CIS 634 Object-Oriented Software Engineering

Project Title: Power Management System

Instructor: Prof. Weidong Xiong

Group Number: 7

Group Members:

Tapan Desai	CSU ID: 2777437
Shaishav Shah	CSU ID: 2835285
Kinjal Patel	CSU ID: 2794458
Mitee Patel	CSU ID: 2817313

SOFTWARE DESIGN SPECIFICATION

1.0 Introduction

The Software Design Specification (SDS) for the Power Management System. The SDS will break down the project into components to describe in detail what the purpose of each component is and how it will be implemented. The SDS will also serve as a tool for verification and validation of the final product.

1.1 Goals and objectives

To achieve software design document, our team can establish exact expectations for the project before starting to code. SDS will help to streamline the code process.

1.2 Statement of scopes

The scope of the Power Management System includes its distinct features, its benefits, and its limitations. The system's distinct features allow to view the customer panel, look up electricity usage, pay the pending bills, and look up previous bills. The system enables the user to determine if there are any payments due.

1.3 Software context

Software required for this project are:

Software	Role
Eclipse	Used to design Backend API
VS Code	Graphical User Interface, front end and react
Xampp	Servers (Apache and MySQL)

1.4 Major constraints

The main constraint for this project will be that as we don’t have access to get all the actual bill data from an existing database. So, we will instead work with dummy data inputted by ourselves for this project.

2.0 Data design

Configuration of a DBMS, Avoiding duplication of data. Each database should be automatically verifying user access to settings, networks, and data. We will be using relational databases store information with columns, rows, and tables. We will be using query to store the data on specific entity.

2.1 Data structures

Structure of Data

1. Input- view user details in the users' list
2. Output – customer details

2.2 Database description

Each table of the database has a selected key that identifies the statistics withinside the table. To join one table to another, foreign keys are used. A foreign key's then related to a primary key.

- The name of each variable
- Data type (integer, text, date, count, etc.)
- Possible values or range of values, especially if coded (M = male, F=female, O=Other)

New Tables
List any new tables that will be needed, for each one including table name, table description, and related tables.

1. User_role

TABLE NAME	FOREIGN KEY	DATA TYPE	ALLOW NULLS	FIELD DESCRIPTION
USER	ID	Varchar(50)	No	USER IS USED TO IDENTIFY CUSTOMER AND ADMIN
USER_ROLE	**ID**	**VARCHAR(50)** **INT** **DATE**	**NO**	ROLE IS IDENTIFIED AS CUSTOMER OR ADMIN

2. Customer

TABLE NAME	FOREIGN KEY	DATA TYPE	ALLOW NULLS	FIELD DESCRIPTION
ADMINUSER	ID	Varchar(50) INT DATE	No	GET THIS FIELD FROM THE DATABASE TO VIEW USER DETAILS
USER	ID	Varchar(50) INT DATE	No	VIEW DETAILS / CAN ONLY EDIT PERSONAL INFORMATION
USERADDRESS	USERID	Varchar(100) INT	No	THIS WILL TIE TO USER
USER PAYMENT	USERID	INT DATE	NO	USER CONFIDENTIAL INFORMATION

3. Bill

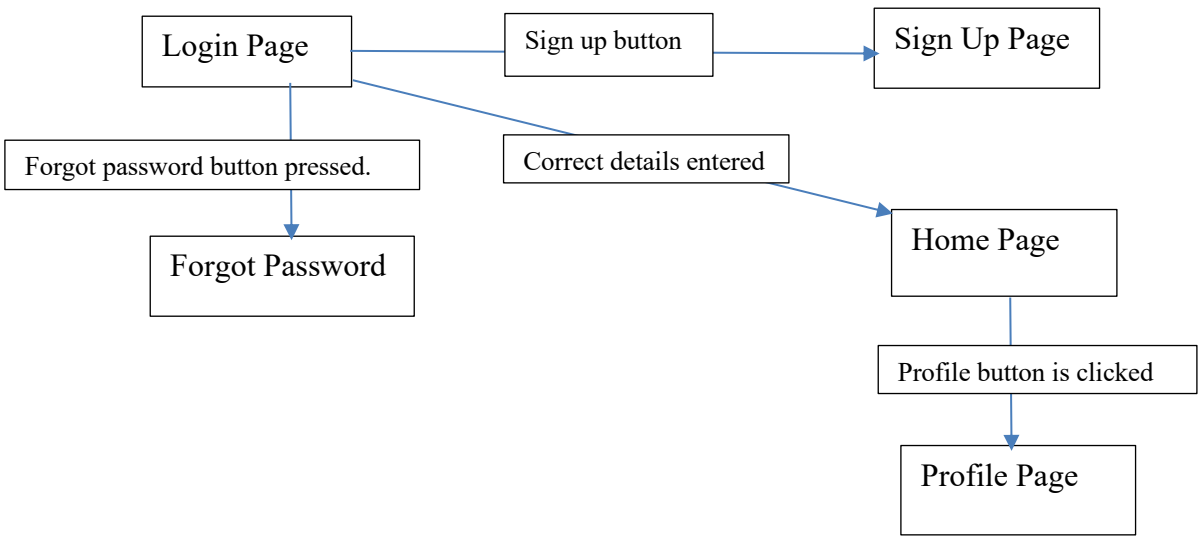
TABLE NAME	FOREIGN KEY	DATA TYPE	ALLOW NULLS	FIELD DESCRIPTION
BILL_ID	ID	Varchar(50)	No	UNIQUE ID OF EACH INDIVISUAL BILL
AMOUNT	ID	Varchar(50) INT	No	AMOUNT IN USD THAT IS DUE

BILL_DATE	ID	Varchar(50) DATE	no	THE DATE OF ISSUE OF THAT PERTICULAR BILL
DUE_DATE	Id	Varchar(50) DATE	no	THE DUE DATE OF THAT PERTICULAR BILL
STATUS	ID	Varchar(50) INT	No	STATES IF THE BILL IS PAID OR UNPAID
UNITS	ID	VARCHAR(50) INT	NO	STATES THE TOTAL UNITS USED

3.0 Architectural and component-level design

Software architecture is how components should behave and communicate in the system, set the physical location of components, and finally choose the tools in order to create components.

3.1 Architecture diagram



3.2 Description for Components

A description of major software components contained within the architecture is presented.

3.2.1 Component n description

3.2.1.1 Interface description

Identification	NewUserAccountScreen
Input	Sign up/ Log in
Output	This screen contains links to the following screen: • Login Screen
Interfaces	The links are contained in the bottom half of the screen. The screen is designed to be easy to view using the resolution standard on the PDA.

3.2.3.2 Static models

N/A

3.2.3.3 Dynamic models

N/A

3.3 External Interface Description

N/A

4.0 User interface design

This Design represents what the user is going to see on the web application. The UI should be kept simple for user of every sector to interact with it with ease. There are no secondary features for the customer other than Payment or Viewing the previous records. All the update of the data i.e., Customer details or Bill change credentials must ONLY be done by administrator.

4.1 Description of the user interface

4.1.1 Customer UI

The customer can log-in using his credentials. If the user credentials are incorrect, he can request a password retrieval by contacting the administrator directly. The user can view his payment due, his information and a link to view his previous Bills. By clicking on the link, other web page opens which has filters for sorting the previous bills and a drop-down menu to select the month and year. Through the bill payment page, a payment gateway will open through which the payment can be made.

4.1.2 Administrator UI

The Administrator can log-in using the credentials provided. The administrator can view each customer and his details regarding the Bill Payment. Admin can enter the units used by each customer manually. The admin can view requests for Password Retrieval and provide with the password form the database. The admin can view the overdue bills of the user.

4.2 Interface design rules

The interface would be designed in HTML+CSS. Following are some design rules that must be followed up to a certain level. It not mandatory to follow all the rules.

4.2.1 Consistency

In comparable scenarios, equivalent action sequences should be needed; identical language should be used in prompts, menus, and help screens; and consistent color, layout, capitalization, typefaces, and so on should be utilized throughout. Exceptions, such as required confirmation of the delete command or no password echoing, should be understandable and restricted in number.

4.2.2 Universal Usability

Universal Usability is the design of things such that they are useful to as many people as possible. Recognize the demands of varied users and design for flexibility, allowing for content modification. The spectrum of criteria that leads design is enriched by novice to expert distinctions, age ranges, impairments, international variations, and technology variety. Including elements for both beginners and specialists, such as explanation and quicker pace, enhances the interface design and boosts perceived quality.

4.2.3 Confirmation design for Closing

Action sequences should be divided into groups with a beginning, middle, and end. Informative feedback upon the conclusion of a group of tasks provides users with a sense of success, a signal to abandon contingency preparations, and an indicator to prepare for the next series of actions. For this web application, for example, guide customers through payment confirmation, concluding with a clear confirmation page that completes the payment.

4.2.4 Error Prevention

Design the interface so that users can't make big mistakes as much as possible. For example, gray out menu items that are inappropriate and do not permit alphabetic characters in numerical entry fields. The user interface ought to provide straightforward, helpful, and specific recovery instructions if a user makes a mistake. If a user enters an incorrect username, for instance, they should not be required to retype the entire name-and-address form; rather, they should be directed to repair only the defective component. The interface's state should remain unchanged, or the interface should provide instructions for restoring the state in the event of an error**.**

4.2.5 Reduce the load on short-term memory

Designers should avoid creating interfaces that require users to remember information from one display and then use that information on another because of humans' limited capacity for information processing in short-term memory (the rule of thumb is that people can remember "seven plus or minus two chunks" of information).It means that long forms should be reduced to fit a single display, that phone numbers should not need to be entered again, and that web addresses should remain visible. For each web page, these fundamental principles must be interpreted, improved, and extended. They have some drawbacks, but they are a good place to start for web, desktop, and mobile designers.

5.0 Restrictions and limitations

5.1 Restriction

The Major restriction of this project is that it only involves interaction between the user and administrator. It does not involve any third party for technical support and maintenance.

5.2 Limitation

The data is manually entered into the database. Thus, the data can be un-synchronized or not structured. The retrieval of the data must be done manually by Administrator. For example, if the customer wants to change the DOB, or address it won’t be possible by the customer himself, but a request can be sent

to the administrator, and he/she can change it and then sent the details back to the customer.

5.2.1 Limited Mobile Access

The display and features might not be synchronized or not ideal for viewing on Mobile or tablet. For example, the drop-down menu to select the month or the period of the bill generation – it would not be synchronized properly and thus the UI would not be easy to navigate.

5.2.2 Storage limit

The storage would be limited for the web Application. It includes the number of customers, the period for bill generation NOT exceeding 3 months for each customer and technical and maintenance request NOT exceeding 2 per customer.

5.2.3 Reduced Speed

A web program will most likely run at a somewhat slower speed than one hosted on a local server. As far as Browser Support is concerned, we don't all use the same browser, which is unfortunate. This implies that throughout development, you must guarantee that your software is compatible with a wide range of browsers.

5.3 Manually Updating Required

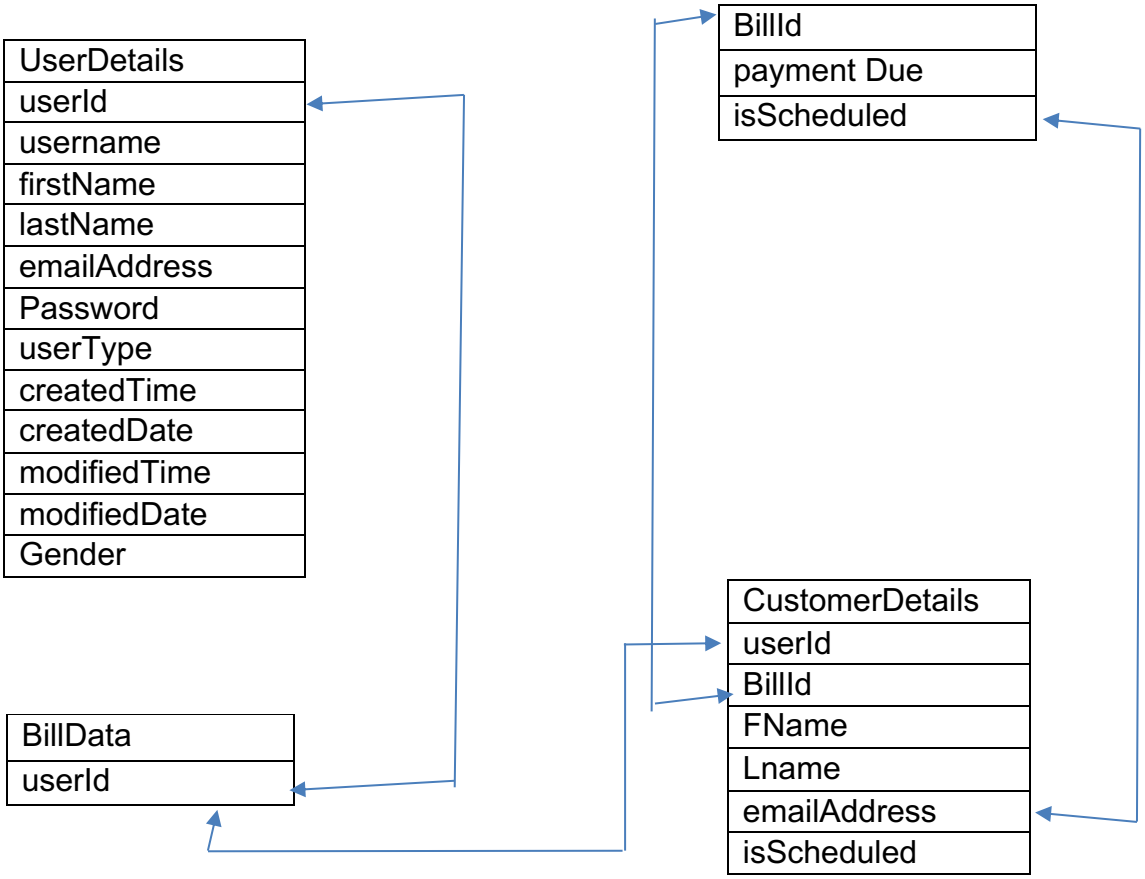
As it is a web application, all the updating would be done by the administrator manually. For example if there is a change in per Unit charge or there are some changes in the records of customer, it would be done by the administrator himself. There is no UI available that can change the records or units. It must be done by the administrator on the required Database.

6.0 Appendices

Appendix A

Customer Information	Bill Information
• First name • Middle initial • Last name • DOB • Address • Payment details	• Unique bill ID • Payment Due • Payment not issued / not paid

6.1 Requirements traceability matrix



6.2 Implementation issues

N/A