SYSTEM DESIGN DESCRIPTION

INFO 2413: System Development Project (A10)

FAM BUDGETLY

An Online Budget Management System – Web Application

Submitted By
Ali Manghat – 100356935
Akshit Thakur – 100394694
Ramandeep Singh – 100394782
Gurpreet Singh – 100400870
Sunpreet Kaur – 100410018

OCTOBER 17, 2022

Contents

1. Introduction 2
   1.1 Purpose 2
   1.2 Scope 2
   1.3 Intended Audience 2
   1.4 References 2
   1.5 Summary 2
2. Definitions 3
3. Design Viewpoints 4
   3.1 Logical Viewpoint 4
      3.1.1 Class Diagram 4
      3.1.2 Class Diagram Summary 4
      3.1.3 Relationship Summary 6
   3.2 Information viewpoint 7
      3.2.1 User Information Table 7
      3.2.2 User Credential Table 7
      3.2.3 Login Data Table 8
      3.2.4 Category Table 8
      3.2.5 Spending Data Table 9
      3.2.6 Budget Data Table 9

1. Introduction

1.1 Purpose

The purpose of this Software Design Document is to help the software development team of this project understand and visualize the design of the project. In order to help us do this, we have developed a class diagram that lists the classes, the properties of the classes, and useful methods for the classes for this system. Also, an understanding of the database is crucial to have a well-functioning system. Many people would consider the database to be the most important part of an application since it has such sensitive information about users such as usernames and passwords, and other important data. Therefore, a database schema has also been developed in this document along with database table info. Having all this information readily available before the code is written will help us save time and write more efficient code.

1.2 Scope

This Software Design Document gives a clear and precise view of the budgeting system. It shows the thinking behind the system and the logical structure of how the software development team is planning to design the system. This is shown through the class diagram and database schema that will be used in this project.

1.3 Intended Audience

The intended audience for the Software Design Document is the software development team. The document is intended to help us understand what we are trying to achieve and what is the most efficient way to get there.

1.4 References

- "UML Class Diagram Tutorial" https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/

- 2 September 2022 -"Primary Key" https://en.wikipedia.org/wiki/Primary_key

- 12 October 2022 - "Foreign Key" https://en.wikipedia.org/wiki/Foreign_key

1.5 Summary

This document contains all the structural and database designs for the application. The Logical viewpoint covers everything related to the structure of classes, their methods, their relationships, and a UML class diagram to help understand the layout in an easy way. Information Viewpoint covers all the structural details of the database and the tables to store information in real-time. All this design information will help us understand the application's backend functioning.

2. Definitions

a) Primary Key - Relational databases use primary keys to uniquely identify tuples in relationships, and they are made up of a set of attributes.

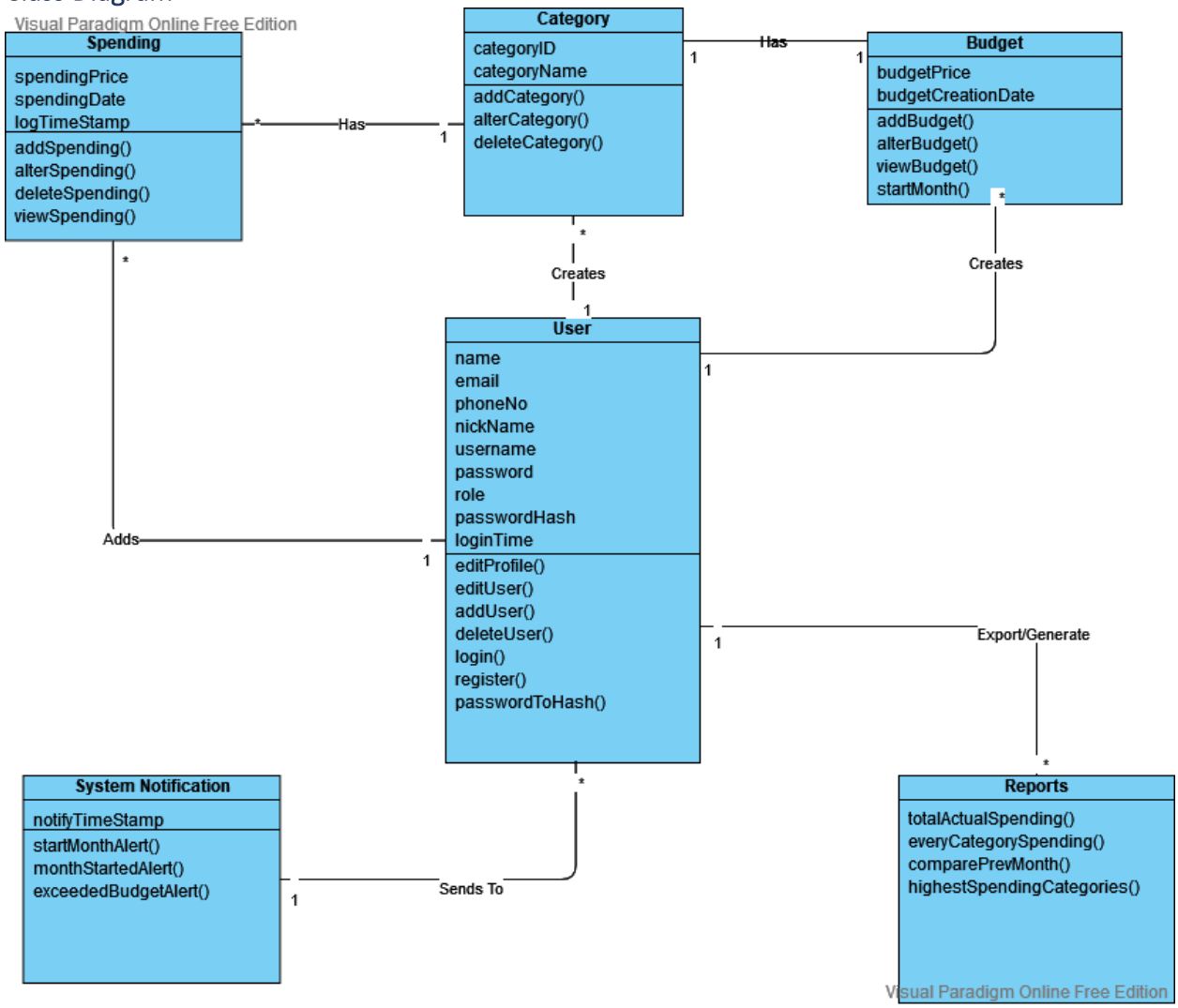
b) Foreign Key - Foreign keys refer to attributes in a table that correspond to the primary keys in other tables. These two tables are linked by a foreign key.

c) UML Diagram - A UML diagram is a diagram derived from the UML (Unified Modeling Language) that illustrates an entire system, along with its roles, actions, artifacts or classes, for the purpose of understanding, altering, maintaining, or documenting its contents.

3. Design Viewpoints

3.1 Logical Viewpoint

3.1.1 Class Diagram



3.1.2 Class Diagram Summary

a) *User Class*

The user class will have following attributes and operations:
Name: Full name of the user
Email: Email address of the user
PhoneNo: Phone number of the user
NickName: Nick name of the user
username: Unique username of the user
password: Password for the user
Role: User's role i.e., Super user or Regular User

passwordHash: An attribute to store hash code of the password
loginTime(): time at which a user will log into his/her account
editProfile(): A function to change personal profile information.
editUser(): A function to edit any regular user's profile.
addUser(): A function to add a new regular user.
deleteUser(): A function to delete an existing regular user.
login(): A function for users to log into the website using username and password.
register(): A function to register an account.
passwordToHash(): A function to convert password to a hash code

IMPORTANT NOTES:
a) Functions such as editUser(), addUser(), deleteUser() and startMonth() are only accessible depending on the role of the user who logged into his/her account. Here, only Super User can access them.
b) All the new users registering on the website will have their role set as "Regular Users" by default.

b) *Category Class*

The category class will have the following attributes and operations:
categoryID: Unique identifier for each category
categoryName: Name for the category
addCategory(): A function to add a new category
alterCategory(): A function to alter an existing category
deleteCategory(): A function to delete an existing category

c) *Budget Class*

The budget class will have the following attributes and operations:
budgetPrice: Budget price set for each category at the start of the month.
budgetCreationDate: Date and time when the budget is created
addBudget(): A function to add budget to a given category by the Super User
alterBudget(): A function to alter budget of a given category by the Super User
viewBudget(): A function to view the budget set for the month for each category.
startMonth(): A function to start the budget of the month.

d) *Spending Class*

The spending class will have the following attributes and operations:
spendingPrice: Spending price added by any user for a specific category.
spendingDate: The date on which spending occurred.
logTimeStamp: Date and Time an which spending is logged into the system
addSpending: A function to add spending for a category by any user
alterSpending: A function to alter spending for a category by any user
deleteSpending: A function to delete a specific spending by any user
viewSpending: A function to list and view all the spending data to the user.

e) *Reports Class*

The reports class will have the following operations:
totalActualSpending(): A function for comparing the monthly budget and the actual spending in total
everyCategorySpending(): A function for comparing the actual spending with every category's budget.
comparePrevMonth(): A function for analysing the current month in comparison with the previous month
highestSpendingCategories(): A function for the top 3 categories for which the spending has exceeded 90% or more of the budget in the last 3 months.

f) *System Notification Class*

The system notification class will have the following attributes and operations:
notifyTimeStamp: Date and Time at which the notification will be sent.
startMonthAlert(): A function to send alert notification starting the 2nd day of the month if the month is not started by the Super User. Only sent to Super Users.
monthStartedAlert(): A function to notify all users once the Super Users start the month.
exceededBudgetAlert(): A function to notify all the users that the budget has exceeded 80% for a specific category.

3.1.3 Relationship Summary

a) User and Categories: (1 to Many)
Users (Only Super Users) can create multiple categories.
b) User and Budget: (1 to Many)
Users (Only Super Users) can create a budget to start the month.
c) User and Spending: (1 to Many)
Users (All Users) can add spending once the month is started.
d) User and Reports: (1 to Many)
Users (Only Super Users) can generate different comparison reports.
e) Category and Budget: (1 to 1)
A Category has a maximum budget assigned to it when the month starts.
f) Category and Spending: (1 to Many)
A Category also has spending which is added by the User.
System Notification and Users: (1 to Many)
System Notifications are sent to Users.

3.2 Information viewpoint

3.2.1 User Information Table

This table will contain all the personal information of the users.

a) User_ID: Unique ID for each user
b) Full Name: User's full name
c) Email: User's email address
d) Phone_Number: User's contact information
e) Nick_Name: User's nickname

Primary Key: user_id

```
9  SELECT * FROM user_information;
```

Data Output | Explain | Messages | Notifications

user_id [PK] integer	full_name character varying (100)	email character varying (100)	phone_number character varying (100)	nick_name character varying (100)

3.2.2 User Credential Table

This table will contain login credentials for login access to the website and the user's role.

a) User_ID: User's unique ID for identifying the user.
b) Username: User's unique login-name for authentication.
c) Password_Hash: The user's password is stored as a hashcode for login access.
d) User_Role: Either "Super" or "Regular" to provide authentication for different privileges.

Primary Key: user_id
Foreign Key: user_id from "User Information" table

```
8  SELECT * FROM user_credential;
```

Data Output | Explain | Messages | Notifications

user_id [PK] integer	username character varying (100)	password_hash character varying (500)	user_role character varying (100)

3.2.3 Login Data Table

This table will contain the time at which a user logs into the website.

a) User_ID: User's Unique ID
b) Time_Logged_In: Time and Date at which the user logs in.

Composite Primary Key: user_id and time_logged_in
Foreign Key: user_id (from "User Information" table)

```
7  SELECT * FROM login_data;
```

Data Output | Explain | Messages | Notifications

user_id [PK] integer	time_logged_in [PK] timestamp without time zone

3.2.4 Category Table

This table will contain all the categories, e.g. food, entertainment, mobile_bill etc.

a) Category_ID: Unique ID to identify category
b) Category_Name: Name of the category

Primary Key: category_id

```
6  SELECT * FROM category;
```

Data Output | Explain | Messages | Notifi

category_id [PK] integer	category_name character varying (100)

3.2.5 Spending Data Table

This table will contain information regarding the spending done by all the users.

a) User_ID: Unique ID to identify the user who added the spending data
b) Time_of_Entry: Time and Date at which a user enters the spending data
c) Category_ID: Unique ID to identify the category for which spending data is entered
d) Spending: Spending amount
e) Spending_Date: Date entered by the user on which spending was done.

Composite Primary Key: user_id and time_of_entry
Foreign Key: user_id (from "User Information" table)
category_id (from "Category" table)

```
SELECT * FROM spending_data;
```

Data Output | Explain | Messages | Notifications

user_id [PK] integer	time_of_entry [PK] timestamp without time zone	category_id integer	spending numeric (20,2)	spending_date date

3.2.6 Budget Data Table

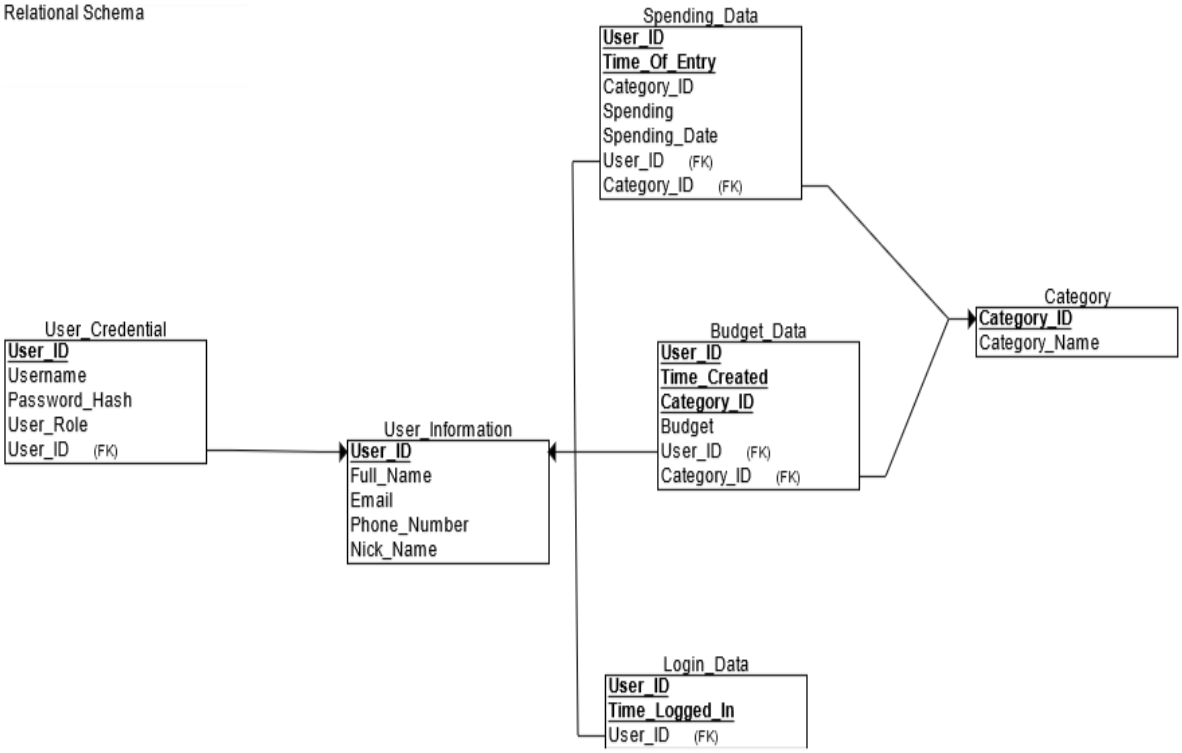
This table will contain the budget data which is set by the super-user at the start of each month.

a) User_ID: Unique ID to identify the user who starts the month
b) Time_Created: Date and Time at which the super-user starts the month
c) Category_ID: Unique ID to identify the category for which the budget is set
d) Budget: Maximum budget set by the user for each category

Composite Primary Key: user_id, time_created and category_id
Foreign Key: user_id (from "User Information" table)
category_id (from "Category" table)

```
SELECT * FROM budget_data;
```

Data Output | Explain | Messages | Notifications

user_id [PK] integer	time_created [PK] timestamp without time zone	category_id [PK] integer	budget numeric (20,2)

Relational Schema