

## **TABLE OF CONTENTS**

<b>ABSTRACT</b>	.....
<b>Chapter 1</b>	.....
1.1 Introduction	.....
1.2 Proposed Timeline Table	.....
1.3 Requirements Traceability Matrix	.....
1.4 Tools	.....
1.5 Code Level	.....
<b>Chapter 2</b>	.....
2.1 Objective of the Project	.....
2.2 Scope of the Project	.....
<b>Chapter 3</b>	.....
3.1 Background	.....
3.2 Existing and Proposed System	.....
<b>Chapter 4</b>	.....
4.1 Overview	.....
4.2 Feasibility Study	.....
<b>Chapter 5</b>	.....
5.1 Software Requirement And Specification	.....
5.2 Hardware Requirement And Specification	.....
<b>Chapter 6</b>	.....
6.1 System Design	.....
6.2 ER Diagram	.....
6.3 Schema Diagram	.....
<b>Chapter 7</b>	.....
7.1 System Implementation	.....
7.2 Implementation of SQL Statements	.....
7.3 Algorithm of Implementation	.....
<b>Chapter 8</b>	.....

<b>8.1 Testing Process</b>	.....
<b>8.2 Graphs</b>	.....
<b>8.3 Tables</b>	.....
<b>Chapter 9</b>	.....
<b>9.1Snapshot</b>	.....
<b>Chapter 10</b>	.....
<b>10.1 Future Enhancement</b>	.....
<b>Chapter 11</b>	.....
<b>11.1 Conclusion</b>	.....
<b>Chapter 12</b>	.....
<b>12.1 Bibliography</b>	.....

### **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 unitsof energy of the current month in their account. The Admin must feed the systemwith the electricity usage data into respective user's account. The system then calculates the electricity bill for every user and updates the information into theiraccount every month. Users can then view their electricity bill and pay before themonth end.

## **CHAPTER 1**

### **1.1 INTRODUCTION**

Electricity Billing System is a software-based application.

- i. This project aims at serving the department of electricity by computerizing the billing system.
- ii. It mainly focuses on the calculation of units consumed during the specified time and the money to be charged by the electricity offices.
- iii. 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.

**1.2 Proposed Timeline Table**

Proposed timeline is the timeline which are made by guess in a time bound and actual timeline means the work history which is make after completed work.

**Table 01: Proposed Timeline Table**

<b>Task</b>	<b>Deadline</b>
Proposal	Within 5 <sup>th</sup> week of November
Requirement Analysis, Specification	Within 20 <sup>th</sup> December
Designing, Study	Within 3 <sup>rd</sup> January
Coding	Within 1st February
Final Testing	Within 5 <sup>th</sup> week of February

**Table 02: Actual Timeline Table**

<b>Task</b>	<b>Deadline</b>
Proposal	Within 5 <sup>th</sup> week of November
Requirement Analysis, Specification	Within 20 <sup>th</sup> December
Designing, Study	Within 3 <sup>rd</sup> January
Coding	Within 1st February
Final Testing	Within 5 <sup>th</sup> week of February

### 1.3 Requirements Traceability Matrix

A Traceability Matrix is a document that co-relates any two-baseline documents that require a many-to-many relationship to check the completeness of the relationship.

Req1 = 2D Animation

Req2 = Objectives

Selection Req3 =

Moving wall

Req4 = Collision

Detection Req5 =

Moving Background

Req6 = Score Counting

**Table 03: Requirement Traceability Table**

<b>Requirement Test Case</b>	<b>Req 1</b>	<b>Req 2</b>	<b>Req 3</b>	<b>Req 4</b>	<b>Req 5</b>	<b>Req 6</b>
Test Case 1						
Test Case 2	✓					
Test Case 3	✓	✓				
Test Case 4	✓	✓				
Test Case 5	✓	✓	✓	✓		
Test Case 6	✓	✓	✓	✓	✓	✓

### **1.4 Tools**

**Language:** JAVA

**IDE:** An IDE (Integrated Development Environment) contains a code editor, a compiler or interpreter, and a debugger, accessed through a single graphical user interchange (GUI). Our IDE is NetBeans.

### **1.5 Code Level**

**LOC** - Line of code in the whole project. The Total number of code lines in whole project.

**NCLOC** - Non-comment line of code. The line which are not commented and this code accomplish our objective.

**CLOC** - Comment line of code. The line of code which are not working to calculate output.

**Density of Comments** - It means the number of comment line proportion to average code.

**Average LOC in a Class** - The average number of line of code in each class.

**Table 0: Code Level Table**

LOC	970
NCLOC	778
CLOC	192
Average LOC	88
Density of Comments	19.8 %

## **Chapter 2**

### **2.1 Objective of the Project:**

The main objective of the project on the Electricity Billing System is to manage the details of Electricity. Unit of Energy, Bill, Store Record, Electricity Board. It manages all the information about Electricity, Connections, Electricity Board, Electricity. The Project is totally build at administrative and thus only the administrator is granted the access. The purpose of the project is to build an application program to reduce the manual work for managing the Electricity, Unit of energy, Connections, Bill. It tracks all the details about the Bill, Store Record, Electricity Board.

#### **Functionalities provided by Electricity Billing System are as follows:**

- Provides the searching facilities based on various factors. Such as Electricity, Bill, Store Record, Electricity Board.
- Electricity Billing System also manage the Connections details online for Store Record Details, Electricity Board Details, Electricity.
- It tracks all the information of Unit of Energy, Connections, Store Record, etc.
- Manage the information of Unit of Energy.
- Shows the information and description of the Electricity Bill.
- To increase efficiency of managing the Electricity, Unit of Energy.
- It deals with monitoring the information and transactions of Store Record.
- Manage the Information of Electricity.
- Editing, adding and updating of Records is improved which results in proper resource management of Electricity Data.
- Manage the information of Store Record.
- Integration of all records of Electricity Board.



## **2.2 Scope of the Project:**

It may help collecting perfect management in details. In a very short time, the collection will be obvious, simple and sensible. It will help a person to know the management of passed year perfectly and vividly. It also helps in current all works relative to Electricity Billing System. It will be also reduced the cost of collecting the management & collection procedure will go on smoothly.

Our project aims at Business process automation, i.e. we have tried to computerize various processes of Electricity Billing System.

- In computer system the person has to fill the various forms & number of copies of the forms can be easily generated at a time.
- In computer system, it is not necessary to create the manifest but we can directly print it, which saves our time.
- To assist the staff in capturing the effort spent on their respective working areas.
- To utilize resources in an efficient manner by increasing their productivity through automation.
- The system generates types of information that can be used for various purposes.
- It satisfy the user requirement.
- Be easy to understand by the user and the operator.
- Be easy to operate.
- Have a good user interface.
- Be expandable.
- Delivered on schedule within the budget.

## **Chapter 3**

### **3.1 INTRODUCTION**

Electricity Billing System is a software-based application.

- iv. This project aims at serving the department of electricity by computerizing the billing system.
- v. It mainly focuses on the calculation of units consumed during the specified time and the money to be charged by the electricity offices.
- vi. 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.

## **3.2 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.

## Chapter 4

### JAVA

**Java** is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let programmers *write once, run anywhere* (WORA), meaning that compiled Java code can run on all platforms that support Java without the need to recompile. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture.

The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages. As of 2019, Java was one of the most popular programming languages in use according to GitHub, particularly for client-server web applications, with a reported 9 million developers.

Java was originally developed by James Gosling at Sun Microsystems. It was released in May 1995 as a core component of Sun Microsystems' Java platform. The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun had relicensed most of its Java technologies under the GPL-2.0-only license. Oracle offers its own HotSpot Java Virtual Machine, however the official reference implementation is the OpenJDK JVM which is free open-source software and used by most developers and is the default JVM for almost all Linux distributions.

## HISTORY OF JAVA

James Gosling, Mike Sheridan, and Patrick Naughton initiated the Java language project in June 1991. Java was originally designed for interactive television, but it was too advanced for the digital cable television industry at the time. The language was initially called Oak after an oak tree that stood outside Gosling's office. Later the project went by the name Green and was finally renamed Java, from Java coffee, a type of coffee from Indonesia. Gosling designed Java with a C/C++-style syntax that system and application programmers would find familiar.<sup>[25]</sup>

Sun Microsystems released the first public implementation as Java 1.0 in 1996. It promised write once, run anywhere (WORA) functionality, providing no-cost run-times on popular platforms. Fairly secure and featuring configurable security, it allowed network- and file-access restrictions. Major web browsers soon incorporated the ability to run Java applets within web pages, and Java quickly became popular. The Java 1.0 compiler was re-written in Java by Arthur van Hoff to comply strictly with the Java 1.0 language specification. With the advent of Java 2 (released initially as J2SE 1.2 in December 1998 – 1999), new versions had multiple configurations built for different types of platforms. J2EE included technologies and APIs for enterprise applications typically run in server environments, while J2ME featured APIs optimized for mobile applications. The desktop version was renamed J2SE. In 2006, for marketing purposes, Sun renamed new J2 versions as Java EE, Java ME, and Java SE, respectively.

In 1997, Sun Microsystems approached the ISO/IEC JTC 1 standards body and later the Ecma International to formalize Java, but it soon withdrew from the process. Java remains a de facto standard, controlled through the Java Community Process. At one time, Sun made most of its Java implementations available without charge, despite their proprietary software status. Sun generated revenue from Java through the selling of licenses for specialized products such as the Java Enterprise System.

On November 13, 2006, Sun released much of its Java virtual machine (JVM) as free and open-source software (FOSS), under the terms of the GPL-2.0-only license. On May 8, 2007, Sun finished the process, making all of its JVM's core code available under free software/open-source distribution terms, aside from a small portion of code to which Sun did not hold the copyright

### FEATURES AND BENEFITS OF JAVA

- **Java is relatively easy to use.**
- **Java is an object-oriented programming language.**
- **Java is secure.**
- **Java is inexpensive to maintain.**
- **Java is inexpensive to maintain.**
- **Java is a high-level programming language.**
- **Java is super portable.**
- **Java is stable.**
- **Java supports multi-threading.**
- **Java is a distributed language.**

### WHAT CAN JAVA DO?

The general-purpose, high-level Java programming language is a powerful software platform. Every full implementation of the Java platform gives you the following features:

- **Development Tools:** The development tools provide everything you'll need for compiling, running, monitoring, debugging, and documenting your applications. As a new developer, the main tools you'll be using are the javac compiler, the java launcher, and the javadoc documentation tool.
- **Application Programming Interface (API):** The API provides the core functionality of the Java programming language. It offers a wide array of useful classes ready for use in your own applications. It spans everything from basic objects, to networking and security, to XML generation and database access, and more. The core API is very large; to get an overview of what it contains, consult the Java Platform Standard Edition 8 Documentation.
- **Deployment Technologies:** The JDK software provides standard mechanisms such as the Java Web Start software and Java Plug-In software for deploying your applications to end users.
- **User Interface Toolkits:** The JavaFX, Swing, and Java 2D toolkits make it possible to create sophisticated Graphical User Interfaces (GUIs).
- **Integration Libraries:** Integration libraries such as the Java IDL API, JDBC API, Java Naming and Directory Interface (JNDI) API, Java RMI, and Java Remote Method Invocation over Internet Inter-ORB Protocol Technology (Java RMI-IIOP Technology) enable database access and manipulation of remote objects.

### WHY JAVA?

- Java a great programming language.
- Java is designed to allow for better performance on mobile devices or embedded systems with lower CPU power.
- Java applets are platform independent that work equally well on any machine running java installed so long as they have the plug-ins you need, which will usually be the case.

### WHAT IS SQL?

SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in a relational database.

SQL is the standard language for Relational Database System. All the Relational Database Management Systems (RDMS) like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

Also, they are using different dialects, such as –

- MS SQL Server using T-SQL,
- Oracle using PL/SQL,
- MS Access version of SQL is called JET SQL (native format) etc.

### WHY SQL?

SQL is widely popular because it offers the following advantages –

- Allows users to access data in the relational database management systems.
- Allows users to describe the data.
- Allows users to define the data in a database and manipulate that data.
- Allows to embed within other languages using SQL modules, libraries & pre-compilers.
- Allows users to create and drop databases and tables.
- Allows users to create view, stored procedure, functions in a database.
- Allows users to set permissions on tables, procedures and views.

### BRIEF HISTORY OF SQL

- **1970** – Dr. Edgar F. "Ted" Codd of IBM is known as the father of relational databases. He described a relational model for databases.
- **1974** – Structured Query Language appeared.

- **1978** – IBM worked to develop Codd's ideas and released a product named System/R.
- **1986** – IBM developed the first prototype of relational database and standardized by ANSI. The first relational database was released by Relational Software which later came to be known as Oracle.



## 4.2 FEASIBILITY STUDY

### **Feasibility Study:**

After doing the project Electricity Billing System, study and analyzing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible - given unlimited resources and infinite time.

Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements.

### **A. Economical Feasibility**

This is a very important aspect to be considered while developing a project. We decided the technology based on minimum possible cost factor.

- ☐ All hardware and software cost has to be borne by the organization.
- ☐ Overall we have estimated that the benefits the organization is going to receive from the proposed system will surely overcome the initial costs and the later on running cost for system.

### **B. Technical Feasibility**

This included the study of function, performance and constraints that may affect the ability to achieve an acceptable system. For this feasibility study, we studied complete functionality to be provided in the system, as described in the System Requirement Specification (SRS), and checked if everything was possible using different type of frontend and backend platforms.

### **C. Operational Feasibility**

No doubt the proposed system is fully GUI based that is very user friendly and all inputs to be taken all self-explanatory even to a layman. Besides, a proper training has been conducted to let know the essence of the system to the users so that they feel comfortable with new system. As far our study is concerned the clients are comfortable and happy as the system has cut down their loads and doing.

## CHAPTER 5

### 5.1 SOFTWARE REQUIREMENT SPECIFICATION

Operation System	Windows 10
Database	MySQL Workbench 8.0
Software	NetBeans IDE 16
Front End Tool	Java Core/Swings
Language	Java

### 5.2 HARDWARE REQUIREMENT SPECIFICATION

Processor	Intel(R) Core(TM) i5-3340M CPU @ 2.70GHz
Processor Speed	2.70GHz
Monitor	LCD Monitor
Hard Disk	512GB SSD
Ram	8.00 GB
Mouse	Compatible Mouse
Keyboard	Standard Keyboard

## **CHAPTER 6**

# **SYSTEM DESIGN**

### **6.1 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.

#### **6.1.1 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.

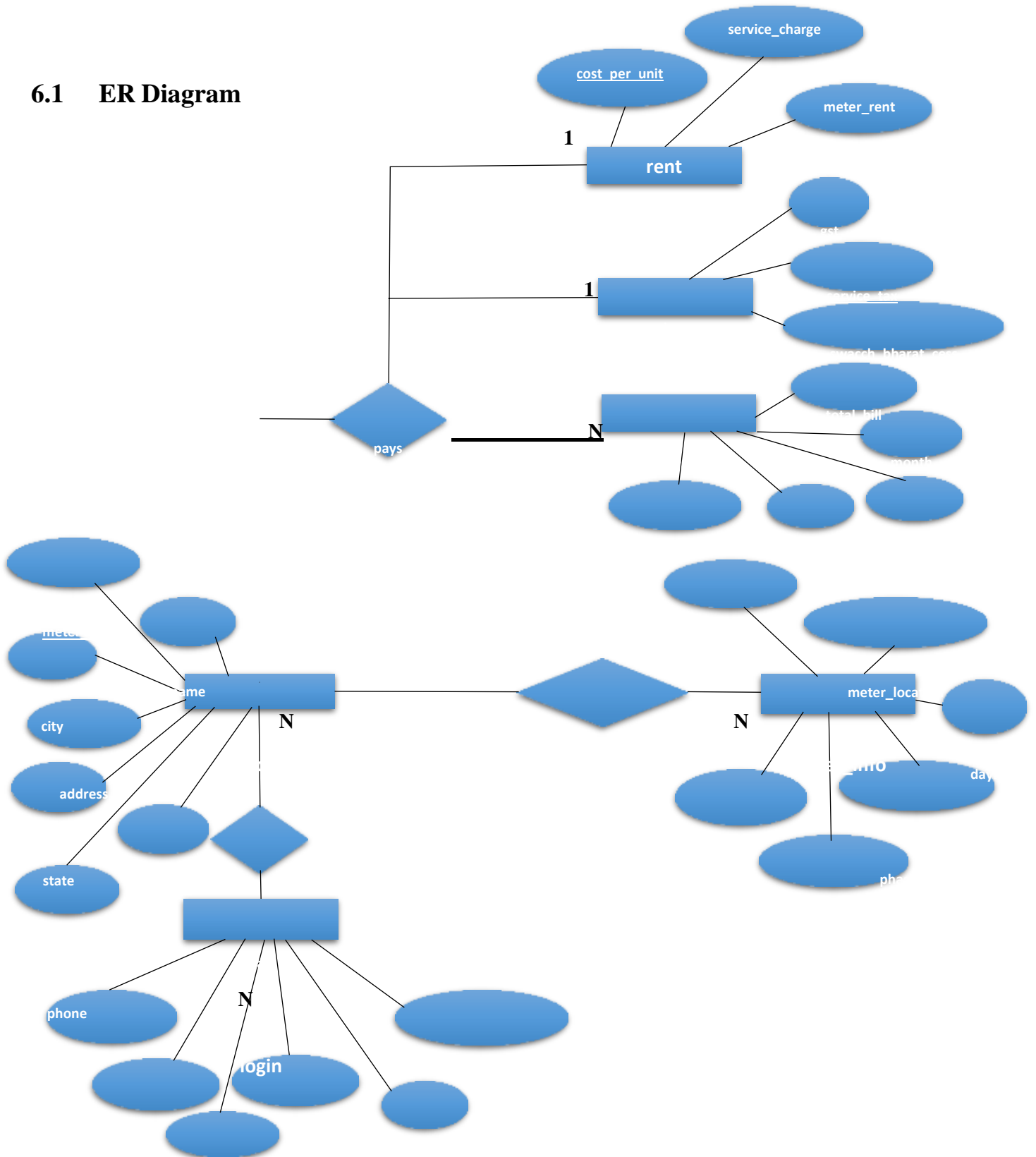
An ER diagram is a means of visualizing how the information a system produces is related. There are five main components of an ERD:

- ❖ 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 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.

## 6.1 ER Diagram



## 6.2 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".

### 6.3.1 Schema Diagram

#### Login

meter_no	Username	password	user	question	Answer
----------	----------	----------	------	----------	--------

#### customer

Name	<u>meter_no</u>	Address	City	state	Email	phone
------	-----------------	---------	------	-------	-------	-------

#### rent

<u>cost_per_unit</u>	meter_rent	service_rent
----------------------	------------	--------------

#### tax

<u>Sservice tax</u>	swacch_bharat_cess	gst
---------------------	--------------------	-----

#### bill

meter_no	Month	units	total_bill	Status
----------	-------	-------	------------	--------

#### meter\_info

meter_no	meter_location	meter_type	phase_code	bill_type	Days
----------	----------------	------------	------------	-----------	------

**FIG 6.3.1: Schema diagram of Electricity Billing System**

## CHAPTER 7

### SYSTEM IMPLIMENTATION

#### 7.1 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.

#### 7.2 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 ("snehal","12345"," btm","Bhandup",  
"Mumbai", "[snehal@gmail.com](mailto:snehal@gmail.com)", "7718096358").

##### 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= [snehal@gmail.com](mailto:snehal@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\_name WHERE 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: CREATE TABLE table\_name (column1 datatype, column2 datatype, 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));

## 7.3 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.

## CHAPTER 8

# TESTING,GRAPHS,TABLES

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

### 8.1 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.

### 8.2 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.

### 8.3 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.

#### 8.1.1 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 8.1: Negative test case for phone number insertion**

Function Name	Input	Expected Output	Error	Resolved
Input phone number	77180	Phone number is invalid	Length of phone number is not equal to 10	Consume ()
Input phone number	77180sdk	Phone number is invalid	Alphabets are being taken as input for phone number	—

**Table 8.2: Positive test case for phone number insertion**

Function Name	Input	Expected Output	Error	Resolved
Input phone number	7718096358	Expected output is seen	—	—

**Table 8.3: Negative test case for email insertion**

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

**Table 8.4: Positive test case for email insertion**

Function Name	Input	Expected Output	Error	Resolved
Input email	<a href="mailto:snehal123@gmail.com">snehal123@gmail.com</a>	Expected output is seen	—	—

**Table 8.5: Negative test case for customer name insertion**

Function Name	Input	Expected Output	Error	Resolved
Input customer name	snehal123	Name is invalid	Numbers are being taken as input for name	Consume ()

**Table 8.6: Positive test case for customer name insertion**

Function Name	Input	Expected Output	Error	Resolved
Input customer name	Snehal	Expected output is seen	—	—

### 8.1.2 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 8.7: 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 8.8: Test case on basis of depositedetails**

<b>Function Name</b>	<b>Input</b>	<b>Expected Output</b>	<b>Error</b>	<b>Resolved</b>
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	—	—

### 8.1.3 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 8.9: Test cases for the project**

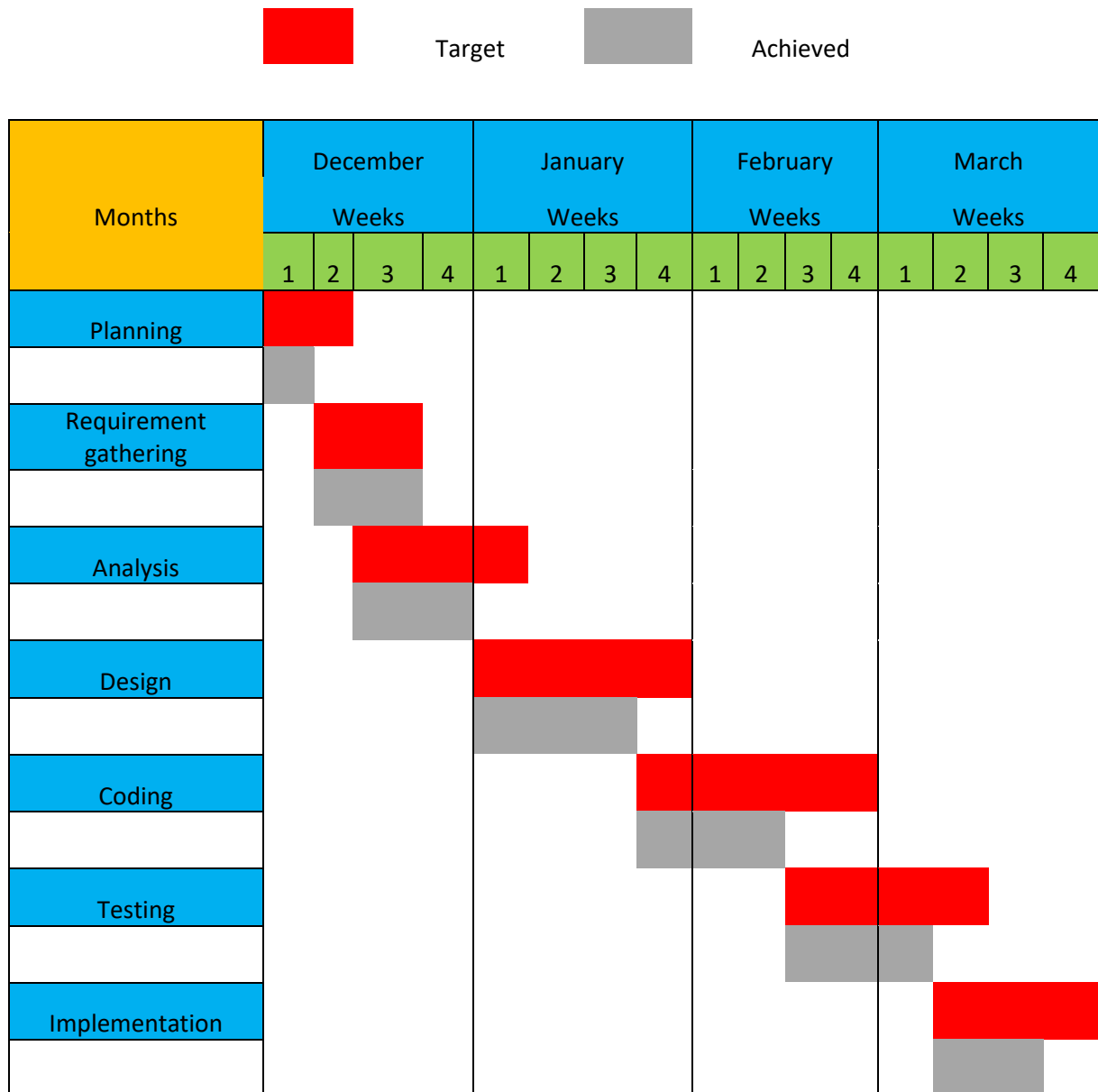
<b>Steps</b>	<b>Action</b>	<b>Expected output</b>
Step1 choice	The screen appears when the users run the program. 1.If admin login 2.If customer login	A page with different menu's appears.  1.Admin panel opens and 2.Customer panel opens
Step 2	The screen appears when the admin logs in and selects any 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	

## ELECTRICITY BILLING SYSTEM

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	

## 8.2 GRAPH

### GANTT CHART



## 8.3 TABLES

### 8.3.1 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.

```
mysql> show tables;
+-----+
| Tables_in_elect |
+-----+
| bill              |
| customer          |
| login             |
| meter_info        |
| rent              |
| tax               |
+-----+
6 rows in set (0.03 sec)
```

**FIG 8.10:List of tables**

#### Login Table:

```
mysql> desc login;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| meter_no   | varchar(30)   | YES  |     | NULL    |       |
| username   | varchar(30)   | YES  |     | NULL    |       |
| password   | varchar(30)   | YES  |     | NULL    |       |
| user       | varchar(30)   | YES  |     | NULL    |       |
| question   | varchar(40)   | YES  |     | NULL    |       |
| answer     | varchar(30)   | YES  |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.00 sec)
```

**FIG 8.11:login table description**

**Customer Table:**

```
mysql> desc customer;
```

Field	Type	Null	Key	Default	Extra
name	varchar(30)	YES		NULL	
meter_no	varchar(20)	NO	PRI	NULL	
address	varchar(50)	YES		NULL	
city	varchar(20)	YES		NULL	
state	varchar(30)	YES		NULL	
email	varchar(30)	YES		NULL	
phone	varchar(30)	YES		NULL	

7 rows in set (0.00 sec)

**FIG 8.12: customer table description****Tax Table:**

```
mysql> desc tax;
```

Field	Type	Null	Key	Default	Extra
service_tax	int	NO	PRI	NULL	
swacch_bharat_cess	int	YES		NULL	
gst	int	YES		NULL	

3 rows in set (0.00 sec)

**FIG 8.13: tax table description****Rent Table:**

```
mysql> desc rent;
```

Field	Type	Null	Key	Default	Extra
cost_per_unit	int	NO	PRI	NULL	
meter_rent	int	YES		NULL	
service_charge	int	YES		NULL	

3 rows in set (0.00 sec)

**FIG 8.14: rent table description****Bill Table:**

```
mysql> desc bill;
```

Field	Type	Null	Key	Default	Extra
meter_no	varchar(20)	YES	MUL	NULL	
month	varchar(20)	YES		NULL	
units	int	YES		NULL	
total_bill	int	YES		NULL	
status	varchar(40)	YES		NULL	

5 rows in set (0.01 sec)

**FIG 8.15: bill table description**

**Meter\_Info Table:**

```
mysql> desc meter_info;
```

Field	Type	Null	Key	Default	Extra
meter_no	varchar(30)	YES	MUL	NULL	
meter_location	varchar(10)	YES		NULL	
meter_type	varchar(15)	YES		NULL	
phase_code	int	YES		NULL	
bill_type	varchar(10)	YES		NULL	
days	int	YES		NULL	

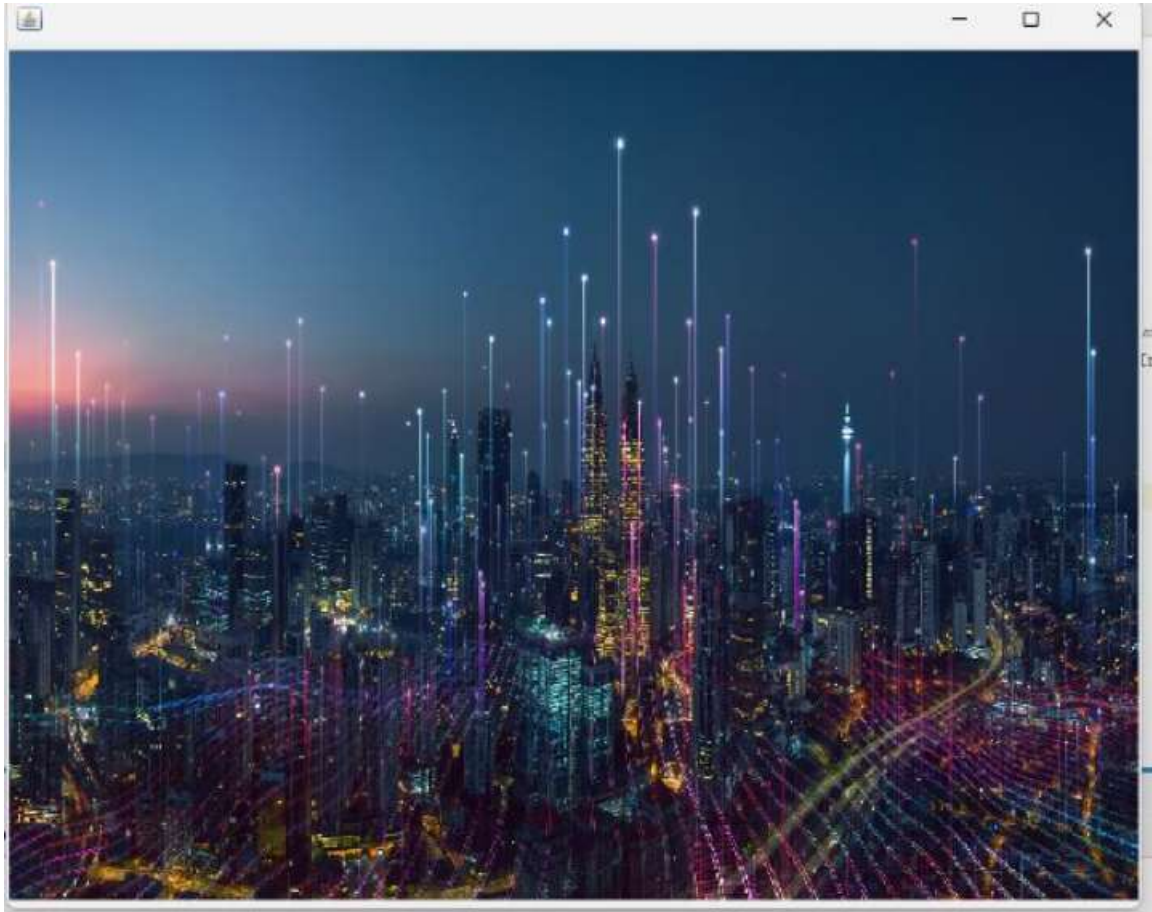
6 rows in set (0.00 sec)

**FIG 8.16: meter\_info table description**

## CHAPTER 9

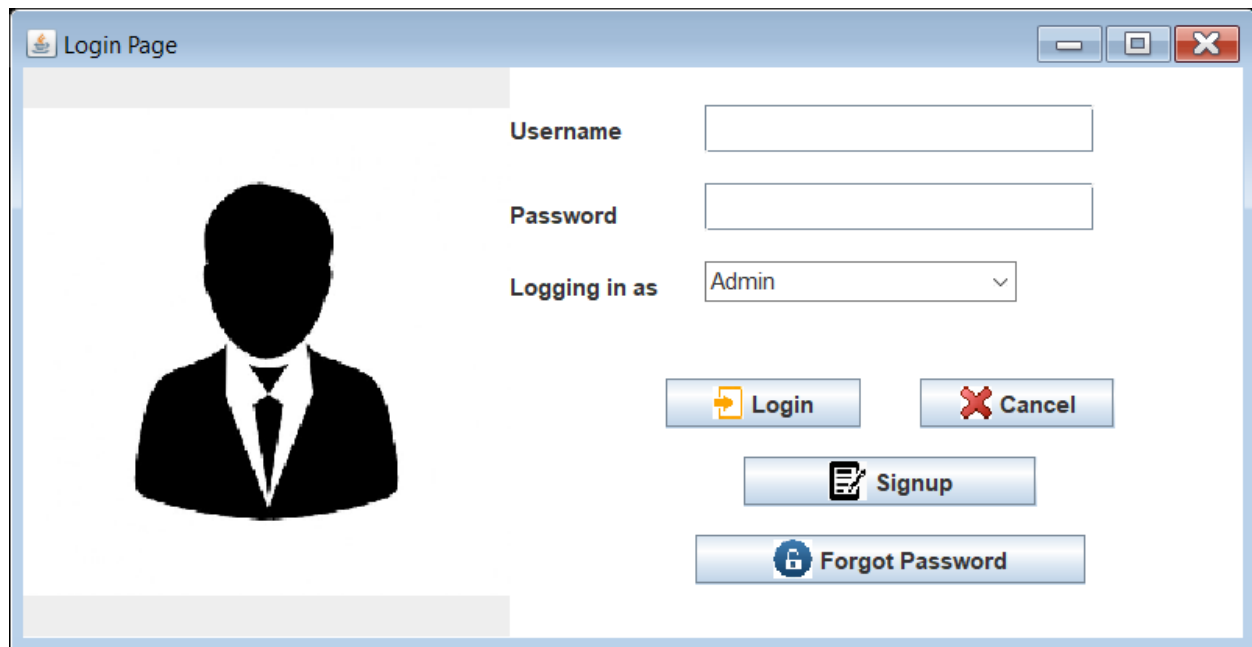
### DISCUSSION AND SNAPSHOTS

#### 9.1 SNAPSHOTS:



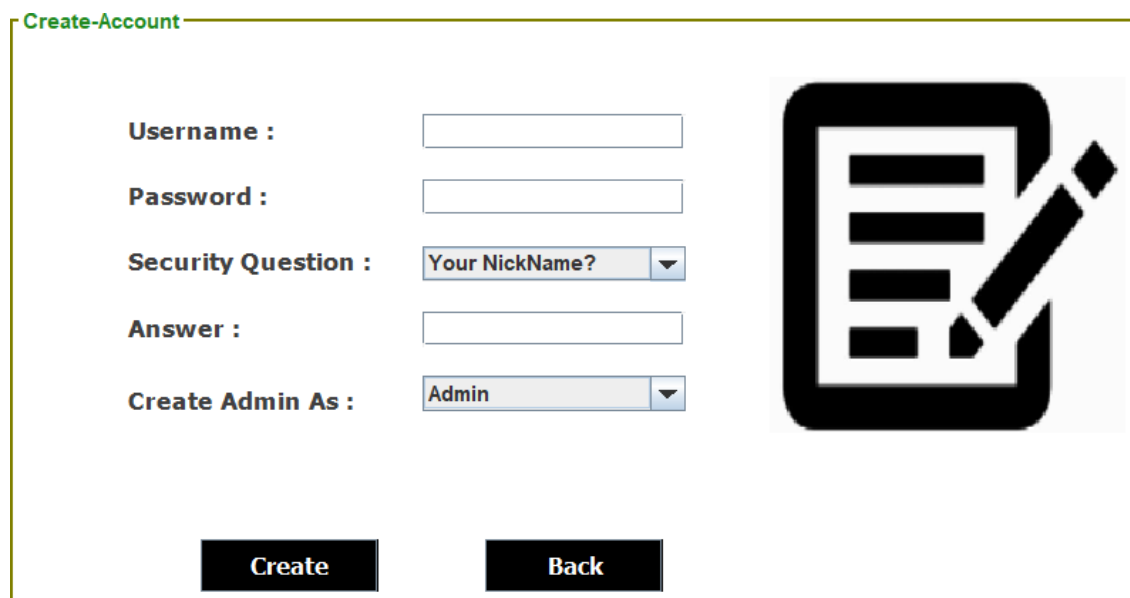
9.2

**FIG 9.1: Splash page of Electricity Billing System**



The screenshot shows a web browser window titled "Login Page". On the left is a silhouette of a person in a suit. To the right are three input fields: "Username", "Password", and "Logging in as" (a dropdown menu currently showing "Admin"). Below these fields are three buttons: "Login" (with a right arrow icon), "Cancel" (with a red X icon), and "Signup" (with a document and pencil icon). At the bottom is a "Forgot Password" button (with a lock icon).

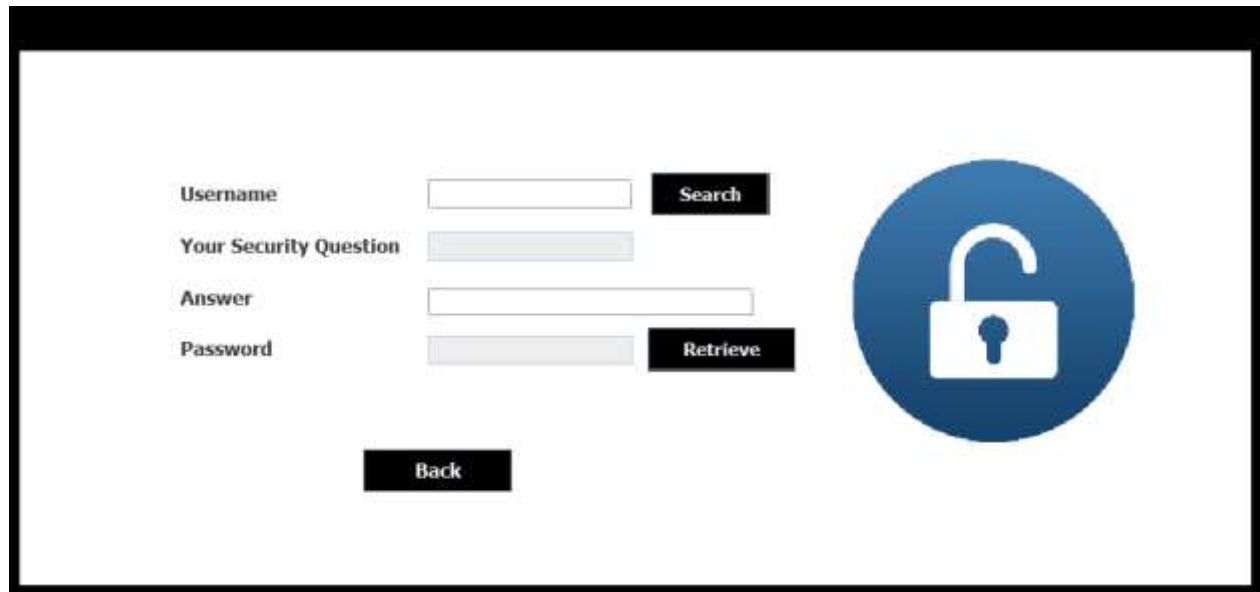
**FIG 9.2: Login page**



The screenshot shows a web page titled "Create-Account". On the left are five input fields: "Username :", "Password :", "Security Question :" (a dropdown menu showing "Your NickName?"), "Answer :", and "Create Admin As :" (a dropdown menu showing "Admin"). To the right is a large icon of a document with a pencil. At the bottom are two buttons: "Create" and "Back".

**FIG 9.3: Signup page**





The image shows a web form for a 'Forgot Password' page. It features four input fields: 'Username', 'Your Security Question', 'Answer', and 'Password'. The 'Username' field is followed by a 'Search' button. The 'Password' field is followed by a 'Retrieve' button. A 'Back' button is located at the bottom left. To the right of the form is a large blue circular icon containing a white padlock with a keyhole.

Username

Your Security Question

Answer

Password

**FIG 9.4: ForgotPassword page**



**FIG 9.5: Admin home page**



### New Customer

Customer Name

Meter No 673692

Address

City

State

Email

Phone Number

**FIG 9.6: New customer page**

MeterInfo Page

### Meter Information

Meter Number 904764

Meter Location


Meter Type

Phase Code

Bill Type


Days 30

Note: By default Bill is calculated for 30 days only

A photograph showing a person's hand holding a clipboard with a checklist, standing in front of an electrical meter mounted on a wall.

**FIG 9.7: Meter Info page**

---



### Calculate Electricity Bill

Meter Number:

Name:

Address:

Units Consumed:

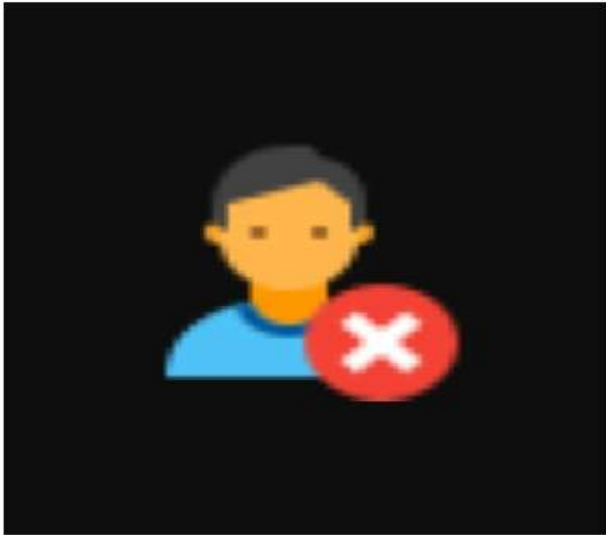
Months:

**FIG 6.17: Calculate Bill page**

DeleteDetails Page

DELETE CUSTOMER DETAILS

Meter\_no :



**FIG 6.18: Delete Customer page**



**FIG 6.19: Customer Home page**

A screenshot of a web application window titled 'VIEW CUSTOMER INFORMATION'. The window contains a form with the following fields and values:

Name	Snehal Dilip Ka...	State	Maharashtra
Meter Number	889200	Email	SK@143
Address	Bhandup,Marin...	Phone	7718096358
City	Mumbai		

Below the form, there is a 'Cancel' button. At the bottom of the window, there is a decorative illustration of a group of diverse people standing and holding up large yellow thumbs-up icons, signifying approval or success.

**FIG 6.21: View Customer Details page**



The screenshot shows a web application window titled "Electricity Bill". It contains a form with the following fields and values:

Field	Value
Meter Number	889200
Name	Snehal Dilip Kapse
Month	July
Units	95
Total Bill	981
Status	Paid

Below the form are two buttons: "Pay" and "Back". To the right of the form is a large graphic of a blue bill icon with a red checkmark and the word "BILL" in red.

FIG 6.23: Pay Bill page



FIG 6.24: Paytm page

[illegible]

**ELECTRICITY BILL GENERATED FOR THE MONTH OF March, 2022**

Customer Name: Snehal Dilip Kapse  
 Meter Number : 889200  
 Address : Bhandup, Marine Drive  
 City : Mumbai  
 State : Maharashtra  
 Email : SK@143  
 Phone : 7718096358

Meter Location: Inside  
 Meter Type: Smart Meter  
 Phase Code: 011  
 Bill Type: Normal  
 Days: 30

Cost Per Unit: 9  
 Meter Rent: 9  
 Service Charge: 22  
 Service Tax: 22  
 Swacch Bharat Cess: 6  
 Fixed Tax: 18

**Total Bill: 1158 INR**

**FIG 6.26: Generate Bill page**

Sort By Meter Number: 413098      Sort By Month: January

Search      Print

meter_no	month	units	total_bill	status
413098	January	25	375	Paid

**FIG 6.27: Deposit Details page**

**About Project**

About Project

Electricity Billing System is a software-based application developed in Java programming language. The project aims at serving the department of electricity by computerizing the billing system. It mainly focuses on the calculation of Units consumed during the specified time and the money to be paid to electricity offices. This computerized system will make the overall billing system easy, accessible, comfortable and effective for consumers.

Exit

**FIG 6.28: About page**



## CHAPTER 10

### **FUTURE ENHANCEMENT**

1. Splash Screen.
2. Bird skins and Background Textures will change according to Levels and User choice.
3. The status and the history will be saved and will be dynamically shown in a graph where user can see his total performance.
4. Drawing assets that actually work in a game and look good is super time-consuming.
5. Real pixel art is way harder than it looks (by real I mean no cheating, no animation programs, no diagonal pixels. Just drawing pixel by pixel).
6. If there are no obstacles at the top of the screen, the player has no reason to go down. That's why they made Helicopter Game inside a cave! (This seems obvious in retrospect, but at the time I was absorbed in the whole process rather than focused on the design itself).
7. Game design is a full time job and you can't just pull ideas out of your butt and make them into a fun game.
8. Gamemaker is not magical. It's hard, and you still need to think like a programmer to make good use of it.
9. Add power ups and other features once the base game is done.
10. Algorithm for image moving.
11. We will add more feature to the game and will change the bird and background scenery according to user choice. The status and the history will be saved and we show a graph where user can see his total performance whether it increasing or decreasing.

## **CHAPTER 11**

### **CONCLUSION**

After all the hard work is done for electricity bill management system is here. It is a software which helps the user to work with the billing cycles, paying bills, managing different DETAILS under which are working etc.

This software reduces the amount of manual data entry and gives greater efficiency. The User Interface of it is very friendly and can be easily used by anyone.

It also decreases the amount of time taken to write details and other modules.

## CHAPTER 12

# BIBLIOGRAPHY

## REFERENCES

### Book Reference

Database Management Systems 3rd Edition by Raghu Ramakrishnan (TEXTBOOK).

### Websites

- [www.youtube.com](http://www.youtube.com)
- [www.stackoverflow.com](http://www.stackoverflow.com)
- [www.google.com](http://www.google.com)
- <http://www.javatpoint.com/>