**Project Presentation**

on

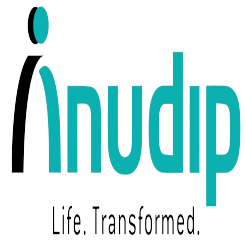
### “ELECTRICITY BILLING SYSTEM”

Submitted by

**Mr. Gaurav Marathe**

Under the guidance of

#### Trainer :- Priti Yadav



**Academic Year - 2023-2024**

### ABSTRACT

Electricity consumers are often faced with the problem of inaccuracy and delay in monthly billing due to some drawbacks. Thus, it is essential to have an efficient system for such purposes via electronic platform with consideration to proximity. The proposed system automates the conventional process of paying electricity bill by visiting the Electricity Board which is tiresome and time consuming. It is also designed to automate the electricity bill calculation and payment for user convenience. The system is developed with Java swings as the base programming language which can be used to develop websites, web applications and web services. The Microsoft Structured Query Language (SQL) server is also used for creating back-end database. The system would be having two logins: the administrative and user login. The administrator can view the user's account details and can add the customer's information of consuming units of energy of the current month in their account. The Admin must feed the system with the electricity usage data into respective user’s account. The system then calculates the electricity bill for every user and updates the information into their account every month. Users can then view their electricity bill and pay before the month end.

### TABLE OF CONTENTS

**Abstract i**

Acknowledgment ii

[Table of contents iii](#_TOC_250026)

1. Introduction
   1. [Preamble 1-2](#_TOC_250024)
   2. [Problem statement 2](#_TOC_250023)
   3. [Proposed solution 2](#_TOC_250022)
2. Analysis and System Requirements
   1. [Existing and Proposed System 3](#_TOC_250021)
   2. [Software & Hardware Requirements 3](#_TOC_250020)
3. System Design and Modelling
   1. [Preliminary Design](#_TOC_250019)
      1. [Entity-Relationship Diagram 4-6](#_TOC_250018)
      2. [Schema Diagram 7-8](#_TOC_250017)
   2. [Normalization](#_TOC_250016)
      1. [First normal form(1NF) 9](#_TOC_250015)
      2. [Second normal form(2NF) 9](#_TOC_250014)
      3. [Third normal form(3NF) 9](#_TOC_250013)
4. Implementation
   1. [Implementation of operations 10](#_TOC_250012)
   2. [Implementation of SQL statements 10-11](#_TOC_250011)
   3. [Algorithm or pseudocode of implementation 12-15](#_TOC_250010)
5. Testing
   1. [Testing process 16](#_TOC_250009)
   2. [Testing objectives 16](#_TOC_250008)
   3. [Levels of Testing](#_TOC_250007)
      1. [Unit Testing 16-18](#_TOC_250006)
      2. [Integration testing 18-19](#_TOC_250005)
      3. [System testing 19-20](#_TOC_250004)
6. Discussion and Snapshots
   1. [Tables 21-23](#_TOC_250003)

### iii

# INTRODUCTION

Electricity Billing System is a software-based application.

1. This project aims at serving the department of electricity by computerizing the billing system.
2. It mainly focuses on the calculation of units consumed during the specified time and the money to be charged by the electricity offices.
3. This computerized system will make the overall billing system easy, accessible, comfortable, and effective for consumers.

To design the billing system more service oriented and simple, the following features have been implemented in the project. The application has high speed of performance with accuracy and efficiency.

The software provides facility of data sharing, it does not require any staff as in the conventional system. Once it is installed on the system only the meter readings are to be given by the admin where customer can view all details, it has the provision of security restriction.

The electricity billing software calculates the units consumed by the customer and makes bills, it requires small storage for installation and functioning. There is provision for debugging if any problem is encountered in the system.

The system excludes the need of maintaining paper electricity bill, administrator does not have to keep a manual track of the users, users can pay the amount without visiting the office. Thus, it saves human efforts and resources.

## Preamble

We, the owners of our project, respect all customers and make them happy with our service.

The main aim of our project is to satisfy customer by saving their time by payment process, maintaining records, and allowing the customer to view his/her records and permitting them to update their details.

The firm handles all the work manually, which is very tedious and mismatched.

The objectives of our project are as follows:

* + - To keep the information of Customer.
    - To keep the information of consuming unit energy of current month.
    - To keep the information of consuming unit energy of previous month.
    - To calculate the units consumed every month regularly.
    - To generate the bills adding penalty and rent.
    - To save the time by implementing payment process online.

## Problem Statement

The manual system is suffering from a series of drawbacks. Since whole of the bills is to be maintained with hands the process of keeping and maintaining the information is very tedious and lengthy to customer. It is very time consuming and laborious process because, staff need to be visited the customers place every month to give the bills and to receive the payments. For this reason, we have provided features Present system is partially automated(computerized), existing system is quite laborious as one must enter same information at different places.

## Proposed Solution

* This project system excludes the need of maintaining paper electricity bill as all the electricity bill records are managed electronically.
* Administrator doesn't have to keep a manual track of the users. The system automatically calculates fine.
* Users don't have to visit to the office for bill payment.
* There is no need of delivery boy for delivery bills to user's place.
* Thus, it saves human efforts and resources.

# ANALYSIS AND SYSTEM REQUIREMENT

## Existing and Proposed System

The conventional system of electricity billing is not so effective; one staff must visit each customer’s house to note the meter readings and collect the data. Then, another staff must compute the consumed units and calculate the money to be paid. Again, the bills prepared are to be delivered to customers. Finally, individual customer must go to electricity office to pay their dues.

Hence, the conventional electricity billing system is uneconomical, requires many staffs to do simple jobs and is a lengthy process overall. In order to solve this lengthy process of billing, a web based computerized system is essential. This proposed electricity billing system project overcomes all these drawbacks with the features. It is beneficial to both consumers and the company which provides electricity.

With the new system, there is reduction in the number of staffs to be employed by the company. The working speed and performance of the software is faster with high performance which saves time. Furthermore, there is very little chance of miscalculation and being corrupted by the staffs.

## Software & Hardware Requirements

#### Hardware Requirements:

* + - Hardware Specification: -Processor Intel Pentium V or higher
    - Clock Speed: -1.7 GHz or more
    - System Bus: -64 bits
    - RAM: -16GB
    - HDD: -2TB
    - Monitor: -LCD Monitor
    - Keyboard: -Standard keyboard
    - Mouse: -Compatible mouse

#### Software Requirements:

* + - Operating System: -Windows 10
    - Software: -Microsoft SQL Server
    - Front End: -Java core/swings (NetBeans, Eclipse)
    - Back End: -MySQL

# SYSTEM DESIGN AND MODELLING

## Preliminary Design

System design is an abstract representation of a system component and their relationship and which describe the aggregated functionally and performance of the system. It is also the plan or blueprint for how to obtain answer to the question being asked. The design specifies various type of approach.

Database design is one of the most important factors to keep in mind if you are concerned with application performance management. By designing your database to be efficient in each call it makes and to effectively create rows of data in the database, you can reduce the amount of CPU needed by the server to complete your request, thereby ensuring a faster application.

## Entity-Relationship Diagram

An entity relationship diagram (ERD) shows the relationships of entity sets stored in a database. An entity in this context is an object, a component of data. An entity set is a collection of similar entities. These entities can have attributes that define its properties.

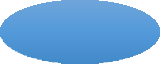
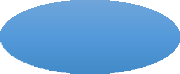
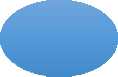
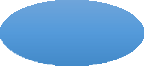
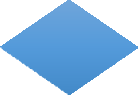
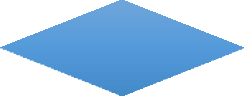
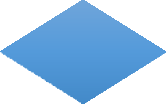
There are two reasons to create a database diagram. You’re either designing a new schema or you need to document our existing structure.

If you have an existing database you need to document, you create a database diagram using data directly from your database. You can export your data base structure as a CSV file (there are some scripts on how to do this here), then have a program generate the ERD automatically.

An ER diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

* + - * Entities, which are represented by rectangles. An entity is an object or concept about which you want to store information.
      * A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.
      * Actions, which are represented by diamond shapes, show how two entities share information in the database.
      * In some cases, entities can be self-linked. For example, employees can supervise other employees.
      * Attributes, which are represented by ovals. A key attribute is the unique, distinguishing characteristic of the **no** entity.
      * A multivalued attribute can have more than one value. For example, an employee entity can have multiple skill values.
      * A derived attribute is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.
      * Connecting lines, solid lines that connect attributes to show the relationships of entities in the diagram.
      * Cardinality specifies how many instances of an entity relate to one instance of another entity. Ordinality is also closely linked to cardinality.

Figure 3.1.1 describes the ER diagram of Electricity Billing System. It has 5 entities namely login, customer, tax, bill, and meter info. The entities have attributes which are primary and foreign and attributes. The primary attributes are underlined.



**user**

Service\_charge

**meter\_**

#### ER Diagram

**cost\_per\_unit**

**1**

**rent**

**meter\_rent**

**gst**

**1 service\_tax**

**tax**

**swacch\_bharat\_cess**

**N**

**pays**

**total\_bill**

**month**

**meter\_no**

**bill**

**units**

**status**

**meter\_no**

**meter\_no**

**city**

**address**

**name**

**N N**

**manages**

**Customer**

**meter\_location**

**days**

**meter\_info**

**state**

**bill\_type**

**phase\_code**

**email**

has

**login**

## Schema Diagram

Database schema is described as database connections and constraints. It contains attributes. Every database has a state instances represent current set of databases with values. There are different types of keys in a database schema.

A primary key is a table column that can be used to uniquely identify every row of the table. Any column that has this property, these columns are called candidate key. A composite primary key is a primary key consisting of more than one column. A foreign is a column or combination of columns that contains values that are found in the primary key of some table.

All the attributes of each table are interconnected by foreign key which is primary key in another column and composite key. Primary key cannot be null. The fact that many foreign key values repeat simply reflects the fact that its one-to-many relationship. In one-to-many relationship, the primary key has the one value and foreign key has many values.

Figure 3.1.2 is a Schema diagram of Electricity Billing System which has six tables i.e., login, customer, tax, rent, bill, and meter\_info where each table contain attributes some with primary key, foreign key. In the login table there are 6 attributes "meter\_no”, "username”, “password”, "user”, "question”, "answer”. The customer table has 7 attributes "name”, "meter\_no"(primary key), "address”, "city”, "state”, “email”, "phone”. The rent table has 3 attributes "cost\_per\_unit"(primary key), " meter\_rent”, "service\_charge". The tax table has 3 attributes " service\_tax”, "swacch\_bharat\_cess”, "gst”. The bill table has 5 attributes "meter\_no"(foreign key that references the primary key of the customer table meter\_no), "month”, "units”,"total\_bill”, "status”. The meter\_info table has 6 attributes "meter\_no"(foreign key that references the primary key of the customer table meter\_no), "meter\_location”, "meter\_type”, "phase\_code”, " bill\_type”, "days ".

#### 3.1.2 Schema Diagram Login

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| meter\_no | Username | Password | user | question | Answer |

**customer**

**rent**

**tax**

**bill**

**meter\_info**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | meter\_no | Address | City | state | Email | phone |

|  |  |  |
| --- | --- | --- |
| cost\_per\_unit | meter\_rent | service\_rent |

|  |  |  |
| --- | --- | --- |
| service\_tax | swacch\_bharat\_cess | Gst |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| meter\_no | Month | units | total\_bill | Status |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| meter\_no | meter\_location | meter\_type | phase\_code | bill\_type | Days |

**FIG 3.1.2: Schema diagram of Electricity Billing System**

## 

# IMPLIMENTATION

## Implementation of operations

* + - **Adding Customer:** Here admin can add new customer to the customer list who started using electricity bill system.
    - **Searching Deposit Details:** Here admin can search according to meter number and month to view deposit details.
    - **Viewing Details**: Here admin and user can view customer details and about details.
    - **Adding Tax:** Here admin can add tax details.
    - **Updating Customer:** Here customer can update his/her details by using meter\_no of the customer.
    - **Delete Customer:** Here admin can delete details based on meter number.

## Implementation of SQL statements

#### Insert statement:

* The INSERT INTO statement is used to insert new records in a table.
* The INSERT INTO syntax would be as follows: INSERT INTO table\_name VALUES (value1, value2, value3, ...).
* The following SQL statement insert's a new record in the “customer” table: Insert into customer VALUES (“sai”,”12345”,” btm”,” Bangalore”, “Karnataka”, “[sai@gmail.com](mailto:sai@gmail.com)”, “9876543333”).

#### Update statement:

* An SQL UPDATE statement changes the data of one or more records in a table. Either all the rows can be updated, or a subset may be chosen using a condition.
* The UPDATE syntax would be as follows: UPDATE table\_name SET column\_name =value, column\_name=value... [WHERE condition].
* The following SQL statement update's a new record in the “customer” table: UPDATE TABLE customer SET email= [su@gmail.com](mailto:su@gmail.com) WHERE meter\_no

=”12345”.

#### Delete statement:

* The DELETE statement is used to delete existing records in a table.
* The DELETE syntax would be as follows: DELETE FROM table\_nameWHERE condition.
* The following SQL statement delete's a record in the “customer” table: delete from customer where meter\_no=12345.

#### Create statement:

* The CREATE TABLE Statement is used to create tables to store data. Integrity Constraints like primary key, unique key, foreign key can be defined for the columns while creating the table.
* The syntax would be as follows: CREATETABLE table\_name (column1datatype, column2datatype, column3 datatype, columnN datatype, PRIMARY KEY (one or more columns)).
  + The following SQL statement creates a table “customer” table: create table customer (name varchar (30), meter\_no varchar (20) primary key, address varchar (50), city varchar (20), state varchar (30), email varchar (30), phone varchar (30));
  + The following SQL statement creates a table “login” table: create table login (meter\_no varchar (30), username varchar (30), password varchar (30), user varchar (30), question varchar (40), answer varchar (30));
  + The following SQL statement creates a table “tax” table: create table tax (cost\_per\_unit int (20) primary key, meter\_rent int (20), service\_charge int (20), service\_tax int (20), swacch\_bharat\_cess int (20), gst int (20));
  + The following SQL statement creates a table “bill” table: create table bill (meter\_no varchar (20), foreign key(meter\_no) references customer(meter\_no) on delete cascade, month varchar (20), units int (20), total\_bill int (20), status varchar (40));
  + The following SQL statement creates a table “meter\_info” table: create table meter\_info (meter\_no varchar (30), foreign key(meter\_no) references customer(meter\_no) on delete cascade, meter\_location varchar (10), meter\_type varchar (15), phase\_code int (5), bill\_type varchar (10), days int (5));

## Algorithm or pseudocode of implementation

#### Explanation of Algorithm or pseudocode of system:

* Start system
* Enter login name and password
* On clicking the login button
* Connect to database
* Query database to know whether user credentials are correct
* If not, deny access and return login page with an error message
* If correct, check if credentials for administrator
* If yes, allow login
* Set admin session, re-direct administrator to admin login page
* If no, allow login set user session
* Re-direct user to user home page

#### Algorithm or pseudocode of admin:

**Login:**

* This program will allow the admin to enter the username and password.
* If the entered credentials are correct, then the login will be successful otherwise need to be signup.
* If admin forgets password, it can be retrieved by giving username and answer for security question.
* After successful login the admin will be redirected to admin portal page where he/she can do following activities.

#### NewCustomer:

* This program will allow the admin to enter the customer details and automatically generates unique meter number.
* If customer name, address, city, state, email and phone number is entered, insert the values into customer

else print error while next=true

enter the meter\_info details else print meter\_info error

Submit the details of customer that has been entered by clicking onto next button.

* If we need to cancel the particulars that has been entered click onto cancel option.
* If we need to submit the particulars that has been entered click onto submit option.

#### CustomerDetails:

* This program will allow the admin to view customer details.
* If we need to print the particulars that has been viewed click onto print option.

#### DepositDetails:

* This program will allow the admin to view bill details. If we need to sort the particulars based on meter\_no and month.
* If we need to search the particulars that has been viewed click onto search option.
* If we need to print the particulars that has been viewed click onto print option.

#### TaxDetails:

* This program will allow the admin to add tax details. insert the values into tax

else print error

Submit the details of tax that has been entered by clicking onto submit button.

* If we need to cancel the particulars that has been entered click onto cancel option.

#### CalculateBill:

* This program will allow the admin to calculate total\_bill when units consumed are inserted where meter\_no and month is selected.

insert the values into bill else print error

Submit the details of tax that has been entered by clicking onto submit button.

* If we need to cancel the particulars that has been entered click onto cancel option.

#### DeleteBill:

* This Program will allow the admin to delete the customer info when meter\_no is selected.
* If we need to delete the particulars that has been saved click onto delete option.
* If we need to cancel the particulars that has been entered click onto back option.

#### About:

* This program will allow the admin to view details of the project in short.
* If we need to exit the particulars that has been viewed click onto exit option.

#### Algorithm or pseudocode of Customer:

**Login:**

* This program will allow the customer to enter the username and password. If the entered credentials are correct, then the login will be successful otherwise need to be signup with the meter\_no which is given by admin.
* If customer forgets password, it can be retrieved by giving username and answer for security question. After successful login the customer will be redirected to customer portal page where he/she can do following activities.

#### UpdateInfo1:

* This program will allow the customer to update the customer details. If customer address, city, state, email and phone number is updated,

update the values into customer else print error

update the details of customer that has been updated by clicking onto update button.

* If we need to cancel the particulars that has been updated, click onto back option.

#### ViewInfo:

* This program will allow the customer to view his/her own details.
* If we need to go back from the particulars that has been viewed click onto back option.

#### PayBill:

* This program will allow the customer to view bill details and redirects to pay
* the bill where status will be updated.
* If we need to cancel the particulars that has been viewed click onto back option.
* If we need to pay the bill amount that has been viewed click onto pay option.

#### BillDetails:

* This program will allow the customer to view bill details.
* If we need to print the particulars that has been viewed click onto print option.

#### GenerateBill:

* This program will allow the customer to generate bill when meter\_no and month is selected.
* Generate the details by clicking on generatebill button.

#### About:

* This program will allow the customer to view details of the project in short.
* If we need to exit the particulars that has been viewed click onto exit option.

NOTE: Utility (notepad, browser, calculator),query and logout is given to both customer and admin portals.

# TESTING

This chapter gives the outline of all the testing methods that are carried out to get a bug free application.

## Testing process

Testing is an integral part of software development. Testing process, in a way certifies, whether the product, that is developed, compiles with the standards, that it was designed to. Testing process involves building of test cases, against which, the product has to be tested. In some cases, test cases are done based on the system requirements specified for the product/software, which is to be developed.

## Testing objectives

The main objectives of testing process are as follows:

* Testing is a process of executing a program with the intent of finding an error.
* A good test case is one that has high probability of finding an as yet undiscovered error.
* A successful test is one that uncovers an as yet undiscovered error.

## Levels of Testing

Different levels of testing are used in the testing process; each level of testing aims to test different aspects of the system. The basic levels are unit testing, integration testing, system testing and acceptance testing.

#### Unit Testing

Unit testing focuses verification effort on the smallest unit of software design the module. The software built, is a collection of individual modules. In this kind of testing exact flow of control for each module was verified. With detailed design consideration used as a guide, important control paths are tested to uncover errors within the boundary of the module.

#### Table 5.1: Negative test case for phone number insertion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input phone number | 98977 | Phone number is invalid | Length of phone number is not equal to  10 | Consume () |
| Input phone number | 98977agv | Phone number is invalid | Alphabets are being taken as input for phone  number | \_ |

**Table 5.2: Positive test case for phone number insertion**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input phone number | 8888161301 | Expected output is seen | \_ | \_ |

#### Table 5.3: Negative test case for email insertion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input email | aksh.in | Email is invalid | Email is not in a format given | Consume () |

**Table 5.4: Positive test case for email insertion**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input email | akshay@gmail.com | Expected output is seen | \_ | \_ |

#### Table 5.5: Negative test case for customer name insertion

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input customer name | akshay123 | Name is invalid | Numbers are being taken as input for name | Consume () |

**Table 5.6: Positive test case for customer name insertion**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Input customer  Name | Akshay | Expected output is seen | **\_** | **\_** |

#### Integration testing

The second level of testing is called integration testing. In this, many class-tested modules are combined into subsystems, which are then tested. The goal here is to see if all the modules can be integrated properly. We have been identified and debugged.

#### Table 5.7: Test case on basis of generation of bill

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Negative searching of total\_bill | 12334(meter\_no) January(month) | Details seen but not total\_bill | Output not seen | Consume () |
| Positive searching of total\_bill | 12334(meter\_no) January(month) | Must display full generated bill with  total\_bill | **\_** | **\_** |

**Table 5.8: Test case on basis of depositedetails**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Function Name** | **Input** | **Expected Output** | **Error** | **Resolved** |
| Negative searching of depositedetails | 12334(meter\_no) January(month) | Details not seen | Output not seen | Consume () |
| Positive searching of total\_bill | 12334(meter\_no) January(month) | Must display depositedetails | \_ | \_ |

#### System testing

Here the entire application is tested. The reference document for this process is the requirement document, and the goal is to see IF the application meets its requirements. Each module and component of ethereal was thoroughly tested to remove bugs through a system testing strategy. Test cases were generated for all possible input sequences and the output was verified for its correctness.

**Table 5.9: Test cases for the project**

|  |  |  |
| --- | --- | --- |
| **Steps** | **Action** | **Expected output** |
| Step1 choice | The screen appears when the users run the program.  1.If admin login 2.If customer login | A page with different menu’s appears.  1.Admin panel opens and 2.Customer panel opens |
| Step 2 | The screen appears when the admin logs in and selects any one of the  menus from the click of the mouse. | A window for adding new customer, inserting tax, calculate bill, view deposit details etc |
| Selection 1 | * New Customer * Customer Details   + Deposit Details     - Calculate Bill       * Tax Details   + Delete Customer |  |

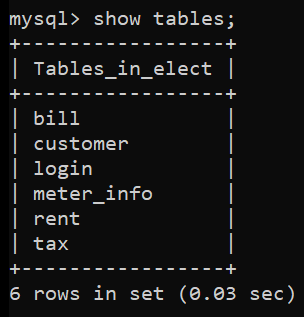
|  |  |  |
| --- | --- | --- |
| Step 2.1 | The screen appears when the customer login and selects any one of the menus from the click of  the mouse | A window for generating bill, update customer details, view details, generating bill |
| Selection 2 | * Update Details   + View Details |  |
| Selection 2a | * Generate Bill |  |
| Selection 2b | * Pay Bill * Bill Details |  |

# DISCUSSION AND SNAPSHOTS

#### TABLES:

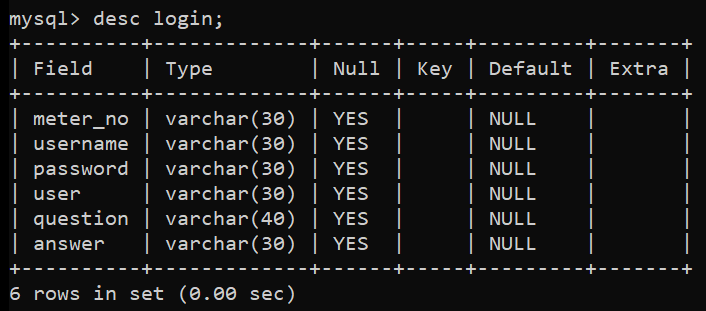
The given below table is a snapshot of backend view of the localhost and the structures of the tables present in Electricity Billing System. The tables present are login, customer, tax, bill, meter\_info.

* + - The login is used to store the details of login’s admin and customer with meter\_no.
    - The customer is used to store details of customer.
    - The tax is used to store tax values.
    - The rent is used to store rent values.
    - The bill is used to store details of bill of meter.
    - The meter\_info is used to store information of meter placed.



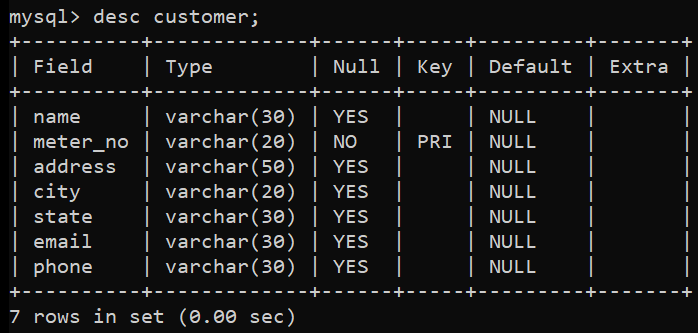
#### FIG 6.1:List of tables

**Login Table:**



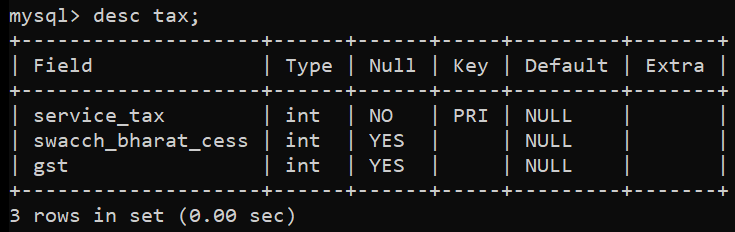
#### FIG 6.2:login table description

**Customer Table:**



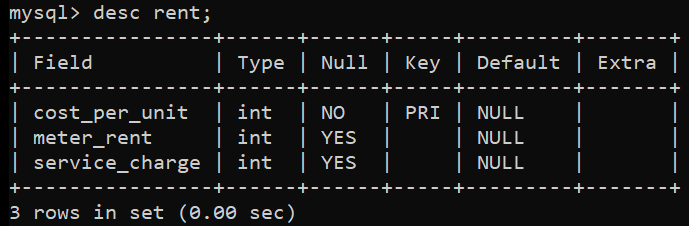
#### FIG 6.3: customer table description

**Tax Table:**



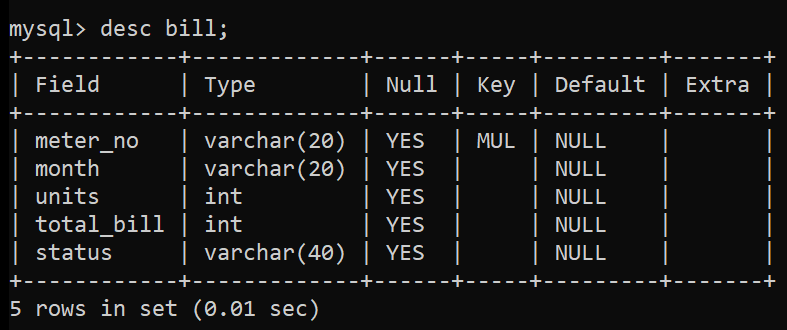
#### FIG 6.4: tax table description

**Rent Table:**

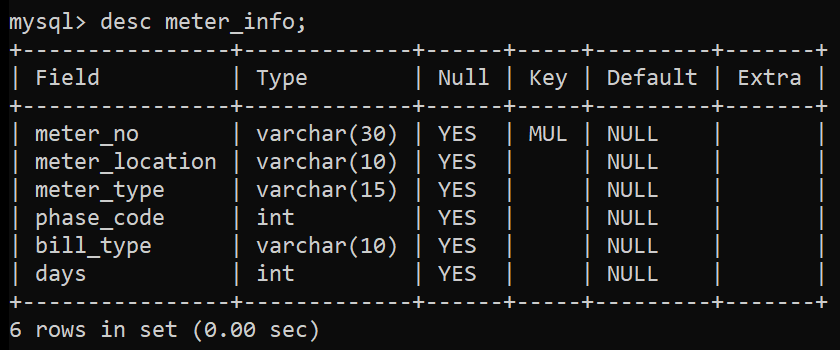


#### FIG 6.5: rent table description

**Bill Table:**



#### FIG 6.6: bill table description

**Meter\_Info Table:**

**FIG 6.7: meter\_info table description**