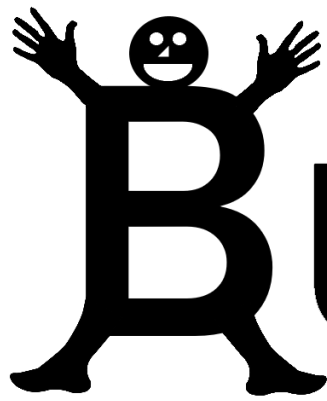


CSCE 361 Software Engineering



Budget Pal

Kyle Crowder, Jacob Norton, Harrison Hruby, Jon
Glodowski (Team 5)

Design and Design Specification

Document

Version: (1.0)

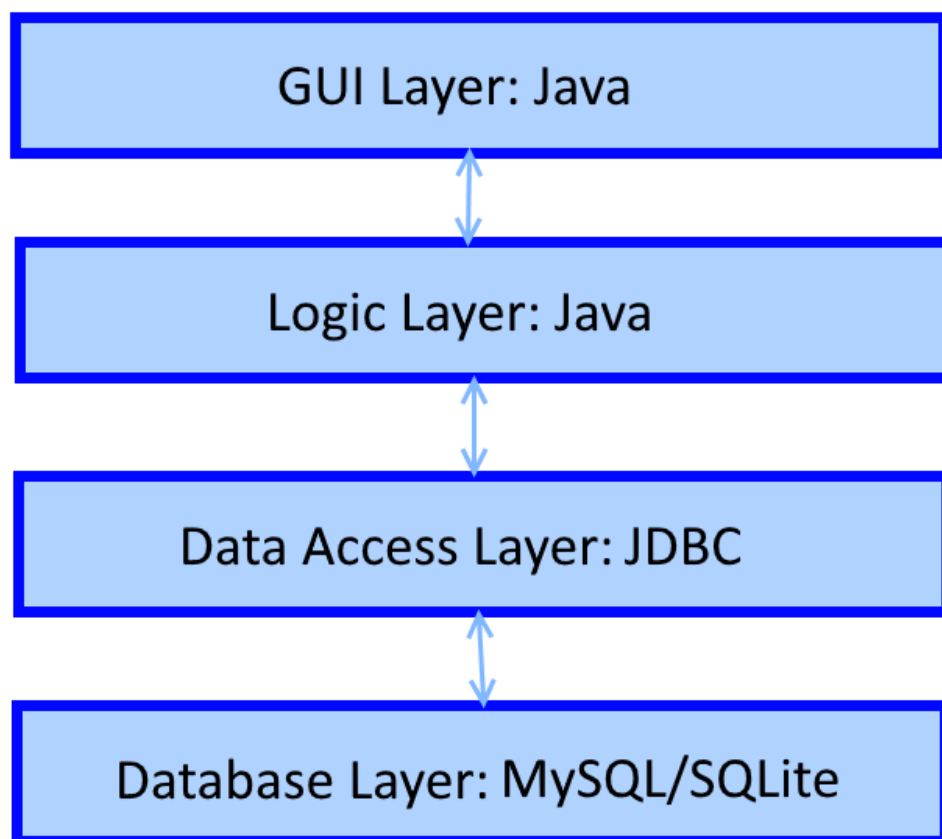
Date: (02/22/2017)

1. Introduction

This design document will demonstrate and describe at a high level the architecture and relations between the entities that make up the BudgetPal application. This document will include diagrams to show relations between different parts of the application. This document will also include diagrams to demonstrate the relationships from the the parts to the database used by the application. The audience for this document is those who will be implementing and providing maintenance to the application.

2. Architecture

2.1. Introduction



The architecture of this application will be a layered model. The lowest layer of this model will be the MySQL/SQLite database to store all data. The data will be accessible via the data access layer which will retrieve the data for use in upper layers. The logic layer will handle all the logic and functions required for the top layer. The top layer will be the GUI layer which will be where all user interactions take place.

2.2. Modules

2.2.1. Database Layer

Paraphrase: This layer stores all of the data that is used by the system and manages the relationships between the data. A MySQL/SQLite database will be used to store the data and Android Studio will easily allow for this incorporation. The relationships between the data will be discussed further in the upcoming sections.

2.2.2. Data Access Layer

This layer controls communication between the Logic & GUI layers and the Database layer. It stores the data that is sent by the Logic & GUI layers into the Database. Any communication to and from the Database layer will be required to pass through this layer. This layer also transforms the information between its database representation and its representation as objects in Java. It will be able to update the respective formats in both directions. JDBC will be used to implement this layer.

2.2.3. Logic Layer

This layer does the majority of the required budgeting and date logic to the Java objects. The data calculated in this layer will then be displayed in their respective formats using the GUI layer and will also be sent to the data access to update the database.

2.2.4. GUI Layer

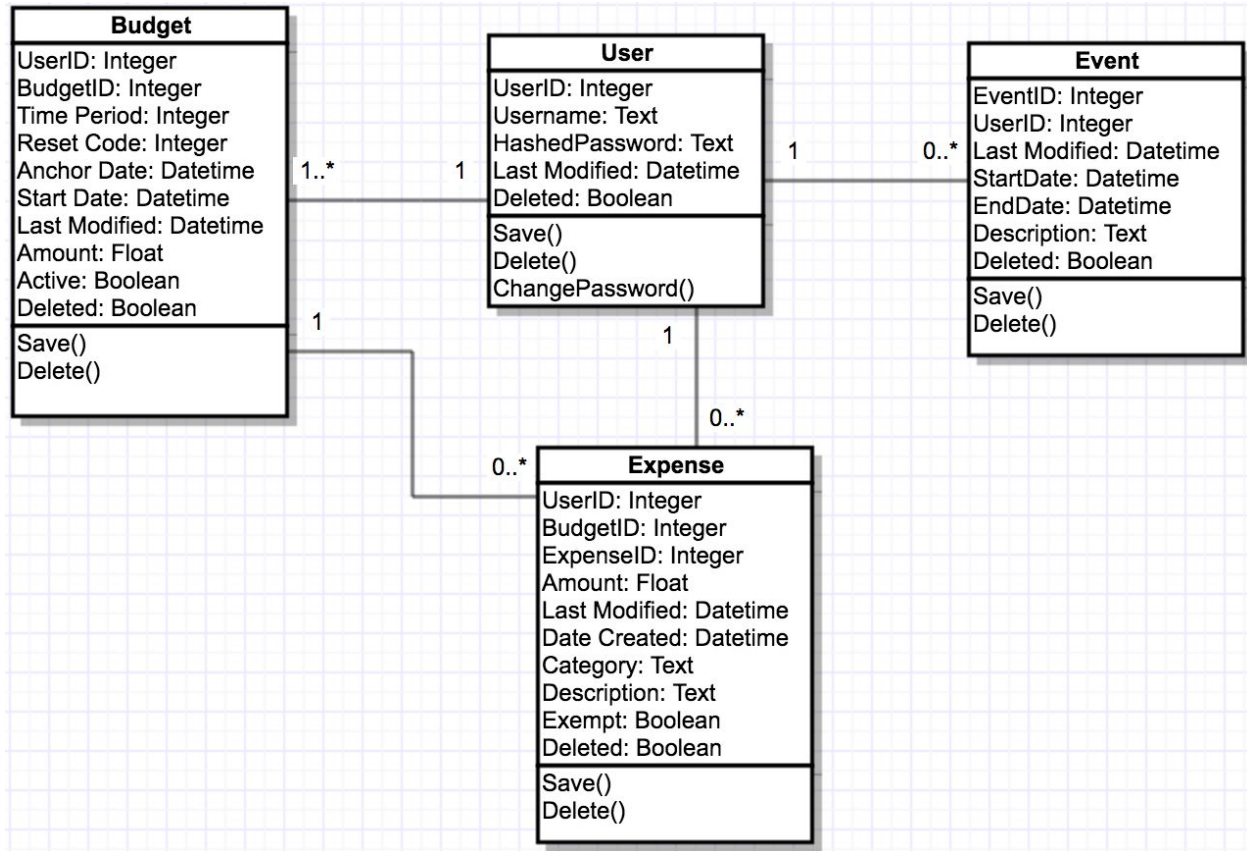
The user will interact with this layer. It will create everything the user will be able to see and touch. The user will use this layer to login to the budgeting app. The interface will be able to adapt to the changing data. Users will be able to interact with the GUI layer to create new expenses and events and allow them to set their budget. These modifications will be carried on through the layers to the database layer for storage. In the future when a web app is developed, the GUI layer will change but all other layers will remain unchanged.

3. Class Diagrams

3.1. Data Table Classes

The system will be using an SQL/SQLite database to store and persist all relevant class data. The major classes include Users, Expenses, Events and Budgets. These classes, their relevant data, as well as their relationships is shown in the figure below.

3.1.1. Schema



3.1.2. Schema Information

Users: This table represents all users of the system. Users will be associated with an auto-generated ID, a username, a hash-protected password, a last-modified date, and a deleted boolean.

Expenses: This table represents the expenses that each User will incur. Expenses will be associated with an auto-generated ID, and amount, a last-modified date, a create date, a category, a description, an exempted boolean, and a deleted boolean. Expenses will also contain foreign keys in the form of an associated Budget ID and an associated User ID.

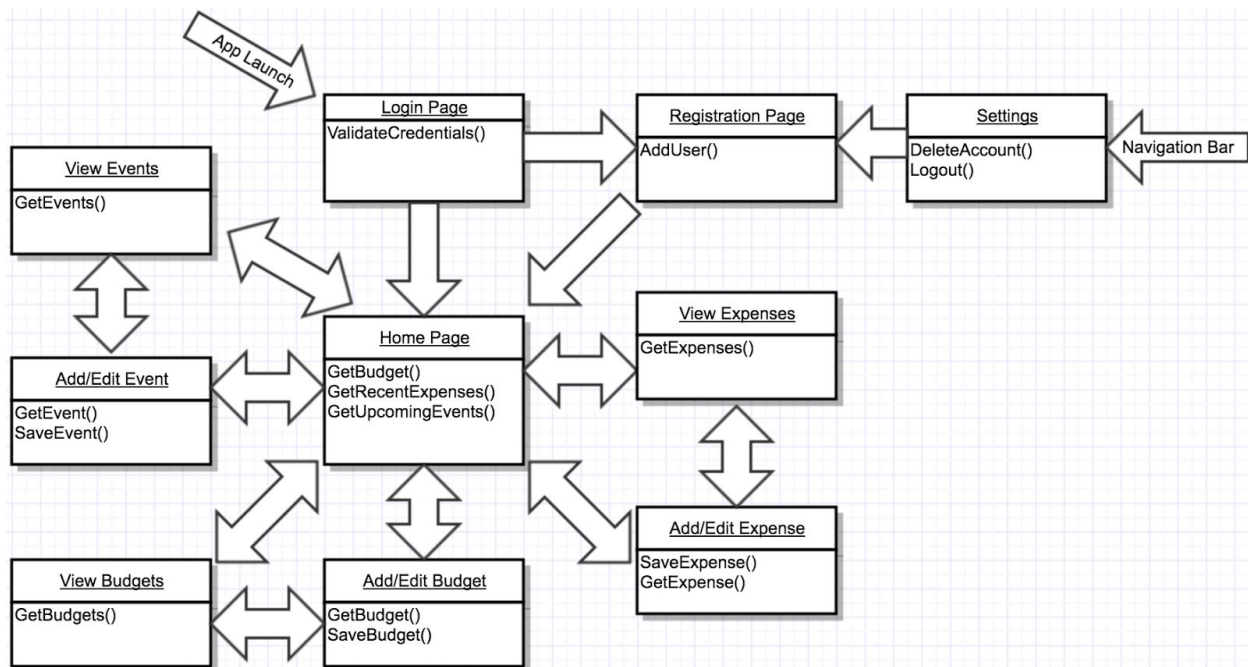
Events: This table represents the various deadlines and financial events that a User will create throughout their experience with the system. Events will be associated with an auto-generated ID, a last-modified date, a start date, an end date, a text description and a deleted boolean. Events will also contain foreign keys in the form of an associated User ID.

Budgets: This table represents the amount of money a User is willing to spend throughout their designated time. Budgets will be associated with an auto-generated ID, a time period integer, a reset code integer, an anchor date, a start date, a last-modified date, an amount, an active boolean, and a deleted boolean. Budgets will also contain foreign keys in the form of an associated User ID.

3.2. Class Information

The Classes will be implemented in JDBC the data access layer of the layered system and used throughout the BudgetPal application. Classes will replicate the database schema above and each table will relate to a specific JDBC class. Then the upper logic layer will take these classes and pass the need information to the upper GUI layer where it is then used in displaying information to the user.

3.3. GUI Layer



Login Page is the opening page and will allow the user to input their credentials and upon correctly input credentials will direct them to the Landing Page. From the Landing Page, Users will be able to navigate to the Budget Page. There the Users shall be allowed to set and update their budgets as well as their cycling time periods. The Event Page is also accessible from the Landing Page. Here, the any and all User created Event objects will be displayed. Users will also be able to add/delete Event objects which will update the Event Page accordingly. The Navigation Drawer will be available from all pages (omit: Login Screen) and, when activated, will allow the User to navigate to any other page. These actions will ease the navigation of the app.