ELECTRICITY BILL MANAGEMENT SYSYTEM

This project is designed to simplify electricity bill calculation and storage. It allows users to generate bills based on units consumed using slab rates, store consumer details in a MySQL database, and view all saved bills. The system ensures quick, accurate, and organized billing.

1.Consumer.java

```
 // Consumer.java

public class Consumer {

    private String consumerName;

    private int consumerID;

    private double units;

    private double billAmount;


    public Consumer(String name, int id, double units, double
billAmount) {

        this.consumerName = name;

        this.consumerID = id;

        this.units = units;

        this.billAmount = billAmount;

    }


    public String getConsumerName() {

        return consumerName;

    }
```

```
    public int getConsumerID() {
        return consumerID;
    }

    public double getUnits() {
        return units;
    }

    public double getBillAmount() {
        return billAmount;
    }

    public void displayBill() {
        System.out.println("\n--- Electricity Bill ---");
        System.out.println("Consumer ID   : " + consumerID);
        System.out.println("Consumer Name : " + consumerName);
        System.out.println("Units Consumed: " + units);
        System.out.println("Total Bill    : Rs. " + billAmount);
        System.out.println("   ");
    }
}
```

Consumer.java is a model class that represents one customer.

It stores: name, ID, units, bill amount.

It has a constructor to create a consumer with these details.

Getter methods let other classes read the details.

displayBill() prints the bill in a nice format.

In simple words It's just a blueprint to hold and show one customer's electricity bill.

2.BillCalculator.java

```
// BillCalculator.java
 public class BillCalculator {
 public static double
 calculateBill(double units) {
double amount = 0;
if (units <= 100) {
amount = units * 1.5;
 } else if (units <= 200) {
 amount = (100 * 1.5) + (units - 100) * 2.5;
 } else if (units <= 300) {
 amount = (100 * 1.5) + (100 * 2.5) + (units - 200) * 3.5;
} else {
amount = (100 * 1.5) + (100 * 2.5) + (100 * 3.5) + (units - 300) *
5.0;
 }
return amount;
 }
```

```
}
```

BillCalculator.java

Handles the calculation of electricity bills using slab rates. It takes the number of units consumed and returns the total bill amount.

3.DBHandler.java

```
// DBHandler.java

 import java.sql.;

import java.util.;

public class DBHandler {

 private static final String URL= "jdbc:

mysql://localhost:3306/ElectricityDB";

 private static final String USER = "root"; // your MySQL username
private static final String PASSWORD = "root"; // your MySQL password

// Insert new consumer bill into DB
public static void

 saveToDB(Consumer consumer) {
    try (Connection conn = DriverManager.getConnection(URL, USER,
PASSWORD)) {
        String query = "INSERT INTO bills (consumer_id, consumer_name,
units, bill_amount) VALUES (?, ?, ?, ?)";
        PreparedStatement ps = conn.prepareStatement(query);
        ps.setInt(1, consumer.getConsumerID());
        ps.setString(2, consumer.getConsumerName());
        ps.setDouble(3, consumer.getUnits());
        ps.setDouble(4, consumer.getBillAmount());
        ps.executeUpdate();
        System.out.println("✅ Bill saved to database!");
```

```
    } catch (SQLException e) {
        System.out.println("⚠ Error saving to DB: " + e.getMessage());
    }
}

// Fetch all saved bills
public static List<Consumer> readFromDB() {
    List<Consumer> consumers = new ArrayList<>();
    try (Connection conn = DriverManager.getConnection(URL, USER,
PASSWORD)) {
        String query = "SELECT * FROM bills";
        Statement stmt = conn.createStatement();
        ResultSet rs = stmt.executeQuery(query);

        while (rs.next()) {
            int id = rs.getInt("consumer_id");
            String name = rs.getString("consumer_name");
            double units = rs.getDouble("units");
            double bill = rs.getDouble("bill_amount");
            consumers.add(new Consumer(name, id, units, bill));
        }
    } catch (SQLException e) {
        System.out.println("⚠ Error reading from DB: " +
e.getMessage());
    }
    return consumers;
}


}
```

DBHandler.java

Manages the database connection with MySQL. It saves consumer details and bills into the database and retrieves all stored bills when needed.

4.Main.java

```
// Main.java import java.util.*;
```

```
public class Main {

 public static void main(String[] args) {

 Scanner sc = new Scanner(System.in);

 int choice;

   do {
      System.out.println("\n===== Electricity Bill Management System
=====");
        System.out.println("1. Generate New Bill");
        System.out.println("2. View Saved Bills");
        System.out.println("3. Exit");
        System.out.print("Enter your choice: ");
        choice = sc.nextInt();
        sc.nextLine();

 switch (choice) {
            case 1:
                System.out.print("Enter Consumer Name: ");
                String name = sc.nextLine();
                System.out.print("Enter Consumer ID: ");
                int id = sc.nextInt();
                System.out.print("Enter Units Consumed: ");
                double units = sc.nextDouble();

                double billAmount =
BillCalculator.calculateBill(units);
                Consumer consumer = new Consumer(name, id, units,
billAmount);

                consumer.displayBill();
                DBHandler.saveToDB(consumer);
                break;

            case 2:
                System.out.println("\n--- Saved Bills from Database --
-");
                List<Consumer> consumers = DBHandler.readFromDB();
                for (Consumer c : consumers) {
```

```
                    c.displayBill();
                }
                break;

            case 3:
                System.out.println("Exiting... Thank you!");
                break;

            default:
                System.out.println("⚠ Invalid choice! Try again.");
        }
    } while (choice != 3);

    sc.close();
}


}
```

Main.java

The starting point of the program. It shows a menu, takes input from the user, creates Consumer objects, calculates bills, saves them to the database, and displays saved bills.