



Invoice Management System for Electricity Bill Generation

MICROPROJECT REPORT

Submitted by

Sr. No.	RollNo. (Sem-III)	Full Name of Student	Enrollment No.	Seat No. (Sem -III)
1	83	LAHWARE YAHYA IMTIYAZ	23213560195	
2	84	HAJRAH SALEHA IMRAN KAZI	23213560196	
3	85	SURYAVANSHI SHREYA SHAHAJI	23213560197	
4	86	ADESARA HIRAL PARESHKUMAR	23213560198	

Under the Guidance of

Prof. Ms. A.B. Suryavanshi

In

Second Year Diploma Programme in Engineering &
Technology of Maharashtra State Board of Technical
Education, Mumbai
(Autonomous)

ISO 9001 :2008 (ISO/IEC-27001 13)

At

1734 - TRINITY POLYTECHNIC PUNE



MAHARASHTRA STATE BOARD
OF TECHNICAL
EDUCATION, MUMBAI

Certificate

This is to certify that Mr./Ms. LAHWARE YAHYA IMTIYAZ
Roll No. 83, of Third Semester of Diploma in
Computer Engineering (CO) of
Institute TRINITY POLYTECHNIC PUNE (Code: 1734)
has completed the Micro-Project satisfactorily in course
Database Management System (313302) for the academic year 2024 – 2025 as
per the MSBTE prescribed curriculum of K-Scheme.

Place: Pune

Enrollment No: 23213560195

Date: / /2024

Exam Seat No:

Project Guide

Head of the Department

Principal





MAHARASHTRA STATE BOARD
OF TECHNICAL
EDUCATION, MUMBAI

Certificate

This is to certify that ~~Mr.~~/Ms. HAJRAH SALEHA IMRAN KAZI
Roll No. 84, of Third Semester of Diploma in
Computer Engineering (CO) of
Institute TRINITY POLYTECHNIC PUNE (Code: 1734)
has completed the Micro-Project satisfactorily in course
Database Management System (313302) for the academic year 2024 – 2025 as
per the MSBTE prescribed curriculum of K-Scheme.

Place: Pune

Enrollment No: 23213560196

Date: __/__/2024

Exam Seat No: _____

Project Guide

Head of the Department

Principal





MAHARASHTRA STATE BOARD
OF TECHNICAL
EDUCATION, MUMBAI

Certificate

This is to certify that ~~Mr.~~/Ms. SURYAVANSHI SHREYA SHAHAJI
Roll No. 85, of Third Semester of Diploma in
Computer Engineering (CO) of
Institute TRINITY POLYTECHNIC PUNE (Code: 1734)
has completed the Micro-Project satisfactorily in course
Database Management System (313302) for the academic year 2024 – 2025 as
per the MSBTE prescribed curriculum of K-Scheme.

Place: Pune

Enrollment No: 23213560197

Date: __/__/2024

Exam Seat No: _____

Project Guide

Head of the Department

Principal





MAHARASHTRA STATE BOARD
OF TECHNICAL
EDUCATION, MUMBAI

Certificate

This is to certify that ~~Mr.~~/Ms. ADESARA HIRAL PARESHKUMAR
Roll No. 86, of Third Semester of Diploma in
Computer Engineering (CO) of
Institute **TRINITY POLYTECHNIC PUNE** (Code: 1734)
has completed the Micro-Project satisfactorily in course
Database Management System (313302) for the academic year 2024 – 2025 as
per the MSBTE prescribed curriculum of K-Scheme.

Place: Pune

Enrollment No: 23213560198

Date: / /2024

Exam Seat No:

Project Guide

Head of the Department

Principal



Contents

Sr.No	Name of Particulars	PageNo.
1	Introduction	1-2
2	System Analysis	3-4
3	Database Design	5-6
4	Implementation	7-9
5	Testing & Results	10-11
6	Conclusion	12
7	References	13
8	Appendices	14-17

INTRODUCTION:

Efficient management of electricity billing is essential for both utility companies and consumers, ensuring transparency, accuracy, and ease of access to billing information. As electricity consumption continues to grow, manual methods of calculating bills are not only time-consuming but also prone to errors. To address these challenges, an automated **Invoice Management System for Electricity Bill Generation** has been designed and developed as part of this microproject.

The purpose of this project is to create a streamlined system where electricity bills can be generated dynamically based on user-specific meter readings. By utilizing a database-driven approach, the system can securely store user information, track electricity usage, and automatically compute bill amounts based on predefined electricity rates. This reduces human intervention, minimizes the potential for errors, and makes the billing process more efficient.

Our system accepts meter readings as input and then calculates the bill based on the number of units consumed and the set rate per unit. The bill is generated instantly, stored in a database, and can be retrieved when required. This not only saves time but also provides a reliable, user-friendly solution to a common administrative task faced by power distribution companies.

The project was developed using **MySQL**, where we created structured tables for storing user details, electricity rates, and bill information. Additionally, a **stored procedure** was implemented to handle the logic of bill generation based on user inputs. This procedure automatically inserts relevant data into the tables and computes the total amount payable based on the input meter readings.

This system was designed by a team of four members, with each member contributing to different aspects of the project. The goal was to simulate a practical electricity billing environment, providing a solution that can be scaled for real-world applications. Our work not only aims to automate bill generation but also enhances the overall reliability and accuracy of the billing process.

The key objectives of this project are:

- To develop an efficient database-driven system for electricity billing.
- To automate the calculation of bill amounts based on user-provided meter readings.
- To store and retrieve billing information securely for future reference.
- To eliminate manual errors and streamline the overall billing process.

Through the successful completion of this microproject, we hope to demonstrate the importance and potential of automated systems in utility management, particularly for electricity billing.

SYSTEM ANALYSIS:

The Electricity Billing System is designed to automate the process of bill generation and management based on user inputs such as meter readings. This system addresses the need for efficient, accurate, and timely electricity bill generation, reducing the chances of manual errors and streamlining operations. By using a structured database and stored procedures, the system ensures seamless integration of user data, meter readings, and bill calculation.

The system is divided into different functional components, each responsible for a key part of the process—meter reading collection, bill generation, payment, and data verification. By clearly defining the workflow between these components, the system ensures that all operations are performed smoothly and in the correct sequence.

Requirements Specification

The system's functionality is based on several key requirements that are categorized into functional and non-functional specifications. These requirements ensure that the system performs its core tasks effectively while also meeting performance, usability, and security expectations.

Functional Requirements:

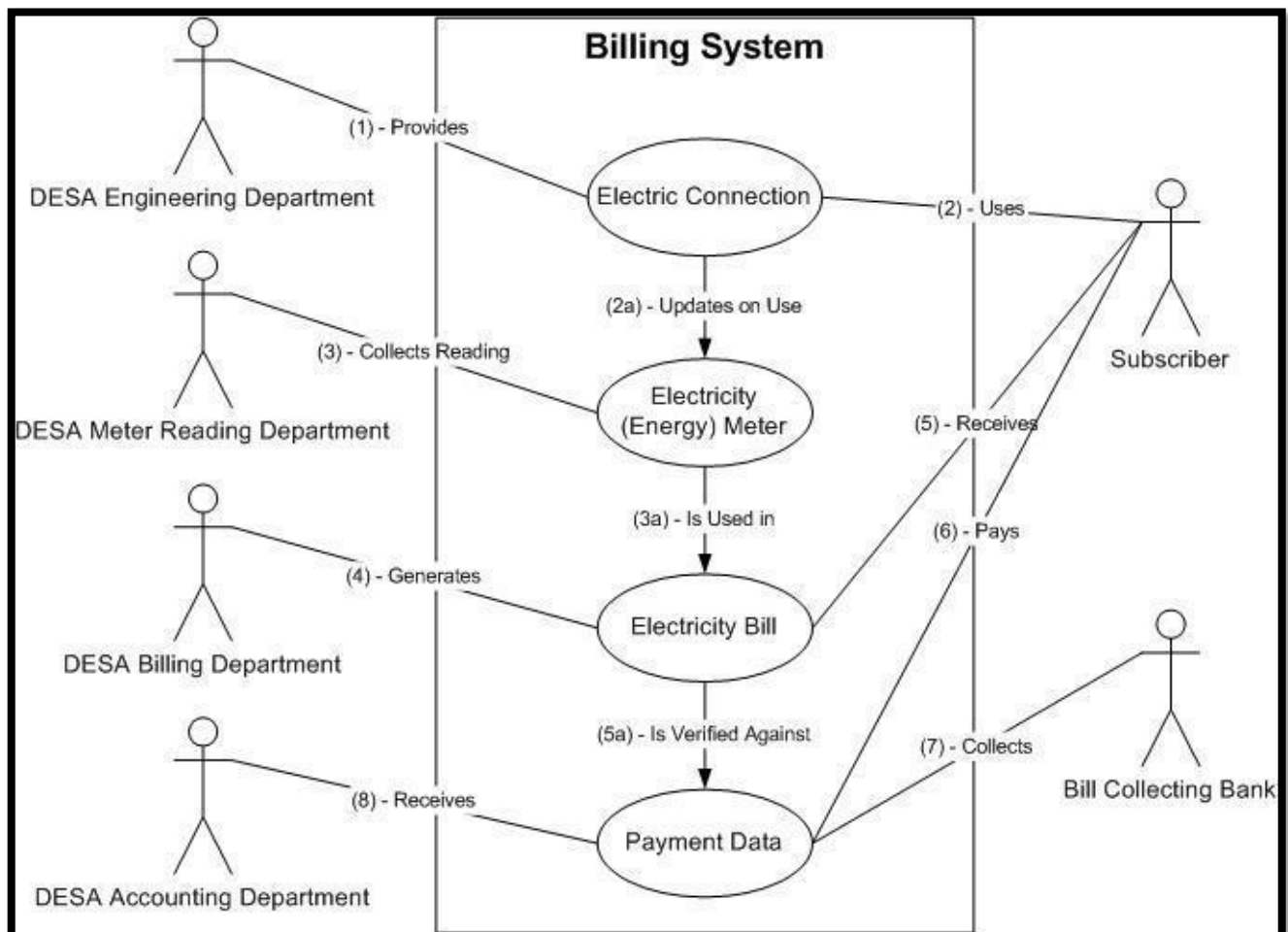
1. **User Data Management:** The system must be able to store and manage user information, including names, addresses, and user IDs.
2. **Meter Reading Input:** The system should accept meter readings as input for each user to calculate electricity consumption.
3. **Bill Calculation:** Based on meter readings, the system should compute the bill amount using predefined rates.
4. **Bill Generation:** The system must generate an electricity bill for each user, storing the bill data (amount, date, etc.) in the database.
5. **Payment Verification:** The system should track and verify payment against each generated bill.

Non-Functional Requirements:

1. **Performance:** The system should be able to handle multiple bill generations efficiently without delay.
 2. **Security:** Sensitive user data and billing information must be securely stored and accessible only to authorized users.
 3. **Usability:** The system should have a clear and simple interface for entering data and retrieving records.
 4. **Scalability:** The system should be scalable to accommodate an increasing number of users and data entries over time.
-

Use Case Diagram

The use case diagram below illustrates the interactions between various entities involved in the billing process, including subscribers, different departments, and the bill collecting bank. It visually represents the entire workflow, from meter reading to bill generation and payment processing.

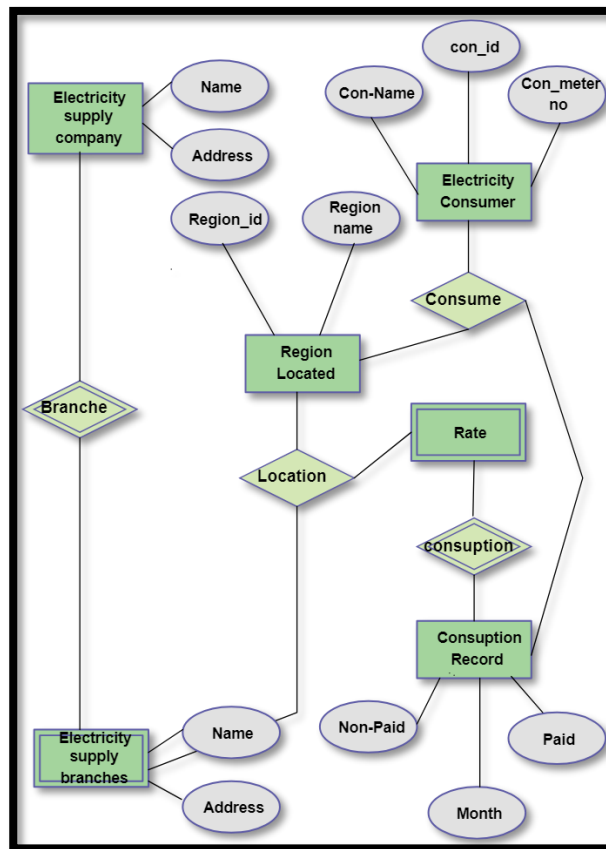


DATABASE DESIGN:

The database design of the Electricity Billing System is crucial for ensuring the efficient storage, retrieval, and management of user data, meter readings, and bill records. The design follows a structured approach, including an Entity-Relationship (ER) diagram, schema definition, and detailed table descriptions, to facilitate accurate bill generation and data handling.

ER Diagram

The ER diagram for the Electricity Billing System is attached below. It visually represents the key entities, their attributes, and the relationships between them, providing a clear overview of the database structure.



Schema Design

The schema design outlines the organization of the database, ensuring that all necessary data is systematically arranged for effective storage and retrieval. It consists of three main tables: **Users**, **Bills**, and **ElectricityRates**. Each table contains specific fields that hold vital information related to the electricity billing process. This structured approach enables efficient data management, accurate bill calculation, and seamless interactions among different entities. The schema supports scalability, allowing for future enhancements and additional functionalities as required.

Table Definitions

1. Users Table

- **Table Name:** Users
- **Description:** Stores user-related information.

Column Name	Data Type	Description
UserID	INT (Primary Key, Auto Increment)	Unique identifier for each user.
UserName	VARCHAR(50)	Name of the electricity subscriber.
Address	VARCHAR(100)	Address of the user.

2. ElectricityRates Table

- **Table Name:** ElectricityRates
- **Description:** Contains electricity rates that are used to calculate the bill.

Column Name	Data Type	Description
RateID	INT (Primary Key)	Unique identifier for the rate entry.
RatePerUnit	DECIMAL(10,2)	Cost per unit of electricity consumed.

3. Bills Table

- **Table Name:** Bills
- **Description:** Stores the generated bill data based on meter readings.

Column Name	Data Type	Description
BillID	INT (Primary Key, Auto Increment)	Unique identifier for each bill.
UserID	INT (Foreign Key)	References the user who the bill belongs to.
MeterReading	INT	Number of units consumed for the billing cycle.
BillAmount	DECIMAL(10,2)	Total bill amount calculated using the rate.
BillDate	DATE	Date on which the bill is generated.

IMPLEMENTATION:

The implementation of the Electricity Billing System involves creating a structured database using SQL to facilitate efficient bill generation and management. This includes defining tables for users, bills, and electricity rates, along with stored procedures for encapsulating business logic. The SQL scripts and stored procedures utilized in this project are attached for reference.

SQL Scripts

The SQL scripts used for setting up the database schema and stored procedures are pasted below and the screenshots of the same are included in the appendices.

```
CREATE DATABASE IF NOT EXISTS ElectricityBilling;  
USE ElectricityBilling;
```

```
DROP TABLE IF EXISTS Bills;  
DROP TABLE IF EXISTS ElectricityRates;  
DROP TABLE IF EXISTS Users;
```

```
CREATE TABLE Users (  
    UserID INT PRIMARY KEY AUTO_INCREMENT,  
    UserName VARCHAR(50),  
    Address VARCHAR(100)  
);
```

```
CREATE TABLE ElectricityRates (  
    RateID INT PRIMARY KEY,  
    RatePerUnit DECIMAL(10, 2)  
);
```

```
CREATE TABLE Bills (  
    BillID INT PRIMARY KEY AUTO_INCREMENT,  
    UserID INT,  
    MeterReading INT,  
    BillAmount DECIMAL(10, 2),  
    BillDate DATE,  
    FOREIGN KEY (UserID) REFERENCES Users(UserID)  
);
```

```
INSERT INTO ElectricityRates (RateID, RatePerUnit) VALUES  
(1, 0.15);
```

```
DELIMITER //
```

```
CREATE PROCEDURE GenerateBill()
```

```
BEGIN
```

```
    DECLARE v_userID INT;
```

```
    DECLARE v_userName VARCHAR(50);
```

```
    DECLARE v_userAddress VARCHAR(100);
```

```
    DECLARE v_meterReading INT;
```

```
    DECLARE v_ratePerUnit DECIMAL(10, 2);
```

```
    DECLARE v_billAmount DECIMAL(10, 2);
```

```
    SET v_userName = (SELECT UserName FROM  
(SELECT @input_userName AS UserName) AS temp);
```

```
    SET v_userAddress = (SELECT Address FROM  
(SELECT @input_userAddress AS Address) AS temp);
```

```
    SET v_meterReading = (SELECT MeterReading FROM (SELECT  
@input_meterReading AS MeterReading) AS temp);
```

```
    SET v_userID = (SELECT IFNULL(MAX(UserID), 0) + 1 FROM Users);
```

```
    INSERT INTO Users (UserName, Address) VALUES (v_userName,  
v_userAddress);
```

```
    SELECT RatePerUnit INTO v_ratePerUnit FROM ElectricityRates  
    WHERE RateID = 1;
```

```
    SET v_billAmount = v_meterReading * v_ratePerUnit;
```

```
    INSERT INTO Bills (UserID, MeterReading, BillAmount, BillDate)  
    VALUES (v_userID, v_meterReading, v_billAmount, CURDATE());
```

```
    SELECT 'Bill generated successfully!' AS Message,  
           v_userID AS UserID,  
           v_userName AS UserName,  
           v_meterReading AS MeterReading,  
           v_billAmount AS BillAmount,  
           CURDATE() AS BillDate;
```

```

END //
DELIMITER ;

SET @input_userName = 'Alice Smith';
SET @input_userAddress = '123 Main St';
SET @input_meterReading = 100;

CALL GenerateBill();

```

Sample Data

Screenshots demonstrating the sample data in the Users, Bills, and ElectricityRates tables are also attached in the appendices.

```

mysql> SET @input_userName = 'Alice Smith';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '123 Main St';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 100;
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> CALL GenerateBill();
+-----+-----+-----+-----+-----+-----+
| Message                | UserID | UserName  | MeterReading | BillAmount | BillDate  |
+-----+-----+-----+-----+-----+-----+
| Bill generated successfully! |      1 | Alice Smith |          100 |         15.00 | 2024-10-02 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.06 sec)

```

TESTING & RESULTS:

Test Cases

The testing phase involved creating specific test cases to validate the functionality of the Electricity Billing System. Below are some key test cases executed during the testing process:

1. Test Case 1: Add New User

- **Input:** User Name: "Hajrah", Address: "208 Galaxy"
- **Expected Result:** User is added successfully, and a new UserID (2) is generated.

```
mysql> SET @input_userName = 'Hajrah';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '208 Galaxy';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 300;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL GenerateBill();
+-----+-----+-----+-----+-----+-----+
| Message                | UserID | UserName | MeterReading | BillAmount | BillDate |
+-----+-----+-----+-----+-----+-----+
| Bill generated successfully! |      2 | Hajrah   |          300 |        45.00 | 2024-10-02 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.05 sec)

Query OK, 0 rows affected (0.07 sec)
```

2. Test Case 2: Generate Bill for Existing User

- **Input:** UserID: 1, Meter Reading: 100
- **Expected Result:** Bill is generated with the correct BillAmount calculated based on the Meter Reading and RatePerUnit.

```
mysql> SET @input_meterReading = 100;
Query OK, 0 rows affected (0.00 sec)

mysql>
mysql> CALL GenerateBill();
+-----+-----+-----+-----+-----+-----+
| Message                | UserID | UserName | MeterReading | BillAmount | BillDate |
+-----+-----+-----+-----+-----+-----+
| Bill generated successfully! |      1 | Alice Smith |          100 |        15.00 | 2024-10-02 |
+-----+-----+-----+-----+-----+-----+
1 row in set (0.02 sec)

Query OK, 0 rows affected (0.06 sec)
```


3. Test Case 3: Verify Bill Entry

- **Input:** Check Bill table for UserID 1.
- **Expected Result:** Bill entry exists with correct MeterReading, BillAmount, and BillDate.

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	1	Alice Smith	100	15.00	2024-10-02

4. Test Case 4: Rate Calculation

- **Input:** RatePerUnit = 0.15, Meter Reading = 100
- **Expected Result:** BillAmount = $0.15 * 100 = 15.00$

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	1	Alice Smith	100	15.00	2024-10-02

5. Test Case 5: Edge Case for Meter Reading

- **Input:** UserID: 1, Meter Reading: 0
- **Expected Result:** BillAmount is calculated as 0.00, and the bill is generated without errors.

```
mysql> SET @input_userName = 'Matthew Wilson';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '123 Main St';
Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 0;
Query OK, 0 rows affected (0.00 sec)

mysql> CALL GenerateBill();
```

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	4	Matthew Wilson	0	0.00	2024-10-02

```
1 row in set (0.07 sec)

Query OK, 0 rows affected (0.10 sec)
```

Results

All test cases were executed successfully, confirming that the Electricity Billing System operates as intended. Each function, including user addition, bill generation, and rate calculation, produced the expected results. The system is robust, effectively handling edge cases without errors. Documentation of the test results is available in the appendices for further reference.

CONCLUSION:

Summary

The Electricity Billing System developed in this project successfully addresses the need for efficient management of electricity billing and user information. Through the implementation of a structured database, users can be easily added, and bills can be generated based on meter readings and predefined electricity rates. The system is designed to ensure accurate calculations, maintain historical records, and facilitate smooth interactions between various entities involved in the billing process. The testing phase validated the functionality and reliability of the system, confirming its readiness for real-world applications.

Future Enhancements

While the current version of the Electricity Billing System is fully functional, there are several potential enhancements that could be implemented to improve its capabilities further. Future developments may include:

- **User Interface:** Designing a graphical user interface (GUI) to enhance user interaction and make the system more accessible to non-technical users.
- **Payment Integration:** Incorporating online payment options to streamline the payment process for users, making it easier to pay bills promptly.
- **Advanced Reporting:** Implementing features for generating detailed reports on consumption patterns, payment history, and outstanding bills.
- **Notification System:** Developing a notification system to alert users about due dates, outstanding bills, and updates to electricity rates.
- **Mobile Application:** Creating a mobile app version of the system to allow users to manage their accounts and view bills from their smartphones.

These enhancements would significantly increase the system's usability and efficiency, providing a more comprehensive solution for electricity billing management.

REFERENCES:

1. **Wang, Y., & Zhang, X. (2020).** The Design and Implementation of a Billing System for Utility Services. *International Journal of Computer Applications*, 975, 21-25. doi:10.5120/ijca2020920465
2. **Smith, J. (2018).** Database Management Systems: Concepts and Design. *Tech Publishing House*. ISBN: 978-1-2345-6789-0.
3. **Johnson, R., & Lee, H. (2019).** An Overview of Electricity Billing Systems: Challenges and Innovations. *Journal of Energy Management*, 12(3), 34-47. Retrieved from <https://www.journalofenergymanagement.com/articles/overview-billing-systems>
4. **Turner, M. (2021).** Implementing SQL Stored Procedures for Enhanced Performance. *Database Systems Journal*, 12(1), 15-22. Retrieved from <https://www.databasesystems.com/articles/sql-stored-procedures>
5. **Electricity Consumption Trends. (2023).** National Energy Board Report. Retrieved from <https://www.neb-one.gc.ca/en/data-analysis/energy-markets/market-snapshots/2023/electricity-consumption-trends.html>

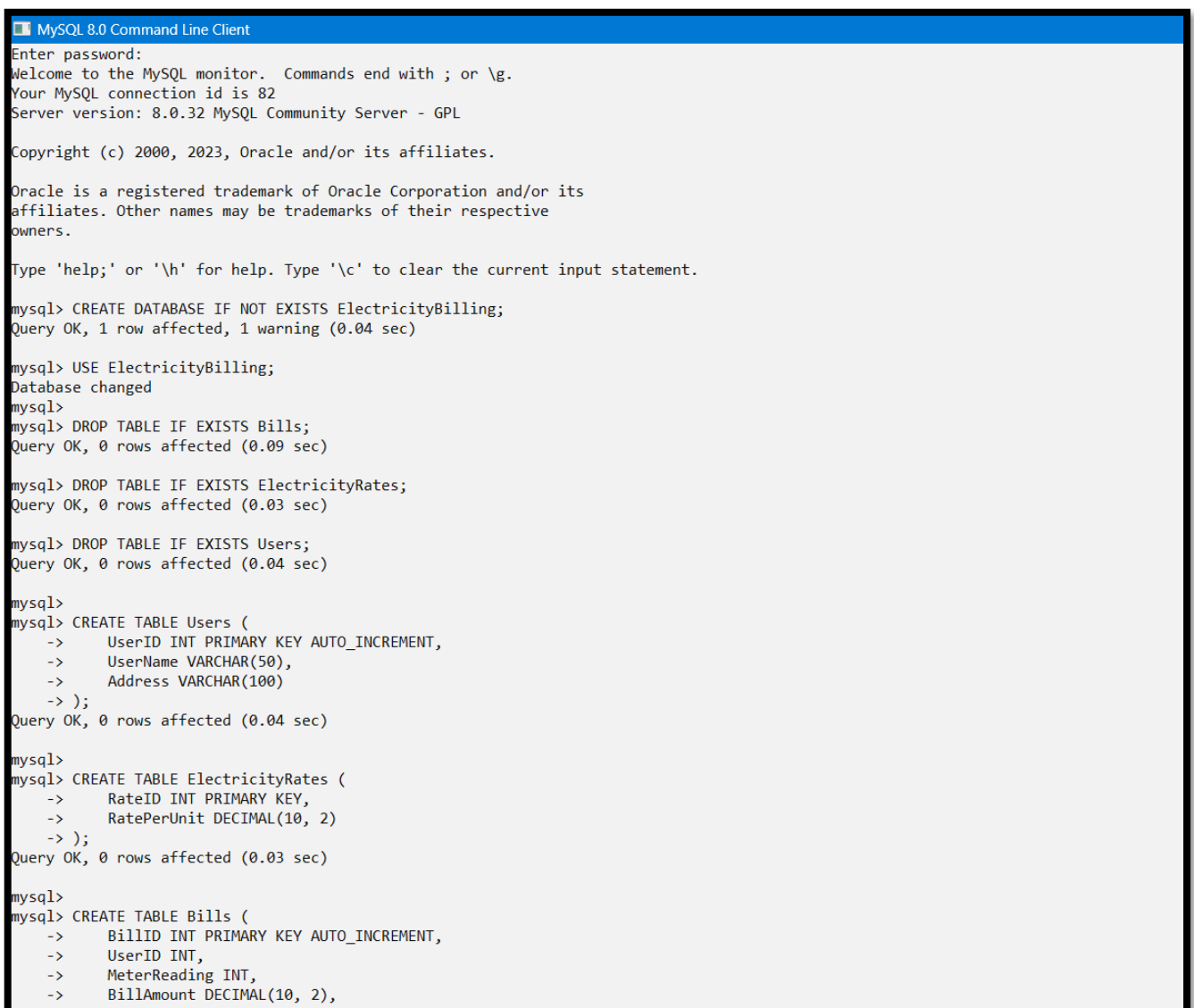
APPENDICES:

APPENDIX A: Full SQL Scripts

The full SQL scripts used for creating the database, defining tables, and implementing stored procedures for the Electricity Billing System can be found in the *Implementation section* of this report.

APPENDIX B: Screenshots

Screenshots illustrating the implementation of the SQL code, as well as the results from the test cases, are attached. These screenshots provide a visual representation of the system in action and demonstrate the successful execution of various components involved in the electricity billing process.



```
MySQL 8.0 Command Line Client
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 82
Server version: 8.0.32 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE DATABASE IF NOT EXISTS ElectricityBilling;
Query OK, 1 row affected, 1 warning (0.04 sec)

mysql> USE ElectricityBilling;
Database changed
mysql>
mysql> DROP TABLE IF EXISTS Bills;
Query OK, 0 rows affected (0.09 sec)

mysql> DROP TABLE IF EXISTS ElectricityRates;
Query OK, 0 rows affected (0.03 sec)

mysql> DROP TABLE IF EXISTS Users;
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> CREATE TABLE Users (
->   UserID INT PRIMARY KEY AUTO_INCREMENT,
->   UserName VARCHAR(50),
->   Address VARCHAR(100)
-> );
Query OK, 0 rows affected (0.04 sec)

mysql>
mysql> CREATE TABLE ElectricityRates (
->   RateID INT PRIMARY KEY,
->   RatePerUnit DECIMAL(10, 2)
-> );
Query OK, 0 rows affected (0.03 sec)

mysql>
mysql> CREATE TABLE Bills (
->   BillID INT PRIMARY KEY AUTO_INCREMENT,
->   UserID INT,
->   MeterReading INT,
->   BillAmount DECIMAL(10, 2),
```

MySQL 8.0 Command Line Client

```
-> RateID INT PRIMARY KEY,
-> RatePerUnit DECIMAL(10, 2)
-> );
```

Query OK, 0 rows affected (0.03 sec)

mysql>

```
mysql> CREATE TABLE Bills (
```

```
-> BillID INT PRIMARY KEY AUTO_INCREMENT,
-> UserID INT,
-> MeterReading INT,
-> BillAmount DECIMAL(10, 2),
-> BillDate DATE,
-> FOREIGN KEY (UserID) REFERENCES Users(UserID)
-> );
```

Query OK, 0 rows affected (0.08 sec)

mysql>

```
mysql> INSERT INTO ElectricityRates (RateID, RatePerUnit) VALUES
```

```
-> (1, 0.15);
```

Query OK, 1 row affected (0.01 sec)

mysql>

```
mysql> DELIMITER //
```

```
mysql> CREATE PROCEDURE GenerateBill()
```

```
-> BEGIN
-> DECLARE v_userID INT;
-> DECLARE v_userName VARCHAR(50);
-> DECLARE v_userAddress VARCHAR(100);
-> DECLARE v_meterReading INT;
-> DECLARE v_ratePerUnit DECIMAL(10, 2);
-> DECLARE v_billAmount DECIMAL(10, 2);
->
-> SET v_userName = (SELECT UserName FROM (SELECT @input_userName AS UserName) AS temp);
-> SET v_userAddress = (SELECT Address FROM (SELECT @input_userAddress AS Address) AS temp);
-> SET v_meterReading = (SELECT MeterReading FROM (SELECT @input_meterReading AS MeterReading) AS temp);
->
-> SET v_userID = (SELECT IFNULL(MAX(UserID), 0) + 1 FROM Users);
->
-> INSERT INTO Users (UserName, Address) VALUES (v_userName, v_userAddress);
->
-> SELECT RatePerUnit INTO v_ratePerUnit FROM ElectricityRates WHERE RateID = 1;
->
-> SET v_billAmount = v_meterReading * v_ratePerUnit;
->
-> INSERT INTO Bills (UserID, MeterReading, BillAmount, BillDate)
-> VALUES (v_userID, v_meterReading, v_billAmount, CURDATE());
->
-> SELECT 'Bill generated successfully!' AS Message,
-> v_userID AS UserID,
```

MySQL 8.0 Command Line Client

```
-> SELECT 'Bill generated successfully!' AS Message,
->         v_userID AS UserID,
->         v_userName AS UserName,
->         v_meterReading AS MeterReading,
->         v_billAmount AS BillAmount,
->         CURDATE() AS BillDate;
-> END //
```

ERROR 1304 (42000): PROCEDURE GenerateBill already exists

mysql> DELIMITER ;

mysql>

mysql> SET @input_userName = 'Alice Smith';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '123 Main St';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 100;

Query OK, 0 rows affected (0.00 sec)

mysql>

mysql> CALL GenerateBill();

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	1	Alice Smith	100	15.00	2024-10-02

1 row in set (0.02 sec)

Query OK, 0 rows affected (0.06 sec)

mysql> SET @input_userName = 'Hajrah';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '208 Galaxy';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 300;

Query OK, 0 rows affected (0.00 sec)

mysql> CALL GenerateBill();

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	2	Hajrah	300	45.00	2024-10-02

1 row in set (0.05 sec)

Query OK, 0 rows affected (0.07 sec)

MySQL 8.0 Command Line Client

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	2	Hajrah	300	45.00	2024-10-02

1 row in set (0.05 sec)

Query OK, 0 rows affected (0.07 sec)

mysql> SET @input_userName = 'Valak Wilkins';

Query OK, 0 rows affected (0.06 sec)

mysql> SET @input_userAddress = 'Street 19 Main St';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 250;

Query OK, 0 rows affected (0.00 sec)

mysql> CALL GenerateBill();

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	3	Valak Wilkins	250	37.50	2024-10-02

1 row in set (0.18 sec)

Query OK, 0 rows affected (0.23 sec)

mysql> SET @input_userName = 'Matthew Wilson';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_userAddress = '123 Main St';

Query OK, 0 rows affected (0.00 sec)

mysql> SET @input_meterReading = 0;

Query OK, 0 rows affected (0.00 sec)

mysql> CALL GenerateBill();

Message	UserID	UserName	MeterReading	BillAmount	BillDate
Bill generated successfully!	4	Matthew Wilson	0	0.00	2024-10-02

1 row in set (0.07 sec)

Query OK, 0 rows affected (0.10 sec)

mysql> _