Design and Development of Electricity Bill Management System (EBMS)

by
Most. Fatema Tuj Zohra
ID: CSE1901016013

Supervised by
Prof. Bulbul Ahamed

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
May 2024

APPROVAL

The project titled "**Design and Development of Electricity Bill Management System (EBMS)**" submitted by Most. Fatema Tuj Zohra (CSE1901016013)) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board of Examiners

Prof. Bulbul Ahamed — **Supervisor**
Head
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 1**
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 2**
Department of Computer Science and Engineering
Sonargaon University (SU)

(Examiner Name and Signature) — **Examiner 3**
Department of Computer Science and Engineering
Sonargaon University (SU)

DECLARATION

I, hereby, declare that the work presented in this report is the outcome of the investigation performed by me under the supervision of **Prof. Bulbul Ahamed, Head,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, in Bangladesh. I reaffirm that no part of this project has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Prof. Bulbul Ahamed)
Supervisor

Signature

Most. Fatema Tuj Zohra
ID: CSE1901016013

ABSTRACT

The Design and Development of Electricity Bill Management System (EBMS) is a comprehensive software solution designed to streamline and automate the process of managing electricity bills for residential, commercial, and industrial consumers. The system aims to alleviate the complexities associated with monitoring electricity consumption, calculating bills accurately, and facilitating efficient payment processes. EBMS integrates advanced metering technologies, data analytics, and user-friendly interfaces to provide users with real-time insights into their electricity usage patterns. Through smart metering devices, the system captures consumption data at regular intervals, allowing consumers to monitor their energy usage on a granular level. Leveraging predictive analytics algorithms, EBMS forecasts future electricity demands, enabling proactive energy management strategies to optimize consumption and minimize costs.Moreover, EBMS offers robust billing functionalities, automating the generation of accurate invoices based on consumption data, tariff rates, and any applicable taxes or surcharges. The system supports various billing structures, including flat-rate, time-of-use, and demand-based pricing models, catering to the diverse needs of consumers. Additionally, EBMS facilitates secure online payment options, ensuring convenient and timely settlement of bills through electronic channels.

ACKNOWLEDGMENT

At the very beginning, I would like to express my deepest gratitude to Almighty Allah for giving me the ability and strength to finish the task successfully within the scheduled time.

I am auspicious that I had the kind association as well as supervision of **Bulbul Ahamed**, Head, Department of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out my project.

I would like to convey my special gratitude to **Brig. Gen. (Retd) Prof. Habibur Rahman Kamal, NDC, PSC,** Dean, Faculty of Science and Engineering for his kind concern and precious suggestions.

I am also thankful to all my teachers during my whole education, for exposing me to the beauty of learning.

Finally, my deepest gratitude and love to my parents for their support, encouragement, and endless love.

LIST OF ABBREVIATIONS

CDN	Content Delivery Network
CSS	Cascading Style Sheet
Docker	Containerization Platform
DNS	Domain Name System
DOM	Document Object Model
Git	Distributed Version Control System
HTML	HyperText Markup Language
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IDE	Integrated Development Environment
JWT	JSON Web Token
JSON	JavaScript Object Notation
JS	JavaScript
MERN	MongoDB, Express.js, React, Node.js
NoSQL	Not Only SQL
Npm	Node Package Manager
OOP	Object-Oriented Programming
OAuth	Open Authorization
ORM	Object-Relational Mapping
PaaS	Platform as a Service
REST	Representational State Transfer
SASS	Syntactically Awesome Stylesheets
SEO	Search Engine Optimization
SSO	Single Sign-On
SQL	Structured Query Language
SPA	Single Page Application
SSL	Secure Sockets Layer

TABLE OF CONTENTS

Title		Page No.
DECLARATION		iii
ABSTRACT		iv
ACKNOWLEDGEMENT		v
LIST OF ABBREVIATIONS		vi
CHAPTER 1		1 – 5
INTRODUCTION OF E-BILL		
1.1	Introduction	1
1.2	Objectives	1
1.3	Proposed	2-3
1.4	Benefits	3-4
1.5	Overview	4-5
CHAPTER 2		6-8
BACKGROUND STUDY		
2.1	Discussion With Client	6
2.2	Discussion With Supervisor	7
2.3	Web Search	7
2.4	Using MERN Stack	8
CHAPTER 3		9-11
METHODOLOGY		
3.1	Objective	9
3.2	System Analysis	9
3.3	Values, Principles	9-10
3.4	Reason of Using Agile Methodology	10-11
3.5	Advantages of Agile Method	11
3.6	Disadvantages of Agile	11
CHAPTER 4		12-15
SYSTEM DESIGN		
4.1	Requirement Specification	12-13
4.2	System Flowchart	14-15

CHAPTER 5 16-22
IMPLEMENTATION AND OUTCOME
5.1 Public View ……..…….. 16-18
5.2 Customer View ……..…….…… 18-19
5.3 Dashboard 20-22

CHAPTER 6 23-25
CONCLUSION, LIMITATIONS AND FUTURE WORKS
6.1 Conclusion ……..……. 23
6.2 Limitations ……..……..……. 23-24
6.3 Future Wokrs…..……..…… 24-25

REFERENCES ……..……. 26
APPENDIX …… 27-32

LIST OF FIGURES

Figure No.	Title	Page No.
Fig.2.1	Client Discussion	6
Fig.2.2	Google Search	7
Fig.2.3	MERN Stack	8
Fig.3.1	Agile Model	10
Fig.4.1	Node Js	12
Fig.4.2	VS Code	12
Fig.4.3	System Flowchart	14
Fig.4.4	Admin Flowchart	15
Fig.5.1	Home Page	16
Fig.5.2	About Page	17
Fig.5.3	Login Page	18
Fig.5.4	Pay Bill	18
Fig.5.5	My Bill	19
Fig.5.6	Bill Calculator	19
Fig.5.7	Pending Payment	20
Fig.5.8	Approved Payment	20
Fig.5.9	Rejected Payment	21
Fig.5.10	Set Unit Price	21
Fig.5.11	View Unit Price	22
Fig.5.12	Edit Unit Price	22

CHAPTER 1

INTRODUCTION OF E-Bill

1.1 Introduction

Welcome to the introduction of "e-bill" – an innovative Electricity Bill Management System (EBMS) project aimed at revolutionizing the way consumers and utility providers interact with electricity billing processes. In today's dynamic energy landscape, the efficient management of electricity consumption and billing procedures is paramount for both consumers and service providers.[1] "e-bill" emerges as a comprehensive solution to address these challenges, offering convenience, transparency, and sustainability in electricity billing management. The primary objective of the "e-bill" project is to develop a user-centric platform that simplifies the complexities associated with electricity billing, enhances transparency, and promotes energy efficiency. By leveraging cutting-edge technologies and robust software architecture, "e-bill" aims to empower consumers to monitor, analyze, and optimize their electricity usage while facilitating seamless billing and payment processes. The proliferation of smart metering infrastructure, coupled with advancements in data analytics and digital communication technologies, forms the backbone of the "e-bill" system. Through smart meter integration, "e-bill" enables real-time monitoring of electricity consumption, providing users with actionable insights into their energy usage patterns. This granular visibility empowers consumers to make informed decisions regarding their energy consumption habits, ultimately leading to cost savings and environmental sustainability. Moreover, "e-bill" offers a range of features designed to enhance the billing experience for consumers and utility providers alike. From automated billing generation based on consumption data to customizable tariff plans and secure online payment options, the platform ensures accuracy, efficiency, and convenience at every step of the billing cycle. Additionally, "e-bill" facilitates seamless communication between consumers and utility providers, fostering trust and collaboration in the energy ecosystem.[3] In essence, "e-bill" represents a paradigm shift in electricity billing management, embracing digitization, automation, and user empowerment to create a more sustainable and equitable energy future. By harnessing the power of technology, "e-bill" aims to redefine the relationship between consumers and electricity billing processes, paving the way for greater efficiency, transparency, and environmental stewardship in the energy sector.

1.2 Objectives

The objectives of "e-bill" encompass enhancing the user experience, promoting transparency and accountability in billing processes, optimizing costs, automating billing procedures, offering customization and flexibility in tariff plans, fostering energy conservation, ensuring secure payment transactions, integrating with smart grid infrastructure, facilitating efficient communication channels between consumers and utility providers, and striving for continuous improvement through feedback and technological advancements.

1.3 Purpose

The purpose of e-bill is to revolutionize the traditional approach to managing electricity bills by leveraging technology to create a more efficient, transparent, and user-friendly system. At its core, e-bill aims to transform the often cumbersome and complex process of electricity billing into a seamless and intuitive experience for consumers and utility providers alike.

By offering a user-centric platform, e-bill seeks to enhance the overall experience of managing electricity bills, making it easier for consumers to access relevant information, monitor their energy usage, and make informed decisions about their consumption habits. This emphasis on user experience is crucial in fostering trust and engagement with the platform.

Transparency and accountability are also central to the purpose of e-bill. By providing consumers with detailed insights into their energy usage patterns, billing calculations, and tariff structures, e-bill promotes transparency in billing processes, empowering consumers to understand and manage their electricity expenses effectively. Moreover, by automating billing procedures, e-bill minimizes the potential for errors and discrepancies, further enhancing transparency and trust. Cost optimization is another key objective of e-bill. Through real-time monitoring of energy consumption and personalized recommendations for energy efficiency improvements, e-bill helps consumers optimize their electricity costs, potentially leading to significant savings over time.[3] This focus on cost optimization not only benefits consumers but also contributes to the overall sustainability of the energy ecosystem. In addition to cost optimization, e-bill aims to encourage energy conservation and sustainability. By raising awareness about energy consumption patterns and providing consumers with tools and resources to reduce their environmental footprint, e-bill empowers individuals to make environmentally conscious choices and contribute to a more sustainable future.[5] Security is paramount in the design of e-bill. With the implementation of robust encryption protocols and secure payment gateways, e-bill ensures the security of payment transactions, protecting consumers' financial information and instilling confidence in the reliability of the platform. Furthermore, e-bill is designed to integrate seamlessly with smart grid infrastructure and IoT devices, leveraging advanced metering technologies to optimize energy distribution and management. By harnessing the power of smart grid technology, e-bill enhances grid reliability, supports renewable energy integration, and enables more efficient energy usage. Efficient communication channels are also a key aspect of e-bill. By facilitating seamless communication between consumers and utility providers, e-bill enables prompt resolution of billing queries, dissemination of important information, and fosters stronger relationships between stakeholders. Ultimately, the purpose of e-bill extends beyond mere billing management; it represents a commitment to innovation, sustainability, and empowering consumers to take control of their energy usage in a digital age. Through continuous improvement and innovation, e-bill strives to redefine the electricity billing experience and pave the way for a more efficient, transparent, and sustainable energy future.

1.4 Benefits

The benefits of e-bill are extensive and encompass various aspects of electricity billing management, both for consumers and utility providers:

1. Convenience: e-bill offers consumers the convenience of managing their electricity bills online, eliminating the need for paper-based invoices and physical visits to payment centers. Users can access their billing information anytime, anywhere, through a user-friendly digital platform.

2. Cost Savings: By providing real-time insights into energy consumption patterns and offering personalized recommendations for energy efficiency improvements, e-bill helps consumers optimize their electricity costs, potentially leading to significant savings over time.

3. Transparency: e-bill promotes transparency in electricity billing processes by providing consumers with detailed insights into their energy usage, billing calculations, and tariff structures. This transparency fosters trust and accountability between consumers and utility providers.

4. Accuracy: With automated billing procedures and advanced metering technologies, e-bill ensures the accuracy of billing calculations, minimizing the potential for errors and discrepancies in electricity bills.

5. Efficiency: e-bill streamlines billing processes, including bill generation, tariff calculation, and payment processing, reducing administrative overhead for both consumers and utility providers. This increased efficiency translates to cost savings and improved operational performance.

6. Environmental Sustainability: By encouraging energy conservation and promoting environmentally friendly practices, e-bill contributes to environmental sustainability efforts. By reducing energy consumption, e-bill helps minimize carbon emissions and mitigate the environmental impact of electricity usage.

7. Security: e-bill prioritizes the security of payment transactions, implementing robust encryption protocols and secure payment gateways to protect consumers' financial information. This ensures the integrity and confidentiality of sensitive data, instilling confidence in the reliability of the platform.

8. Customer Satisfaction: With its user-centric design and emphasis on convenience, transparency, and efficiency, e-bill enhances customer satisfaction and loyalty. By offering a seamless and intuitive billing experience, e-bill strengthens the relationship between consumers and utility providers.

9. Data-driven Insights: Through advanced analytics and reporting functionalities, e-bill provides consumers and utility providers with valuable data-driven insights into energy usage patterns, billing trends, and opportunities for optimization. This information empowers

stakeholders to make informed decisions and drive continuous improvement in energy management practices.

10. Integration with Smart Grid Infrastructure: e-bill integrates seamlessly with smart grid infrastructure and IoT devices, leveraging advanced metering technologies to optimize energy distribution and management. This integration enhances grid reliability, supports renewable energy integration, and enables more efficient energy usage.

The benefits of e-bill extend beyond mere billing management; they encompass convenience, cost savings, transparency, accuracy, efficiency, environmental sustainability, security, customer satisfaction, data-driven insights, and integration with smart grid infrastructure. By leveraging technology to enhance the electricity billing experience, e-bill delivers tangible value to both consumers and utility providers, ultimately contributing to a more sustainable and efficient energy future.

1.5 Overview

"e-bill" is a cutting-edge Electricity Bill Management System (EBMS) designed to revolutionize the way consumers and utility providers interact with electricity billing processes.[5] It represents a comprehensive solution that streamlines and modernizes the management of electricity bills, offering a range of features and benefits to enhance efficiency, transparency, and sustainability in billing management.

At its core, "e-bill" aims to simplify the complexities associated with electricity billing, providing consumers with a user-friendly platform to manage their bills conveniently and effectively. Through its intuitive interface, users can access their billing information, monitor their energy usage, and make payments securely from anywhere, at any time.

One of the key highlights of "e-bill" is its emphasis on transparency and accountability. By providing consumers with detailed insights into their energy consumption patterns, billing calculations, and tariff structures, the system fosters trust and understanding between consumers and utility providers. This transparency not only enhances consumer confidence but also promotes responsible energy usage and billing practices.

Moreover, "e-bill" leverages advanced metering technologies and data analytics to offer personalized recommendations for energy efficiency improvements. By analyzing energy usage data in real-time, the system helps consumers optimize their electricity costs and reduce their environmental footprint, contributing to overall sustainability efforts.

In addition to its consumer-facing features, "e-bill" offers numerous benefits for utility providers as well. The system automates billing processes, including bill generation, tariff calculation, and payment processing, reducing administrative overhead and improving operational efficiency.[7] Furthermore, "e-bill" integrates seamlessly with smart grid infrastructure and IoT devices, optimizing energy distribution and management for enhanced grid reliability and performance.

Overall, "e-bill" represents a paradigm shift in electricity billing management, embracing digitization, automation, and user empowerment to create a more sustainable and equitable energy future. By providing consumers and utility providers with a modern, efficient, and transparent billing solution, "e-bill" aims to drive positive outcomes for all stakeholders involved in the energy ecosystem.

CHAPTER 2

BACKGROUND STUDY

2.1 Discussion With Client



Fig: 2.1 Client Discussion

Before making this project we had to discuss with some teachers that what is their requirement? Why they need this site, what are the feature they only want? Good client communication often involves actively listening to your customers to interpret their emotions and better cater to their needs Being transparent with a client helps facilitate trust Customers may appreciate learning more about the production process so they can shop consciously We did this thing only because there are so many Website at the market for sell but those are may have more feature that for the school of suburb area and those will cost them much more where they don't want to invest Client communication is important because it establishes and maintains trust between the client and the business When customers trust a brand, they may be more likely to remain loyal Open client communication can also help limit misunderstandings, lead to greater customer satisfaction and make clients more likely to recommend a business to others So this was crucial for us to take this project though this had been created before so many time Having adaptable solutions for communicating can strengthen the relationship between the client and us When a change occurs, flexibility helps both parties continue to communicate The clients are eager to implement this system at their schools for save time management if it costs in a budget range like 20 to 30 thousands BDT Also they want this thing in a very simple way that they can use So This step of work were fruitful for this project

2.2 Discussion With Supervisor

Effective communication with supervisor is a key element of your eventual success in the workplace Therefore, it is important that you feel comfortable and prepared to approach your supervisor in a timely manner to discuss concerns, request assistance, or report work progress As we have got an amazing supervisor to our group we had discuss with her about our simple concept She is good at communication She is clear to her speech, and never did underestimate us She offered us by brief discussion By that way we were able to take our decision Through some discussion we were able to keep this as our final proposal We didn't choose any thesis like project because. With all of the obstacle our supervisor **Prof. Bulbul Ahamed**, Head department of computer science and engineering of Sonargaon University helped us to do the at least job which is in front of us now she is really an amazing woman as she never scolded us rather she did inspire all of us to do at least basic of the job as this is our first project

2.3 Web Search

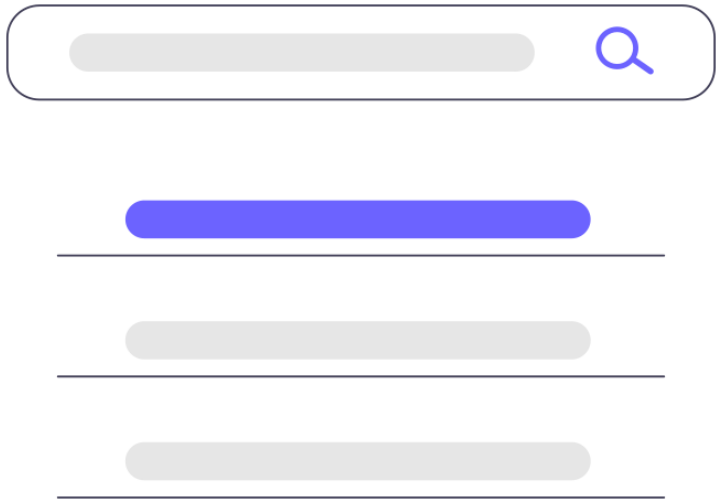


Fig: 2.2 Google Search

Searching through website is very important now a day In the web there is almost everything Every project build with three or more Technology There need more than one language and a server need to connect there So, as we mentioned earlier that we had the lack of knowledge to build a complete website So we had to see some website and inspected here code from browser After this we had to see some tutorials from youtube how they do the job for a project and read some content from some site.

2.4 Using MERN

The stack of the project is MERN we mainly use MongoDB ,Express Js,React Js,Node Js

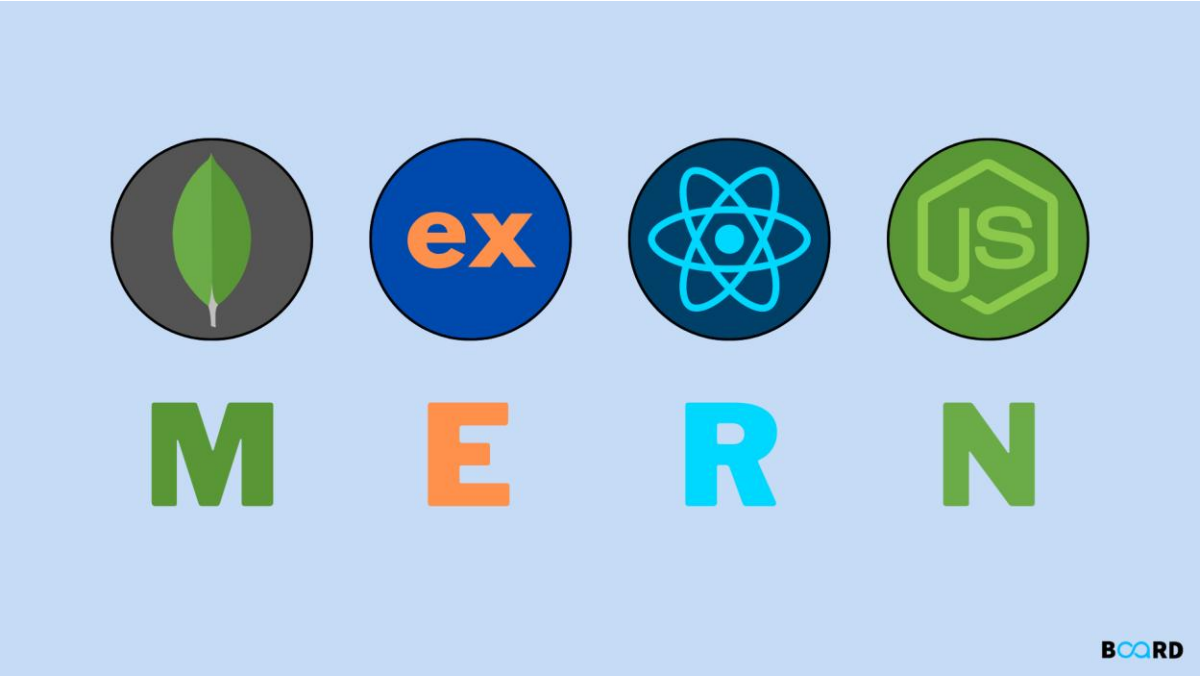


Fig: 2.3 MERN Stack

MEAN is a free and open-source JavaScript software stack for building dynamic web sites and web applications. A variation known as MERN replaces Angular with React. is a collection of technologies that enables faster application development. It is used by developers worldwide. The main purpose of using MERN stack is to develop apps using JavaScript only. This is because the four technologies that make up the technology stack are all JS-based. Thus, if one knows JavaScript (and JSON), the backend, frontend, and database can be operated easily. Together, these technologies provide a full-stack JavaScript solution for building modern web applications. MongoDB is used for data storage, Express.js for server-side logic and routing, React for building interactive user interfaces, and Node.js for server-side execution. This stack is particularly popular for its flexibility, performance, and the ability to develop both the front-end and back-end of an application using JavaScript, which can streamline development and maintenance processes.

CHAPTER 3

METHODOLOGY

3.1 Objective

Methodology refers to the overarching strategy and rationale of your research project It involves studying the methods used in your field and the theories or principles behind them, in order to develop an approach that matches your objectives The methodology is the general research strategy that outlines the way in which research is to be undertaken and, among other things, identifies the methods to be used in it These methods, described in the methodology, define the means or modes of data collection or, sometimes, how a specific result is to be calculated Methodology does not define specific methods, even though much attention is given to the nature and kinds of processes to be followed in a particular procedure or to attain an objective A methodology does not set out to provide solutions it is therefore, not the same as a method Instead, a methodology offers the theoretical underpinning for understanding which method, set of methods, or best practices can be applied to a specific case, for example, to calculate a specific result There are various types of methodology Online However, we choose to use AGILE method for our project Online Voting System

3.2 System Analysis

Systems analysis is a process of looking at end-user implementation of a software package or product; looking in-depth at source code to define the methodologies used in building software; or taking feasibility studies and other types of research to support the use and production of a software product, among other things methodologies used in building software; or taking feasibility studies and other types of research to support the use and production of a software product, among other things System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recorded improvements on the system It is a problem solving activity that requires intensive communication between the system user and system developer System analysis or study is an important phase of any system development process A detailed study of the process must be made by various techniques like interviews, questionnaires etc The data collected by these sources must be scrutinized to arrive to conclusion every synthesis is built upon the results of a preceding analysis and every analysis requires a subsequent

3.3 Values, Principles

There are some specific values and principles in Agile Method Those are given Below:

Values:

1. Working software over comprehensive documentation
2. Individuals and interactions over processes and tools
3. Customer collaboration over contract negotiation
4. Responding to change over following a plan

Principles:

1. Customer satisfaction through early and continuous software delivery

2. Accommodate changing requirements throughout the development process
3. Attention to technical detail and design enhances agility
4. Simplicity
5. Working software is the primary measure of progress
6. Agile processes to support a consistent development pace
7. Enable face-to-face interactions
8. Support, trust, and motivate the people involved
9. Frequent delivery of working software
10. Collaboration between the business stakeholders and developers throughout the project

Because of its addictiveness, Agile methodology is commonly used to deliver more complex projects It uses six main deliverables to track progress and create the product which are the product vision statement, product road map, product backlog, release plan, Sprint backlog, and increment With these features, it establishes itself as a methodology that places an emphasis on collaboration, flexibility, continuous improvement, and high quality results

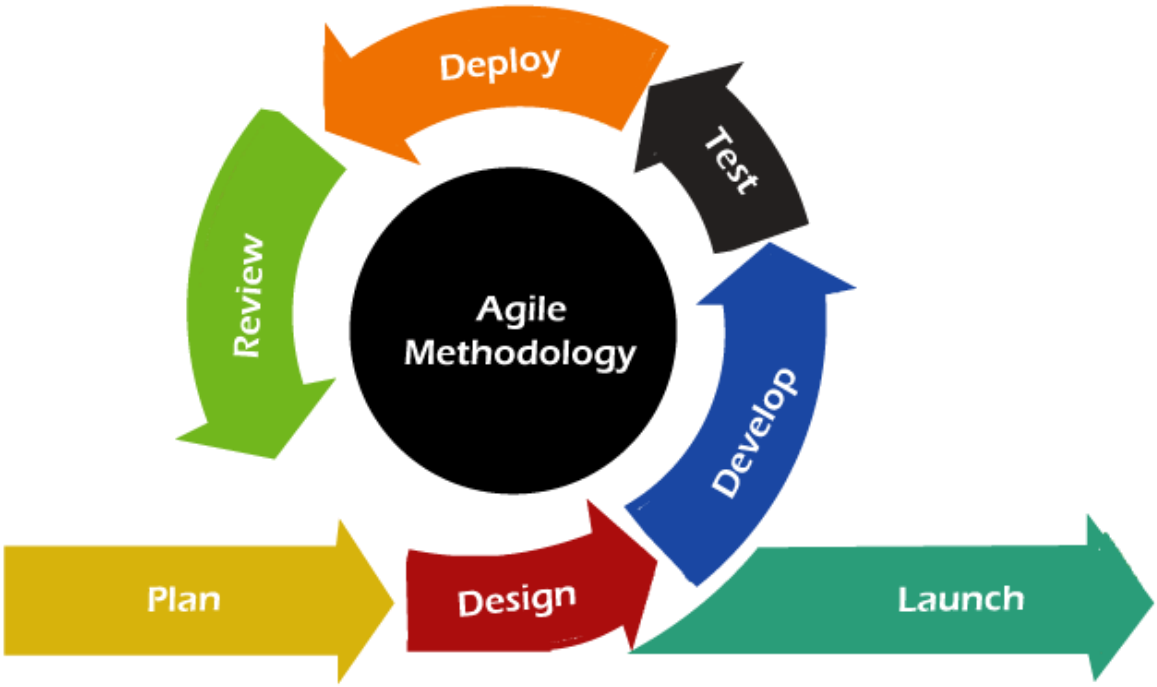


Fig: 3.1 Agile Methodology

This methodology works in a very simple way by breaking down the Project into smaller part and continuously delivering them in a cycle called iterations.

3.4 Reason of Using Agile Methodology

Software project management is a discipline where we may need to coordinate many various projects of different scope and deal with different customers who need different results Naturally, we will not be able to use the same management methods for delivering a basic application developed by a team of four and a complex software solution which needs

professionals from various departments Besides, most customers will require us to yield results along the project, so that they can see what kind of product they will get at the end To be a successful project manager, we need to master different software delivery methodologies Basically, there are two of them widely used in software development: Waterfall and Agile Agile Manifesto was put together back in 2001 by a group of developers utilizing different delivery methods Eleven years after, a study by the Standish group was conducted showing that Agile methods yield successful projects three times more often compared to the traditional Waterfall projects The gist of Agile trade is to continually develop shippable software pieces to our client Along the way, our client monitors progress, gives feedback and supervises direction of your work Unlike Waterfall methodology which takes a rigid, sequential structure, Agile methods are more flexible and involve constant cooperation with the client.

3.5 Advantages of Agile Method

Some Advantage of Agile method are given below:

1. More flexibility More productivity
2. More transparency
3. Products of superior quality
4. Decreased risk of missed goals
5. Faster implementation of solutions
6. Waste reduction thanks to the minimization of the resources
7. Greater flexibility and adaptability to change
8. More success thanks to more focused efforts
9. Faster delivery times
10. Faster detection of problems and defects; Optimized development processes
11. A lighter/less complicated structure
12. Excellent project control
13. Greater attention to specific customer need
14. Increased collaboration frequency and feedback

3.6 Disadvantages of Agile

Some disadvantages of Agile method are given below:

1. In case of some software deliverables, especially the large ones, it is difficult to assess the effort required at the beginning of the software development life cycle
2. There is lack of emphasis on necessary designing and documentation
3. The project can easily get taken off track if the customer representative is not clear what final outcome that they want
4. Only senior programmers are capable of taking the kind of decisions required during the development process Hence it has no place for newbie programmers, unless combined with experienced resources

CHAPTER 4

SYSTEM DESIGN

4.1 Requirement Specification

To create ReactJs project we have to install node js in our laptop/Desktop. After installing node js we have to install npx create react app.

Node js: nodejs is an open-source, cross-platform JavaScript runtime environment and library for running web applications outside the client's browser It is used for server-side programming, and primarily deployed for non-blocking, event-driven servers, such as traditional websites and back-end API services, but was originally designed with real-time, pushbased architectures in mind.



Fig: 4.1 Node JS

Visual Studio Code: After installing node js we have to install an IDE(Integrated Development Environment) we use visual studio code for this project.



Fig: 4.2 Visual Studio Code

To Run the Project:

[9] There need to install Node jsto confirm the installtion we have to write cmd in node --version comnmand then if it will show us the version we can confirm the install of node js

then write command npm install create-react-app after completing the react create-app we have to create new create react app the command of react are given below

1. Npx create-react-app appName
2. Cd appName
3. Npm start

Then it will open default port localhost:3000

now there need to enter following URL in browser installed in the pc

https://localhost:3000

we have built this project where we used:

1. Visual studio code
2. Css
3. Jsx
4. React js
5. Express js
6. Node js
7. Mongodb
8. Material-ui
9. Bootstrap
10. Javascript

4.2 System Flow

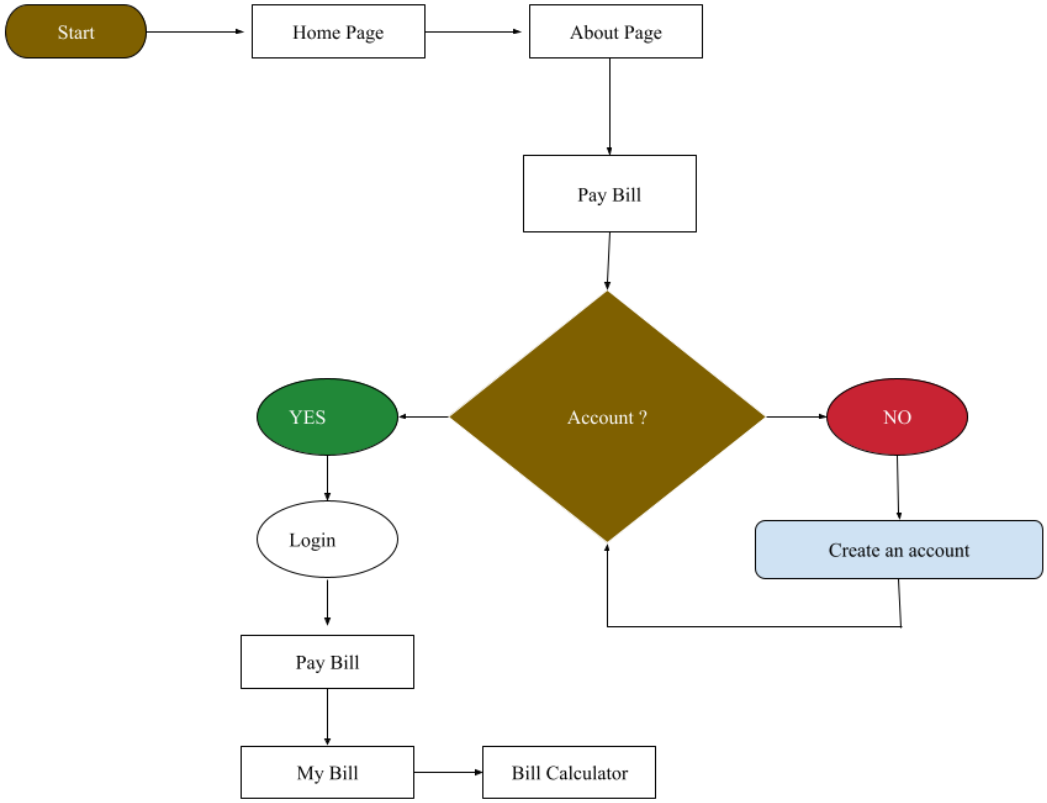


Fig: 4.3 User Flow

Admin Flow

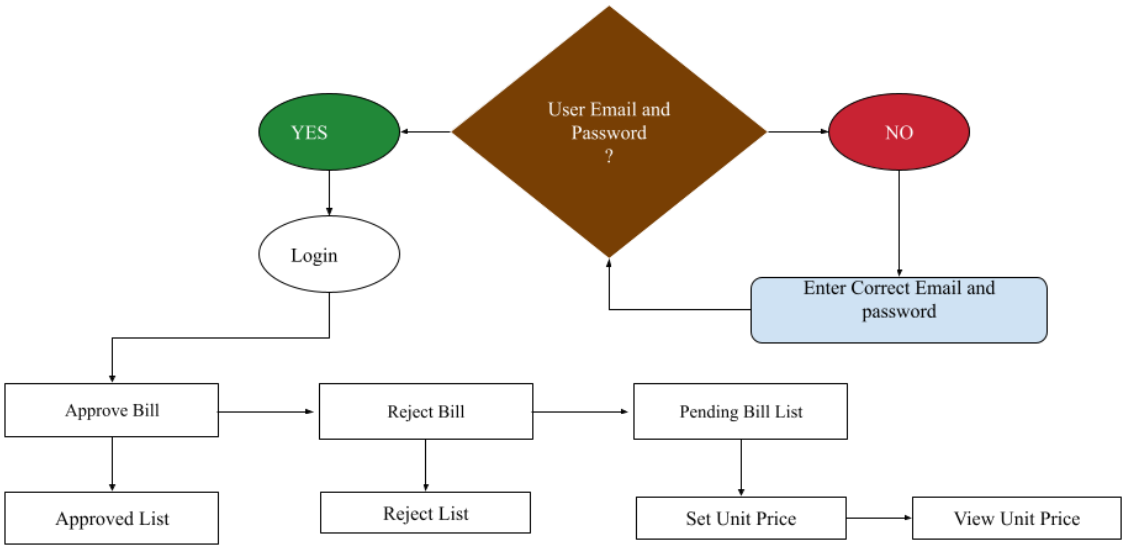


Fig: 4.4 Admin Flow

CHAPTER 5

IMPLEMENTATION AND OUTCOME

5.1 Public View

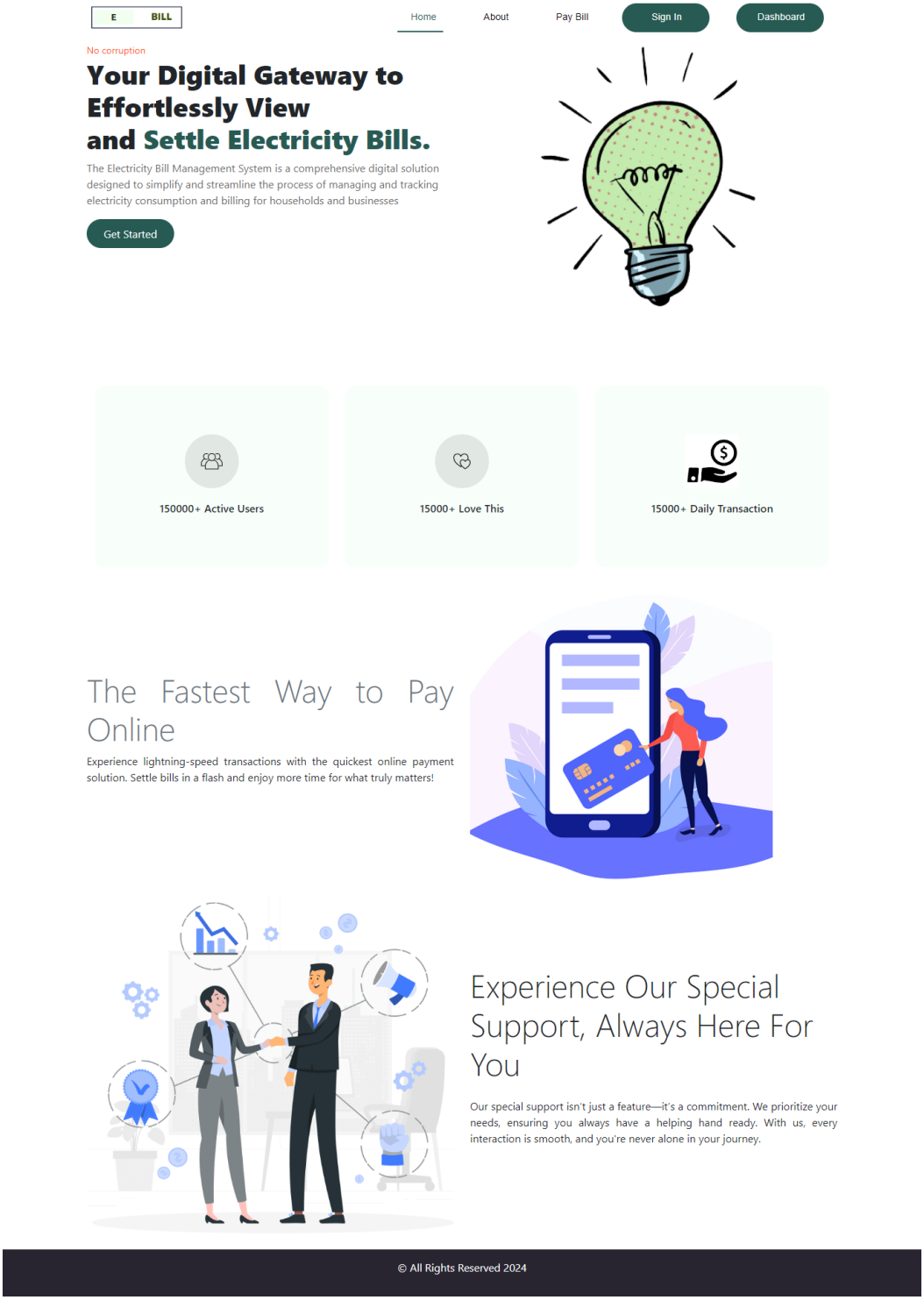


Fig: 5.1 Home Page

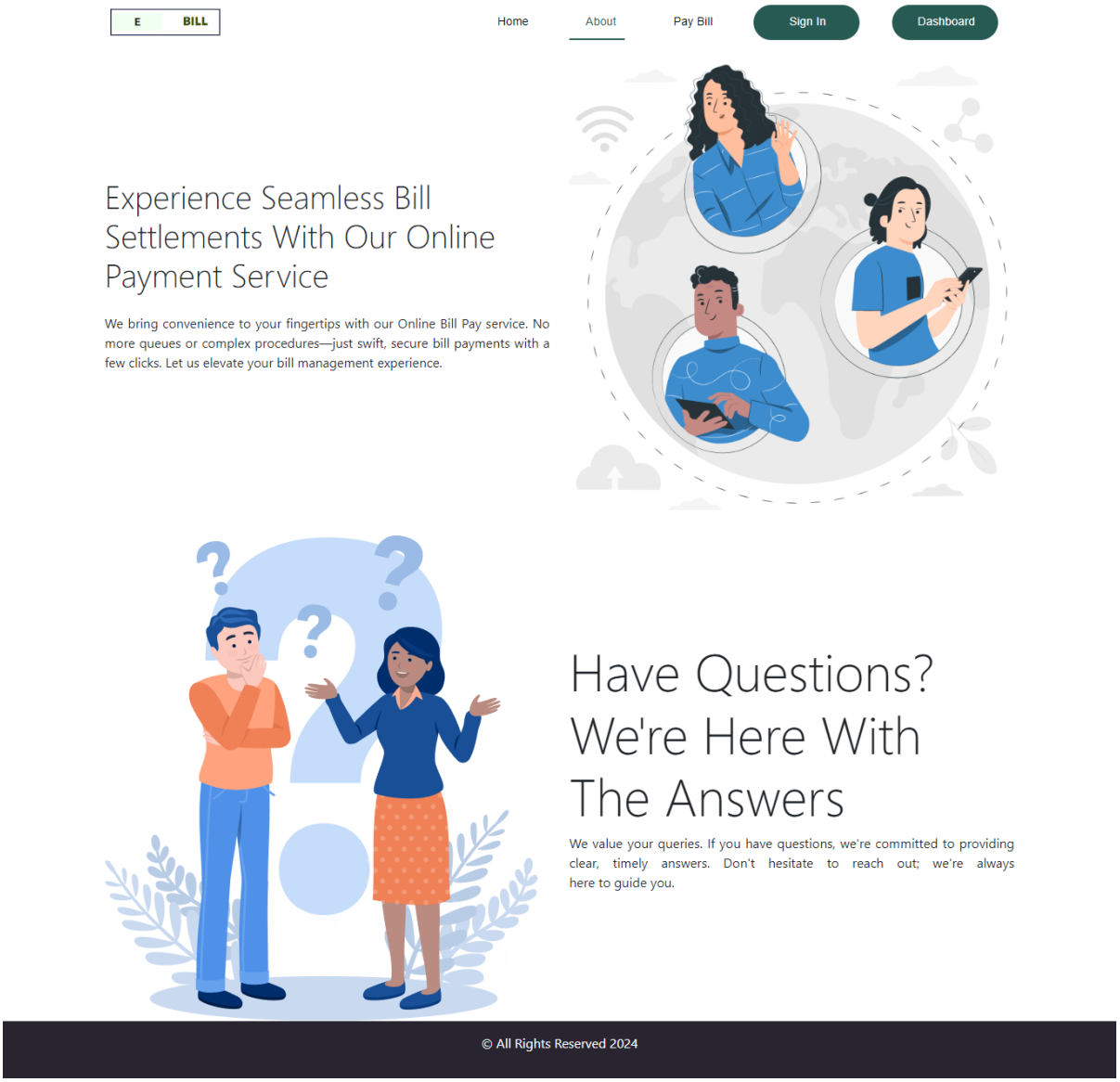


Fig: 5.2 About Page

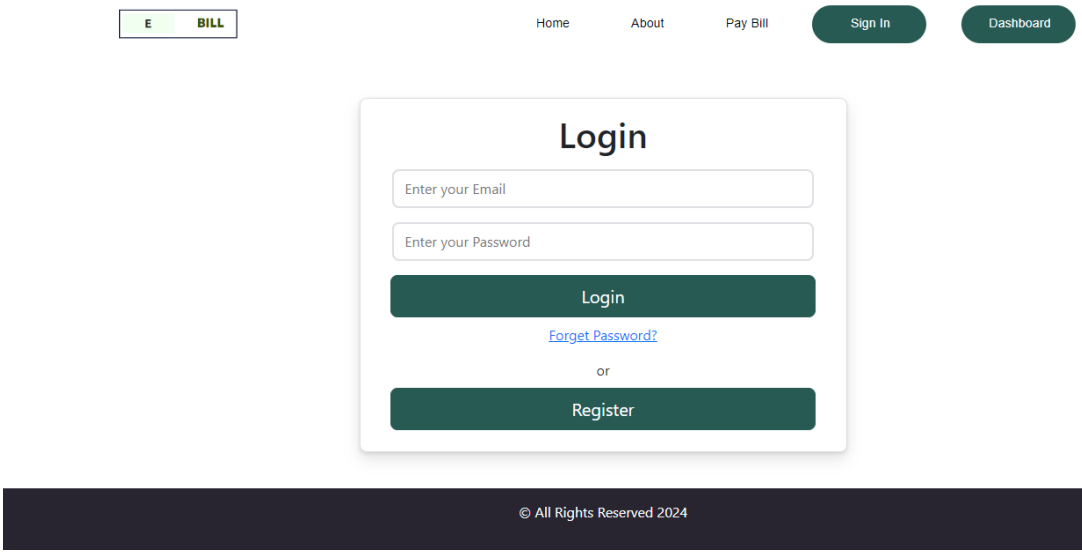


Fig: 5.3 Login Page

5.2 Customer View

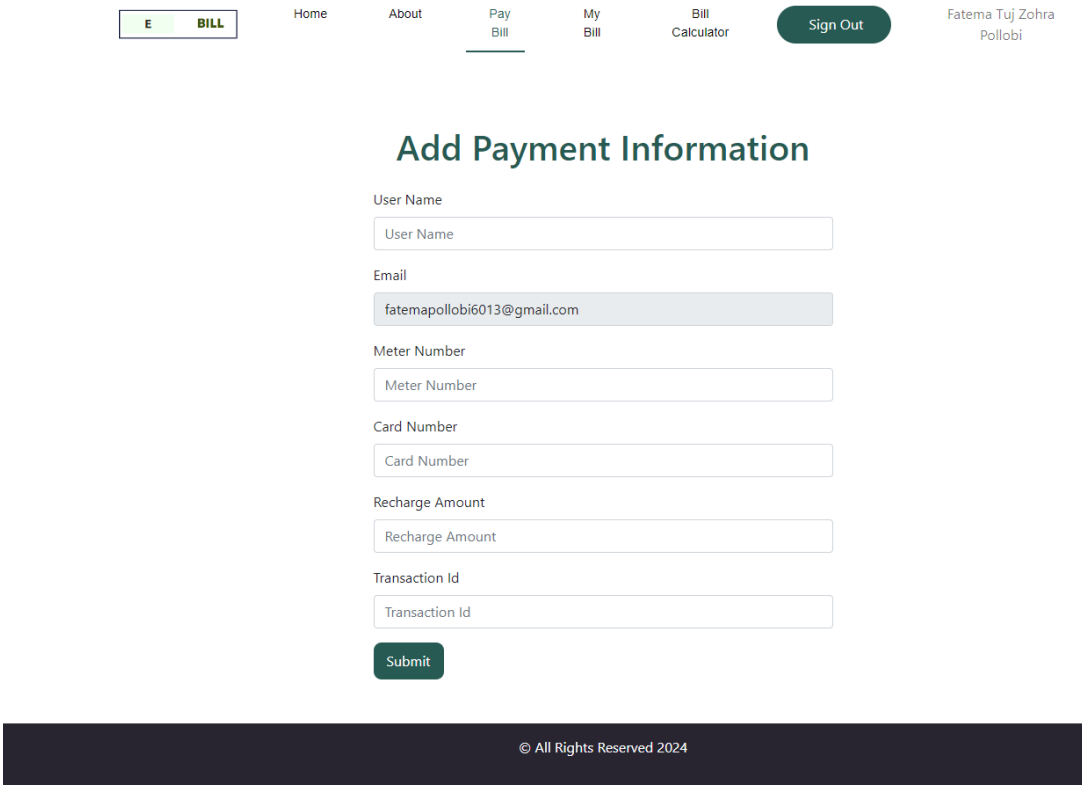


Fig: 5.4 Pay Bill

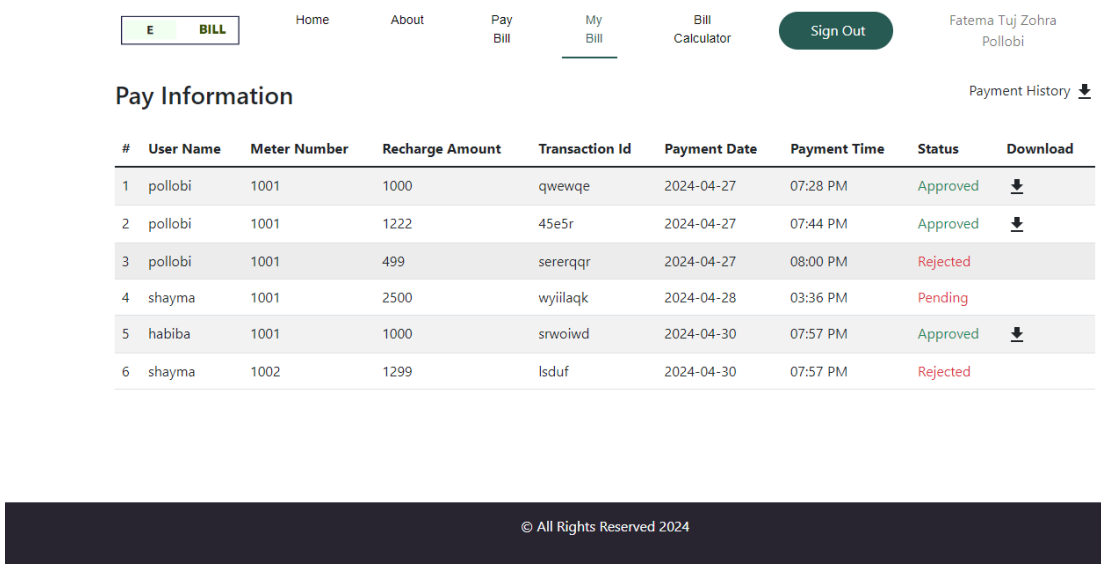


Fig: 5.5 My Bill

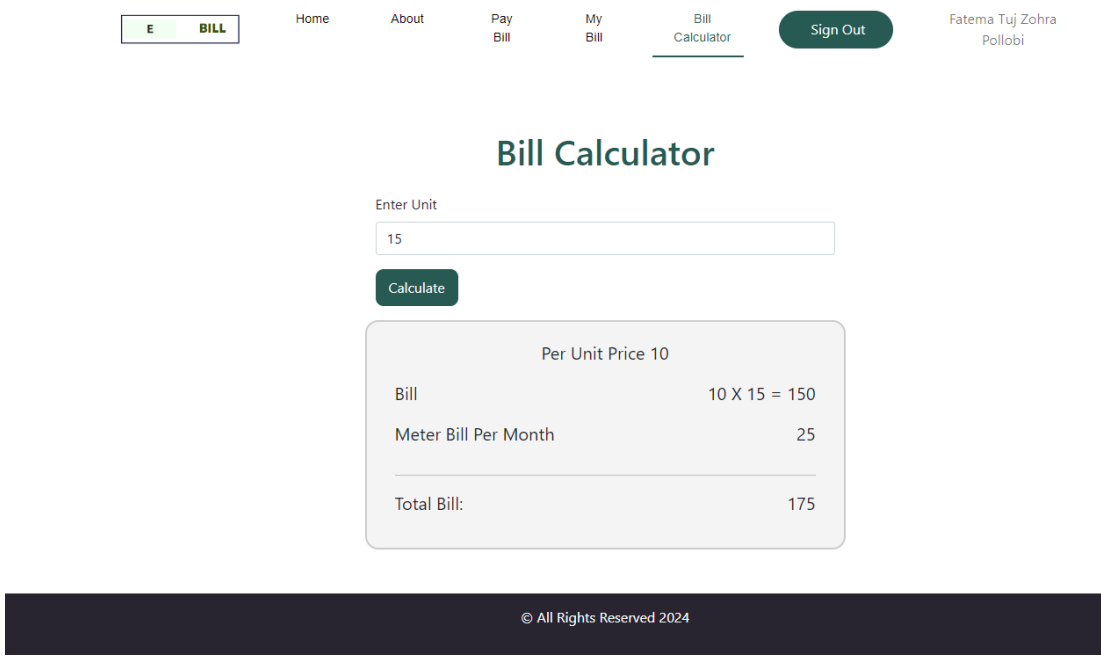


Fig: 5.6 Bill Calculator

5.3 Admin View

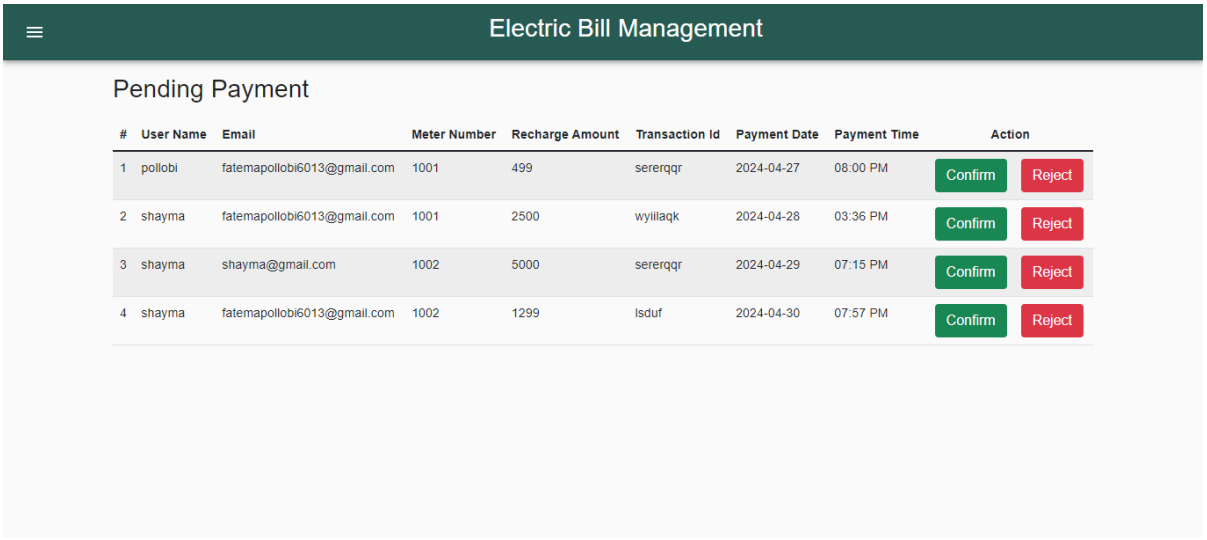


Fig: 5.7 Pending Payment

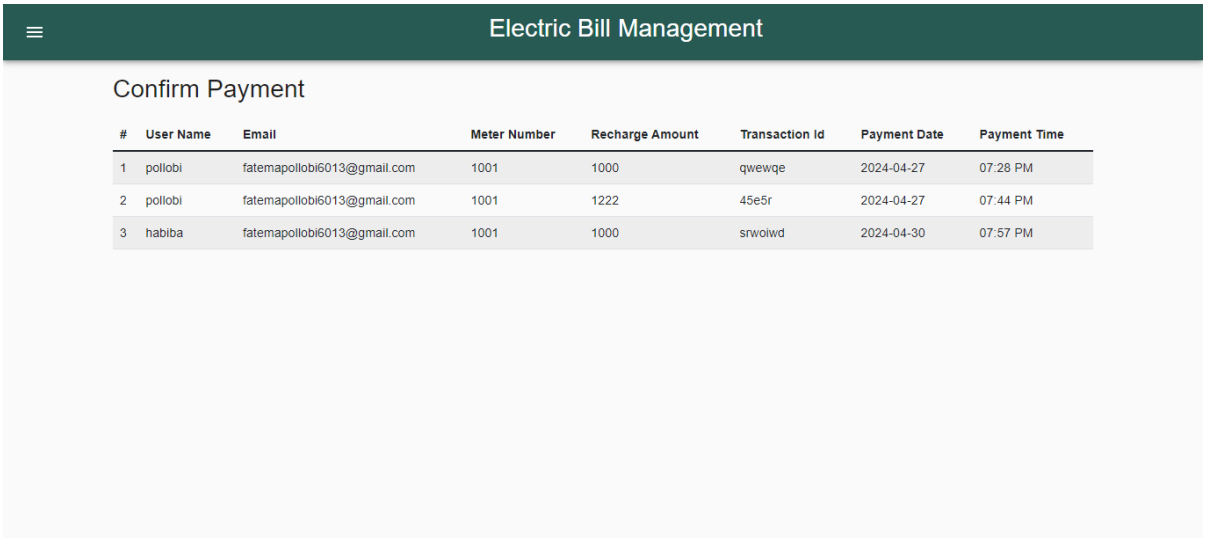


Fig: 5.8 Approved Payment

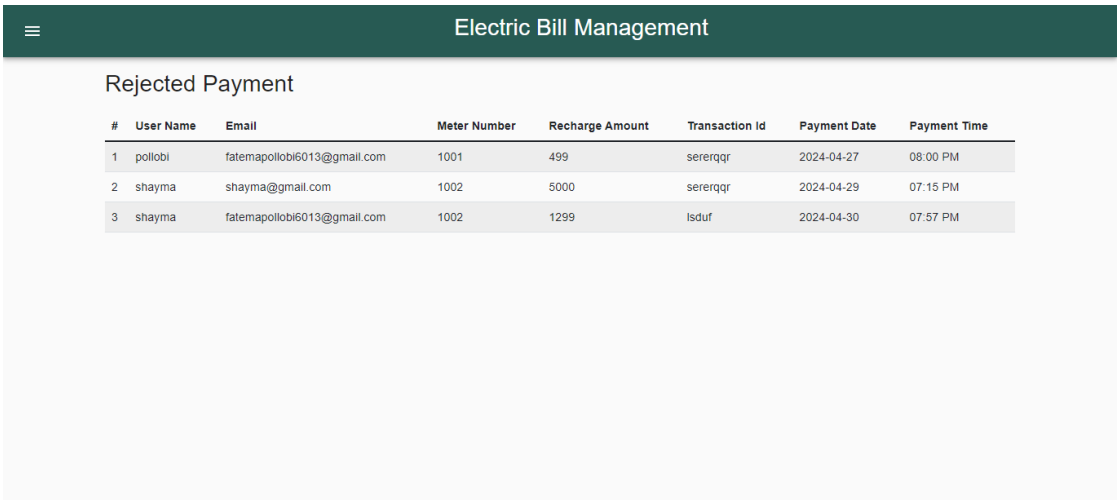


Fig: 5.9 Rejected Payment

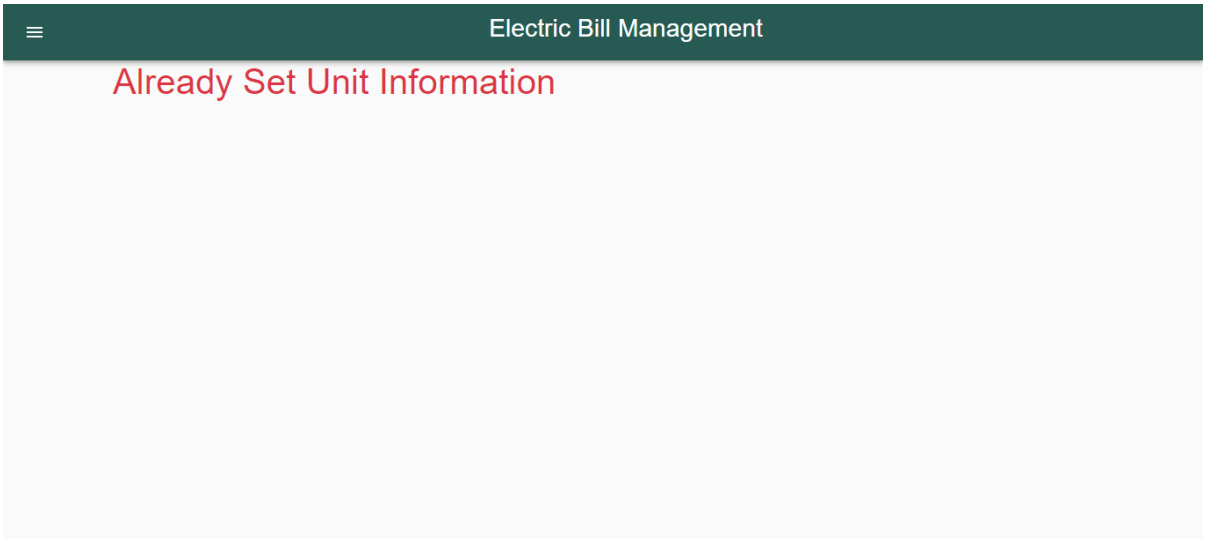


Fig: 5.10 Set Unit Price

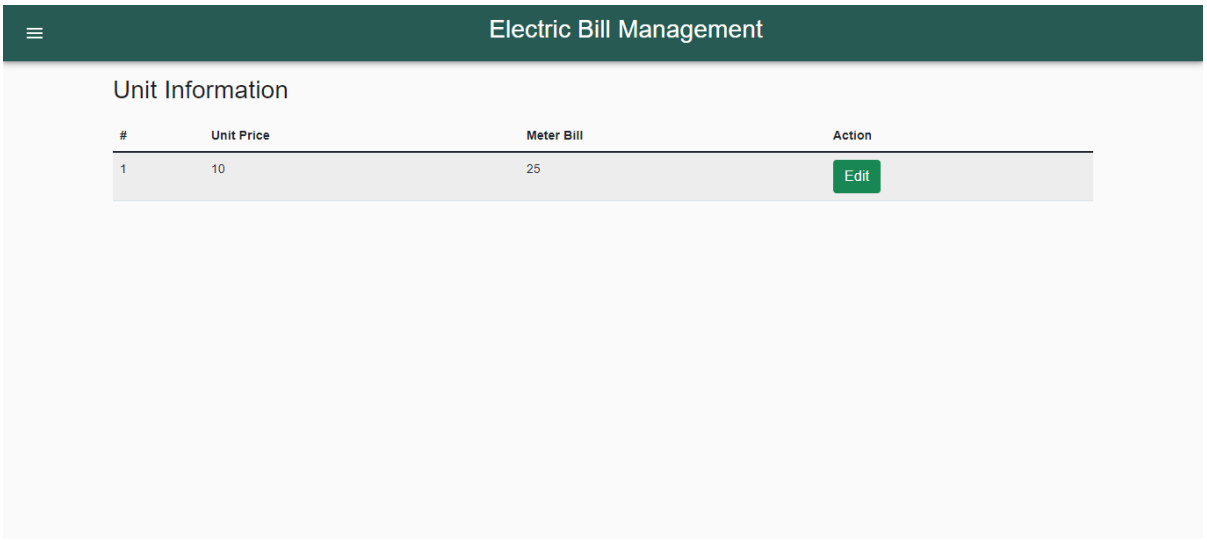


Fig: 5.11 View Unit Price

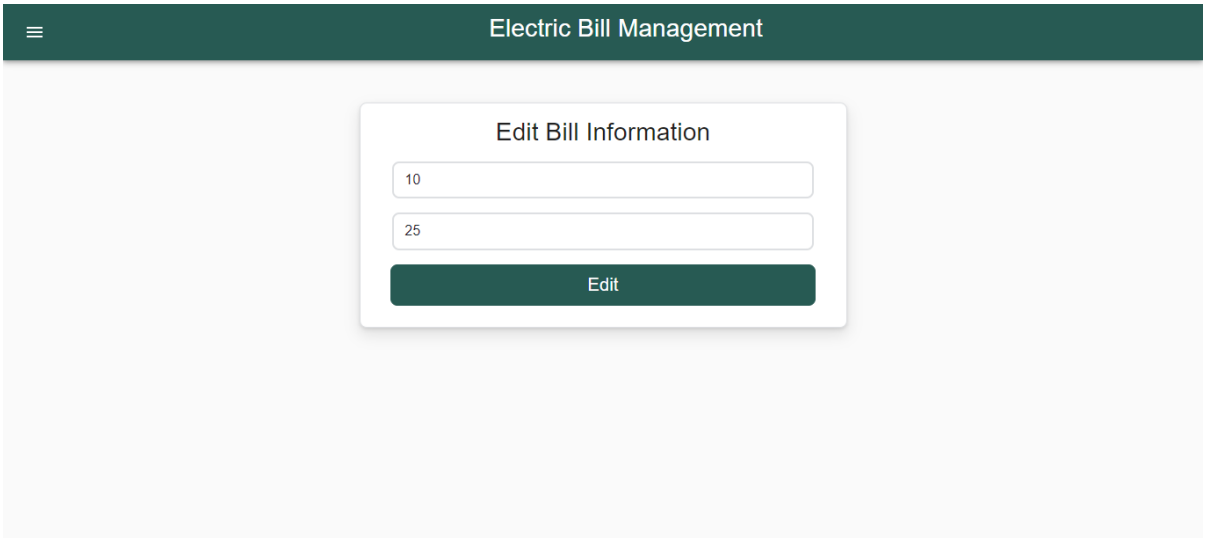


Fig: 5.12 Edit Unit Price

CHAPTER 6

CONCLUSION, LIMITATIONS AND FUTURE WORKS

6.1 Conclusion

In conclusion, "e-bill" represents a transformative solution in the realm of electricity billing management, offering a comprehensive platform that revolutionizes the way consumers and utility providers interact with billing processes. Through its emphasis on user empowerment, transparency, efficiency, and sustainability, "e-bill" not only simplifies the complexities of electricity billing but also drives positive outcomes for all stakeholders involved.

By providing consumers with real-time insights into their energy usage, personalized recommendations for efficiency improvements, and convenient payment options, "e-bill" enhances the overall billing experience, fostering greater trust, satisfaction, and accountability. Moreover, its integration with smart grid infrastructure, advanced analytics capabilities, and continuous development initiatives ensure that "e-bill" remains at the forefront of innovation, delivering value-added services and solutions to meet the evolving needs of users.

From the perspective of utility providers, "e-bill" streamlines administrative processes, reduces overhead costs, and enhances operational efficiency, ultimately leading to improved service delivery and customer satisfaction. Furthermore, its focus on environmental sustainability, through energy conservation initiatives and support for renewable energy sources, aligns with broader efforts to combat climate change and build a more sustainable energy future.

In summary, "e-bill" embodies the convergence of technology, sustainability, and consumer-centric design, offering a compelling vision for the future of electricity billing management. As it continues to evolve and expand its capabilities, "e-bill" holds the potential to drive significant positive impact across the energy ecosystem, empowering consumers, optimizing utility operations, and contributing to a more resilient and sustainable energy infrastructure.

6.2 Limitations

1. Data Privacy and Security: Ensuring the privacy and security of consumer data within the Electricity Bill Management System (EBMS) presents a significant challenge. Safeguarding sensitive information against cyber threats and unauthorized access requires robust encryption protocols and stringent security measures.

2. Infrastructure Compatibility: EBMS may encounter limitations in compatibility with existing infrastructure, particularly in regions with outdated or incompatible systems. Integration with legacy systems and interoperability challenges may need to be addressed to ensure seamless implementation.

3. Initial Setup and Training: Implementing EBMS requires initial setup and training for both consumers and utility providers. Ensuring user adoption and proficiency may pose challenges, requiring comprehensive training programs and ongoing support initiatives.

4. Regulatory Compliance: Compliance with regulatory requirements and industry standards presents another limitation. EBMS must adhere to regulatory frameworks governing data privacy, billing practices, and consumer rights, which may vary across jurisdictions.

5. Cost and Resource Allocation: Implementing EBMS involves significant costs and resource allocation for development, deployment, and maintenance. Limited financial

resources and competing priorities may pose challenges for organizations considering adoption.
6. Reliability and Performance: Maintaining the reliability and performance of EBMS under varying conditions is critical. System downtime, latency issues, and scalability constraints may impact user experience and operational efficiency.
7. User Adoption and Engagement: Encouraging user adoption and engagement is essential for the success of EBMS. Overcoming resistance to change, addressing user concerns, and fostering a culture of transparency and accountability may require targeted communication and outreach efforts.
8. Technological Advancements: Rapid technological advancements may render certain features or components of EBMS obsolete over time. Continuously monitoring and adapting to emerging trends and innovations is necessary to ensure the system remains relevant and effective.
9. Environmental Factors: Environmental factors such as natural disasters, extreme weather events, and infrastructure vulnerabilities may disrupt the operation of EBMS. Implementing contingency plans and disaster recovery strategies is essential to mitigate risks and ensure business continuity.
10. Consumer Awareness and Education: Limited consumer awareness and understanding of EBMS may hinder adoption rates. Educating consumers about the benefits, functionalities, and usage of EBMS is crucial for maximizing its potential impact and value.
Navigating these limitations requires careful planning, collaboration, and ongoing evaluation to address challenges effectively and maximize the benefits of EBMS for all stakeholders involved.

6.3 Future Works

As Electronic Bill Management Systems (EBMS) continue to evolve, there are several avenues for future development and improvement. These areas of focus aim to enhance user experience, integrate emerging technologies, expand renewable energy integration, and promote sustainability. Here are some potential directions for future work:

1. Enhanced User Experience: Further refinement of the user interface and experience of EBMS to ensure ease of use, accessibility, and engagement for all users.
2. Integration with Emerging Technologies: Exploration of integration with emerging technologies such as artificial intelligence (AI), machine learning (ML), and Internet of Things (IoT) devices to enable advanced features like predictive billing and smart energy management.
3. Blockchain for Security and Transparency: Investigation into the use of blockchain technology to enhance security, transparency, and traceability within EBMS, ensuring the integrity of billing data and transactions.
4. Expansion of Renewable Energy Integration: Expansion of the integration of EBMS with renewable energy sources such as solar and wind power, enabling users to optimize their renewable energy generation and consumption effectively.
5. Mobile Applications: Development of dedicated mobile applications for EBMS to provide users with on-the-go access to billing information, energy usage data, and payment options.
6. Advanced Analytics and Reporting: Enhancement of EBMS with advanced analytics and reporting capabilities to provide users with deeper insights into energy usage patterns, billing trends, and cost-saving opportunities.

7. Scalability and Interoperability: Continuous optimization of the scalability and interoperability of EBMS to support growing user bases, evolving technological standards, and regulatory requirements.
8. Customer Engagement Initiatives: Implementation of customer engagement initiatives within EBMS to educate, incentivize, and empower users to adopt energy-efficient behaviors.
9. Global Expansion and Localization: Expansion of EBMS into new markets and regions, adaptation to local regulatory frameworks and cultural norms, and empowerment of communities worldwide to manage their electronic bills effectively.
10. Partnerships and Collaborations: Exploration of partnerships and collaborations with other stakeholders in the energy ecosystem to drive innovation, address common challenges, and promote the adoption of EBMS on a larger scale.

REFERENCES

[1] Brad Traversy, MERN Stack Front To Back: Full Stack React, Redux & Node.js, August 2020. [Online Course] Available: Udemy.

[2] Max Schwarzmüller, React - The Complete Guide (incl Hooks, React Router, Redux), September 2021. [Online Course] Available: Udemy.

[3] Stephen Grider, Node with React: Fullstack Web Development, September 2021. [Online Course] Available: Udemy.

[4] Colt Steele, The Web Developer Bootcamp 2021, September 2021. [Online Course] Available: Udemy.

[5] Shaun Wassell, Build Responsive Real World Websites with HTML5 and CSS3, September 2021. [Online Course] Available: Udemy.

[6] Robin Wieruch, The Road to learn React: Your journey to master plain yet pragmatic React.js, August 2021. [E-book] Available: leanpub.

[7] Daniel Shiffman, The Coding Train, February 2021. [YouTube Channel] Available: YouTube.

[8] David Katz, Full-Stack React Projects: Modern web development using React 16, Node, Express, and MongoDB, October 2021. [E-book] Available: Packt.

[9] Adam Boduch, Full-Stack React, TypeScript, and Node: Build scalable, full-stack JavaScript apps using the latest technologies, August 2021. [E-book] Available: Packt.

App.js

```
import { createContext, useEffect, useState } from "react";
import {
  BrowserRouter as Router,
  Switch,
  Route
} from "react-router-dom";
import { Redirect } from "react-router";
import './App.css';
import Home from './Components/HomePage/Home';
import Login from "./Components/login/login";
import PrivateRoute from "./Components/PrivateRoute/PrivateRoute";
import Register from "./Components/register/register";
import About from "./Components/About/About";
import Dashboard from "./Components/Dashboard/Dashboard";
import ForgetPassword from "./Components/ForgetPassword/ForgetPassword";
import UpdatePassword from "./Components/UpdatePassword/UpdatePassword";
import PayBill from "./Components/PayBill/PayBill";
import MyBill from "./Components/MyBill/MyBill";
import BillCalculator from "./Components/BillCalculator/BillCalculator";

export const loggedInContext = createContext();

function App() {

  const [loggedInUser, setLoggedInUser] = useState([]);

  useEffect(() => {
    setLoggedInUser(JSON.parse(localStorage.getItem('userInfo')))

  }, [])
  const saveLocalStorage = (user) => {
    localStorage.setItem("userInfo", JSON.stringify(user))
    setLoggedInUser(user)
  }

  const time = new Date();
  time.setSeconds(time.getSeconds() + 59);
  return (
    <div className="">
      <loggedInContext.Provider value={[loggedInUser, saveLocalStorage]} >
        <Router>
          <Switch>
            <Route path="/home">
              <Home />
            </Route>

            <Route path="/dashboard">
              <Dashboard />
            </Route>
            <Route exact path="/login"
              render={() => (loggedInUser && loggedInUser.name ? <Redirect to="/"
/> : (<Login />)
              )}
            />
            <Route exact path="/register"
              render={() => (loggedInUser && loggedInUser.name ? <Redirect to="/"
/> : (<Register />)
              )}
            />
            <Route exact path="/about" >
              <About />
            </Route>
            <PrivateRoute exact path="/pay-bill">
              <PayBill />
            </PrivateRoute>
            <PrivateRoute exact path="/my-bill">
```

```
              <MyBill />
            </PrivateRoute>
            <PrivateRoute exact path="/bill-calculator">
              <BillCalculator />
            </PrivateRoute>
            <Route exact path="/forget-password" >
              <ForgetPassword />
            </Route>
            <Route exact path="/update-password/:id" >
              <UpdatePassword />
            </Route>
            <Route exact path="/" >
              <Home />
            </Route>
            <Route path="*">
              <h3 className="text-center">Page Not Found </h3>
            </Route>
          </Switch>
        </Router>
      </loggedInContext.Provider>
    </div>
  );
}

export default App;
```

BillCalculator.js

```
import React, { useEffect, useState } from 'react';
import Layout from '../Layouts/Layout';
import { url } from '../../Data/URL';
import "./BillCalculator.css"

const BillCalculator = () => {
    const [data, setData] = useState([]);
    const [bill, setBill] = useState(0);
    const [unitValue, setUnitValue] = useState(""); // Store the unit input
separately

    useEffect(() => {
        fetch(`${url}/unit-price`)
            .then(res => res.json())
            .then(data => setData(data))
    }, [])

    const handleChange = (e) => {
        const { name, value } = e.target;

        if (name === 'unit' && value < 0) {
            alert("Please, Enter a Valid Unit");
            setUnitValue(""); // Clear the unit input field
            return;
        } else {
            setUnitValue(value); // Update the unit input value
        }
    }

    const calculateBill = (e) => {
        e.preventDefault();
        if (unitValue === "") {
            alert("Please, Enter a Valid Unit");
            return;
        }
        let totalBill = (Number(data[0].unitPrice) * Number(unitValue)) +
Number(data[0].meterBill);
        setBill(totalBill);
    }

    return (
        <div>
            <Layout>
```

```
                <div className="add-rooftop">
                    <div className='my-bill'>
                        <div className="container">
                            <div className="add-bg-color">
                                <div className="container">
                                    <div className="info">
                                        <form className="row g-3">
                                            <h1 className='text-center' style={{
color: "#14D1D0" }}>Bill Calculator</h1>
                                            <div className="col-md-6 offset-md-3">
                                                <label className="form-
label">Enter Unit</label>
                                                <input placeholder="Unit"
value={unitValue} onChange={(e) => handleChange(e)} name='unit' type="number"
className="form-control" required />
                                            </div>
                                            <div className="col-md-6 offset-md-3">
                                                <button onClick={(e) =>
calculateBill(e)} className="submit-btn" type="submit">Calculate</button>
                                            </div>
                                            <div className="col-md-6 offset-md-3">
                                                {bill !== 0 ? (
                                                    <div className="row
calculator">
                                                        <p className='fs-5 text-
center'>Per Unit Price {data[0].unitPrice} </p>
                                                        <div className="col-md-
12">
                                                            <div className="d-flex
justify-content-between">
                                                                <div>
                                                                    <p
className='fs-5'> Bill  </p>

                                                                </div>
                                                                <div>
                                                                    <p
className='fs-5'>{data[0].unitPrice} X {unitValue} = {Number(data[0].unitPrice) *
Number(unitValue)} </p>

                                                                </div>

                                                            </div>
                                                            <div className="d-flex
justify-content-between">
                                                                <div>
                                                                    <p
className='fs-5'>Meter Bill Per Month</p>

                                                                </div>
                                                                <div>
                                                                    <p
className='fs-5'>{data[0].meterBill} </p>

                                                                </div>
                                                            </div>
                                                        </div>
                                                        <div className="col-md-
12">
                                                            <hr />
                                                        </div>
                                                        <div className=" col-md-12
result">
                                                            <div className='d-flex
justify-content-between'>
                                                                <p className='fs-
5'>Total Bill:</p>
                                                                <p className='fs-
5'>{bill}</p>
                                                            </div>
                                                        </div>
                                                    </div>
                                                ) : null}
                                            </div>
```

```
                                        </form>
                                    </div>
                                </div>
                            </div>
                        </div>
                    </div>
                </Layout>
            </div>
        );
};

export default BillCalculator;
```

Dashboard.js

```
import { createContext, useEffect, useState } from "react";
import {
    BrowserRouter as Router,
    Switch,
    Route
} from "react-router-dom";
import { Redirect } from "react-router";
import Home from './HomePage/Home';
import Login from './login/login';
import SideNavbar from './SideNavbar/SideNavbar';
import './Dashboard.css'
import ApprovedPayment from "./Payment/ApprovedPayment";
import PendingPayment from "./Payment/PendingPayment";
import SetBill from "./Bill/SetBill";
import ViewElectricBill from "./Bill/ViewElectricBill";
import EditBill from "./Bill/EditBill";
import RejectedPayment from "./Payment/RejectedPayment";

export const AdminContext = createContext();


const Dashboard = () => {
    const [loggerInUser, setLoggedInUser] = useState([])

    useEffect(() => {
        setLoggedInUser(JSON.parse(localStorage.getItem('admin-info')))
    }, [])

    const saveLocalStorage = (admin) => {
        localStorage.setItem("admin-info", JSON.stringify(admin))
        setLoggedInUser(admin)
    }

    return (
        <div>
            <div className="">
                <AdminContext.Provider value={[loggerInUser, saveLocalStorage]}>

                    {
                        loggerInUser && loggerInUser._id ? <Router>
                            <SideNavbar />
                            <Switch>
                                <Route path="/dashboard/home">
                                    <Home />
                                </Route>
                                <Route path="/dashboard/login"
                                    render={() => (loggerInUser &&
loggerInUser._id ? <Redirect to="/dashboard/home" /> : (<Login />)
                                    )}
                                />
                                <Route exact path="/dashboard/approve-payment">
                                    <ApprovedPayment />
                                </Route>
                                <Route exact path="/dashboard/pending-payment">
                                    <PendingPayment />
```

```
                                    </Route>
                                    <Route exact path="/dashboard/rejected-payment">
                                        <RejectedPayment />
                                    </Route>
                                    <Route exact path="/dashboard/unit-price">
                                        <SetBill />
                                    </Route>
                                    <Route exact path="/dashboard/view-unit-price">
                                        <ViewElectricBill />
                                    </Route>
                                    <Route exact path="/dashboard/edit-bill/:id">
                                        <EditBill />
                                    </Route>
                                    <Route exact path="/dashboard/home">
                                        <Home />
                                    </Route>
                                    <Route path="*">
                                        <h3 className="text-center">Page Not Found
</h3>
                                    </Route>
                                </Switch>
                            </Router> : <Router>
                                <Switch>
                                    <Route path="/dashboard/login">
                                        <Login />
                                    </Route>
                                </Switch>
                            </Router>
                        }

                    </AdminContext.Provider>
                </div>
            </div>
        );
};

export default Dashboard;
```

Login.js

```
import React, { useContext, useState } from "react"
import "./login.css"
import axios from "axios"
import { Link, useHistory } from "react-router-dom"
import { useLocation } from "react-router"
import { loggedInContext } from "../../App"
import Layout from "../Layouts/Layout"
import { url } from "../../Data/URL"

const Login = () => {

    const history = useHistory();
    const location = useLocation();
    const [loggedInUser, setLoggedInUser] = useContext(loggedInContext)

    let { from } = location.state || { from: { pathname: "/" } };

    const [user, setUser] = useState({
        email: "",
        password: "",
    })

    const handleChange = e => {
        const { name, value } = e.target
        setUser({
            ...user,
            [name]: value
        })
```

```
    }

    const login = () => {
        axios.post(`${url}/login`, user)
            .then(res => {
                setUser(res.data)
                if (res.data.user) {
                    setLoggedInUser(res.data.user)
                    history.push(from)
                }
                else {
                    history.push('/login')
                }
            })
    }

    return (
        <Layout>
            <div className="h-70">
            <div className="my-5">
                <div className="student-login">
                    <h1> Login </h1>
                    <input type="text" name="email" value={user.email}
onChange={handleChange} placeholder="Enter your Email"></input>
                    <input type="password" name="password" value={user.password}
onChange={handleChange} placeholder="Enter your Password" ></input>
                    <div className="button" onClick={login}>Login</div>
                    <Link to="/forget-password">Forget Password?</Link>
                    <p className="text-danger" > {user.message} </p>
                    <div>or</div>
                    <div className="button" onClick={() =>
history.push("/register")}>Register</div>
                </div>
            </div>
            </div>
        </Layout>
    )
}

export default Login
```