

PROJECT REPORT on
Smart Utility Billing

Submitted in partial fulfilment of the requirements
for the award of degree

MASTER OF COMPUTER APPLICATIONS

of

KLE TECHNOLOGICAL UNIVERSITY

by

Ms. Anusha Hadimani
(SRN: 01FE22MCA021)



DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS
KLE TECHNOLOGICAL UNIVERSITY
Vidyanagar, Hubballi-580031 Karnataka.

September- 2024

A Project Report on
Smart Utility Billing

submitted in partial fulfilment of the requirements
for the award of degree

MASTER OF COMPUTER APPLICATIONS
of

KLE TECHNOLOGICAL UNIVERSITY

by

Ms. Anusha Hadimani
(SRN: 01FE22MCA021)

under the guidance of

Prof. Shashikala V Budni
Professor,
MCA Dept.



DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS
KLE TECHNOLOGICAL UNIVERSITY
Vidyanagar, Hubballi - 580031 Karnataka.

September- 2024

**DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS
KLE TECHNOLOGICAL UNIVERSITY**



CERTIFICATE

This is to certify that the project work entitled “**Smart Utility Billing**” submitted in partial fulfilment of the requirements for the award of degree of **Master of Computer Applications of KLE Technological University, Hubballi, Karnataka** is a result of the bonafide work carried out by **Anusha Hadimani SRN:**

01FE22MCA021 during the academic year 2023-2024.

Prof. S.V. Budni

Project Guide,

Professor,

MCA Dept.

Dr. P. R. Patil

Head of Department,

Professor,

MCA Dept.

Prof. B. S. Anami

Registrar,

KLE Technological

University.

Viva-Voce Examination

Name of the Examiners

Signature with Date

1. _____

: _____

2. _____

: _____

ACKNOWLEDGEMENT

Every successful completion of any undertaking would be complete only after we remember and thank the almighty, the parents, the teachers, and the personalities, who directly or indirectly helped and guided during the execution of that work. The success of this work is equally attributed to all well-wishers who have encouraged and guided throughout the execution.

I express my deepest gratitude to **Prof. S.V. Budni** Internal the guidance and assistance throughout the project with great interest.

I avail this opportunity to express deepest gratitude to **Dr. P. R. Patil, Professor & HOD** for encouraging us with his valuable tips.

I am grateful to **Prof. B S Anami**, Registrar for his blessings. And we are grateful to **KLE Technological University Hubballi** which has given us a bright future. I would like to thank all the people for their guidance and valuable suggestions throughout the project.

I also thank all teaching and non-teaching staff members of the MCA department for their invaluable cooperation.

Finally, I would like to express our sincere thanks to our **Parents and Friends** for their enormous encouragement and all other who have extended their helping hands towards completion of our project.

Anusha Hadimani

ABSTRACT

Smart Utility Billing is an innovative utility billing system designed to streamline the management of water and electricity billing for both consumers and administrators. The platform features a user-friendly interface that enables users to register, log in, and easily access their billing information, including viewing profiles, outstanding bills, and payment history. By leveraging real-time data integration, Smart Utility Billing ensures accurate and timely updates on billing statuses, facilitating seamless transactions and minimizing discrepancies in billing amounts.

The system incorporates an admin module that empowers administrators to manage user accounts, oversee billing operations, handle notifications, and generate detailed reports on billing trends and customer interactions. This comprehensive management capability enhances operational efficiency and supports informed decision-making for utility providers.

Future enhancements for Smart Utility Billing may include mobile application access, advanced analytics for usage trends, automated payment reminders, and user feedback mechanisms to improve customer engagement. With its emphasis on operational efficiency, user experience, and robust management tools, Smart Utility Billing aims to transform utility billing processes, driving improved service delivery and customer satisfaction.

CONTENTS

Chapter No.	Table of Contents	Page No.
1	INTRODUCTION	
	1.1 Problem Statement	
2	LITERATURE SURVEY	
	2.1 Existing System	
	2.2 Proposed System Overview	
3	SYSTEM REQUIREMENTS	
	3.1 Hardware Requirements	
	3.2 Software Requirements	
4	SOFTWARE REQUIREMENT SPECIFICATION	
	4.1 Introduction	
	4.2 Purpose	
	4.3 Scope	
	4.4 Specific Requirements	
	4.4.1 Functional Requirements	
	4.4.2 Non-Functional Requirements	
5	SYSTEM DESIGN	
	5.1 Architectural Design	
	5.2 Database Design - ER Diagram	
	5.3 Data Flow Diagram - Context, DFD Level 1	
	5.4 Interface Design	
	5.5 UML Design: Use Case, Sequence, Class Diagram, Activity diagram	
6	IMPLEMENTATION	
	6.1 Introduction to New Technologies used in this project	
	6.2 Source code of challenging modules in this project and output screenshots	
7	TEST CASES AND RESULTS	
8	FUTURE ENHANCEMENTS	

9	CONCLUSION	
10	BIBLIOGRAPHY	

Chapter 1 INTRODUCTION

Digitalization has profoundly impacted the service industry across the globe. It offers unmatched convenience to the party, either as a provider or consumer of such services. With the rapidly fast technological process, businesses, and services are rendered in ways that are tough to befall under older systems. Thus, digital solutions are compulsory for the new wave of governance and enterprise operations. People belonging to a particular region expect services such as electricity, water, and other services. But these services are scattered across many portals leading to inefficiencies, causing frustration among the users and increasing operations costs. This project constructs an integrated Smart Utility Billing Digital Service Portal in an effort to put straight processes and enhance the user experience in unifying access to these services.

1.1 Problem Statement

The current utility billing processes for water and electricity are often inefficient, prone to errors, and lack real-time transparency for both consumers and administrators. Customers face difficulties in tracking their bills and payment statuses, while utility providers struggle with managing user accounts, ensuring timely bill generation, and addressing customer queries effectively. One Billing seeks to resolve these challenges by providing a streamlined, user-friendly platform that simplifies billing management, enhances operational efficiency, and improves customer satisfaction through real-time updates and seamless payment integration.

Chapter 2 LITERATURE SURVEY

To this extent, the literature surrounding digital service platforms is apparent that there is indeed a significant move toward integration of services across diverse industries cut across government, utilities, and other essential service providers. This paper, therefore, reviews some of these notable platforms currently in use and their limitations afterward overview of the proposed Smart Utility Billing digital service portal which apparently seeks to overcome those limitations by consolidating services in a user-friendly and secure environment.

2.1 Existing System

1. MyGov(India)

MyGov is the Indian Government's integrated Web portal with all services. It is a platform which includes electrification billing, water management, and even has electronic payment options for taxes. However, despite all these services, it has a few demerits:

Merits:

- It encompasses a vast number of government services under one umbrella.
- Platform for secure payments of bills and services.
- Individual user dashboard to keep a track of the bills.

Demerits:

- Hassle in finding one's way through due to the user interface.
- Service integration varies from region to region.

2. gov.uk(UK)

This website offers various types of public services like utility bills payment, tax services, and registration of cars, etc. It is applied throughout the UK but has some weaknesses:

Merits:

- SSO is provided that means users are able to access vast services by using a single login.
- This website provides transaction protocols to ensure safe data from the user.
- There are various types of services available in one place, so it acts like one-stop shopping for people.

Demerits:

- Cluttered because vast services are available on this website.
- Poor availability of the service, differs depending on location from one region to another, where they tend to depend on the local councils.
- Loading time is slower for during peak seasons such as tax season.

3. eCitizen(Kenya)

eCitizen is Kenyan site, developed to seek to offer various government services ranging from utility payments to property registration services. Although the platform has made it easy for accessibility, there are still ongoing problems:

Merits:

- Central platform offering a variety of services.
- It has an intuitive design that provides easy navigation for users • Mobile first approach, that makes services accessible to more users

Demerits:

- Poor connectivity in remote areas.
- Limited support for many languages, thereby to a great extent, limiting its usage among users who do not understand the basic language.
- Technical support is less accessible; thus, an issue takes more time to be resolved.

4. ServiceNSW(Australia)

Service NSW consolidates state-level services like electricity, water, and council rates into one platform. It has been effective in improving user access but faces integration challenges:

Merits:

- Digital wallet feature allows users to store bills and receipts securely.
- Customizable dashboards for personalized service management.
- Multilingual support, catering to Australia's diverse population.

Demerits:

- Incomplete integration of all services, requiring users to log in to separate systems for some functions.
- Users report occasional delays in service updates and availability due to system maintenance.

5. DubaiNow(UAE)

Dubai Now provides access to over 120 government services, including utility management, vehicle registration, and bill payments. While innovative, it still faces limitations in terms of user experience:

Merits:

- Mobile-first design ensures high accessibility and usability.
- Integration of smart features such as bill reminders and quick pay options.
- Multilingual interface catering to Dubai's international population.

Demerits:

- High dependency on stable internet connectivity, which can be an issue for users in regions with lower bandwidth.
- Complexity for new users due to the breadth of services offered.
- Not all services are fully digitized, requiring physical presence for certain verifications.

2.2 Proposed System

The **Smart Utility Billing Portal** builds on the successes and addresses the shortcomings of the existing systems. It aims to consolidate various essential services—such as electricity, water, and tax payments—into a single, integrated platform that is secure, easy to navigate, and accessible for all users, regardless of their technological proficiency.

Key features of the proposed system include:

- **Centralized Service Access:** A unified portal that brings together multiple services, reducing the need for users to navigate through multiple systems.
- **User-Friendly Interface:** Designed with simplicity in mind, the platform will feature intuitive navigation, personalized dashboards, and responsive design for easy use on both desktop and mobile devices.
- **Real-Time Notifications:** Users will receive notifications for due dates, payment confirmations, and service updates, ensuring they stay informed and compliant with their bills.
- **Scalability:** The system will be designed to handle large user bases, ensuring reliable service even during peak times.
- **Security and Privacy:** The platform will adhere to strict security standards, including data encryption, two-factor authentication, and compliance with data protection regulations such as GDPR.
- **Inclusive Design:** Accessibility will be a core focus, ensuring that users with different abilities or low digital literacy can navigate and use the platform effectively.

By addressing the gaps present in existing systems, **Smart Utility Billing Portal** will create a seamless experience for users, simplifying access to critical services while ensuring secure and efficient operations.

Chapter 3

SYSTEM REQUIREMENTS

The Smart Utility Billing Portal requires a robust and scalable infrastructure to handle multiple services, ensure data security, and provide a smooth user experience. This section outlines the hardware and software requirements necessary to develop, deploy, and maintain the platform effectively.

3.1 Hardware Requirements

Server Hardware:

Operating System: Windows Server (e.g., 2012, 2016, 2019) or other compatible systems like Linux, depending on project needs.

Processor: Dual-core processor or higher (e.g., Intel Core i3 or better).

RAM: Minimum 4GB RAM, recommended 8GB or higher (especially with large datasets or multiple users).

Storage: SSD storage is recommended for faster data access; HDD is an option for more storage capacity.

Network: High-speed internet (fiber, broadband) for reliable data transfer between the server and client.

Client Hardware:

Device: Desktop, laptop device with a modern web browser (Chrome, Firefox, Edge).

Internet: Reliable internet connection to access the web-based system efficiently.

3.2 Software Requirements

a. Server-side (Backend):

- Spring Boot: Latest stable version for creating a robust and efficient backend service. It provides built-in support for RESTful APIs and dependency injection for modular code.
- Hibernate: Latest stable version for Object-Relational Mapping (ORM), facilitating seamless interactions between Java objects and PostgreSQL tables.
- PostgreSQL: Latest stable version for managing relational data efficiently and providing advanced querying capabilities.
- Maven or Gradle: Build tools for managing dependencies, building the project, and running unit tests.
- JWT (JSON Web Tokens): For secure authentication and authorization, ensuring safe transmission of information between the client and server.
- Spring Security: To handle authentication, authorization, and session management, integrating with JWT for securing endpoints.
- Git: Version control system for collaborative development and code management.
- IDE/Text Editor: IntelliJ IDEA, Eclipse, or Visual Studio Code for writing and managing server-side code efficiently.

b. Client-side (Frontend):

- React.js: Latest stable version for building dynamic, interactive user interfaces and handling client-side rendering.
- React Router: For managing client-side routing, enabling seamless navigation between pages.
- Redux (optional): For state management in cases of complex application states.
- HTML5/CSS3: For structuring and designing web pages, ensuring a modern and responsive UI.
- JavaScript (ES6+): A programming language for adding interactivity, handling data fetching, and state management.
- Axios: For making HTTP requests to the backend API endpoints. Provides more features and flexibility than the Fetch API, including automatic JSON parsing.
- Bootstrap or Material UI (optional): CSS frameworks for responsive design and pre-built UI components to speed up development.
- JWT Decoding Library: For decoding JWT tokens on the client-side to manage user sessions and handle authorization in the UI.
- IDE/Text Editor: Visual Studio Code, Sublime Text, or any other preferred editor for writing client-side code.

c. Development Tools:

- Postman: An API development and testing tool for testing RESTful endpoints and verifying backend functionality.
- pgAdmin: A GUI tool for PostgreSQL database management and querying.
- ESLint: JavaScript linter for maintaining code quality and consistency on the client side.
- Prettier: Code formatter for ensuring a consistent coding style across the project.
- Git/GitHub: For version control, collaborative development, and managing the project codebase.

d. Authentication and Authorization:

- JSON Web Tokens (JWT): Used for secure transmission of user authentication and authorization data between the client and server. JWT tokens are generated upon successful login and verified in the backend to protect routes and resources.
- Spring Security with JWT Integration: To implement JWT-based authentication and authorization within the Spring Boot application. This setup secures RESTful endpoints and manages user sessions.

e. Database Connectivity:

- Spring Data JPA: Provides a schema-based solution to interact with PostgreSQL using Hibernate ORM. It allows defining the database schema using entity classes and provides an abstraction over the CRUD operations.
- PostgreSQL JDBC Driver: For seamless communication between Spring Boot and the PostgreSQL database.
- Connection Pooling (HikariCP): Integrated within Spring Boot for efficient management of database connections, enhancing performance.

f. Security:

- **Spring Security:** Comprehensive security framework that integrates with JWT for securing endpoints, managing authentication and authorization.
- **CORS (Cross-Origin Resource Sharing):** Configured in Spring Boot to allow/deny cross-origin requests to the backend server from the client-side application.
- **Helmet (Express.js equivalent in Spring Boot):** Implement security headers using Spring Security's HTTP security configuration, setting headers like Content Security-Policy and X-Content-Type-Options to protect the application against common vulnerabilities.
- **Input Validation and Sanitization:** Use validation annotations in Spring Boot (e.g., `@Valid`, `@NotNull`) to ensure data integrity and prevent injection attacks.

3.3 Non-Functional Requirements

Performance: The system should handle a maximum concurrent user, with a response time of fewer than 2 seconds for standard operations (e.g., viewing bills, making payments).

Scalability: The system must be easily scalable to accommodate increased traffic, services, and users without significant changes to the core architecture.

Reliability: At least 99.9% uptime is required, with failover mechanisms in place to ensure continuous service even in the event of server failure.

Usability: The platform must be easy to navigate for users of all technical skill levels, with responsive design for accessibility on both desktop and mobile devices.

Maintainability: The codebase should be modular and well-documented, enabling easy updates and bug fixes.

Chapter 4

SOFTWARE REQUIREMENT SPECIFICATIONS

SRS stands for Software Requirements Specification. It is a comprehensive document that outlines the functional and non-functional requirements of a software system. The SRS serves as a blueprint for software development, providing detailed information about what the system should do and how it should behave from the perspective of its users.

4.1 Introduction

The Software Requirements Specification (SRS) document serves as a comprehensive guide to the requirements of a software project. It outlines the functional and nonfunctional requirements of the software system, providing a clear understanding of what needs to be developed. The introduction section of the SRS document sets the stage for the rest of the document, providing an overview of the project and its objectives.

4.2 Purpose

The purpose of this document is to define the requirements for **One Digital**, a comprehensive platform designed to streamline private water and electricity billing systems. The Software Requirements Specification (SRS) aims to provide clarity for all stakeholders by outlining the platform's objectives, key functionalities, and technical requirements. This document ensures that the project remains aligned with its goals, preventing scope creep and enabling efficient project management.

By detailing both functional and non-functional requirements, such as bill management, user notifications, payment processing, and system security, the SRS serves as a critical reference for developers during the design and development phases. It also facilitates testing and validation by offering clear criteria for success, ensuring that the final product meets user needs.

Additionally, this document will aid future maintenance efforts by documenting the system's original architecture and design choices.

4.3 Scope

The scope of the **One Digital** project is to develop a comprehensive, web-based platform that streamlines private water and electricity billing services for users. The platform aims to simplify the billing process for both customers and service providers, ensuring transparency, efficiency, and ease of use. Customers will have the ability to view and pay their utility bills online, while service providers can manage billing data and payments seamlessly.

Key features within the scope of this project include user registration, bill generation and viewing, payment processing through secure gateways, and a notification system to alert users of upcoming due dates and payment confirmations. Additionally, the platform will feature an admin dashboard to manage users, oversee operations, and ensure accurate billing and timely updates.

Non-functional aspects such as system performance, data security, privacy, and scalability are also covered to ensure the platform is reliable, secure, and able to handle increased user traffic as it grows. The project will initially focus on web-based functionality, with the potential for mobile app development and integration of more advanced features like real-time customer support in future phases.

4.4 Specific Requirements

Specific requirements detail the essential functionalities, constraints, and quality attributes that the **Smart Utility Billing Portal** must possess to meet the needs of its users. These requirements define the platform's capabilities, ensuring a seamless experience for users managing services like electricity and water billing. Below are the specific functional and non-functional requirements for the **Smart Billing Digital Service Portal**.

4.4.1 Functional Requirements

1. User Registration and Login:

- Users (including service subscribers and administrators) must be able to register by providing details such as name, email, phone number, address, and password.
- The system must allow users to securely log in using their email and password, implementing two-factor authentication (2FA) for added security.

2. Profile Management:

- Users must be able to manage and update their profiles, which will include personal details, payment preferences, and service subscriptions (e.g., electricity, water).
- Administrators should be able to view and update user profiles, manage user privileges, and track user activity history.

3. Service Listings:

- Users must be able to view the available services (electricity, water, tax, etc.) along with detailed descriptions, billing cycles, and payment options.
- The platform must offer search and filter functionalities, allowing users to filter services by billing status, payment method, and due date.

4. Billing and Invoice Management:

- Users must be able to view their bills (e.g., electricity, water) with details such as the bill amount, due date, issued date, and payment status.
- Users should have the option to download invoices and view payment history.
- The system must generate an electronic invoice and allow users to pay bills securely through integrated payment gateways (e.g., Razor pay).

5. Payment Transactions:

- The platform must support multiple payment options (e.g., credit/debit cards, UPI, net banking) and securely process payments for services.
- After successful payment, the system should update the billing status to "Paid" and display the payment details (date, time, and amount).
- If the user revisits a bill that has already been paid, the system must display the paid status and offer an option to download the invoice.

6. Notifications:

- Users must receive notifications for upcoming bill due dates, successful payments, and service-related updates through email and the platform's notification system.
- Admins should also be notified of any system errors, service disruptions, or important user actions.

7. Rating and Feedback System:

- Users should be able to rate the services (electricity, water, etc.) and provide feedback regarding their experience with the platform.
- Ratings and reviews should be visible to other users, helping improve service delivery and user satisfaction.

8. Admin Interface:

- Administrators must have a dashboard to manage users, services, and payment records.
- Admins should be able to approve, suspend, or delete user accounts, manage services, and resolve disputes related to payments or services.

4.4.2 Non-Functional Requirements

1. Performance:

The system should respond to user interactions (e.g., bill viewing, profile updates) within 2 seconds to ensure a smooth and efficient user experience.

2. Scalability:

The platform must be scalable to handle a growing number of users, bills, and services without significant changes to its architecture. It should efficiently manage peak loads, such as during the end of billing cycles.

3. Security:

All user data (personal information, payment details) must be encrypted both in transit and at rest, using SSL/TLS encryption.

The platform should implement role-based access control (RBAC) and two-factor authentication (2FA) to prevent unauthorized access to user and admin accounts.

4. Usability:

The platform must have an intuitive and user-friendly interface, allowing users of all technical skill levels to easily navigate and manage their services. The user interface should be optimized for mobile and desktop usage.

5. Availability:

The system must maintain at least 99.9% uptime, ensuring continuous service access for users. The platform should implement failover and disaster recovery mechanisms to prevent downtime.

6. Maintainability:

The platform codebase should be modular, well-documented, and designed for easy maintenance, allowing future updates, bug fixes, and feature enhancements without disrupting service.

Chapter 5

SYSTEM DESIGN

The system design for the **Smart Utility Billing Portal** follows a client-server architecture where the frontend and backend communicate through REST APIs. The frontend is developed using React, ensuring a responsive and user-friendly interface for all users, including administrators and service subscribers. The backend is built using **Spring Boot**, which provides secure, scalable, and reliable handling of service data such as utility bills, user profiles, and payment transactions.

5.1 Architectural Design

The **Smart Utility Billing Portal** architecture consists of several key components that interact seamlessly to provide a unified platform for managing essential services like electricity and water billing. The architecture is based on a client-server model, with a clear separation between the frontend, backend, and database. This modular approach ensures scalability, security, and performance, while allowing future enhancements and integrations with additional services.

Frontend and Backend:

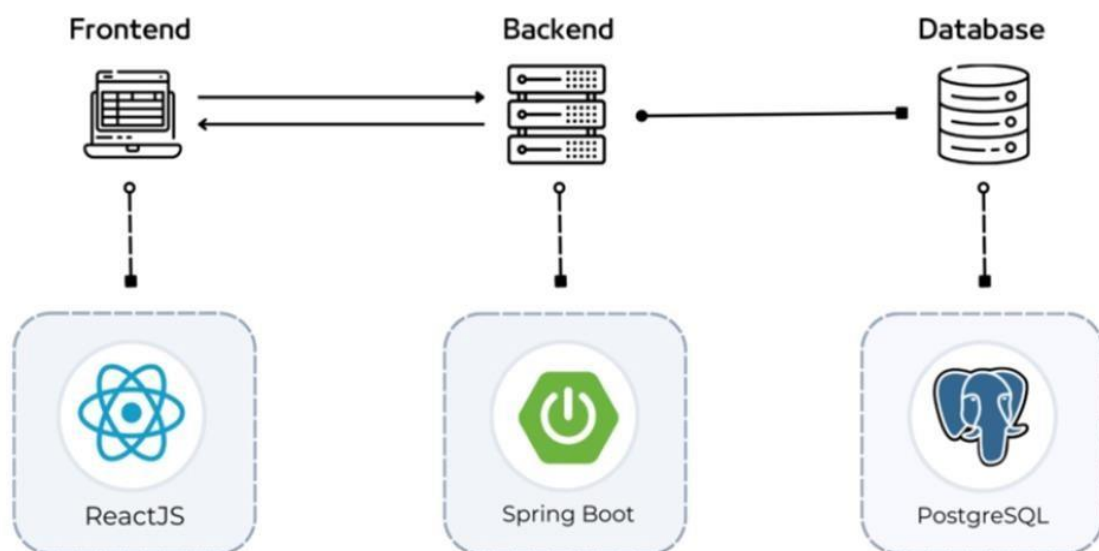


Fig 5.1.1 Architectural Design for Smart Utility Billing System [1].

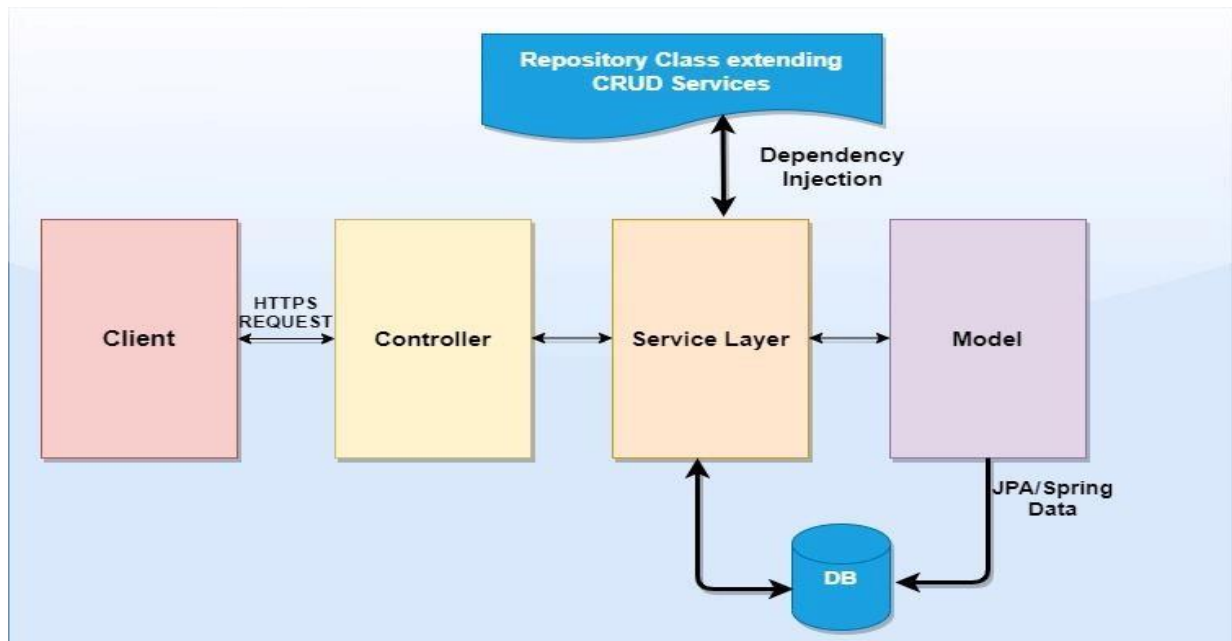
Database and External Services:

Fig 5.1.2 Architectural Design for Smart Utility Billing System [2].

5.2 Database Design – ER Diagram

The ER diagram for One Digital outlines the key entities and their relationships within the utility billing platform. It includes entities such as User, Service, Bill, Payment, and Admin, showcasing their attributes and interactions.

- The User entity represents customers, storing their personal details (name, email, address, etc.) and linking them to bills and payments.
- The Service entity stores information related to the utility services provided, such as water and electricity, including service types and descriptions.
- The Bill entity captures billing details, such as the amount due, issue date, due date, and status (e.g., paid, due).
- The Payment entity tracks payment records, including payment dates, methods, and amounts.
- The admin entity manages users, oversees billing operations, and handles system configurations.
- This ER diagram helps visualize the data structure and the relationships between users, services, billing, and payments, providing a clear understanding of how different components interact within the One Digital system.

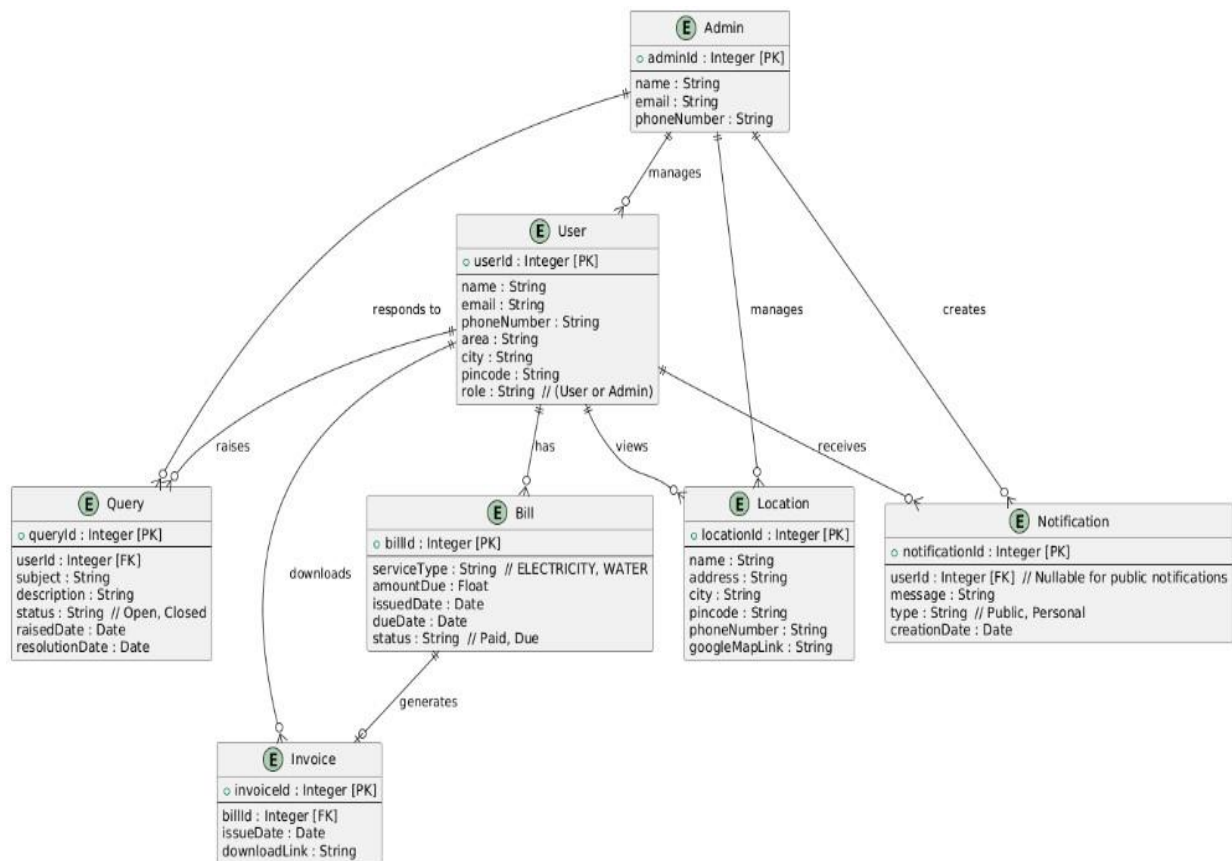


Fig 5.2.1 ER Diagram for Smart Utility Billing System [3].

5.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a visual representation that shows how data flows through a system or process. It is a useful tool for understanding system requirements, identifying processes, defining data flows, and designing systems. DFDs provide a high-level overview of how data moves through the system and are commonly used during the analysis and design phases of system development.

Context Diagram

A context diagram is a type of DFD that provides a high-level view of a system. It typically consists of a central system surrounded by external entities represented as circles or squares. Arrows indicate the flow of data or information between the central system and the external entities.

5.3.1 DFD Level 0

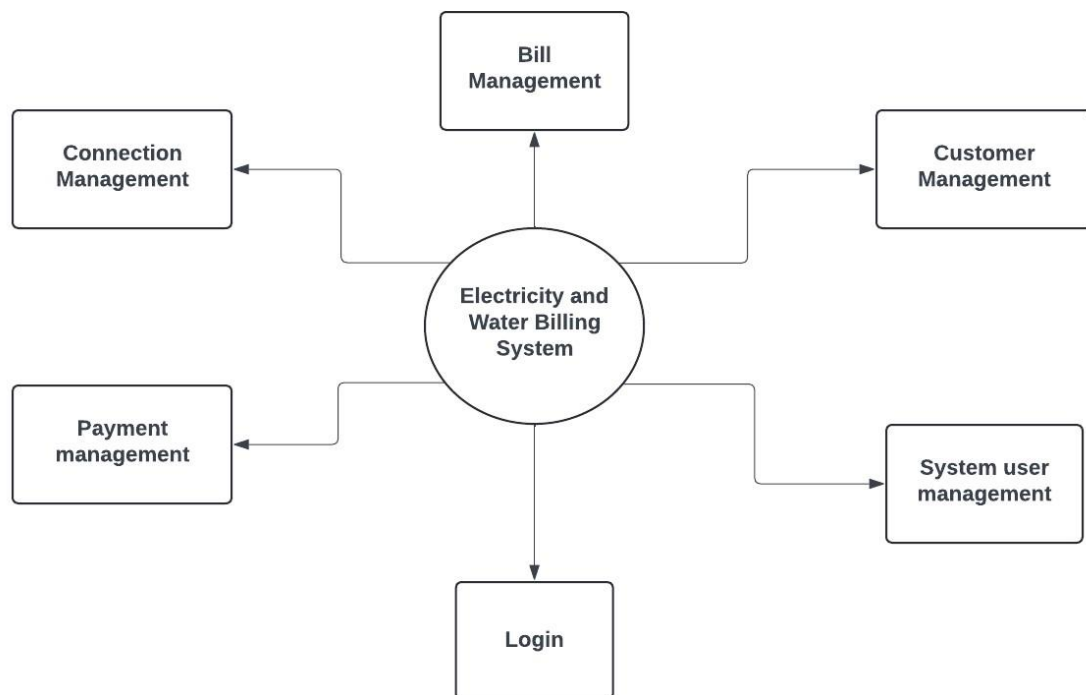


Fig 5.3.1 Architectural Design for Smart Utility Billing System [4].

5.3.2 DFD Level 1

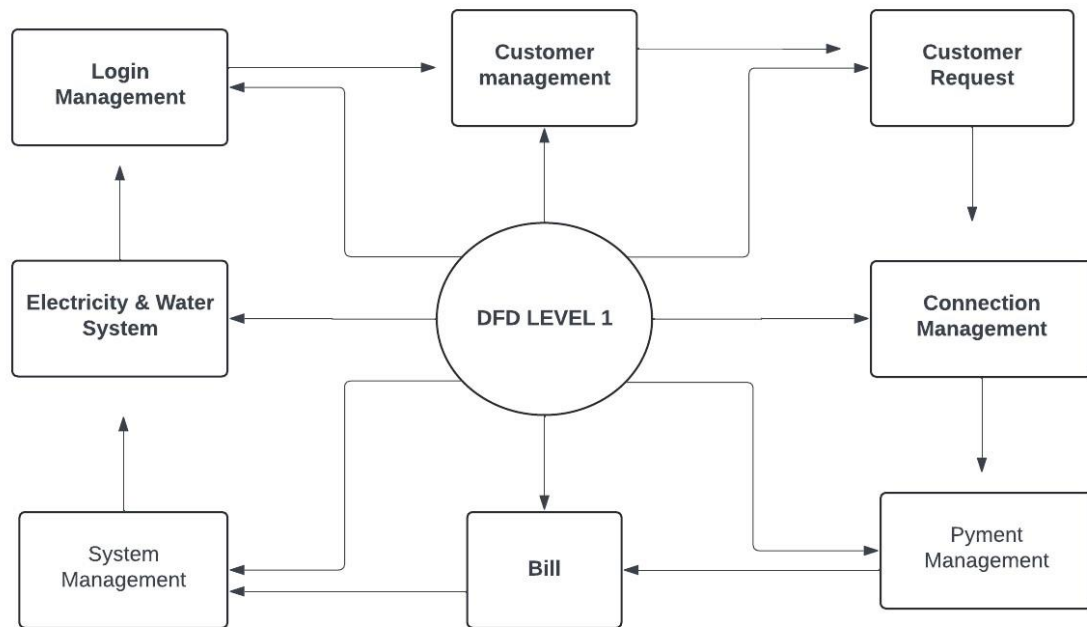


Fig 5.3.2.1 Architectural Design for Smart Utility Billing System [5].

5.4 Interface Design

An Interface Diagram is a visual representation that shows the connections and interactions between various components, systems, or modules within a software or hardware environment. It primarily focuses on how different parts of the system interact or communicate with each other through defined interfaces.

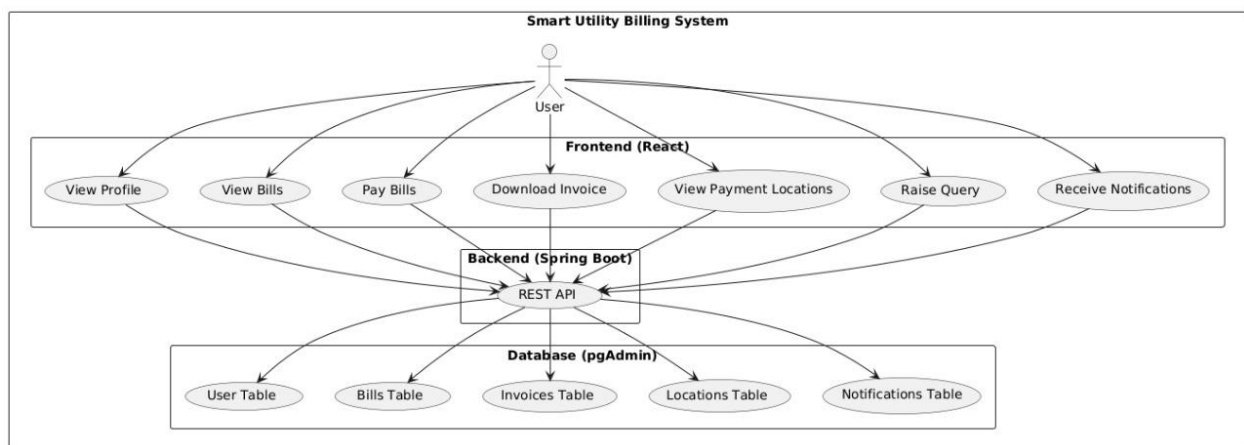


Fig 5.4.1 Interface Design for Smart Utility Billing[6].

5.5 UML Design

A UML (Unified Modelling Language) diagram is a standardized visual tool used to model the structure and behaviour of a system. It provides a way to illustrate relationships, interactions, and functionalities within a system in a clear and concise manner.

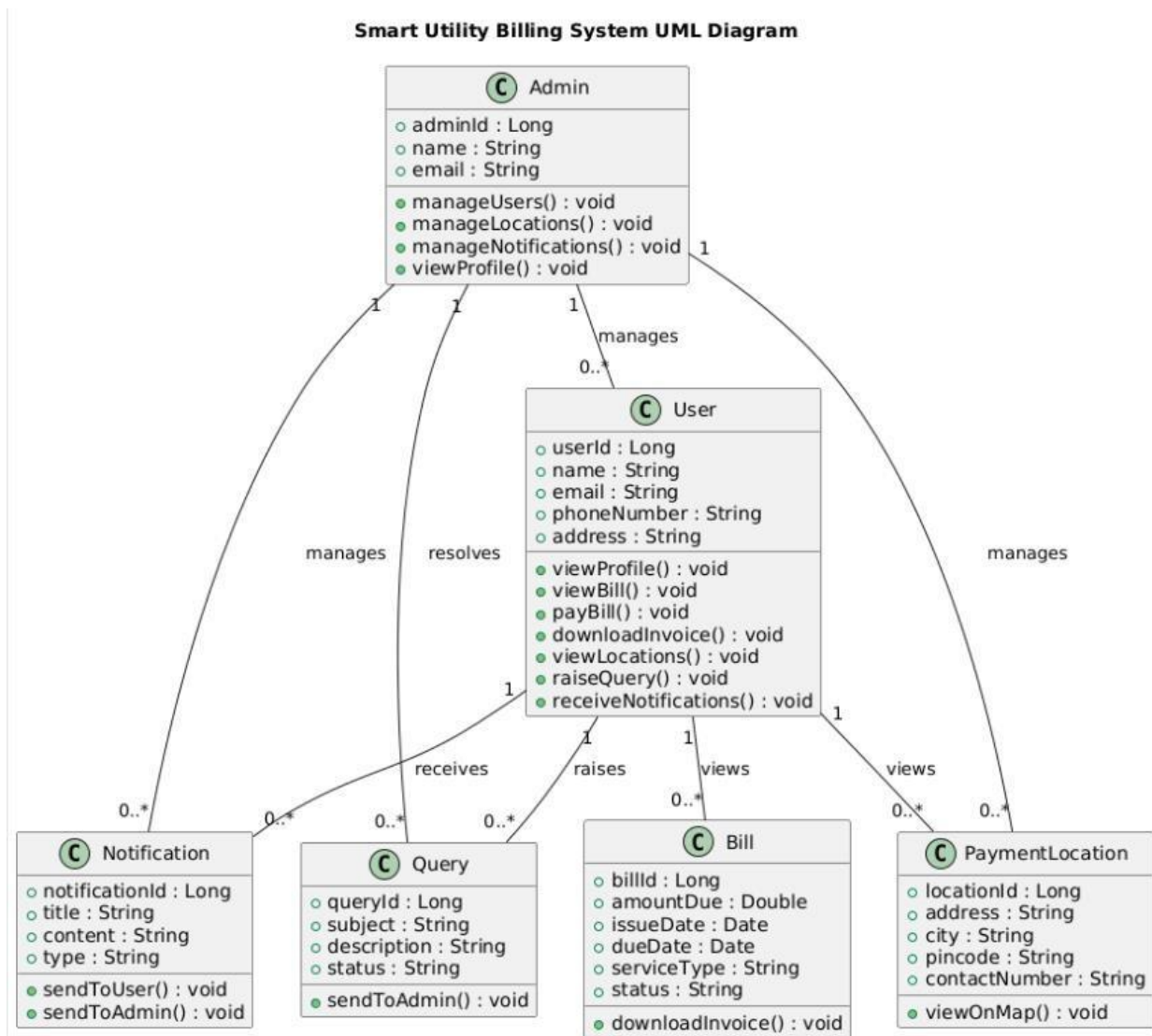


Fig 5.5.1 UML design for Smart Utility Billing[7].

5.5.1 Use Case

A use case diagram visually represents the interactions between users (actors) and the system through various use cases. It illustrates the functional requirements of the system and how different actors engage with these functionalities. This diagram helps in understanding the system's requirements and its scope by highlighting the key processes and user interactions.

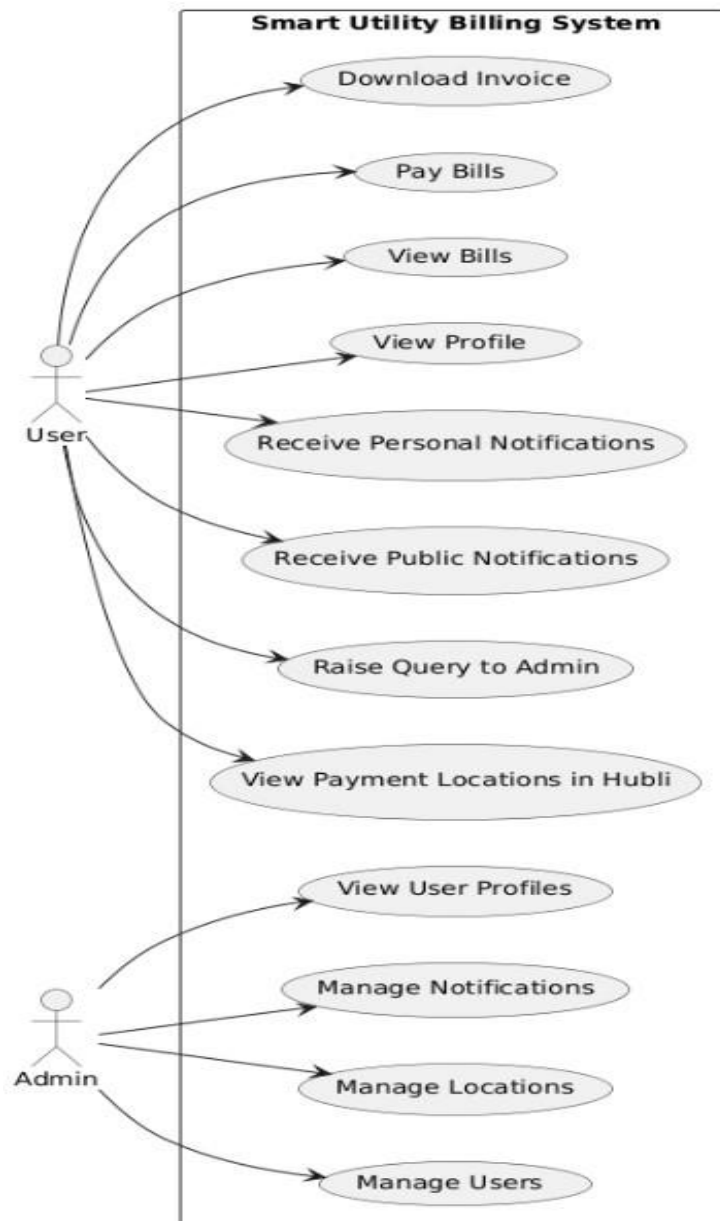
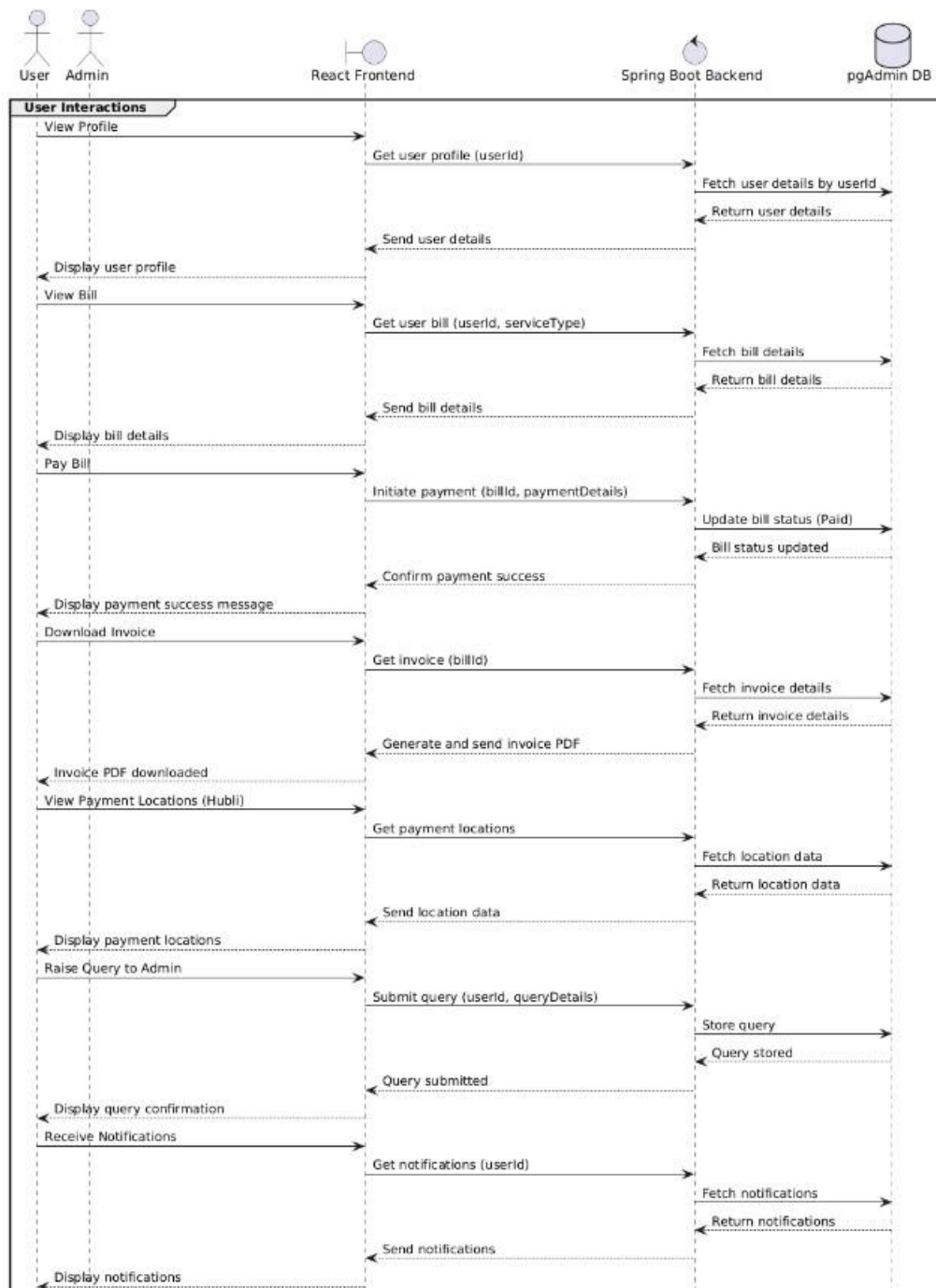


Fig 5.5.1.1 Use Case for Smart Utility Billing[7].

5.5.2 Sequence Diagram

A sequence diagram represents the flow of messages exchanged between different components over a period of time to achieve a specific functionality or scenario within the system.



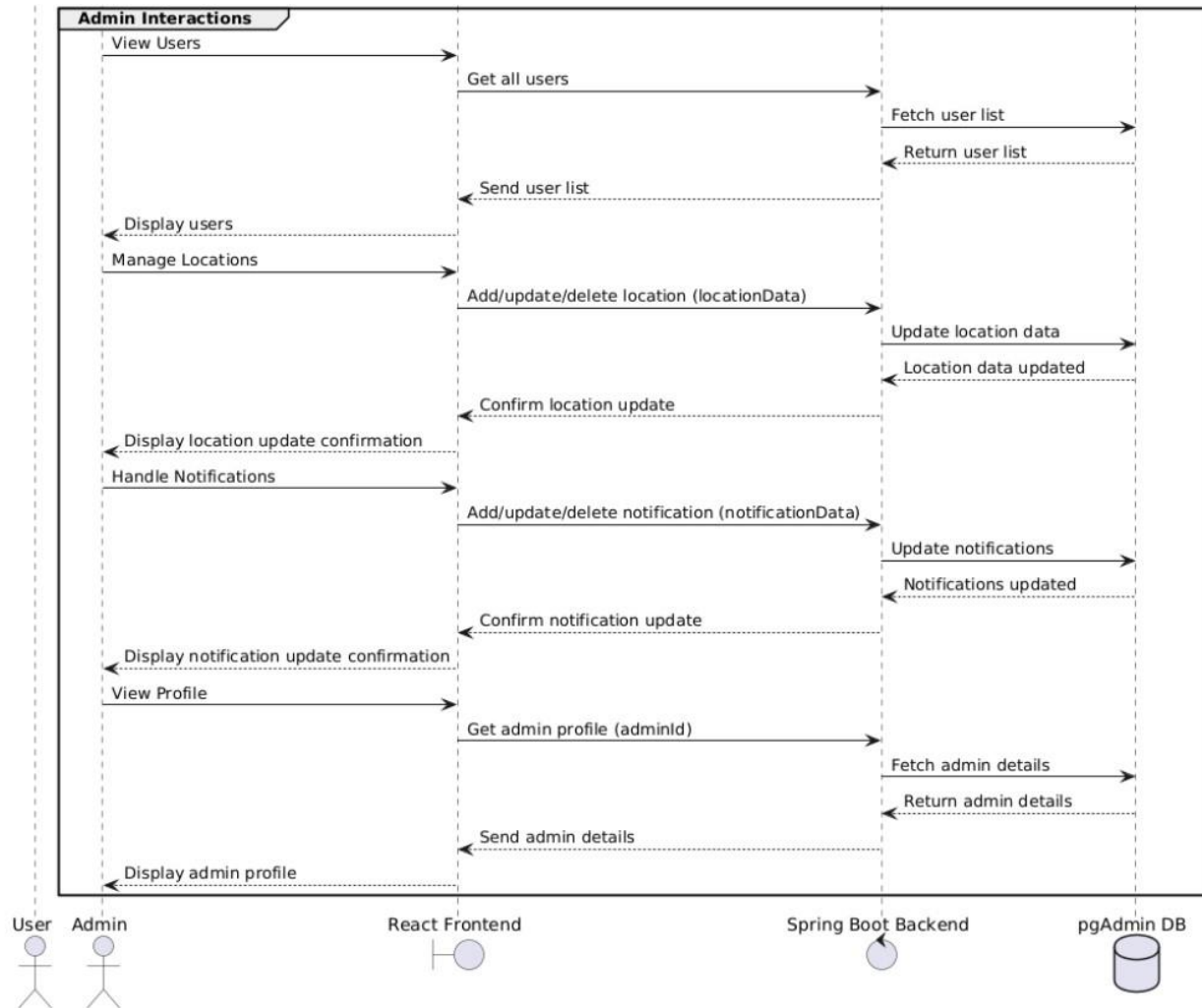


Fig 5.5.2.1 Sequence Diagram for Smart Utility Billing[8].

5.5.3 Class Diagrams

A class diagram is a type of static structure diagram in the Unified Modelling Language (UML) that represents the structure of a system by showing the system's classes, their attributes, methods, and the relationships among the classes. It provides a visual representation of the various entities (classes) within the system and how they interact with each other.

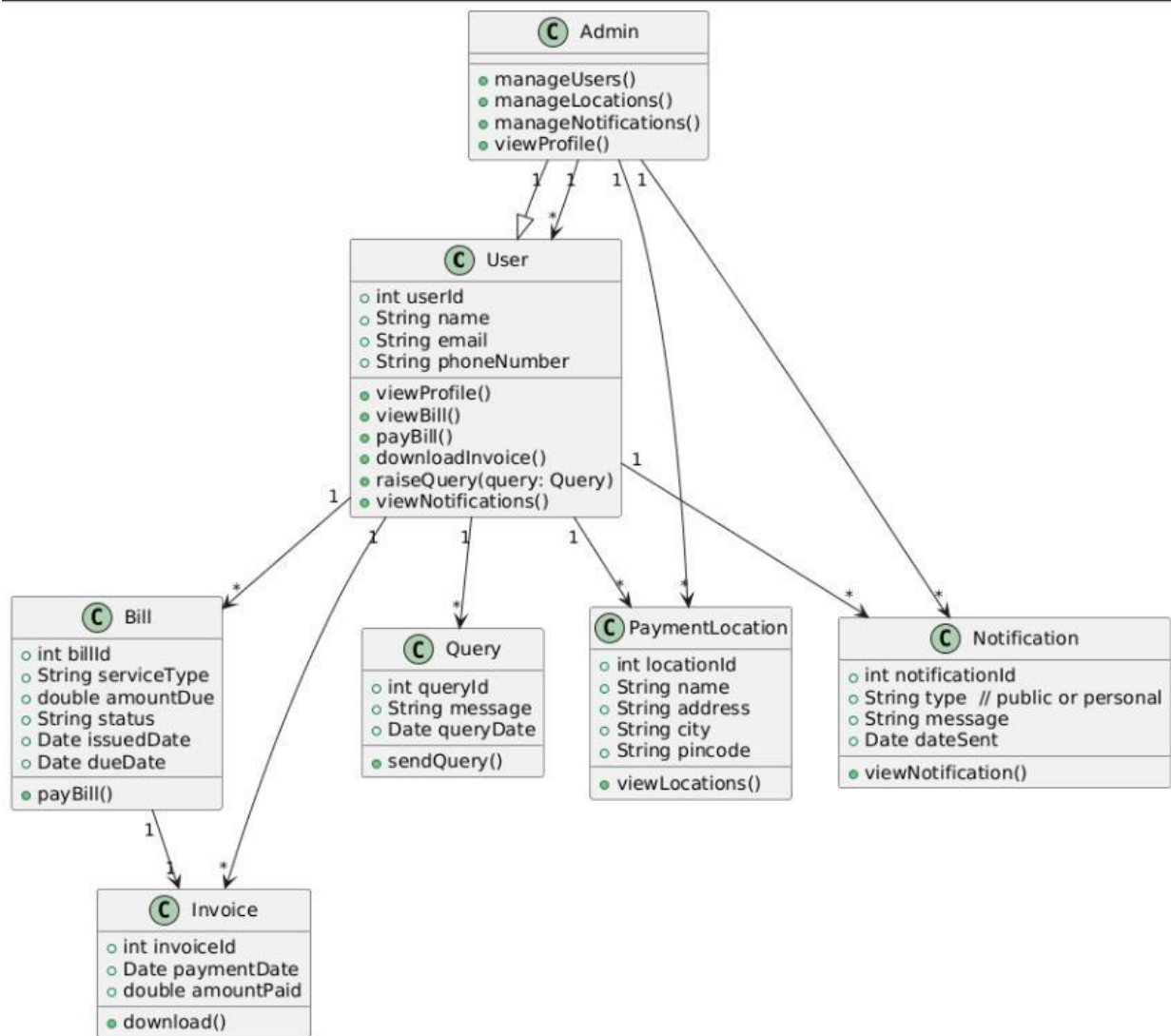
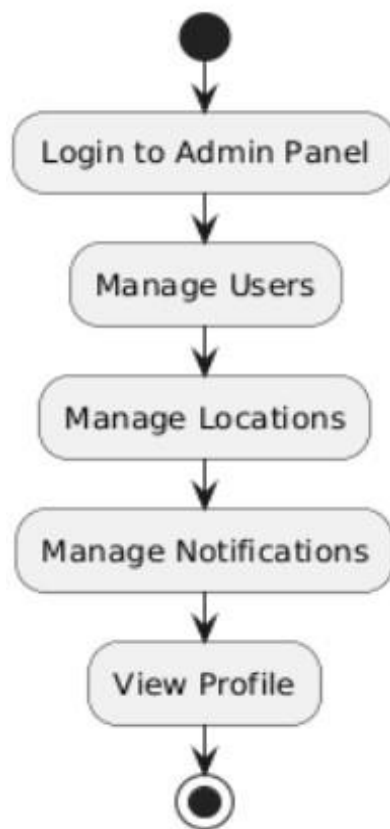


Fig 5.5.3.1 Class Diagram for Smart Utility Billing[8].

5.5.4 Activity Diagram

An Activity Diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration, and concurrency. It depicts the actions, decision points, and transitions involved in various processes or workflows within a system. Activity Diagrams are part of the Unified Modelling Language (UML) and are commonly used in software engineering to model the behaviour of systems, processes, or classes.

Admin Module:



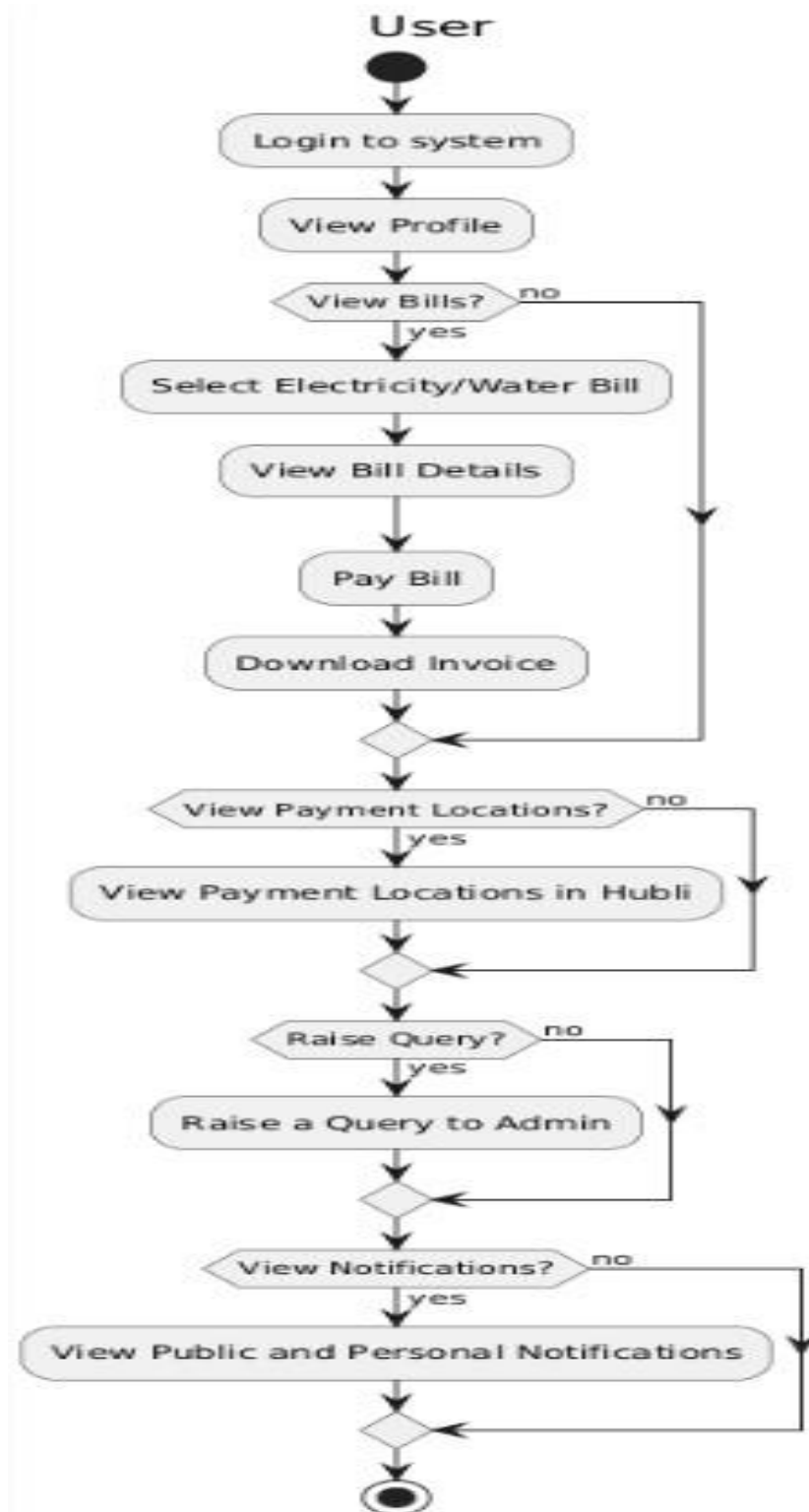
User Module:

Fig 5.5.4.1 Activity Diagram for Smart Utility Billing[9].

Chapter 6

IMPLEMENTATION

The implementation part typically includes detailed descriptions of how the proposed solution or system was developed, including the technologies used, the methodologies followed, and any challenges encountered during the implementation process. Here's how you can structure the implementation section of your project report.

6.1. Introduction To New Technologies Used in This Project Frontend

Development:

ReactJS:

Description: A powerful JavaScript library for building user interfaces, known for its component-based architecture, virtual DOM for performance optimization, and strong community support.

Role in Project: Used to develop the dynamic and interactive user interface of the application. ReactJS enables a modular codebase, making it easier to manage and maintain. Its component-based structure helps in building reusable UI elements, while its state management capabilities support complex interactions and real-time updates.

Backend Development:

Spring Boot:

Description: A framework that simplifies the development of Java-based enterprise applications by providing a comprehensive set of tools and conventions for building robust and scalable backend services.

Role in Project: Serves as the core backend framework, responsible for handling business logic, data processing, and API endpoints. Spring Boot's convention-over-configuration approach streamlines development, while its support for RESTful APIs

and integration with various databases ensures efficient backend operations and easy scalability.

Database Management: pgAdmin:

Description: A powerful open-source administration and management tool for PostgreSQL, offering a user-friendly interface for database management, query execution, and monitoring.

Role in Project: Used to manage and interact with the PostgreSQL database. pgAdmin provides an intuitive interface for database design, query execution, and performance monitoring, facilitating efficient data management and analysis. It supports database administration tasks, including schema design, data import/export, and user management.

Authentication and Authorization:**Spring Security:**

Description: A powerful and customizable authentication and access control framework for Java applications, part of the Spring ecosystem.

Role in Project: Implemented to handle user authentication and authorization within the Spring Boot application. Spring Security provides features such as secure login, role based access control, and protection against common security vulnerabilities. It integrates seamlessly with the backend to enforce security policies and manage user sessions.

JWT (JSON Web Tokens):

Description: A compact, URL-safe means of representing claims to be transferred between two parties. It is commonly used for securely transmitting information between a client and a server.

Role in Project: Used for stateless authentication in the application. JWTs are generated upon user login and used to authenticate requests to the backend. This method simplifies token-based authentication and enhances security by ensuring that sensitive information is not exposed.

Testing:**Unit Testing:**

Description: Testing individual components or functions to ensure they work as expected in isolation.

Role in Project: Applied to both frontend and backend components to verify that each unit of the application performs correctly. In ReactJS, unit testing ensures that individual UI components function as intended. In Spring Boot, unit testing verifies the correctness of business logic and service layers.

Integration Testing:

Description: Testing the interaction between different components or systems to ensure they work together as expected.

Role in Project: Ensures that the frontend and backend integrate smoothly and that data flows correctly between them. Integration testing in Spring Boot involves testing the interaction between the application's various components and external systems, while in ReactJS, it checks the integration of components with API endpoints.

End-to-End Testing:

Description: Testing the complete workflow of the application from the user's perspective to ensure all parts of the system work together as intended.

Role in Project: Used to validate the overall functionality of the application. Tools such as Cypress or Selenium can be employed to simulate user interactions and verify that the entire system, including ReactJS UI and Spring Boot backend, performs correctly from a user's viewpoint.

6.2. Source code of challenging modules:**1. User Registration and Validation**

The major user registration validations for phone number, email format, and password strength to your Signin component. This will ensure proper input handling and validation using custom validation logic.

Challenge:

- Generating strong password and accepting valid email format and phone number.

Solution:

- Email Validation: Ensures the email is in a correct format.
- Password Validation: Ensures the password meets security requirements (minimum length, uppercase, lowercase, digit, special character).
- Phone Number Validation: Checks that the phone number is exactly 10 digits.

Code Snippet:

```
const Signin = () => {  const
navigate=useNavigate();  const
[email,setEmail]=useState("");
const[password,setPassword]=useState("");  const
[errorMessage,ErrorMessage]=useState("");
```

```

const handleSubmit = async (event) => {      event.preventDefault();
    const userDto= JSON.stringify({      email,      password
    });
    console.log(userDto);

    try {
        const response = await axios.post("http://localhost:8081/api/v1/user/signIn",
userDto,{      headers: {
            "Content-Type": "application/json",
        }
    });

        //To check Login successful or not      if (response.status === 200) {
window.alert(response.data);      // console.log(response.headers);
localStorage.setItem("token", response.headers["access_token"]);
navigate('/');      } else {      alert("error in registration: ");
        }
    } catch (error) {      console.error("Error:", error);
    }
};

const handleGoogleSignIn = () => {      try {
const      signInURL      =
"https://accounts.google.com/o/oauth2/v2/auth?redirect_uri=http%3A%2F%2Flocalhost%3A8081%2Fapi%2Fv1%2Fuser%2FsignInWithGoogle&response_type=code&client_id=182116137767-32mo5dvdr9gc7qihacermg15dsqipph.apps.googleusercontent.com&scope=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.email+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fuserinfo.profile+openid&access_type=offline";      const
signInWindow = window.open(
        signInURL,
        "google login",
        "toolbar=no, menubar=no, width=700, height=700, top=100, left=300"
    );

        if (signInWindow) {      const interval = setInterval(() => {      if
(signInWindow.closed) {      clearInterval(interval);
navigate('/about');

```

```
        }  
      }, 1000);  
    }  
  
    } catch (error) {  
      console.error('Error during Google sign-in:', error);  
      window.alert('An error occurred during Google sign-in. Please try again.');
```

2. User Profile Management

In this project second faced challenges in ensuring that the user profile management fetched the correct details entered during the registration phase, specifically linking the phone number automatically without requiring redundant inputs. Here's a brief outline of how you overcame this challenge:

Challenge:

- Fetching user details dynamically from the registration data and automatically linking the phone number to avoid redundant inputs.

Solution:

- By leveraging state management, you persisted user data effectively across different components, allowing the profile page to dynamically fetch and display the correct user details (such as phone number) without requiring the user to re-enter the data manually. This included efficiently passing user data between the registration and profile components, maintaining a smooth user experience.
- The integration of axios for fetching and updating data and useState for managing form input changes were key in solving this issue. Additionally, the user profile was dynamically updated when necessary, such as when editing user details.

```
Code Snippet: const Userprofile = ()
=> {  const [auth, updateAuth] =
useAuth();

  const url = import.meta.env.VITE_ONEBILLING_URL;
const [user, setUser] = useState({  fullName: "",
email: "",  phoneNumber: "",  profilePic: null,
}); // Initially null to handle loading state
const [isEditing, setIsEditing] = useState(false);
const [formData, setFormData] = useState({
fullName: "",  email: "",  phoneNumber: "",
profilePic: "",
});
const [loading, setLoading] = useState(true); // Handle loading state  useEffect(()
=> {

  // Fetch user data from the backend
const fetchUserData = async () => {  try {
const response = await
axios.get(`${url}/user/${auth?.email}`, {
headers: {
  Authorization: auth?.token,
},
});
const  data  =  response.data;
setUser(data);

  setLoading(false); // Stop loading once the data is fetched
} catch (error) {
```

```
        console.error("Error fetching user data:", error);
    }
    setLoading(false);
};
```

```
    fetchUserData();
}, []);
```

```
const handleEdit = () => {
    setIsEditing(true);
};
```

```
const handleCancel = () => {
    setIsEditing(false);
    setFormData(user); // Reset form data on cancel
};
```

```
const handleChange = (e) => {
    setFormData({
        ...formData,
        [e.target.name]: e.target.value,
    });
};
```

```
const handleSubmit = async (e) => {
    e.preventDefault();
```

```
try
{
  // Send updated user data to the server

  const response = await axios.put(`${url}/user/${auth?.userId}`,formData, {
headers: {
  "Content-Type": "application/json",
  Authorization: auth?.token
},
}
);

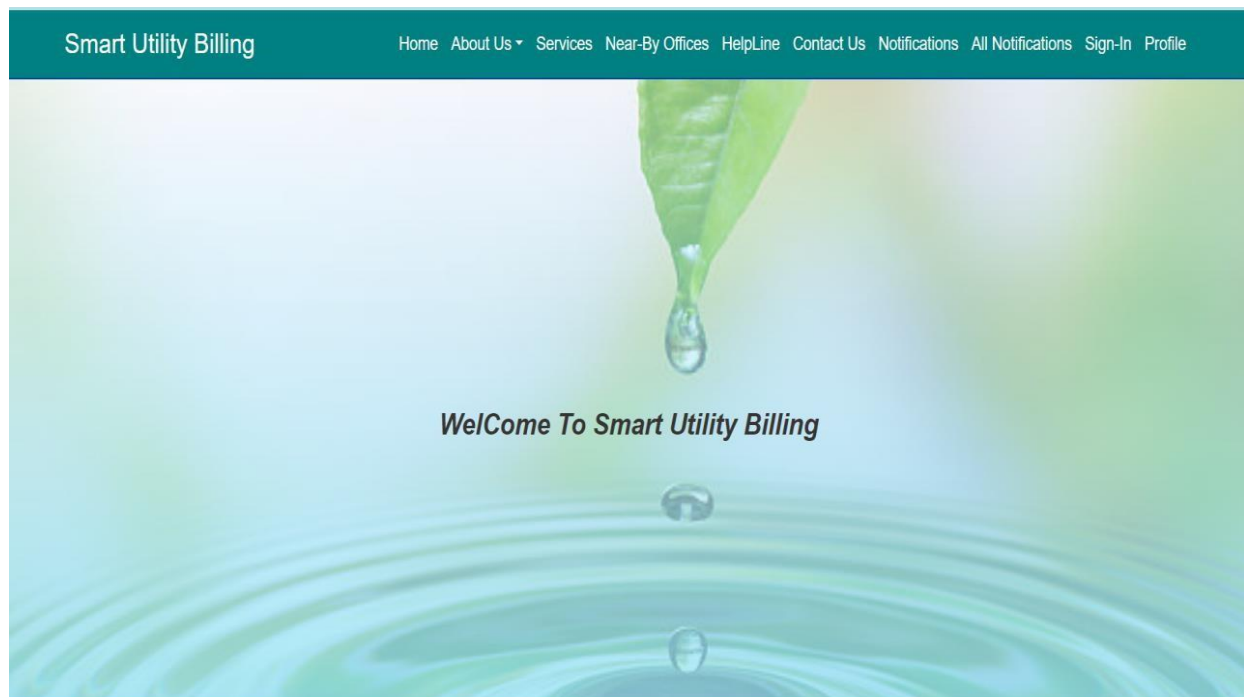
if (response.ok) {
  setUser(formData); // Update the user state with new data

  setIsEditing(false);
} else {
  console.error("Error updating
user profile");
}
} catch (error) {
  console.error("Error
updating profile:", error);
}
};

if (loading) { return <div className="vh-100">Loading...</div>; // Show loading state while
fetching data
}
```

Output Screenshots:

Home Page:



About Smart Utility Billing Page:



Vision Mission Page:

Smart Utility Billing

Home About Us ▾ Services Near-By Offices HelpLine Contact Us Notifications All Notifications Sign-In Profile

Vision & Mission

Vision

"The vision of the One Billing Service Portal is to create an all-inclusive, seamless platform that bridges the gap between citizens and essential services, promoting ease of access, transparency, and efficiency. We aim to empower individuals and businesses with a unified, user-friendly digital experience that enhances service delivery, utilizing cutting-edge technology to transform interactions with government and private entities."

Mission

"The Mission of the One Billing project is to be '**One Click Near To Citizens**'."

ONE BILLING
To provide a single interface for
anytime, anywhere citizen-centric


ABOUT US
Home Page

OTHER PAGES
Sign Up

DIRECT CONTACT
Hubli-Dharwad, Karnataka
560006

Services Page:


Services Offered



Electricity Bill

Manage and pay your electricity bills quickly and easily.

[View Bill](#)



Water Bill

Manage and pay your water bills quickly and easily.

[View Bill](#)

20 September 2024
Fri 04:14 PM (Local time)

Electricity Page:

Smart Utility Billing

[Home](#) [About Us](#) [Services](#) [Near-By Offices](#) [HelpLine](#) [Contact Us](#) [Notifications](#) [All Notifications](#) [Sign-In](#) [Profile](#)

Electricity Bill

RR ID

Account ID

Generate Bill

Go Back

Electricity Bill page:



Hubli Electricity Bill

Phone: [Phone Number] | Email: [Email] | Website: [Website]

User Details

Name: Anusha Sid

Email: anushasid9900@gmail.com

Phone: 6595089268_RANDOM

Area:

City:

Pincode:

Bill Details

Bill ID: 52

RR ID: 1234

Service: Electricity Bill

Status: Pending

Issued Date: 29/08/2024

Due Date: 25/09/2024

Total Amount: ₹500.00

Pay Now

Go Back

Payment Locations Page:

Available Locations to Pay Bill

Center Working Time: 08:00 AM - 07:00 PM

Available Payment Modes: Cash, Cheque, DD, Credit/Debit Card, UPI.

Paymodes may vary from Service to Service. Please contact the respective center for further details.

Enter your pincode here

Navanagar

Address: navanagar

Phone Number: 98785499

Pincode: 580021

Map:
<https://maps.app.goo.gl/f4KWt1cFe56QUUux5>

Vidyanagar

Address: Vidyanagar

Phone Number: 953123121

Pincode: 580020

Map:
<https://maps.app.goo.gl/f4KWt1cFe56QUUux5>

Main offices of Utility Billing Page:

Smart Utility Billing

Home

About Us

Services

Near-By Offices

HelpLine

Contact Us

Notifications

All Notifications

Sign-In

Profile

HUBLI ELECTRICITY OFFICE

Address:

Electricity Department,
Hubli, Karnataka, 580020

Phone: 0836-2213888 / 0836-2213869

Email: hubli.electricity@example.com

HUBLI WATER OFFICE

Address:

Water Supply Department,
Hubli, Karnataka, 580021

Phone: 0836-2233840 / 0836-2233850

Email: hubli.water@example.com

For any other inquiries or issues, please contact us through the respective departments. Our team will be happy to assist you.

Dept of MCA


KLE Technological University, Hubballi

Page 43

Contact Us Page:

Smart Utility Billing

Home About Us Services Near-By Offices HelpLine Contact Us Notifications All Notifications Sign-In Profile



Contact Us


If you have any questions, feel free to reach out to us!

Full Name
Your Name

Email
Your Email

Subject
Your Subject

Message
Your Message

☐ I'm not a robot 

Send Message

Notification Page:

Smart Utility Billing

Home About Us Services Near-By Offices HelpLine Contact Us Notifications All Notifications Sign-In Profile

All Notifications

Water supply alert



There will be some issues with water supply today in hubli vidyanagar





Aug 29, 2024, 12:34:36 PM

ONE BILLING
To provide a single interface for anytime, anywhere citizen-centric services of the government and private businesses in an integrated, convenient, fair, effective, secure, sustainable, and citizen-friendly manner.

ABOUT US
Home Page
Services Offered
Contact Us

OTHER PAGES
Sign Up
Help
Map
Notifications
Website Policies

DIRECT CONTACT
Hubli-Dharwad, Karnataka
0836-2213888/2213869/2213880/2213889
DC Office Dharwad
0836-2233840
anushasid9900@gmail.com
 

Our website is best viewed in:
   

Last reviewed and updated on 18 Aug, 2024

Login Page:

Smart Utility Billing

Home About Us ▾ Services Near-By Offices HelpLine Contact Us Notifications All Notifications Sign-In Profile

Login

Enter a valid email

Enter password


Login

Forgotten password?

Or


G Sign in with Google

Don't have an account? [Register](#)



Profile Page:

Profile Overview



Anusha Sid

Email : anushasid9900@gmail.com

Mobile : 6595089268_RANDOM

Edit Profile

Transaction History & Paid Bills

No transactions found.

Chapter 7 TEST CASES AND RESULTS

Test Plan:

- i Objective: The objective of testing is to ensure the functionality and reliability of the *One Digital* service portal.
- ii Scope: Testing will cover all critical functionalities, including user profile management, service booking, payment integrations, and admin functionalities.
- iii Resources: Testing will be conducted using the Jest testing framework and React Testing Library. Testing environments will include local development and deployment environments.
- iv Schedule: Testing will be conducted iteratively during the development phase and before deployment to ensure all functionalities meet the specified requirements.
- v Testing Environment: Testing will be performed on both development and production environments to ensure compatibility, stability, and reliability across different scenarios.

Test Cases:

1.Unit Testing:

- Definition: Focuses on testing individual components or units of the code, such as functions or methods, to ensure they work as expected in isolation.
- Purpose: Validate that each unit of the software performs as designed.
- Example: Testing if the `handleSubmit` function in the *One Digital* user registration process correctly processes user input and sends it to the backend API.

2.Integration Testing:

- Definition: Ensures that different modules or services used by the application work seamlessly together.
- Purpose: Identify issues that occur when units are combined and tested as a group.

- Example: Testing the integration of the Userprofile component with the user registration data to ensure the user's phone number is correctly fetched and displayed without redundant inputs.

3.System Testing:

- Definition: Validates the complete and integrated software product to ensure it meets the overall system requirements.
- Purpose: Evaluate the system's compliance with the specified requirements.
- Example: Testing the complete user journey on *One Digital*, from user registration to service booking and payment completion, ensuring smooth interaction and error handling.

4.Acceptance Testing:

- Definition: Conducted to determine if the system satisfies the business requirements and is acceptable for deployment.
- Purpose: Validate the system in real-world conditions as per client or stakeholder requirements.
- Example: Testing the *One Digital* portal to verify that users can successfully book services, make payments, and generate invoices, meeting stakeholder expectations for functionality and usability.

Acceptance Test Plan:

An Acceptance Test Plan defines the approach, scope, and criteria for acceptance testing. It details the processes to be followed, tools used, roles and responsibilities, and the acceptance criteria.

Components of an Acceptance Test Plan:

- Objective: Ensure the system meets business requirements and is ready for deployment.
- Scope: Test key features including user registration, service booking, payments, and profile management.
- Entry Criteria: System is fully functional, integrated, and all major defects from previous testing phases are resolved.
- Exit Criteria: No critical defects, key functionalities work as expected, and the system meets performance criteria.
- Responsibilities: QA team for testing, development team for fixes, stakeholders for approval.
- Test Environment: Latest build, server infrastructure, and testing tools (Mocha, Chai).
- Acceptance Criteria: Successful user registration, service booking, payment processing, and matching without major defects.

Unit Test Plan & Test Cases:**Objective:**

The objective of unit testing is to verify that individual functions, methods, or components within the Smart Utility Billing as expected. Each unit test focuses on a small piece of code, such as a function or method, ensuring that it performs correctly in isolation.

Scope:

- The test plan will cover the major functional components of the Smart Utility Billing project.
- Each function, method, or component will be tested for expected behaviour with both valid and invalid inputs.
- Areas to be covered include:
 - Registration form validation (e.g., Phone number, Email, Password validation)
 - Admin operations

Unit Test Cases for Smart Utility Billing:

Component 1: Phone Number, Email and Password Validation in Registration.

Objective: Ensure that the phone number field only accepts valid inputs (10 digits), email format and Password.

Test Case ID	Test Description	Input Data	Expected Output	Actual Output	Pass/Fail
UT-001	Validate phone number with valid input	9876543210	Phone number is accepted and no error message is shown.	Phone number accepted without errors.	Pass
UT-002	Validate phone number with less than 10 digits	98765	Error message "Please enter a valid 10-digit phone number."	Error message shown.	Pass
UT-003	Validate phone number with non-numeric input	abcd12345	Error message "Please enter a valid 10-digit phone number."	Error message shown.	Pass
UT-004	Validate Password with digit, alphabet and special char input	Xyz@12345	Error message "Please enter a valid Password."	Error message shown.	Pass
UT-005	Validate Confirm password input	abcd@12345	Error message "Mismatch password retry".	Error message shown.	Pass

Component 2: Admin Operations (User Management)**Objective:** Ensure that admin actions like deactivating a user work as expected.

Test Case ID	Test Description	Input Data	Expected Output	Actual Output	Pass/Fail
UT-006	Add new Locations.	User ID: 456	“New locations updated successfully”.	User deactivated successfully.	Pass
UT-007	View all the users.	User ID: 999	“Users viewed successfully”	Error message shown.	Pass
UT-008	View Admin profile.	User ID: 456	“View successfully”	Successful	Pass
UT-009	Receive Queries from users.	User ID: 456	“Received successfully”	Successful	Pass
UT-010	Add new notifications	Public/ User ID: 456	“Successfully Added”	Notification successfully.	Pass

Chapter 8

FUTURE ENHANCEMENT

Mobile App Development

- Enhancement: Develop a dedicated mobile application for Android and iOS platforms, complementing the responsive website.
- Benefit: The mobile app will offer users a seamless and more convenient experience, especially on mobile devices, allowing easier access to services on the go and improving overall user satisfaction.
- Service Provider Ratings and Reviews
- Enhancement: Introduce a feature that allows customers to rate and leave reviews for service providers after using their services.
- Benefit: This system will help new users make informed decisions based on real feedback and will motivate service providers to maintain high service standards, fostering a trustworthy environment.

Advanced Filtering Options

- Enhancement: Implement more sophisticated filtering options, such as by customer ratings, availability, and proximity, to fine-tune search results for service providers.
- Benefit: Users can quickly find service providers that best meet their preferences, improving search efficiency and satisfaction by narrowing down choices to the most relevant options.
- Benefit: This enhancement will drive user engagement and retention while encouraging word-of-mouth referrals, expanding the customer base and promoting a sense of community among users.

In-App Messaging System

- Enhancement: Introduce an in-app messaging system, enabling users to communicate directly with service providers to clarify service details.
- Benefit: This feature enhances trust and communication between users and providers, reducing miscommunication and improving the overall experience by allowing real-time discussions.

Analytics Dashboard for Service Providers

- Enhancement: Provide service providers with an analytics dashboard to monitor key performance indicators such as customer reviews.
- Benefit: Service providers will be empowered with insights into their performance, allowing them to track their progress and identify areas for improvement, ensuring long-term engagement with the platform.

Chapter 9

CONCLUSION

For the Smart Utility Billing project, the development of a comprehensive and user-friendly system has been successfully implemented using React for the front end and Spring Boot for the backend, with pgAdmin as the database management tool. The platform enables users to efficiently manage their electricity and water bills by providing functionalities such as viewing profiles, paying bills, downloading invoices, and accessing payment locations in Hubli. Additionally, users can raise queries to the admin and receive both public and personal notifications, enhancing their overall experience.

On the admin side, the system provides robust tools to manage users, locations, notifications, and view user profiles, ensuring seamless administration and operational efficiency. The integration of payment functionality, location services, and an easy-to-navigate interface caters to the needs of both end users and administrators.

The project showcases an effective utility billing solution that leverages modern technologies, providing a scalable and secure platform for utility management. Future enhancements, such as mobile app development and advanced analytics, will further enhance the platform's capabilities, ensuring its long-term viability and user satisfaction.

Chapter 10 BIBLIOGRAPHY

- i. <https://reactjs.org/docs/getting-started.html>
- ii. <https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/>
- iii. <https://www.pgadmin.org/docs/>
- iv. <https://razorpay.com/docs/>
- v. <https://www.baeldung.com/spring-boot-crud/>
- vi. <https://www.utilitybilling.com/blog/best-practices-for-designing-billing-systems>
- vii. <https://www.digitalocean.com/community/tutorials/react-form-handling>
- viii. <https://console.cloud.google.com/>
- ix. <https://www.karnatakaone.gov.in/Info/Public/SiteMap>

SmartBillingReport.docx

ORIGINALITY REPORT

18%

SIMILARITY INDEX

9%

INTERNET SOURCES

8%

PUBLICATIONS

14%

STUDENT PAPERS

PRIMARY SOURCES

1

Submitted to B.V. B College of Engineering
and Technology, Hubli

Student Paper

4%

2

Submitted to University of Greenwich

Student Paper

1%

3

Submitted to University of Houston Clear
Lake

Student Paper

1%

4

Submitted to Exeter College

Student Paper

1%

5

Submitted to University of Wales Institute,
Cardiff

Student Paper

1%

6

Sivaraj Selvaraj. "Mastering REST APIs",
Springer Science and Business Media LLC,
2024

Publication

1%

7

Submitted to Harrisburg University of Science
and Technology

Student Paper

<1%

