Electricity Bill Management System

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Version No. | Date | Prepared by / Modified by | Significant Changes |
|  |  |  |  |
| 1 |  | Varshini N |  |
|  |  |  |  |

Glossary

|  |  |
| --- | --- |
| Abbreviation | Description |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

Table of Contents

[1 Introduction 3](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364881)

[2 Scope of Change 3](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364882)

[3 List of impacted modules 3](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364883)

[4 Design and Detailed technical updates 4](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364884)

[4.1 Process model 4](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364885)

[5 Details of Alternative Design Approach 12](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364892)

[6 Other Technical changes 17](https://hclo365-my.sharepoint.com/personal/varshini_n_hcl_com1/Documents/Microsoft%20Teams%20Chat%20Files/Technical%20Story%20Card%20-%20Supplier%20Finance.doc#_Toc379364893)

7 References …………………………………………………17

# 

# **1 INTRODUCTION**

This system is made to keep the records about the bills of the customers. The admin can manage all the accounts and the registered users like employees. Customer can get their respective electricity bills.

This system helps in maintaining the bills and the payments. A different module is there for employees to check the customer’s details if it is needed.

# **2 SCOPE OF CHANGE**

Change will not be entertained and whatever specifications mentioned in this document is final.

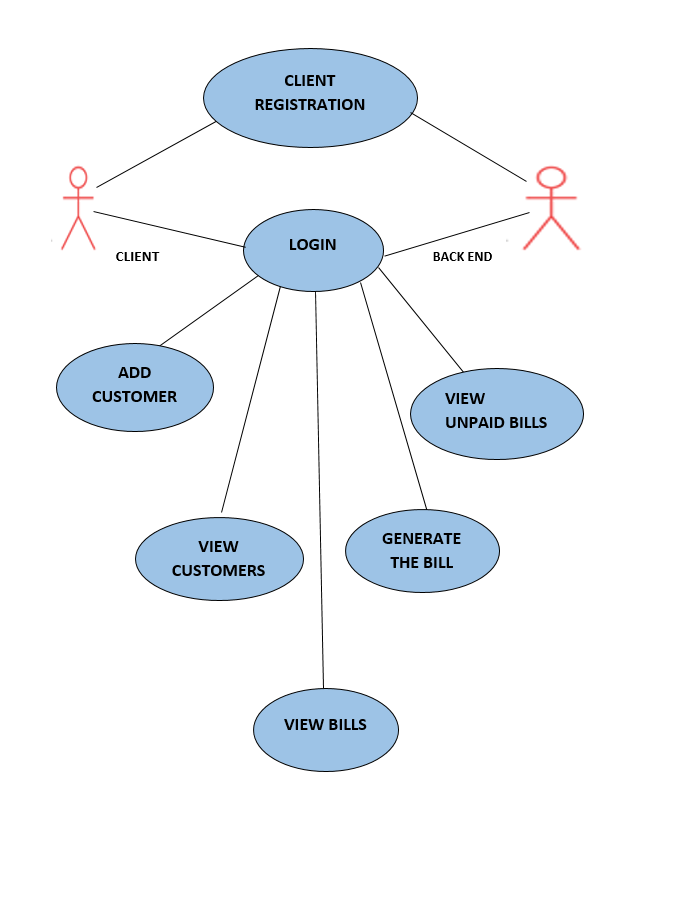
# **3 LIST OF CHANGES**

All the functional modules will be created from scratch

# **4 Design and Detailed technical updates**

## 4.1 Process model

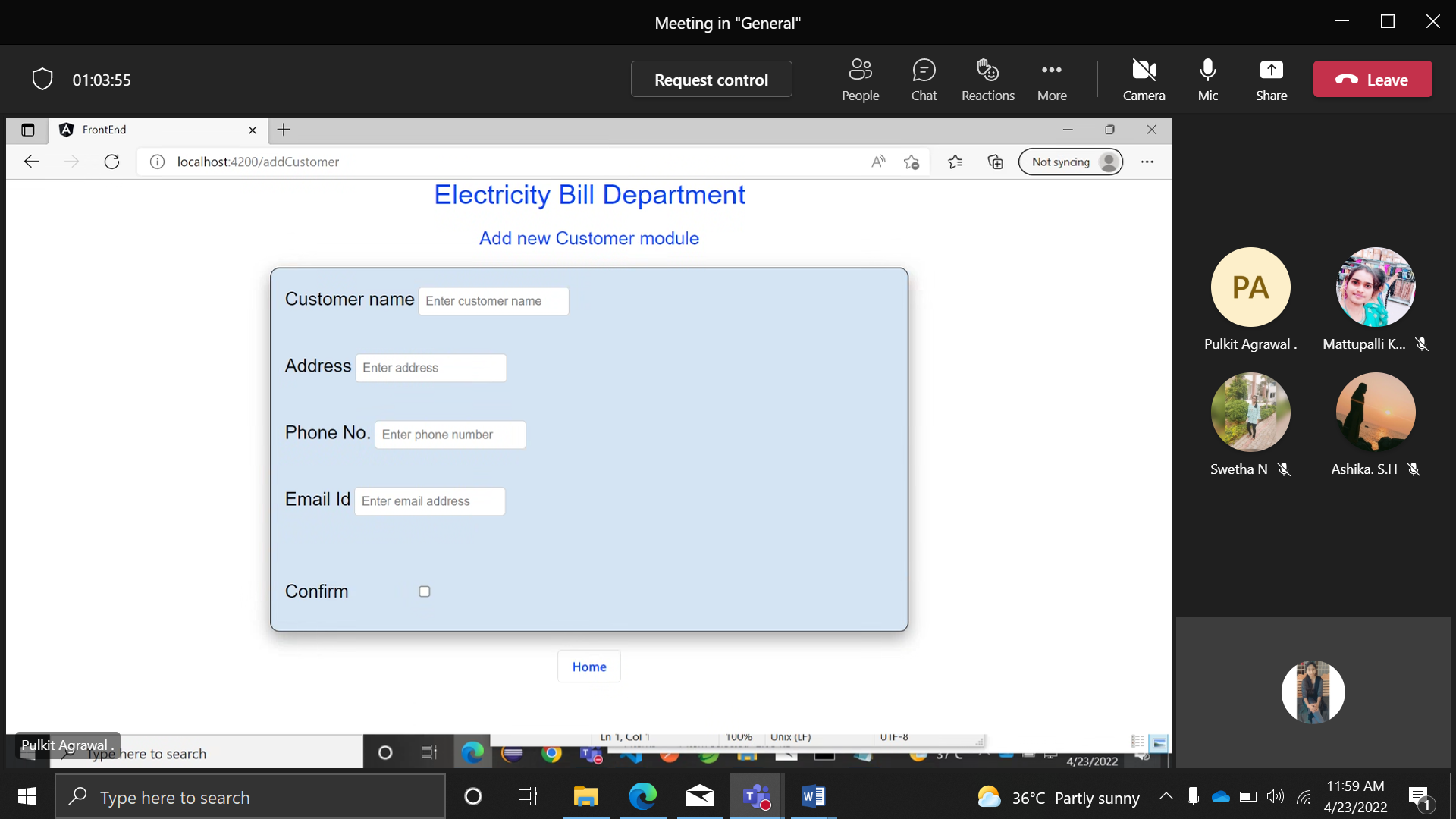
### 4.1.1 Use case Model



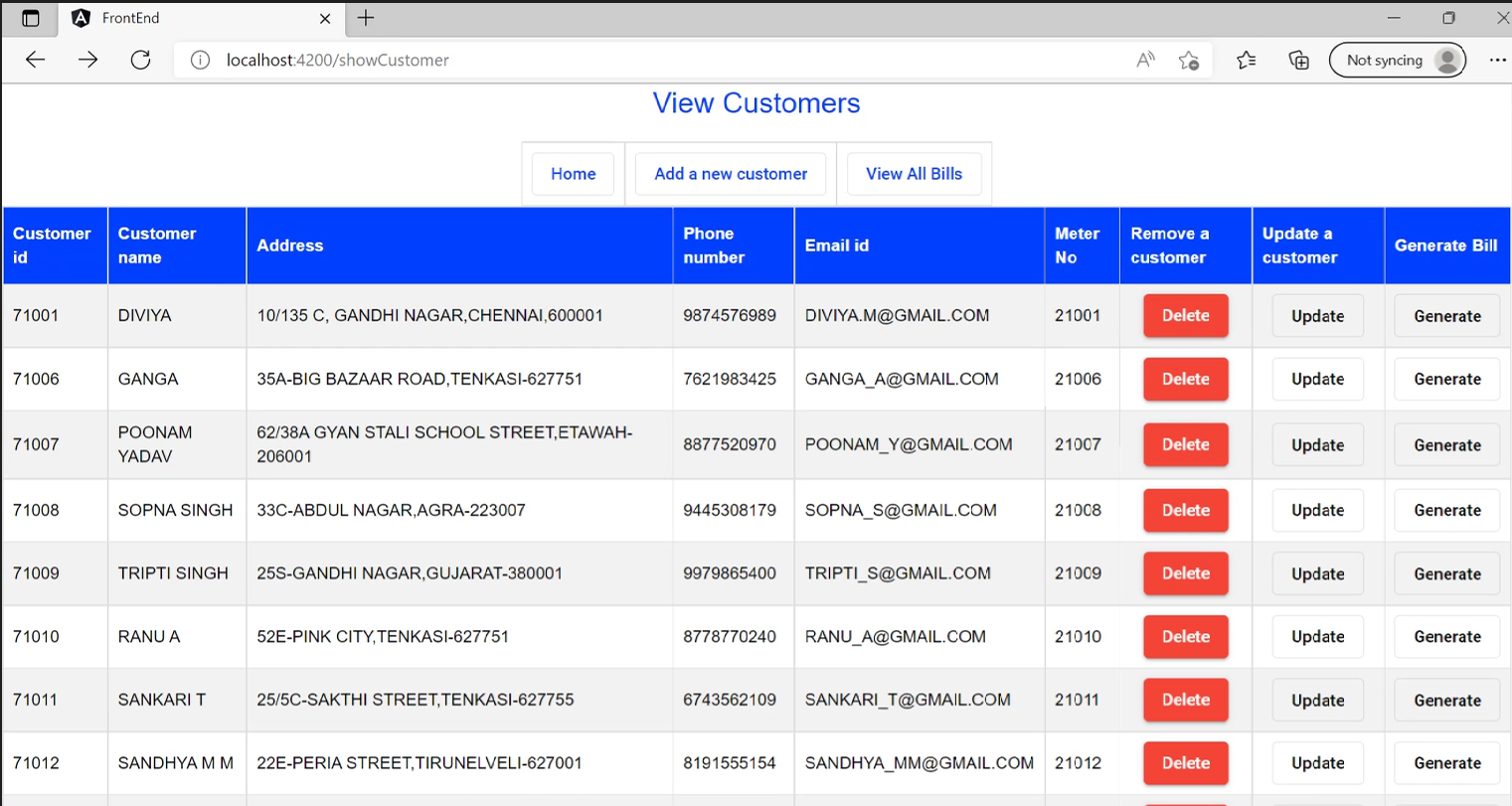
|  |  |
| --- | --- |
| Brief Description | Login |
| Basic Flow | This use case describes how a user log-in in to the system   1. The client has to register himself into the system. 2. After the successful login, user will be taken to the appropriate landing page. 3. The following information is required to login.  * username * password |
| Alternate Flow | 1. The system will validate the credentials provided. If credentials are invalid, login form will be redirected again with error message. |
| Validation | 1. Valid username 2. Valid password |
| Pre-Conditions | User should have network access and Browser with latest updates. |
| Post-Conditions | Landing page has to be displayed. |



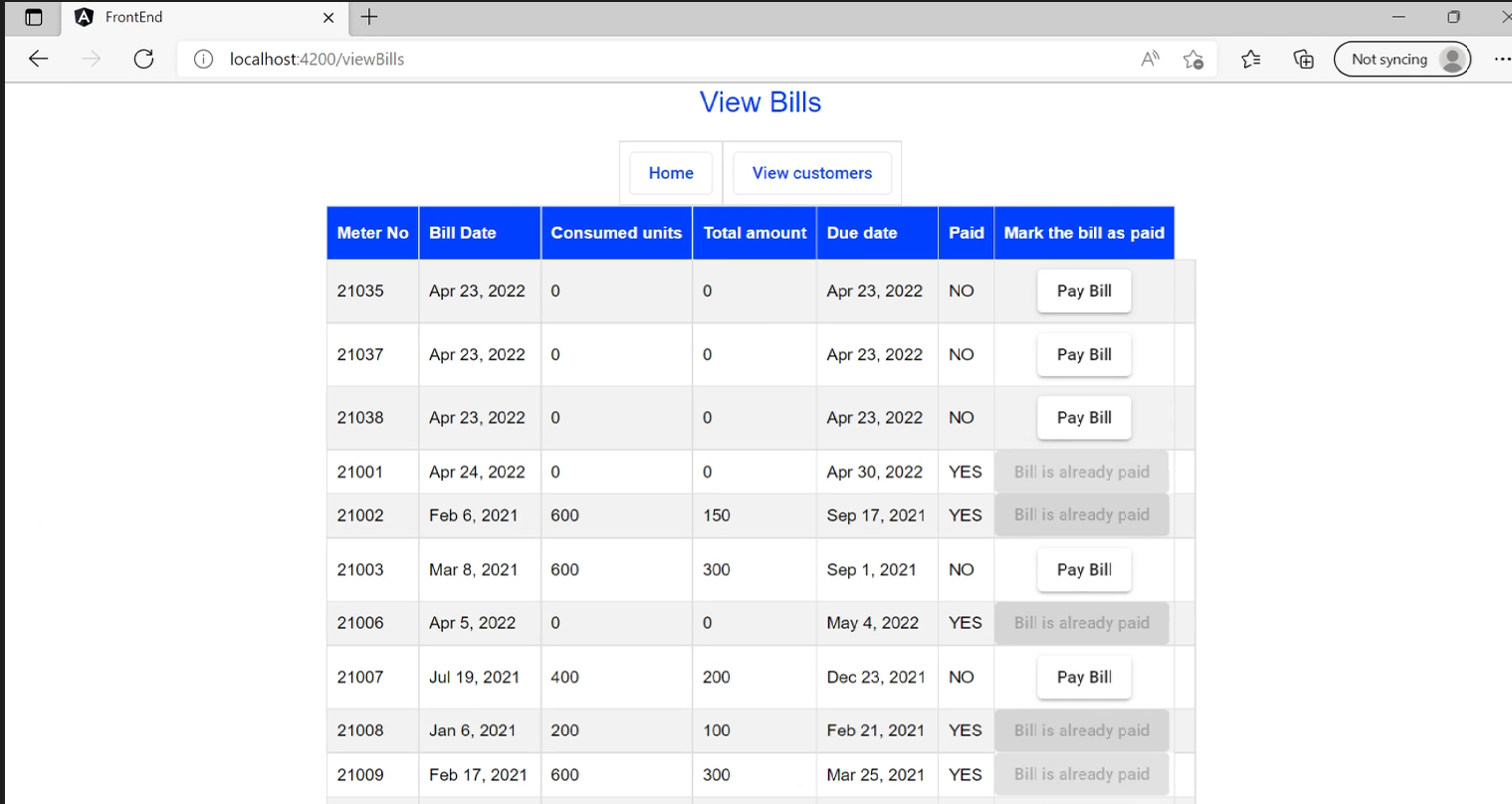
|  |  |
| --- | --- |
| Brief Description | ADD CUSTOMER |
| Basic Flow | This use case describes how a client can add a customer.  The client has to login into the system.   1. The following information is required during adding a customer.  * Customer Name * Address * Phone Number * Email Id |
| Alternate Flow | 1. The system will validate the information provided. If any invalid data is found, the input form will be redirected with error message. |
| Validation | 1. Customer Name should be valid. 2. Address should be valid. 3. Phone Number should be valid 4. Email Id should be valid. |
| Pre-Conditions | User should have already logged into the system. |
| Post-Conditions | Success message should be shown. |



|  |  |
| --- | --- |
| Brief Description | VIEW CUSTOMER |
| Basic Flow | This use case describes how a client can view all the customers.   1. The client has to login into the system. 2. The following information has to be displayed.  * Customer Id * Customer Name * Address * Phone Number * Email Id  1. You can delete the customer. 2. You can update the customer. 3. You can also generate the electricity bill. |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |



|  |  |
| --- | --- |
| Brief Description | VIEW BILLS |
| Basic Flow | This use case describes how a client can view all the bills.  1.The client has to login into the system.  2.The following information has to be displayed.   * Meter Number * Bill Date * Consumed units * Total amount * Due date * Paid   3.You can also mark the bill as paid. |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |



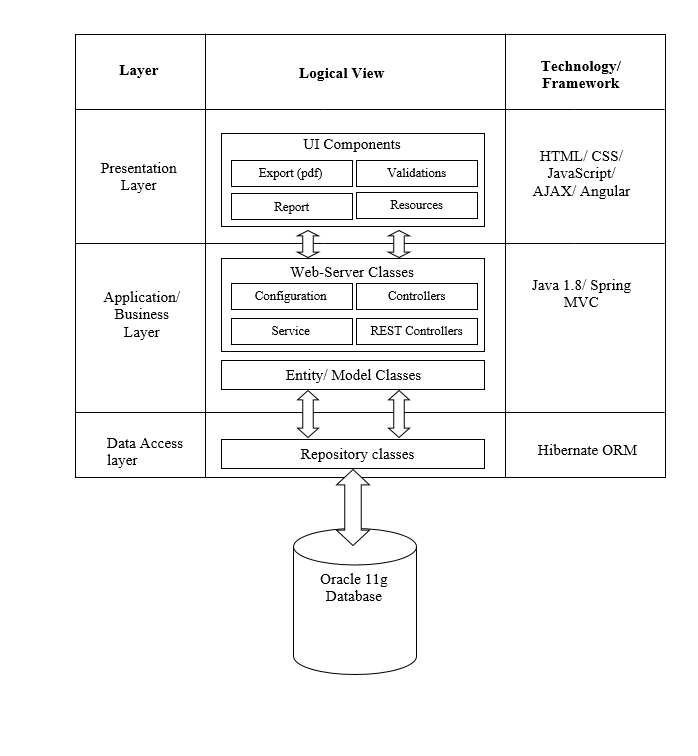
|  |  |
| --- | --- |
| Brief Description | GENERATE ELECTRICITY BILL |
| Basic Flow | This use case describes how an employee can generate a bill for customer.   1. The Employee has to login into the system. 2. Employee has to enter the following attributes.  * Bill date * Previous units * Current units * Due date  1. Status of the invoice has to be updated. |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |
| Pre-Conditions | User should have already logged into the system. |
| Post-Conditions | Success message should be shown. |



|  |  |
| --- | --- |
| Brief Description | VIEW UNPAID BILLS |
| Basic Flow | This use case describes how a client can view all the unpaid bills.  1.The client has to login into the system.  2.The following information has to be displayed.   * Meter Number * Bill Date * Consumed units * Total amount * Due date * Paid |
| Alternate Flow | Not Applicable |
| Validation | Not Applicable |



**Architecture Diagram**



**Presentation Layer:**

This layer consists of all UI components.

* Export component: The plug-in is used to export the reports in pdf format. It helps to reduce the workload on the web/ application server.
* Validation component: All basic level data validations should be done at UI level.
* Report component: All reports to be generated to the user, will be processed and generated at the browser end.
* Resources: All HTML/CSS/ images, which are required for the page design.

**Application Layer:**

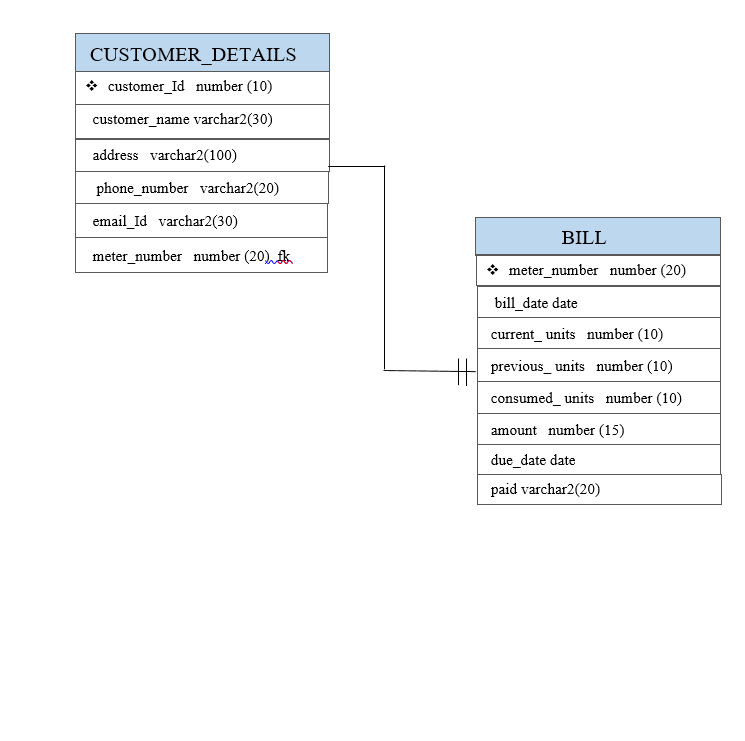
This layer comprises of all server and business classes.

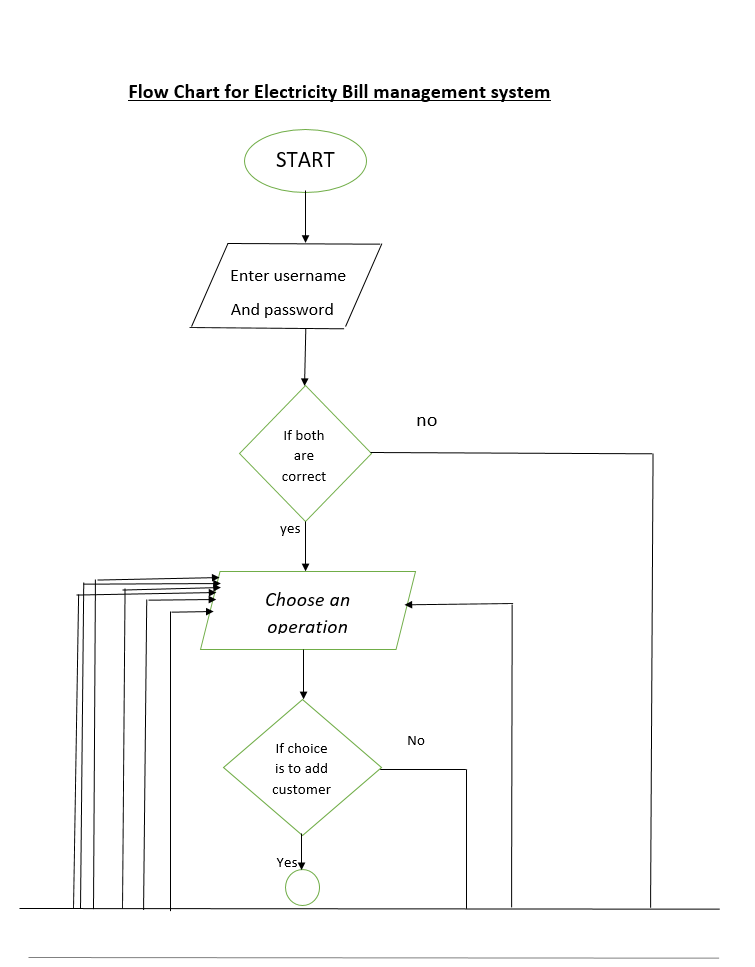
* Configuration settings (XML or Class based) will define the application and server configuration.
* Spring Controllers defines the server classes, which are required to process the incoming http requests.
* Service classes are required to perform the required business services.
* REST Controllers to process the HTTP AJAX requests.
* Model classes are used the define the functions of the entities present in the system.

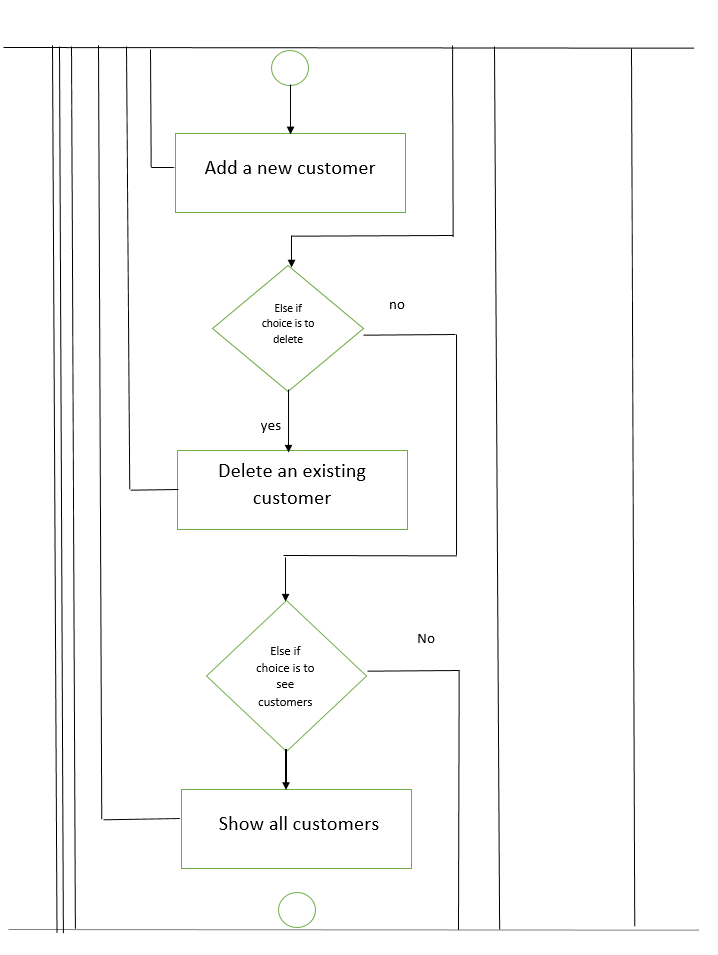
**Data Layer:**

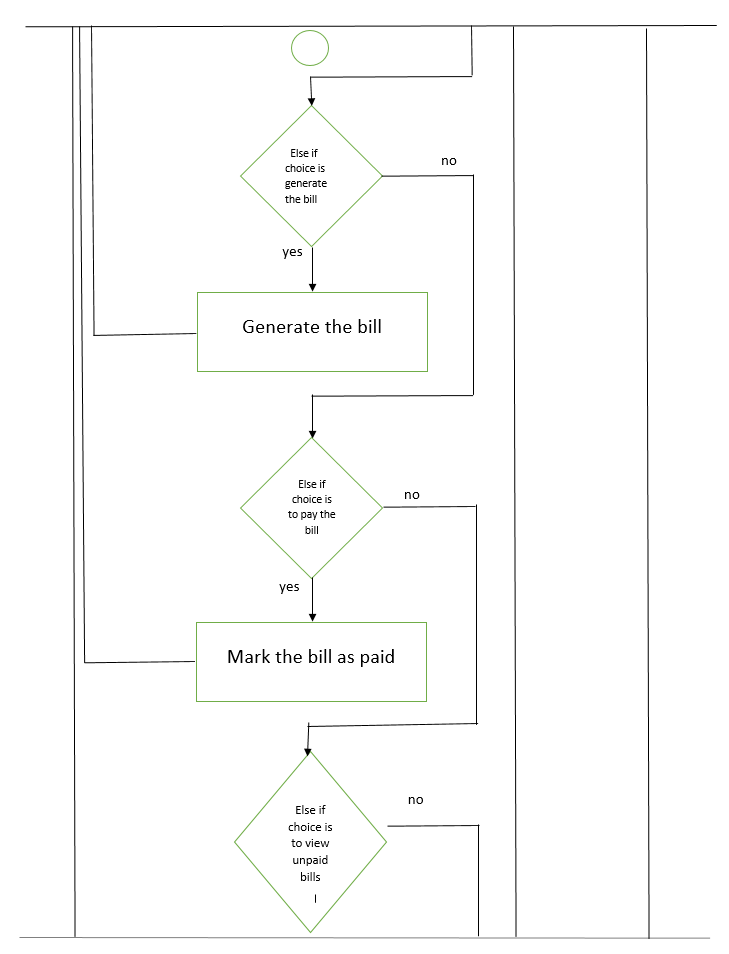
Data layer is implemented through Hibernate ORM. It will contain the repository classes, which provides interface to the table.

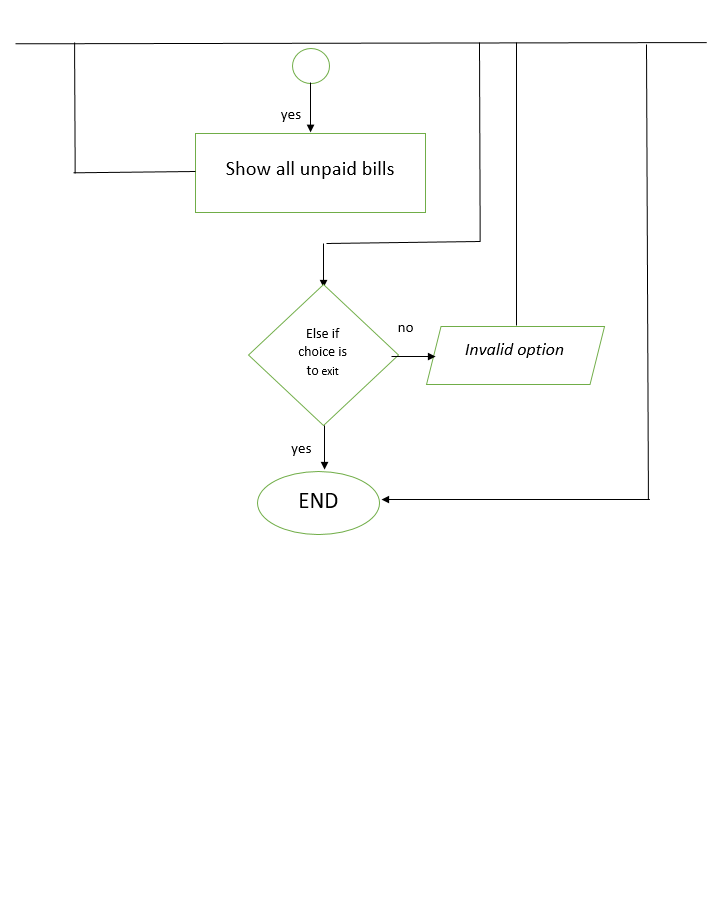
**Class Diagram:**





****

****

****

# **OTHER TECHNICAL CHANGES**

## **6.1 CI/ Build relates tasks**

All the UI part has to be integrated in angular.

## **Non-functional related changes**

System should have proper network.

System should have effective web browser. For example, Google chrome, Microsoft Edge and Firefox etc.,

# **7 REFERENCES**

LMS slides.