BudgetMinder

# Technical Reference Manual

## Tim mcvicker

## Clemson CPSC4820

## Spring 2017

# Table of contents

Introduction 3

Features 4

Database 5

PHP 6

App Code 7

# Introduction

This Manual is intended for use by developers. It will cover the technical details regarding *BudgetMinder* and its multiple components.

*BudgetMinder* is an Android application that will track and remember purchases and incomes that a user enters through the app. It will keep track of how much money a user has (their balance), and help to alert the user when he or she is spending over what they should. It will provide detailed statistics of expenses and suggestions for how to save more money. There will be multiple users and each user will be able to specify a location for their expense. The application will then display statistics based on the sum of all users’ expenses, and will use these statistics to suggest deviations to everyday expenses that will help the user. *BudgetMinder* was developed using Android Studio 2.3 with Java 1.8. It requires a minimum API level of 24.

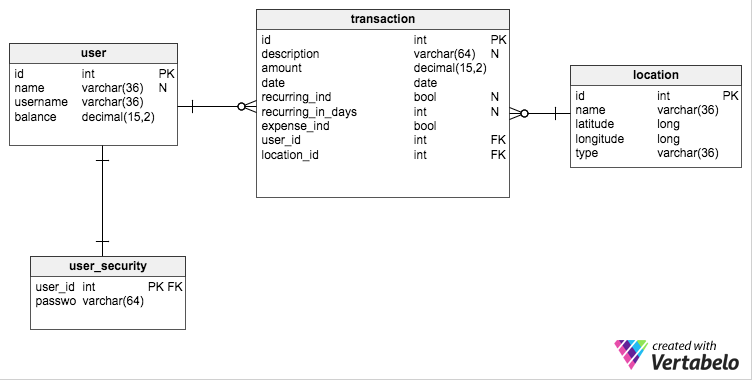
JavaDoc comments will be located at <https://people.cs.clemson.edu/~tmcvick/cpsc4820/files/javadoc_budget/index.html> for download.

# Features

The following are core features of the application:

1. Allow a user to register and create an account
2. Allow a user to change their password/username/name
3. Allow users to enter purchases and incomes
4. Keep track of user's balance and display to the user
5. Display purchase/income history
6. Allow users to use location services to place a location on each transaction
7. Display statistics based on transaction location/type

# Database Design

*BudgetMinder* uses an external and internal database to persist data entered by users.

All four tables are located in the external database. Upon login, all rows from the transaction table for the user are synchronized with the internal database, as well as the user’s id and username in order to facilitate more efficient updates.

Each transaction has a user and a location. Each also has a Boolean that indicates whether the transaction is an expense or income (expense\_ind = true if the transaction is an expense).

In the PHP scripts and the Java code, locations are referred to as “Venues” in order to eliminate confusion with Android Location services.

# PHP SCripts (webservice)

In order to facilitate population of the external database, *BudgetMinder* uses multiple PHP scripts. There is no authoring tool, due to the fact that there should be no modifications or additions to the tables through any medium except the application itself. Each PHP script utilizes query parameters to perform database operations. Links to the scripts are below, with query parameters as examples.

1. **Create User:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/create_user.php?name=Tim&uname=tmcvick&balance=50.34&pword=tim>
2. **Get User by Username:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/get_user.php?uname=tmcvick>
3. **Update User By ID:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/update_user.php?name=Timupdate&uname=tmcvickupdate&balance=55.345&pword=passupdate&id=5>
4. **Get Transactions By UserId:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/get_transactions_for_user.php?id=5>
5. **Create Transaction:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/create_transaction.php?desc=testdesc&amount=50.05&rec=false&days=0&expense=false&user=5&location=1>
6. **Get All Users:** <https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/get_all_users.php>
7. **Get Venues:**

<https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/get_venues.php>

1. **Create Venue:**

<https://people.cs.clemson.edu/~tmcvick/cpsc4820/budget/create_venue.php>

# App Code

The Java code that comprises the *BudgetMinder* app utilizes two bean classes, the Transaction class and the Venue class. These are used to hold the data of each object, and to pass objects of that type throughout the application. The app code also uses two Manager classes, the UserManager and the VenueManager. The UserManager holds all data of the logged in user, including transactions. This class performs operations on each. The VenueManager holds all data of the venues in the database, and performs operations on them. The code utilizes two Helper classes, the BudgetDatabaseHelper and the RestHelper. The BudgetDatabaseHelper performs operations and manages the internal SQLite database, while the RestHelper executes webservice requests (which result in external database operations).