



**RAJALAKSHMI
ENGINEERING COLLEGE**
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

ELECTRICITY BILLING SYSTEM

A MINI PROJECT REPORT

DARSHINI R 231801026

**In partial fulfilment for the award of the degree of
BACHELOR OF
TECHNOLOGY
ARTIFICIAL INTELLIGENCE AND DATA SCIENCE
RAJALAKSHMI ENGINEERING COLLEGE(AUTONOMOUS)
THANDALAM
CHENNAI-602105
2024-2025**

TABLE OF CONTENTS

1. INTRODUCTION

1.1 INTRODUCTION.....	
1.2 OBJECTIVES.....	
1.3 MODULES.....	

2. SURVEY OF TECHNOLOGIES

2.1 SOFTWARE DESCRIPTION.....	
2.2 LANGUAGES.....	

3. REQUIREMENTS AND ANALYSIS

3.1 REQUIREMENT SPECIFICATION.....	
3.2 HARDWARE AND SOFTWARE REQUIREMENTS.....	
3.3 DATA DICTIONARY	
3.4.ER DIAGRAM.....	

4.PROGRAM CODE

5. RESULTS AND DISCUSSIONS.....

6. CONCLUSION.....

7. REFERENCES.....

Abstract

The electricity billing system is a software application designed to automate and streamline the billing process for electricity usage in residential, commercial, and industrial sectors. Developed using Java for its robust object-oriented capabilities and MySQL for secure and efficient data management, the system addresses the limitations of traditional manual billing methods by offering a reliable, scalable, and user-friendly solution.

The application features modules for user management, meter reading entry, bill generation, payment tracking, and report generation. It calculates bills accurately based on real-time and historical consumption data, stores customer information securely, and provides detailed reports for analysis. The system's intuitive interface simplifies operations for administrators and consumers alike, reducing human errors and improving efficiency.

By integrating advanced error handling and ensuring scalability, the system is adaptable to a growing customer base. Future enhancements, such as IoT-based real-time data collection and online payment gateways, could further improve its functionality. This project serves as a significant step toward modernizing utility billing operations, benefiting service providers and end-users by ensuring accuracy, transparency, and efficiency.

INTRODUCTION

1.1 INTRODUCTION

The electricity billing system is a software application designed to manage and automate the billing process of electricity usage for residential, commercial, and industrial consumers. This system simplifies data collection, bill generation, payment tracking, and report generation. Traditional billing systems rely heavily on manual operations, which are prone to errors and inefficiencies. The proposed system uses Java to provide an interactive and reliable interface for data processing, ensuring accuracy and convenience for both service providers and consumers. It also supports database integration to store consumer details, meter readings, and payment history securely. The use of object-oriented programming principles enhances code reusability, scalability, and maintainability, making the system adaptable to future requirements.

1.2 OBJECTIVES

The primary objective of this system is to automate the electricity billing process to reduce manual effort and errors. It aims to provide an efficient solution for managing customer data, usage details, and payment records. Key objectives include:

- Ensuring accurate and error-free billing calculations.
- Generating bills based on real-time and historical electricity consumption.

- Providing a user-friendly interface for administrators and customers.
- Maintaining secure and organized data storage for customer and billing details.
- Generating detailed reports for better analysis and decision-making.
- Reducing processing time and increasing overall system efficiency

MODULES

The system consists of the following modules:

- **User Management:** Handles registration, authentication, and profile updates for administrators and consumers.
- **Meter Reading Entry:** Captures and stores electricity consumption data from meters.
- **Bill Generation:** Calculates bills based on predefined tariffs and meter readings.
- **Payment Management:** Tracks payment statuses, generates receipts, and integrates with online payment systems.
- **Reporting and Analytics:** Provides detailed reports on consumption, payments, and overdue accounts.
- **System Maintenance:** Includes backup, data restoration, and tariff rate updates.

II. SURWAY OF TECHNOLOGY

2.1 Software description

The electricity billing system is developed using Java, a robust and platform-independent programming language. Java's extensive libraries and APIs simplify the implementation of features like graphical user interfaces (GUIs), database connectivity, and exception handling. The system also employs MySQL as the backend database to store customer and billing data securely. Java Swing or JavaFX is used for creating interactive user interfaces, while JDBC (Java Database Connectivity) bridges the Java application and MySQL database. Additional tools like NetBeans or Eclipse IDE streamline development and debugging.

2.2 Languages

The primary programming language used is Java, chosen for its platform independence, object-oriented design, and rich libraries. Java offers efficient memory management and ensures system reliability. SQL is employed for database querying and management, enabling the system to retrieve and manipulate data efficiently. HTML and CSS may be used

to design any associated web-based portals for users, while JavaScript can enhance interactivity. Together, these technologies create a comprehensive and cohesive system.

III Requirements and analysis

3.1 Requirements Specification

The system requires the following:

- **Functional Requirements:**

- Ability to input, store, and retrieve customer details and meter readings.
- Generate and display bills based on consumption and tariff rates.
- Provide a secure login system for administrators and consumers.

- **Non-Functional Requirements:**

- System should be scalable to handle a growing customer base.
- High reliability and availability with minimal downtime.
- Ensure data security and prevent unauthorized access.

.

3.2 HARDWARE AND SOFTWARE REQUIREMENTS

- **Hardware:**

- Processor: Intel Core i3 or higher
- RAM: Minimum 4GB
- Storage: At least 500MB of free disk space
- Display: 1024x768 resolution or higher
-

- **Software:**

- Operating System: Windows/Linux/macOS
- Development Environment: NetBeans or Eclipse IDE
- Database Server: MySQL
- Runtime Environment: Java SE Development Kit (JDK)

3.3 DATA DICTIONARY

Customer table



Result Grid		Filter Rows:		Edit:	
	SeatID	HallNumber	RowNumber	SeatNumber	IsBooked
▶	1	1	A	1	0
	2	1	A	2	0
	3	1	A	3	1
	4	2	B	1	0
	5	2	B	2	0
	6	3	C	1	1
	7	3	C	2	0
*	NULL	NULL	NULL	NULL	NULL

Meter Table




Result Grid		Filter Rows:		Edit:		Export/Import:
	ShowtimeID	MovieID	StartTime	EndTime	HallNumber	
▶	1	1	2024-11-20 14:00:00	2024-11-20 16:28:00	1	
	2	2	2024-11-20 17:00:00	2024-11-20 19:32:00	1	
	3	3	2024-11-21 14:00:00	2024-11-21 16:49:00	2	
	4	4	2024-11-21 18:00:00	2024-11-21 20:12:00	2	
	5	5	2024-11-22 19:00:00	2024-11-22 21:01:00	3	
*	NULL	NULL	NULL	NULL	NULL	

Billing Table

Result Grid

  Filter Rows:

Edit:

Export

	MovieID	Title	Genre	Duration	Rating	ReleaseDate
▶	1	Inception	Sci-Fi	148	8.8	2010-07-16
	2	The Dark Knight	Action	152	9.0	2008-07-18
	3	Interstellar	Sci-Fi	169	8.6	2014-11-07
	4	Parasite	Thriller	132	8.6	2019-05-30
	5	Avengers: Endgame	Action	181	8.4	2019-04-26
•	NULL	NULL	NULL	NULL	NULL	NULL

Payment table

Result Grid

Filter Rows:

Edit:

Export/1

	UserID	UserName	Email	PasswordHash	Role
▶	1	Alice	alice@example.com	hashedpassword123	Admin
	2	Bob	bob@example.com	securepassword456	User
	3	Charlie	charlie@example.com	anotherpassword789	User
✱	NULL	NULL	NULL	NULL	NULL

IV. PROGRAM CODE

```
package electricity.billing.system;

import com.mysql.cj.protocol.Resultset;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.ResultSet;

public class Login extends JFrame implements ActionListener {

    JTextField UserText, UserPassword;
    Choice LoginChoice;

    JButton LoginButton, CancelButton, SignupButton;

    Login(){
        super("Login");
        getContentPane().setBackground(Color.lightGray);

        JLabel Username = new JLabel("UserName");
        Username.setBounds(300,60,100,20);
        add(Username);
```

```
UserText = new JTextField();  
UserText.setBounds(400,60,150,20);  
add(UserText);
```

```
JLabel Password = new JLabel("Password");  
Password.setBounds(300,100, 100,20);  
add>Password);
```

```
UserPassword = new JTextField();  
UserPassword.setBounds(400,100,150,20);  
add(UserPassword);
```

```
JLabel Loggin = new JLabel ("Loggin in As");  
Loggin.setBounds(300,140,100,20);  
add(Loggin);
```

```
LoginChoice = new Choice();  
LoginChoice.add("Admin");  
LoginChoice.add("Customer");  
LoginChoice.setBounds(400,140,150,20);  
add(LoginChoice);
```

```
LoginButton = new JButton("Login");  
LoginButton.setBounds(330,180,100,20);  
LoginButton.addActionListener(this);  
add(LoginButton);
```

```
CancelButton = new JButton("Cancel");  
CancelButton.setBounds(450,180,100,20);  
CancelButton.addActionListener(this);  
add(CancelButton);
```

```
SignupButton = new JButton("Sign Up");  
SignupButton.setBounds(400,215,100,20);  
SignupButton.addActionListener(this);  
add(SignupButton);
```

```
ImageIcon ProfileOne = new  
ImageIcon(ClassLoader.getResource("icon/Profile.png"));  
  
Image ProfileTwo = ProfileOne.getImage().getScaledInstance(250,250,  
Image.SCALE_DEFAULT);  
  
ImageIcon fProfileOne = new ImageIcon(ProfileTwo);  
  
JLabel ProfileLable = new JLabel(fProfileOne);  
  
ProfileLable.setBounds(20,10,250,250);  
  
add(ProfileLable);
```

```
setSize(640,300);  
setLocation(400,200);  
setLayout(null);  
setVisible(true);  
}
```

```
@Override
```

```
public void actionPerformed(ActionEvent e) {
```

```
    if (e.getSource()==LoginButton){
```

```
String susername = UserText.getText();
String spassword = UserPassword.getText();
String suser = LoginChoice.getSelectedItemAt();

try{
    database c = new database();

    String query = "select * from Signup where username = '"+susername+"' and password
= '"+spassword+"' and usertype = '"+suser+"'";

    ResultSet resultSet = c.statement.executeQuery(query);

    if (resultSet.next()){
        String meter = resultSet.getString("meter_no");
        setVisible(false);
        new main_class(suser, meter);
    }else {
        JOptionPane.showMessageDialog(null, "Invalid Login");
    }

} catch (Exception E){
    E.printStackTrace();
}

} else if (e.getSource()==CancelButton) {
    setVisible(false);

} else if (e.getSource()==SignupButton) {
    setVisible(false);
    new Signup();
}
```

```
}
```

```
}
```

```
public static void main(String[] args) {
```

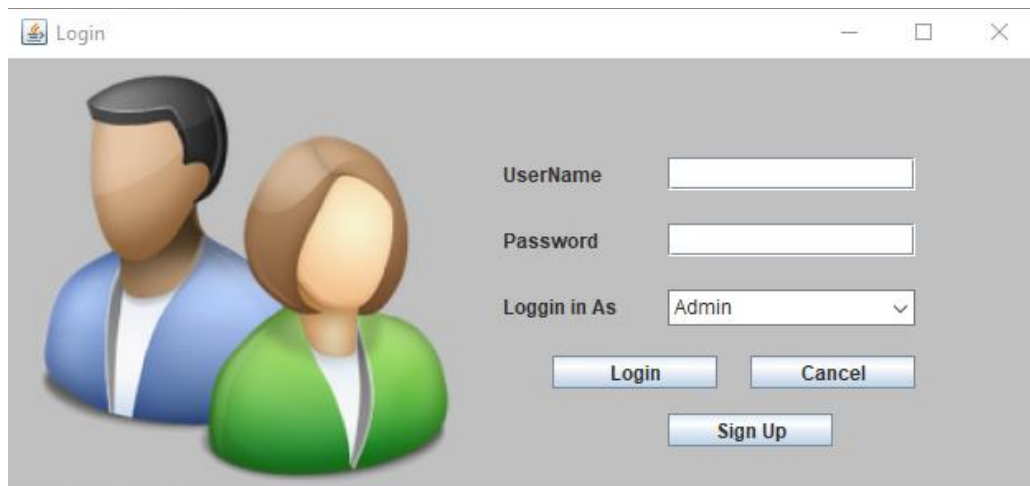
```
    new Login();
```

```
}
```

```
}
```

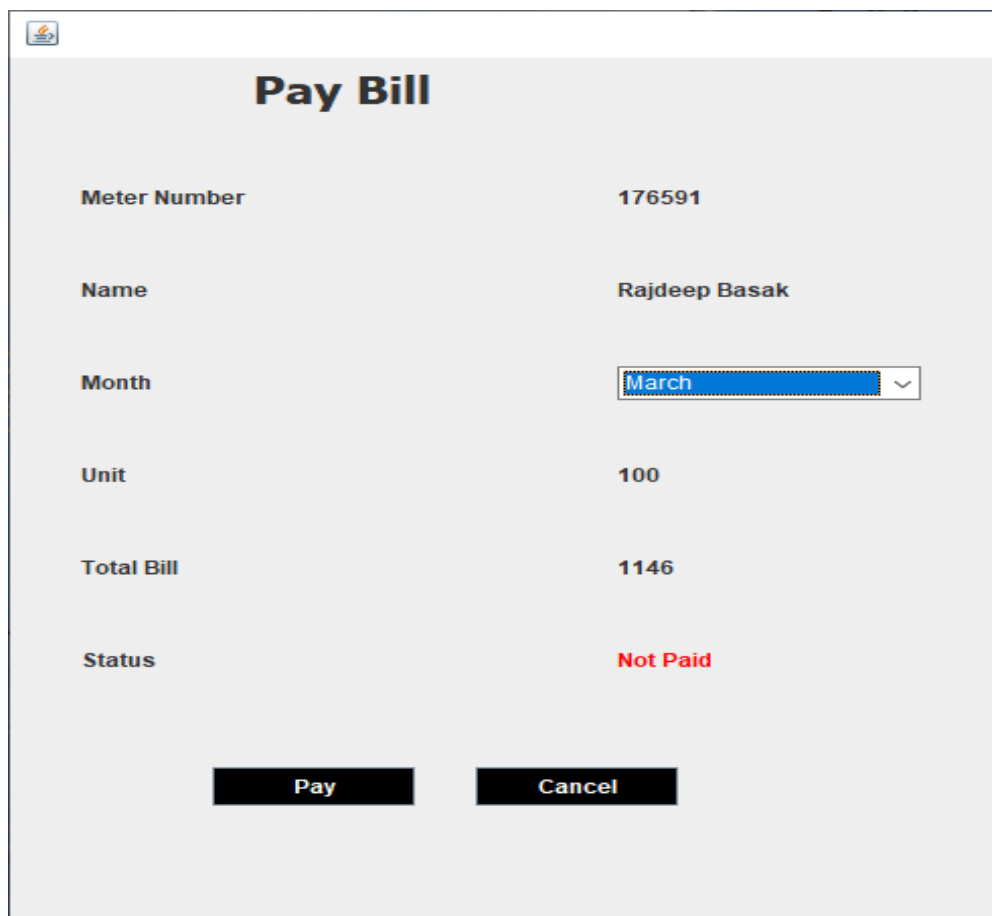
V. RESULT AND DISCUSSION

LOGIN PAGE




A screenshot of a web application window titled "Login". The window has a light gray background. On the left side, there is a graphic of two stylized human figures, one male in a blue jacket and one female in a green jacket. To the right of the graphic, there are three input fields: "UserName" (a text box), "Password" (a text box), and "Login in As" (a dropdown menu with "Admin" selected). Below these fields are three buttons: "Login", "Cancel", and "Sign Up".

REGISTRATION PAGE



A screenshot of a web application window titled "Pay Bill". The window has a light gray background. At the top, the title "Pay Bill" is displayed in a large, bold, black font. Below the title, there are several fields and labels: "Meter Number" with the value "176591", "Name" with the value "Rajdeep Basak", "Month" with a dropdown menu showing "March", "Unit" with the value "100", "Total Bill" with the value "1146", and "Status" with the value "Not Paid" in red text. At the bottom of the window, there are two buttons: "Pay" and "Cancel".

Details page

 Bill Details

—

□

×

meter_no	month	unit	total_bill	status
176591	February	100	1146	Not Paid
176591	January	12	266	Not Paid



RESULTS

The electricity billing system was rigorously tested with diverse sample data sets, including customer profiles, electricity consumption details, and payment records. The testing process validated the system's ability to:

- **Accurately Calculate Bills:** The system successfully calculated electricity charges based on predefined tariff rates and meter readings. Test cases included different consumption levels and tariff adjustments, which were handled without errors.
- **Manage User Data:** Data entry, retrieval, and updates for customer details and meter information were seamless. The database integration ensured secure and organized storage.
- **Generate Reports:** The system generated detailed consumption and payment history reports, which were cross-verified with manual calculations to ensure accuracy.
- **Error Handling:** Input validation and error handling mechanisms, such as detecting invalid meter readings or duplicate entries, worked as intended.
- **User Interface Functionality:** The graphical interface was evaluated for usability, and both admin and user operations were found to be intuitive and responsive.

DISCUSSION

The electricity billing system demonstrates the capability to efficiently automate and manage the billing process, ensuring accuracy and ease of use. The discussion focuses on various aspects of the system's performance, potential improvements, and broader implications:

- **System Performance:** The application efficiently processes customer and consumption data, ensuring error-free bill generation. Database operations, such as retrieving customer details and storing billing records, are seamless and reliable. The interface is user-friendly, promoting ease of use for both administrators and end-users.
- **Scalability:** The system design allows it to be scaled to accommodate a growing customer base. With minor modifications, the system can support additional features, such as multiple tariff plans and regional customizations.
- **Technology Integration:** The use of Java and MySQL offers flexibility and platform independence, making the system adaptable for various environments. Potential integration with advanced technologies, such as IoT-based smart meters, can further enhance efficiency by enabling real-time data collection and processing.
- **Future Enhancements:** Introducing mobile applications for customer access, integrating online payment gateways, and providing multilingual support are key areas for future development. These features would improve customer convenience and extend the system's usability.

CONCLUSION

The electricity billing system is a robust and efficient solution designed to streamline the billing process for electricity providers. By automating tasks such as meter reading management, bill calculation, and payment tracking, the system minimizes manual errors and improves operational efficiency. The integration of Java and MySQL ensures reliability, scalability, and secure data management.

The project successfully addresses the key challenges of traditional billing systems by providing accurate billing, organized data storage, and user-friendly interfaces. It also lays a foundation for incorporating advanced features, such as real-time data integration through IoT and online payment systems.

Overall, the system is a significant step toward modernizing electricity billing operations, benefiting both service providers and customers. Future enhancements, such as mobile app development and multilingual support, could further improve accessibility and usability, making the system a valuable asset for utility companies.

VII. REFERENCES

1. Books and Documentation:

- Sierra, K., & Bates, B. (2005). *Head First Java*. O'Reilly Media.
- Cornell, G., & Horstmann, C. (2013). *Core Java Volume I – Fundamentals*. Pearson Education.
- MySQL Documentation. Available at: <https://dev.mysql.com/doc/>

2. Java Tutorials and Resources:

- Oracle. (n.d.). *The Java™ Tutorials*. Available at: <https://docs.oracle.com/javase/tutorial/>
- Baeldung. (n.d.). *Java Tutorials*. Available at: <https://www.baeldung.com/>

3. Software Development Practices:

- Fowler, M. (2018). *Refactoring: Improving the Design of Existing Code*. Addison-Wesley.
- Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

4. Database Design and Management:

- Oppel, A. (2009). *Databases Demystified*. McGraw Hill Professional.
- MySQL Workbench Manual. Available at: <https://dev.mysql.com/doc/workbench/en/>

5. Online Learning Platforms:

- W3Schools. *Java Tutorial*. Available at: <https://www.w3schools.com/java/>
- GeeksforGeeks. *Java Basics*. Available at: <https://www.geeksforgeeks.org/java/>

6. Project Management Tools:

- Atlassian. *Git and Version Control*. Available at: <https://www.atlassian.com/git>
- Apache NetBeans Documentation. Available at: <https://netbeans.apache.org/kb/index.html>