*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**

|  |  |
| --- | --- |
| Term | Definition |
| API | Application Program Interface. An API is a method by which an application program can access the computer's operating system. |
| GUI | Graphical User Interface. The GUI provides a graphical interface for users to interact with the system. |
| Link | A link is a means of connection between screens. Note: This is not short for “Hyperlink.” |
| SDK | Software Development Kit. A set of programs that allows software developers to create products to run on a particular platform or to work with an API. |

* 1. **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.

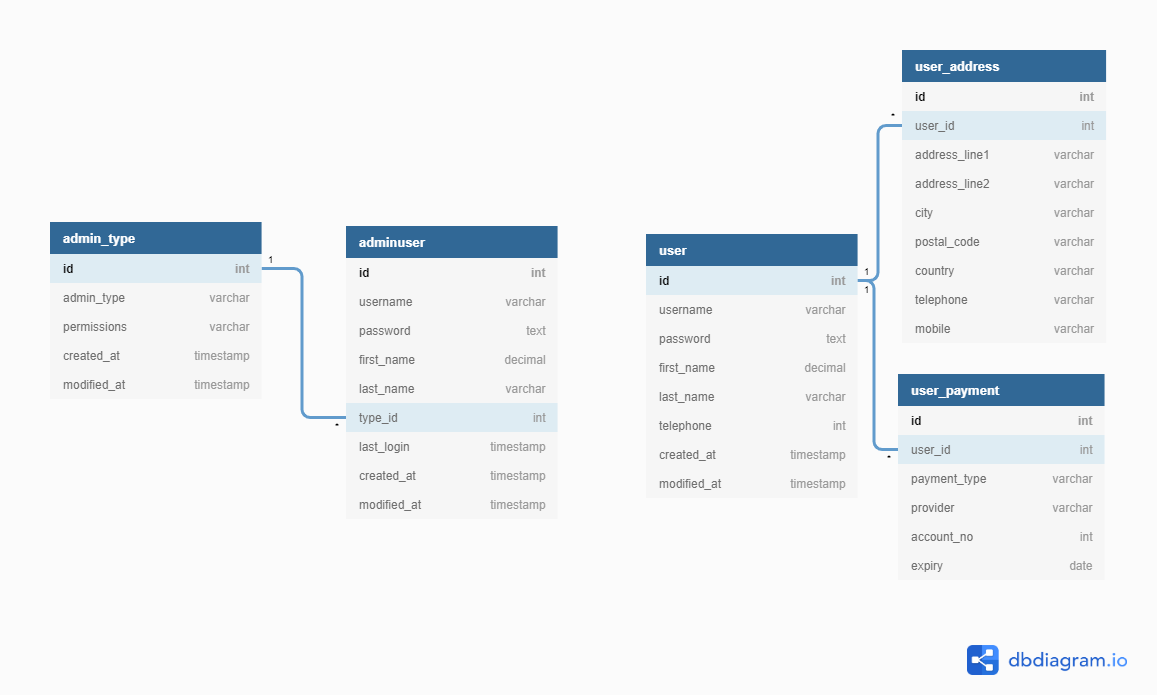
**2.0 Data design**

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.

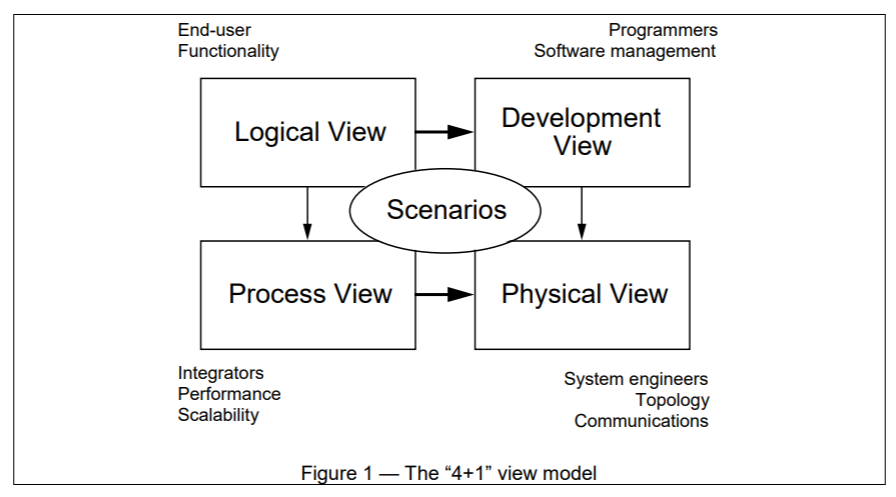
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Table Name | foreign key | Data Type | Allow Nulls | Field Description |
| adminuser | ID | Varchar(50)  INT  DATE | No | Get this field from the database to view user details |
| user | ID | Varchar(50)  INT  DATE | No | view details / can only edit personal information |
| useraddress | USERID | Varchar(100)  INT | No | This will tie to user |
| user payment | UserID | int  DATE | no | user confidential information |

**3.0 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 diagrams**

* Logical view: is concerned with the functionality that the system provides to end-users.
* Development view: illustrates a system from a programmer’s perspective and is concerned with software management.
* Process view: deals with the dynamic aspect of the system, explains the system processes and how they communicate, and focuses on the runtime behavior of the system.
* Physical view: depicts the system from a system engineer's point of view. It is concerned with the topology of software components on the physical layer, as well as communication between these components.



**3.2 Description for Components**

A description of major software components contained within the architecture is presented. Section 3.2.1 is repeated for each of n components.

**3.2.1 Component n description**

**3.2.1.1 Interface description**

|  |  |
| --- | --- |
| Identification | NewUserAccountScreen |
| Input | Sign up |
| Output | This screen contains links to the following screen:   1. 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**

Class diagrams, composite structure diagram, etc.

**3.2.3.3 Dynamic models**

Activity diagrams, sequential diagrams, state diagrams, etc,

**3.3 External Interface Description**

The software's interface(s) to the outside world (other software or hardware systems) are described.

**4.0 User interface design**

A description of the user interface design of the software is presented.

**4.1 Description of the user interface**

A description of user interface including screen images or prototype is presented.

**4.2 Interface design rules**

Conventions and standards used for designing/implementing the user interface are stated.

**5.0 Restrictions, limitations, and constraints**

Special design issues which impact the design or implementation of the software are noted here.

**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**

A matrix that traces stated components and data structures to software requirements is developed.

**6.2 Implementation issues**