ensuring smooth operations at every location. The system provides a centralized platform for managing staff information, including profiles and work assignments. For terminal staff, the system offers secure login access and the ability to view allocated staff members and buses.

Terminal staff can also change their passwords for enhanced security. Additionally, they can assign buses to specific staff members, fostering a clear and organized assignment of responsibilities. The daily collection feature enables staff to record and monitor daily revenue generated by each bus and terminal, offering a comprehensive overview of financial performance.

This information is crucial for financial analysis and future planning. The system incorporates a complaint management module, allowing users to submit complaints and receive timely responses. Terminal staff can view complaints and respond, creating a transparent and communicative environment for issue resolution. Overall, the KSRTC Management System serves as a centralized hub for overseeing and coordinating the various aspects of road transport operations.Its features contribute to enhanced efficiency, data accuracy, and customer satisfaction within the Kerala State Road transport cooperation

7.2 FUTURE SCOPE

- The future scope of the KSRTC Management System can be broad and dynamic, incorporating technological advancements and evolving requirements within the road transport sector. Here are some potential areas of future development and enhancement: Real-Time Monitoring and Tracking: Integration of GPS technology for real-time tracking of buses, enabling administrators and passengers to monitor bus locations, arrival times, and route deviations.
- Predictive Analytics: Implementation of predictive analytics to forecast demand, optimize route planning, and improve resource allocation, leading to more efficient and cost-effective operations. Mobile Applications: Development of mobile applications for passengers, providing real-time information on bus schedules, routes, and bookings. This enhances user experience and accessibility. Digital Ticketing and Payments: Introduction of digital ticketing systems and online payment options to streamline the ticketing process, reduce queues, and enhance the overall convenience for passengers. IoT Integration:
- Smart Infrastructure Integration: Integration with smart city infrastructure, allowing the system to adapt to traffic conditions, road closures, and other external factors that might affect bus routes and schedules. Enhanced Reporting and Analytics:
- Expansion of reporting and analytics capabilities to provide detailed insights into operational performance, financial trends, and passenger behavior. This can aid in data-driven decision-making. Machine Learning for Demand Prediction: Utilization of machine learning algorithms to analyze historical data and predict future demand for specific routes, optimizing bus schedules and resource utilization.
- Enhanced Security Features: Implementation of advanced security features, including biometric authentication for staff, surveillance systems, and cybersecurity measures to protect sensitive data. Regulatory Compliance: Ongoing updates to ensure compliance with evolving transportation regulations and standards, keeping the system aligned with legal requirements.
- The future scope of the KSRTC Management System is dynamic, and continuous innovation and adaptation to emerging technologies will be key to ensuring its relevance and effectiveness in meeting the evolving needs of the transportation industry

.

CHAPTER 8
BIBLIOGRAPHY

REFERENCES:

- AlsoAsked - People Also Ask keyword research tool: A tool to find questions people also ask related to a specific topic.
- Google Support - How to search on Google: Tips on effective Google searches, including finding quick answers.
- Qualtrics - Multiple Choice Question: Information about the multiple-choice question type for surveys.
- MyGreatLearning - 180+ SQL Interview Questions and Answers in 2023: A compilation of SQL interview questions for job seekers

WEBSITES:

- https://www.grafiati.com/
- https://annas-archive.org/
- https://www.ebooks.com/

CHAPTER 9
APPENDIX

9.1 Sample Code

Login

```
{% load static %}
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="utf-8">
  <meta content="width=device-width, initial-scale=1.0" name="viewport">

  <title>Gp Bootstrap Template - Index</title>
  <meta content="" name="description">
  <meta content="" name="keywords">

  <!-- Favicons -->
  <link href="{% static '/img/favicon.png' %}" rel="icon">
  <link href="{% static '/img/apple-touch-icon.png' %}" rel="apple-touch-icon">

  <!-- Google Fonts -->
                      <link                    href="{%                    static
'/https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,7
00,700i|Raleway:300,300i,400,400i,500,500i,600,600i,700,700i|Poppins:300,300i,40
0,400i,500,500i,600,600i,700,700i' %}" rel="stylesheet">

  <!-- Vendor CSS Files -->
  <link href="{% static '/vendor/aos/aos.css' %}" rel="stylesheet">
    <link   href="{%   static   '/vendor/bootstrap/css/bootstrap.min.css'   %}"
rel="stylesheet">
    <link  href="{%  static  '/vendor/bootstrap-icons/bootstrap-icons.css'  %}"
rel="stylesheet">
     <link    href="{%    static    '/vendor/boxicons/css/boxicons.min.css'    %}"
rel="stylesheet">
    <link   href="{%   static   '/vendor/glightbox/css/glightbox.min.css'   %}"
rel="stylesheet">
  <link href="{% static '/vendor/remixicon/remixicon.css' %}" rel="stylesheet">
     <link    href="{%    static    '/vendor/swiper/swiper-bundle.min.css'    %}"
rel="stylesheet">

  <!-- Template Main CSS File -->
  <link href="{% static '/css/style.css' %}" rel="stylesheet">

  <!-- =======================================================
  * Template Name: Gp
  * Updated: Sep 18 2023 with Bootstrap v5.3.2
  * Template URL: https://bootstrapmade.com/gp-free-multipurpose-html-bootstrap-
template/
```

```
  * Author: BootstrapMade.com
  * License: https://bootstrapmade.com/license/
  ======================================================== -->
  <style>
    body
     {
        margin: 0;
        padding: 0;
        font-family: Arial, Helvetica, sans-serif;
         background-image: url('signupimg.jpg'); /* Replace with your own image
path */
        background-size: cover;
        background-position: center;
        background-repeat: no-repeat;
    }

    .signup-container {
        max-width: 500px;
        max-height: 900px;
        margin: 0 auto;
        padding: 5px;
        background-color: rgba(14, 4, 51, 0.5);
        box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.2);
        border-radius: 2px;
        text-align: center;
    }

    .signup-container h2 {
        margin-bottom: 20px;
        color: #333;
    }

    .form-group {
        margin-bottom: 20px;
    }

    .form-group label {
        display: block;
        font-weight: bold;
        color:#ccc

    }

    .form-group input {
        width: 80%;
        padding: 12px;
        margin-bottom: 10px;
        border: 1px solid #ccc;
        border-radius: 5px;
    }

    .error-message {
        color: #e74c3c;
        margin-top: 5px;
        font-size: 12px;
    }

    .success-message {
        color: #27ae60;
        margin-top: 5px;
        font-size: 12px;
    }

    .submit-btn {
        background-color: #333;
        color: #fff;
        border: none;
        border-radius: 5px;
        padding: 15px 20px;
        cursor: pointer;
    }

    @media (max-width: 768px) {
        .signup-container {
            max-width: 90%;
        }

        .form-group input {
            padding: 10px;
        }
    }
</style>
</head>

<body>

  <!-- ======= Header ======= -->
  <header id="header" class="fixed-top ">
    <div class="container d-flex align-items-center justify-content-lg-between">

          <h1  class="logo  me-auto  me-lg-0"><a  href="{%  static  '/index.html'
%}">KSRTC<span>.</span></a></h1>
      <!-- Uncomment below if you prefer to use an image logo -->
           <!--   <a   href="index.html"   class="logo   me-auto   me-lg-0"><img
src="assets/img/logo.png" alt="" class="img-fluid"></a>-->
```

```
      <nav id="navbar" class="navbar order-last order-lg-0">
        <ul>
                              <li><a     class="nav-link     scrollto     active"
href="index.html">Home</a></li>
                                      <li><a      class="nav-link      scrollto"
href="index.html#about">About</a></li>
                                      <li><a      class="nav-link      scrollto"
href="index.html#services">Services</a></li>
          {% if user.is_authenticated %}
          <a href="logout" class="btn btn-primary ml-lg-3">Logout</a>
          {% else %}
                      <a    href="login.html"    class="btn    btn-primary    ml-lg-
3">Login/Register</a>
          {% endif %}
                              <!--     <li><a     class="nav-link     scrollto     "
href="#portfolio">Portfolio</a></li>-->
          <!--<li><a class="nav-link scrollto" href="#team">Team</a></li>
           <li class="dropdown"><a href="#"><span>Drop Down</span> <i class="bi
bi-chevron-down"></i></a>
            <ul>
              <li><a href="#">Drop Down 1</a></li>
                <li class="dropdown"><a href="#"><span>Deep Drop Down</span> <i
class="bi bi-chevron-right"></i></a>
                <ul>
                  <li><a href="#">Deep Drop Down 1</a></li>
                  <li><a href="#">Deep Drop Down 2</a></li>
                  <li><a href="#">Deep Drop Down 3</a></li>
                  <li><a href="#">Deep Drop Down 4</a></li>
                  <li><a href="#">Deep Drop Down 5</a></li>
                </ul>
              </li>
              <li><a href="#">Drop Down 2</a></li>
              <li><a href="#">Drop Down 3</a></li>
              <li><a href="#">Drop Down 4</a></li>
            </ul>
          </li>-->
                                      <li><a      class="nav-link      scrollto"
href="index.html#contact">Contact</a></li>
        </ul>
        <i class="bi bi-list mobile-nav-toggle"></i>
      </nav><!-- .navbar -->

       <li class="dropdown"></li><a href="registration.html" class="get-started-
btn scrollto">Registration</a>

    </div>

  </header><!-- End Header -->
  <!-- ======= Hero Section ======= -->
  <section id="hero" class="d-flex align-items-center justify-content-center">
    <div class="container" data-aos="fade-up">
      <body>
        <br><br><br><br>
        <div class="signup-container">

            <h2>Sign In</h2>
            <span class="mb-0 text-muted">
            </span>
            <form action="" method="POST">
              {% csrf_token %}
                <div class="form-group">
                    <label for="username">username:</label>
                           <input  type="username"  id="username"  name="username"
required>
                    <span id="username-error" class="error-message"></span>
                </div>
                <div class="form-group">
                    <label for="password">Password:</label>
                           <input  type="password"  id="password"  name="password"
required>
                    <span id="password-error" class="error-message"></span>
                </div>
                <button type="submit" class="submit-btn">Sign In</button>
                <span id="success-message" class="success-message"></span>
            </form>
           <!--<p class="signup">Not have an account? Register here.<a href="{%
url 'registration' %}">Register</a></p>-->
             <p><a  href="{%  url  'password_reset'  %}"  class="link"  style="color:
rgb(61, 88, 147);">Forgot Password</a></p>
        </div>

  </div>
  </div>




  </section><!-- End Hero -->
  <!-- ======= Footer ======= -->
<footer id="footer">
  <div class="footer-top">
    <div class="container">
      <div class="row">

        <div class="col-lg-3 col-md-6">
```

```
          <div class="footer-info">
            <h3>KSRTC<span>.</span></h3>
            <p>
              Thiruvanathapuram <br>
              <br><br>
              <strong>Phone:</strong> +91 9876789876<br>
              <strong>Email:</strong> ksrtc@gmail.com<br>
            </p>
            <div class="social-links mt-3">
              <a href="#" class="twitter"><i class="bx bxl-twitter"></i></a>
              <a href="#" class="facebook"><i class="bx bxl-facebook"></i></a>
              <a href="#" class="instagram"><i class="bx bxl-instagram"></i></a>
              <a href="#" class="google-plus"><i class="bx bxl-skype"></i></a>
              <a href="#" class="linkedin"><i class="bx bxl-linkedin"></i></a>
            </div>
          </div>
        </div>

        <div class="col-lg-2 col-md-6 footer-links">
          <h4>Useful Links</h4>
          <ul>
            <li><i class="bx bx-chevron-right"></i> <a href="#">Home</a></li>
                   <li><i  class="bx  bx-chevron-right"></i>  <a  href="#">About
us</a></li>
                              <li><i    class="bx    bx-chevron-right"></i>    <a
href="#">Services</a></li>
                 <li><i  class="bx  bx-chevron-right"></i>  <a  href="#">Terms  of
service</a></li>
                  <li><i  class="bx  bx-chevron-right"></i>  <a  href="#">Privacy
policy</a></li>
          </ul>
        </div>

        <div class="col-lg-3 col-md-6 footer-links">
          <h4>Our Services</h4>
          <ul>
                    <li><i  class="bx  bx-chevron-right"></i>  <a  href="#">Town
services</a></li>
             <li><i class="bx bx-chevron-right"></i> <a href="#">Fast Passenger
Services</a></li>
              <li><i class="bx bx-chevron-right"></i> <a href="#">Super Express
Services</a></li>
               <li><i class="bx bx-chevron-right"></i> <a href="#">Limited Stop
Services</a></li>
                    <li><i  class="bx  bx-chevron-right"></i>  <a  href="#">City
Services</a></li>
          </ul>
        </div>
        <div class="col-lg-4 col-md-6 footer-newsletter">
          <!--<h4>Our Newsletter</h4>
            <p>Tamen quem nulla quae legam multos aute sint culpa legam noster
magna</p>
          <form action="" method="post">
                        <input   type="email"   name="email"><input   type="submit"
value="Subscribe">-->
          </form>

        </div>

      </div>
    </div>
  </div>

  <div class="container">
    <div class="copyright">
      &copy; Copyright <strong><span>ksrtc</span></strong>. All Rights Reserved
    </div>
    <div class="credits">
      <!-- All the links in the footer should remain intact. -->
      <!-- You can delete the links only if you purchased the pro version. -->
      <!-- Licensing information: https://bootstrapmade.com/license/ -->
        <!--  Purchase  the  pro  version  with  working  PHP/AJAX  contact  form:
https://bootstrapmade.com/gp-free-multipurpose-html-bootstrap-template/ -->
           Designed   by   <a   href="{%   static   '/https://bootstrapmade.com/'
%}">BootstrapMade</a>
    </div>
  </div>
</footer><!-- End Footer -->




        <script   src="{%   static   '/vendor/purecounter/purecounter_vanilla.js'
%}"></script>
    <script src="{% static '/vendor/aos/aos.js' %}"></script>
       <script   src="{%   static   '/vendor/bootstrap/js/bootstrap.bundle.min.js'
%}"></script>
    <script src="{% static '/vendor/glightbox/js/glightbox.min.js' %}"></script>
        <script   src="{%   static   '/vendor/isotope-layout/isotope.pkgd.min.js'
%}"></script>
    <script src="{% static '/vendor/swiper/swiper-bundle.min.js' %}"></script>
    <script src="{% static '/vendor/php-email-form/validate.js' %}"></script>

    <!-- Template Main JS File -->
    <script src="{% static '/js/main.js' %}"></script>
```

```
     <!-- Add this JavaScript code inside the <head> section or just before
</body> -->
      <!--<script>
        // Clear email and password fields on page load
        window.onload = function() {
            document.getElementById("email").value = "";
            document.getElementById("password").value = "";
        }
    </script>-->



  </body>

  </html>
```

Daily collection

```
<!DOCTYPE html>
<html>
<head>
  <title>Admin Dashboard</title>
  <link rel="stylesheet" type="text/css" href="css/style.css">
    <link  rel="stylesheet"  href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
  <style>
                                                                        @import
url('https://fonts.googleapis.com/css2?family=Montserrat:wght@300&display=swap')
;
:root{
  --blue:#0046fd;
  --white:#fff;
  --black:#000;
  --lightblue:#b3bbd1;
  --gray:#e5e5e5;
}
*{margin: 0px;  padding: 0px; font-family: 'Montserrat', sans-serif;}
body{height: 100vh;}
.sidebar{
  width: 20%;
  float: left;
  background: var(--blue);
  height: 100vh;
  position: relative;
}
.main{width: 75%;float: left;height: 90vh; padding: 30px;}
.logo{text-align:  center;color:  var(--white);  margin-top:  50px;  font-weight:
800;
margin-bottom: 50px;}
.nav{margin-left: 40px;}
.nav li{list-style: none; padding: 16px 19px;}
.nav li i{ margin-right: 10px;}
.nav li a{color: var(--white); text-decoration: none; font-weight: 600;}
.active{
  background:var(--white);
  border-radius: 27px;
  position: relative;
}
.nav > .active::before, .nav > .active::after {
     content: "";
     position: absolute;
}
.nav > li{
    border-radius: 60px 0 0 60px;
        width: 92%;
    margin-left: -15px;
}
.nav > .active::before {
    top: -80px;
    right: 0px;
    height: 80px;
    width: 62px;
    border-bottom-right-radius: 80px;
    box-shadow: 0 40px 0 0 var(--white);
    transform: rotateZ(360deg);
}
.nav > .active::after {
    top: 51px;
    right: 0px;
    height: 80px;
    width: 62px;
    border-top-right-radius: 80px;
    box-shadow: 0 -40px 0 0 var(--white);
    transform: rotateZ(360deg);

}
.active a{color: var(--blue)!important;}
.social{position: absolute;bottom: 10px; left: 20px;}
.social li{
  list-style: none;
  float: left;
  padding: 10px;
}
.social li a{font-size: 14px; color: var(--white);}
.clearfix{clear: both;}
.col-6{width: 50%; float: left;}
```

```
.hicon{margin-right: 20px; font-size: 22px!important;position: relative;}
.hicon:nth-child(1):after{
  content: '';
  position:absolute;
  width: 7px;
  height: 7px;
  background: red;
  border-radius: 10px;
  left: 13px;
    top: 7px;
    transform: scale(0.5);
  box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
  animation: pulse-red 2s infinite;
  cursor: pointer;

}

@keyframes pulse-red {
  0% {
    transform: scale(0.85);
    box-shadow: 0 0 0 0 rgba(255, 82, 82, 0.7);
  }

  70% {
    transform: scale(1);
    box-shadow: 0 0 0 25px rgba(255, 82, 82, 0);
  }

  100% {
    transform: scale(0.85);
    box-shadow: 0 0 0 0 rgba(255, 82, 82, 0);
  }
}
.user{
  background:var(--lightblue);
    padding: 5px;
    width: 30px;
    margin-bottom: -11px;
    border-radius: 5px;
    cursor: pointer;
}
.profile-div{
  background-color: var(--gray);
  width: 140px;
  border: 1px solid #ddd;
  padding: 10px 0px;
  right: 2%;
  border-radius: 5px;
  position: absolute;
  z-index: 9;
  display: none;
  top: 84px;
  text-align: left;
}
.profile-div:after{
  content: '';
    position: absolute;
    height: 10px;
    width: 10px;
    background-color: var(--gray);
    top: -6px;
    border-left: 1px solid #ddd;
    border-top: 1px solid #ddd;
    transform: rotate(45deg);
    right: 10%;
}
.profile-div p{margin: 0px;padding: 10px 20px;}
.profile-div p:hover{
  margin: 0px;
    padding: 10px 20px;
    background-color: var(--blue);
    border-radius: 0px;
    color: var(--white);
}
.notification-div{
  background-color: var(--gray);
  width: 246px;
  border: 1px solid #ddd;
  padding: 10px 0px;
  right: 9.4%;
  border-radius: 5px;
  position: absolute;
  z-index: 9;
  display: none;
  top: 84px;
  text-align: left;
}
.notification-div:after{
  content: '';
    position: absolute;
    height: 10px;
    width: 10px;
    background-color: var(--gray);
    top: -6px;
    border-left: 1px solid #ddd;
    border-top: 1px solid #ddd;
    transform: rotate(45deg);
    right: 10%;
}
.notification-div p{margin: 0px;padding: 10px 20px;}
.notification-div p:hover{
  margin: 0px;
    padding: 10px 20px;
    background-color: var(--blue);
    border-radius: 0px;
    color: var(--white);
}
.search{ padding: 5px; display: none; outline: none; border:none;
  border-bottom: 1px solid black;
}
table {
  border-collapse: collapse;
  margin: 0;
  padding: 0;
  width: 100%;
  table-layout: fixed;
  border-spacing: 0 15px;
}

table tr {
  background-color: #f8f8f8;
  border: 1px solid #ddd!important;
  padding: .35em;
  margin-bottom: 10px;
}

table th,
table td {
  padding: 1.5em;
  text-align: left;

}

table th {
  font-size: .85em;
  letter-spacing: .1em;
  text-transform: uppercase;
}
.tab-img{
  display: inline-block;
    width: 30px;
    vertical-align: middle;
    border: 1px solid #000;
    border-radius: 18px;
    margin-right: 8px;
}
.ticon{
  padding: 5px;
  background: var(--gray);
  border-radius: 4px;
  width: 15px;
  height: 15px;
  text-align: center;
}
.pe{
  content: '';
  position:absolute;
  width: 7px;
  height: 7px;
  background: red;
  border-radius: 10px;
  left: 10px;
    top: 37px;
    transform: scale(0.5);
  box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
  animation: pulse-red 2s infinite;
  cursor: pointer;
}
.de{
  content: '';
  position:absolute;
  width: 7px;
  height: 7px;
  background: green;
  border-radius: 10px;
  left: 10px;
    top: 37px;
    transform: scale(0.5);
  box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
  animation: pulse-green 2s infinite;
  cursor: pointer;
}
@keyframes pulse-green {
  0% {
    transform: scale(0.85);
    box-shadow: 0 0 0 0 rgba(82, 255, 82, 0.7);
  }

  70% {
    transform: scale(1);
    box-shadow: 0 0 0 25px rgba(82, 255, 82, 0);
  }
  100% {
    transform: scale(0.85);
    box-shadow: 0 0 0 0 rgba(82, 255, 82, 0);
  }
}
.active-tr{
  background: var(--blue);
  color: var(--white);
}
.active-tr .ticon{
  background: #e5e5e54f;
}
.container {
            background-color: #fff;
            padding: 20px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
            width: 400px;
        }

        form {
            display: flex;
            flex-direction: column;
        }

        label {
            margin-bottom: 8px;
        }

        select, input {
            margin-bottom: 16px;
            padding: 8px;
            border: 1px solid #ccc;
            border-radius: 4px;
        }

        button {
            background-color: #007BFF;
            color: #fff;
            padding: 10px;
            border: none;
            border-radius: 4px;
            cursor: pointer;
        }

        button:hover {
            background-color: #0056b3;
        }
  </style>
</head>
<body>

<section>
  <div class="sidebar">
    <h1 class="logo">KSRTC</h1>

    <ul class="nav">
        <li><a  href="{%  url  'myprofile'  %}"><i  class="fa  fa-windows"></i>  my
profile</a></li>
          <li   class="active"><a   href="#"><i   class="fa   fa-shopping-bag"></i>
dashboard</a></li>
      <li><a href="{% url 'daily_collection_sent' user_id=request.user.id %}"><i
class="fa fa-pie-chart"></i> daily collection</a></li>
      <li><a href="{% url 'daily_collection_view' user_id=request.user.id %}"><i
class="fa fa-cube"></i>daily collection view </a></li>
       <li><a  href="{%  url  'complaint'  user_id=request.user.id  %}"><i  class="fa
fa-cube"></i>complaint </a></li>
            <li><a href="{% url 'user_complaints' user_id=request.user.id %}"><i
class="fa fa-cube"></i>view complaint </a></li>

      <li><a>   {% if user.is_authenticated %}
    <a href="{% url 'logout' %}" class="btn btn-primary ml-lg-3">Logout</a>
{% else %}
    <a href="{% url 'login' %}" class="btn btn-primary ml-lg-3">Logout</a>
{% endif %}</a></li>

    </ul>

      <ul class="social">
        <li><a href="#">Facebook</a></li>
        <li><a href="#">Twitter</a></li>
        <li><a href="#">Google</a></li>
      </ul>


  </div>
  <div class="main">

    <div class="head-section">
      <div class="col-6">
        <h2>Order</h2>
        <p>30 orders found</p>
      </div>

      <div class="col-6" style="text-align: right;">
```

```
        <i class="fa fa-bell-o hicon"></i>
        <input type="text" class="search">
        <i class="fa fa-search hicon sicon"></i>

        <img src="image/user.png" class="user">

        <div class="profile-div">
          <p><i class="fa fa-user"></i>   Profile</p>
          <p><i class="fa fa-cog"></i>   Settings</p>
          <p><i class="fa fa-power-off"></i>   Log Out</p>
        </div>

    <div class="notification-div">
      <p>Success!Your registration is now complete!</p>
      <p>Here's some information you may find useful!</p>
    </div>

      </div>

      <div class="clearfix"></div>
    </div>

    <br><br>
    <div class="content">
      <p>All orders</p><br><br>
      <h2>Welcome, {{ user.full_name }}!</h2>


<div class="container">
    <h2>Daily Collection Report</h2>

          <form    method="post"    action="{%    url    'daily_collection_sent'
user_id=request.user.id %}">
        {% csrf_token %}

        <label for="terminal">Select Terminal:</label>
        <select name="staff_allocation" id="staff_allocation">
            {% for staff_allocation in allocated_terminals %}
                  <option value="{{ staff_allocation.terminal.terminal_id }}">{{
staff_allocation.terminal.terminal_name }}</option>
            {% endfor %}
        </select>

        <label for="bus_assignment">Select Bus Assignment:</label>
        <select name="bus_assignment" id="bus_assignment">
            {% for assignment in assigned_buses %}
                                       <option   value="{{   assignment.id   }}">{{
assignment.bus_allocation.bus.registration_number         }}         -         {{
assignment.bus_allocation.bus.bus_number }}</option>
```

```
            {% endfor %}
        </select>

        <label for="collection_date">Collection Date:</label>
        <input type="date" name="collection_date" required>

        <label for="amount_collected">Amount Collected:</label>
        <input type="number" name="amount_collected" step="0.01" required>

        <button type="submit">Submit</button>
    </form>


    </div>



  </div>
</section>



<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script>
$(document).ready(function(){
  $(".user").click(function(){
    $(".profile-div").toggle(1000);
  });
  $(".hicon:nth-child(1)").click(function(){
    $(".notification-div").toggle(1000);
  });
  $(".sicon").click(function(){
    $(".search").toggle(1000);
  });
});
</script>

<script type="text/javascript">
  $('li').click(function(){
    $('li').removeClass("active");
    $(this).addClass("active");
});

</script>
</body>
</html>
```

Post Complaint

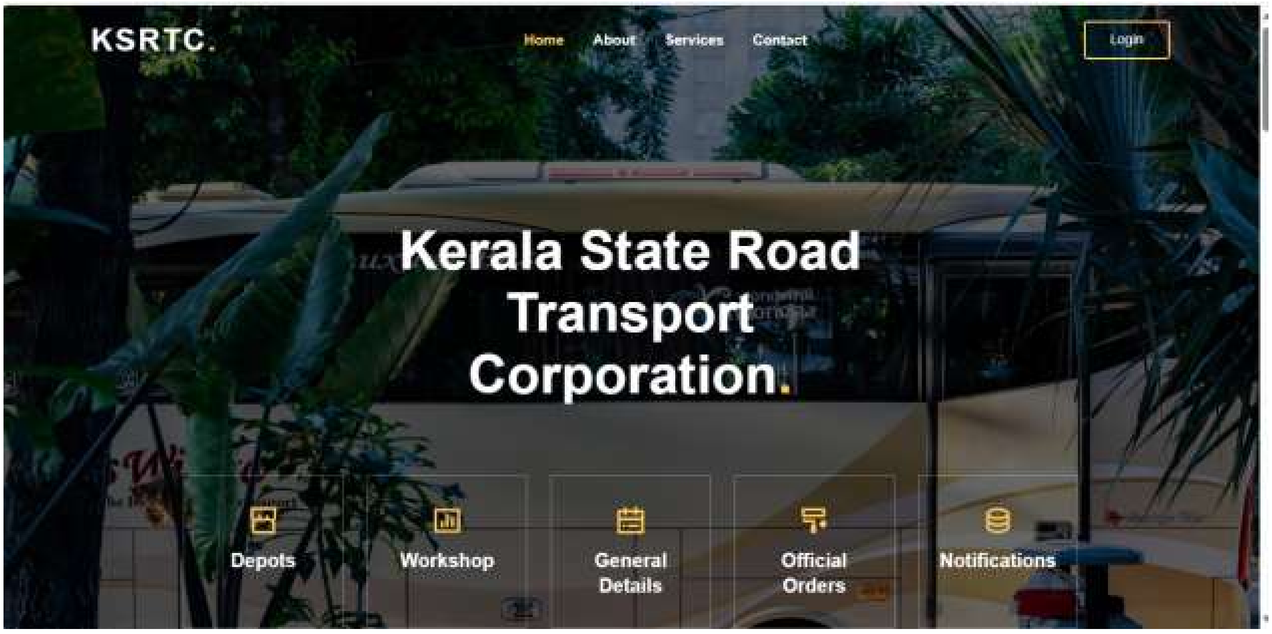
```
<!DOCTYPE html>
<html>
<head>
    <title>Admin Dashboard</title>
    <link rel="stylesheet" type="text/css" href="css/style.css">
      <link  rel="stylesheet"  href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <style>
                                                                          @import
url('https://fonts.googleapis.com/css2?family=Montserrat:wght@300&display=swap')
;
:root{
    --blue:#0046fd;
    --white:#fff;
    --black:#000;
    --lightblue:#b3bbd1;
    --gray:#e5e5e5;
}
*{margin: 0px;  padding: 0px; font-family: 'Montserrat', sans-serif;}
body{height: 100vh;}
.sidebar{
    width: 20%;
    float: left;
    background: var(--blue);
    height: 100vh;
    position: relative;
}
.main{width: 75%;float: left;height: 90vh; padding: 30px;}
.logo{text-align:  center;color:  var(--white);  margin-top:  50px;  font-weight:
800;
margin-bottom: 50px;}
.nav{margin-left: 40px;}
.nav li{list-style: none; padding: 16px 19px;}
.nav li i{ margin-right: 10px;}
.nav li a{color: var(--white); text-decoration: none; font-weight: 600;}
.active{
    background:var(--white);
    border-radius: 27px;
    position: relative;
}
.nav > .active::before, .nav > .active::after {
     content: "";
     position: absolute;
}
.nav > li{
    border-radius: 60px 0 0 60px;
        width: 92%;
    margin-left: -15px;
}
.nav > .active::before {
    top: -80px;
    right: 0px;
    height: 80px;
    width: 62px;
    border-bottom-right-radius: 80px;
    box-shadow: 0 40px 0 0 var(--white);
    transform: rotateZ(360deg);
}
.nav > .active::after {
    top: 51px;
    right: 0px;
    height: 80px;
    width: 62px;
    border-top-right-radius: 80px;
    box-shadow: 0 -40px 0 0 var(--white);
    transform: rotateZ(360deg);

}
.active a{color: var(--blue)!important;}
.social{position: absolute;bottom: 10px; left: 20px;}
.social li{
    list-style: none;
    float: left;
    padding: 10px;
}
.social li a{font-size: 14px; color: var(--white);}
.clearfix{clear: both;}
.col-6{width: 50%; float: left;}
.hicon{margin-right: 20px; font-size: 22px!important;position: relative;}
.hicon:nth-child(1):after{
    content: '';
    position:absolute;
    width: 7px;
    height: 7px;
    background: red;
    border-radius: 10px;
    left: 13px;
    top: 7px;
    transform: scale(0.5);
    box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
    animation: pulse-red 2s infinite;
    cursor: pointer;

}
```

```
@keyframes pulse-red {
    0% {
        transform: scale(0.85);
        box-shadow: 0 0 0 0 rgba(255, 82, 82, 0.7);
    }

    70% {
        transform: scale(1);
        box-shadow: 0 0 0 25px rgba(255, 82, 82, 0);
    }

    100% {
        transform: scale(0.85);
        box-shadow: 0 0 0 0 rgba(255, 82, 82, 0);
    }
}
.user{
    background:var(--lightblue);
    padding: 5px;
    width: 30px;
    margin-bottom: -11px;
    border-radius: 5px;
    cursor: pointer;
}
.profile-div{
  background-color: var(--gray);
  width: 140px;
  border: 1px solid #ddd;
  padding: 10px 0px;
  right: 2%;
  border-radius: 5px;
  position: absolute;
  z-index: 9;
  display: none;
  top: 84px;
  text-align: left;
}
.profile-div:after{
  content: '';
    position: absolute;
    height: 10px;
    width: 10px;
    background-color: var(--gray);
    top: -6px;
    border-left: 1px solid #ddd;
    border-top: 1px solid #ddd;
    transform: rotate(45deg);
    right: 10%;
```

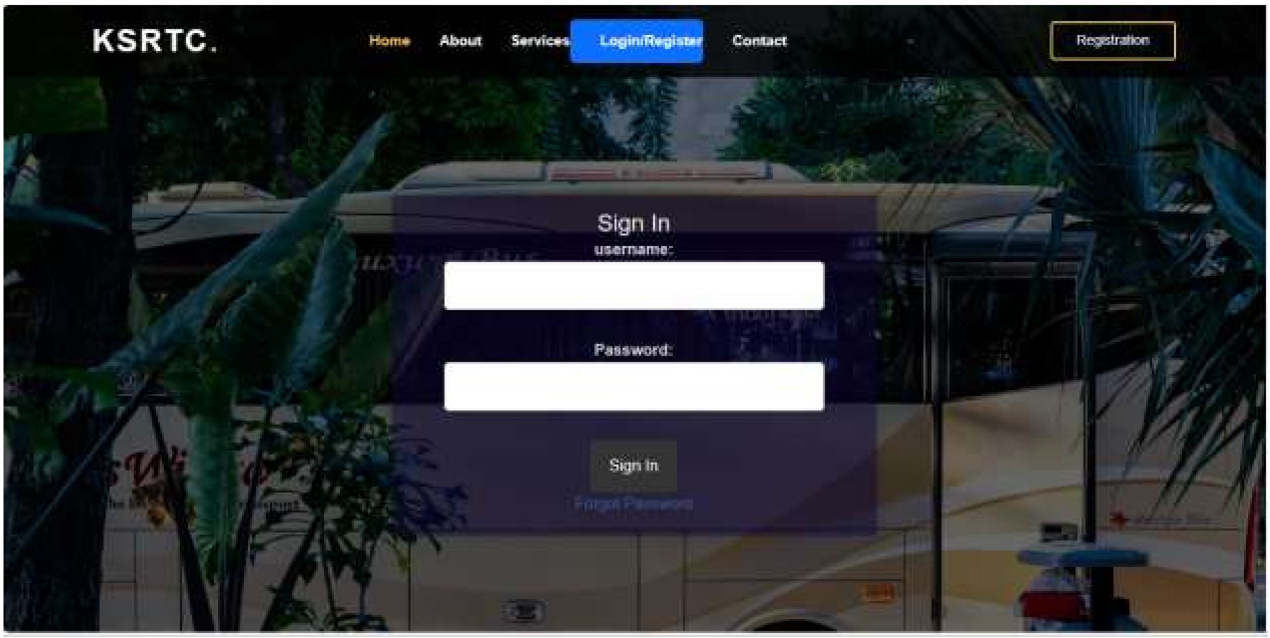
```
}
.profile-div p{margin: 0px;padding: 10px 20px;}
.profile-div p:hover{
  margin: 0px;
    padding: 10px 20px;
    background-color: var(--blue);
    border-radius: 0px;
    color: var(--white);
}
.notification-div{
  background-color: var(--gray);
  width: 246px;
  border: 1px solid #ddd;
  padding: 10px 0px;
  right: 9.4%;
  border-radius: 5px;
  position: absolute;
  z-index: 9;
  display: none;
  top: 84px;
  text-align: left;
}
.notification-div:after{
  content: '';
    position: absolute;
    height: 10px;
    width: 10px;
    background-color: var(--gray);
    top: -6px;
    border-left: 1px solid #ddd;
    border-top: 1px solid #ddd;
    transform: rotate(45deg);
    right: 10%;
}
.notification-div p{margin: 0px;padding: 10px 20px;}
.notification-div p:hover{
  margin: 0px;
    padding: 10px 20px;
    background-color: var(--blue);
    border-radius: 0px;
    color: var(--white);
}
.search{ padding: 5px; display: none; outline: none; border:none;
    border-bottom: 1px solid black;
}
table {
  border-collapse: collapse;
  margin: 0;
  padding: 0;
  width: 100%;
  table-layout: fixed;
  border-spacing: 0 15px;
}

table tr {
  background-color: #f8f8f8;
  border: 1px solid #ddd!important;
  padding: .35em;
  margin-bottom: 10px;
}

table th,
table td {
  padding: 1.5em;
  text-align: left;

}

table th {
  font-size: .85em;
  letter-spacing: .1em;
  text-transform: uppercase;
}
.tab-img{
    display: inline-block;
    width: 30px;
    vertical-align: middle;
    border: 1px solid #000;
    border-radius: 18px;
    margin-right: 8px;
}
.ticon{
    padding: 5px;
    background: var(--gray);
    border-radius: 4px;
    width: 15px;
    height: 15px;
    text-align: center;
}
.pe{
    content: '';
    position:absolute;
    width: 7px;
    height: 7px;
    background: red;
    border-radius: 10px;
    left: 10px;
    top: 37px;
    transform: scale(0.5);
    box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
    animation: pulse-red 2s infinite;
    cursor: pointer;
}
.de{
    content: '';
    position:absolute;
    width: 7px;
    height: 7px;
    background: green;
    border-radius: 10px;
    left: 10px;
    top: 37px;
    transform: scale(0.5);
    box-shadow: 0 0 0 0 rgba(255, 82, 82, 1);
    animation: pulse-green 2s infinite;
    cursor: pointer;
}
@keyframes pulse-green {
    0% {
        transform: scale(0.85);
        box-shadow: 0 0 0 0 rgba(82, 255, 82, 0.7);
    }

    70% {
        transform: scale(1);
        box-shadow: 0 0 0 25px rgba(82, 255, 82, 0);
    }

    100% {
        transform: scale(0.85);
        box-shadow: 0 0 0 0 rgba(82, 255, 82, 0);
    }
}
.active-tr{
    background: var(--blue);
    color: var(--white);
}
.active-tr .ticon{
    background: #e5e5e54f;
}
    </style>
</head>
<body>
```

```
<section>
    <div class="sidebar">
        <h1 class="logo">KSRTC</h1>

        <ul class="nav">
             <li><a href="{% url 'myprofile' %}"><i class="fa fa-windows"></i> my 
profile</a></li>
               <li  class="active"><a  href="#"><i  class="fa  fa-shopping-bag"></i> 
dashboard</a></li>
              <li><a href="{% url 'daily_collection_sent' user_id=request.user.id 
%}"><i class="fa fa-pie-chart"></i> daily collection</a></li>
              <li><a href="{% url 'daily_collection_view' user_id=request.user.id 
%}"><i class="fa fa-cube"></i>daily collection view </a></li>
        <li><a  href="{%  url  'complaint'  user_id=request.user.id  %}"><i  class="fa 
fa-cube"></i>complaint </a></li>
             <li><a href="{% url 'user_complaints' user_id=request.user.id %}"><i 
class="fa fa-cube"></i>view complaint </a></li>

       <li><a>   {% if user.is_authenticated %}
    <a href="{% url 'logout' %}" class="btn btn-primary ml-lg-3">Logout</a>
{% else %}
    <a href="{% url 'login' %}" class="btn btn-primary ml-lg-3">Logout</a>
{% endif %}</a></li>

        </ul>

            <ul class="social">
                <li><a href="#">Facebook</a></li>
                <li><a href="#">Twitter</a></li>
                <li><a href="#">Google</a></li>
            </ul>


    </div>
    <div class="main">

        <div class="head-section">
            <div class="col-6">
                <h2>Order</h2>
                <p>30 orders found</p>
            </div>

            <div class="col-6" style="text-align: right;">
                <i class="fa fa-bell-o hicon"></i>
                <input type="text" class="search">
                <i class="fa fa-search hicon sicon"></i>

                <img src="image/user.png" class="user">
                <div class="profile-div">
                    <p><i class="fa fa-user"></i>   Profile</p>
                    <p><i class="fa fa-cog"></i>   Settings</p>
                    <p><i class="fa fa-power-off"></i>   Log Out</p>
                </div>

        <div class="notification-div">
            <p>Success!Your registration is now complete!</p>
            <p>Here's some information you may find useful!</p>
        </div>

            </div>

            <div class="clearfix"></div>
        </div>

        <br><br>
        <div class="content">
            <p>All orders</p><br><br>
      <h2>Welcome, {{ user.full_name }}!</h2>


<h2>Daily Collection Report</h2>

<form method="post" action="{% url 'complaint' user_id=user.id %}">
    {% csrf_token %}

    <label for="terminal">Select Terminal:</label>
    <select name="staff_allocation" id="staff_allocation">
        {% for staff_allocation in allocated_terminals %}
                <option  value="{{  staff_allocation.terminal.terminal_id  }}">{{
staff_allocation.terminal.terminal_name }}</option>
        {% endfor %}
    </select>
    <br>

    <!-- Select Bus Assignment -->
    <label for="bus_assignment">Select Bus Assignment:</label>
    <select name="bus_assignment" id="bus_assignment">
        {% for assignment in assigned_buses %}
                                <option     value="{{     assignment.id     }}">{{
assignment.bus_allocation.bus.registration_number         }}         -        {{
assignment.bus_allocation.bus.bus_number }}</option>
        {% endfor %}
    </select>
    <br>

    <!-- Collection Date -->
    <label for="collection_date">Collection Date:</label>
    <input type="date" name="collection_date" required>
    <br>

    <!-- Collection Description -->
    <label for="collection_description">Collection Description:</label>
     <textarea id="collection_description" name="collection_description" rows="4"
required></textarea>
    <br>

    <button type="submit">Submit</button>
</form>

</div>

<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/jquery.min.js"></script>

<script>
$(document).ready(function(){
    $(".user").click(function(){
        $(".profile-div").toggle(1000);
    });
    $(".hicon:nth-child(1)").click(function(){
        $(".notification-div").toggle(1000);
    });
    $(".sicon").click(function(){
        $(".search").toggle(1000);
    });
});
</script>

<script type="text/javascript">
    $('li').click(function(){
        $('li').removeClass("active");
        $(this).addClass("active");
    });
</script>
</body>
</html>
```

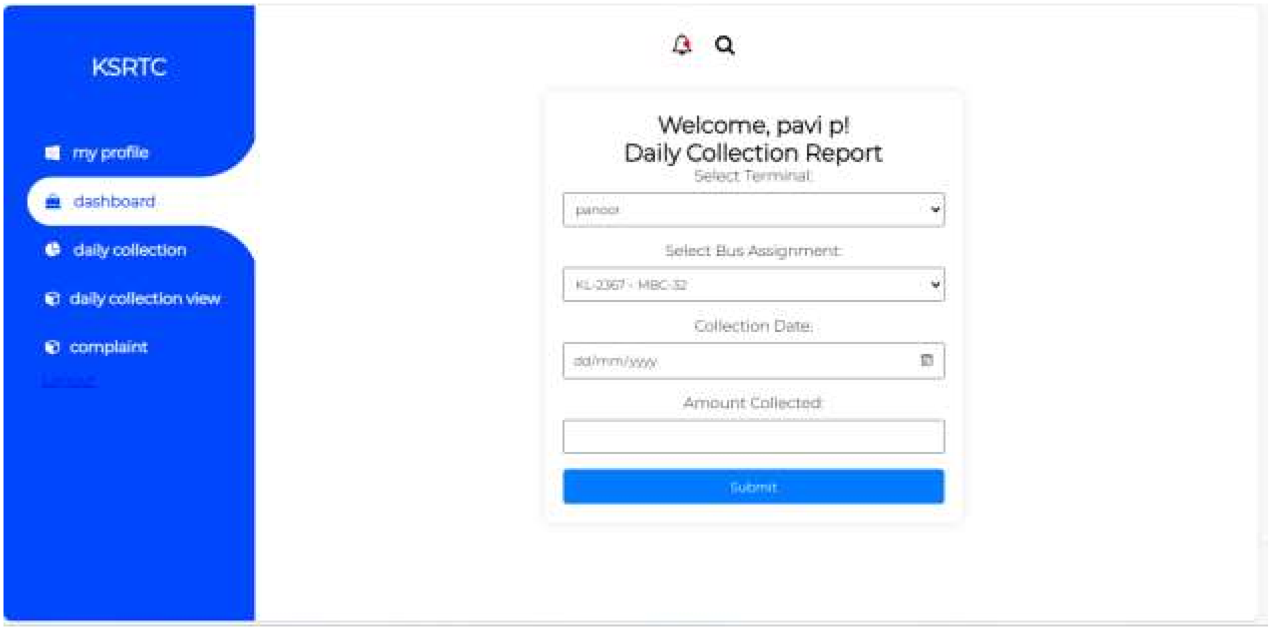
9.1 Screen Shots



Login

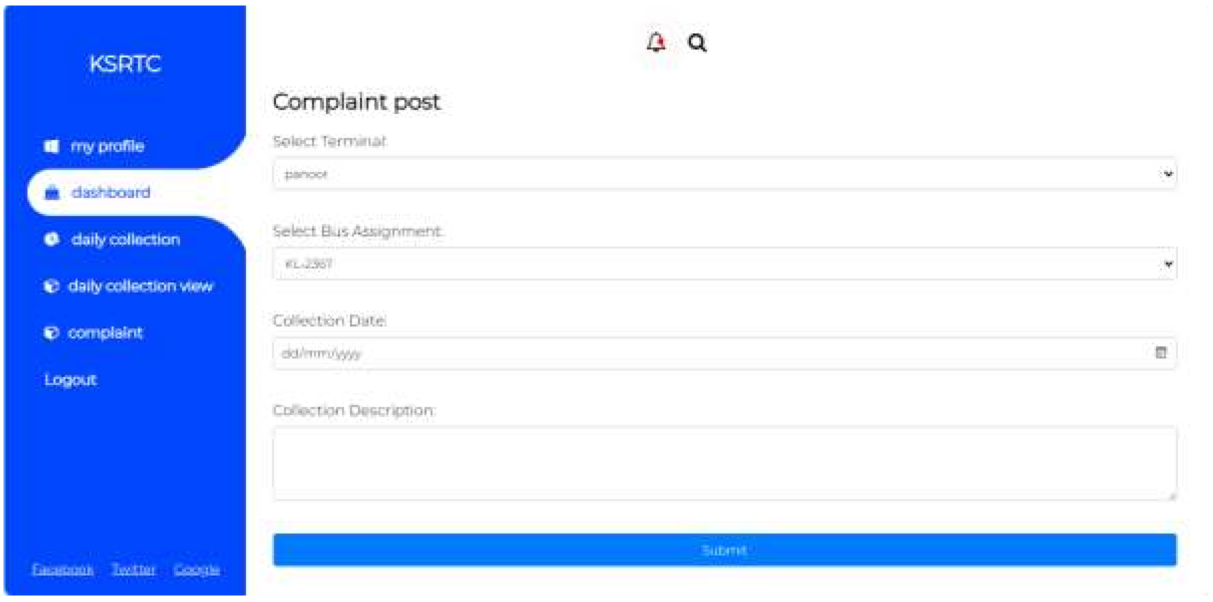


Daily Collection



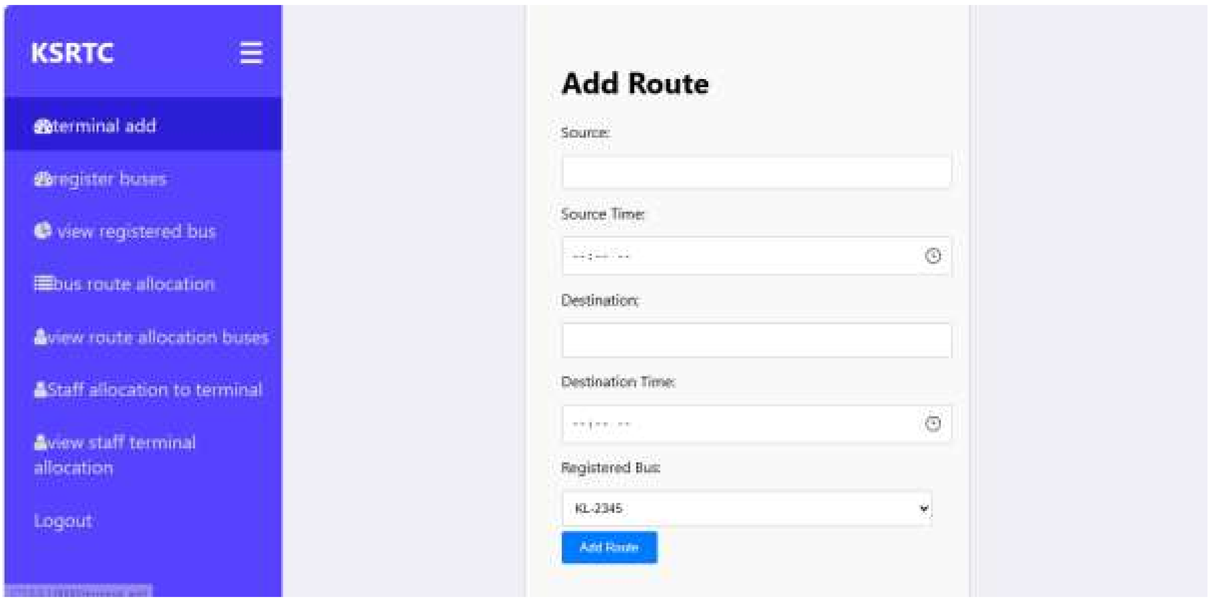
Welcome, pavi p!
Daily Collection Report
Select Terminal:
panoor
Select Bus Assignment:
KL-2367 - MBC-32
Collection Date:
dd/mm/yyyy
Amount Collected:
Submit

Complaint Post



Complaint post
Select Terminal:
panoor
Select Bus Assignment:
KL-2367
Collection Date:
dd/mm/yyyy
Collection Description:
Submit

Bus Route Allocation


KSRTC
terminal add
register buses
view registered bus
bus route allocation
view route allocation buses
Staff allocation to terminal
view staff terminal allocation
Logout

Add Route

Source:

Source Time: --:-- --

Destination:

Destination Time: --:-- --

Registered Bus: KL-2345