**C868 – Software Capstone Project Summary**

**Task 2 – Section C**

|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | http://www.idevnews.com/views/images/uploads/general/wgu_logo.png  Diabetix |
| **Student Name:** | Clayton Dixon |

# Table of Contents

**Application Design** 4-5

Class Diagram 4-5

**UI Design** 6-7

Wireframes 6-7

**Unit Test Plan** 8-15

Introduction 8

Purpose 8

Overview 8

Test Plan 9

Items 9

Features 9

Deliverables 10

Tasks 10

Needs 11

Pass/Fail Criteria 11

Specifications 12

Procedures 13

Results 14

Source Code 14

Link to Live Version 15

**Maintenance Guide** 16-18

Installation and Using the Application 16

Requirements 16

Installation 16-18

**User Guide** 19-44

Introduction 19

Installation 19

Login Credentials 19

Login 20-21

Main Screen 21-22

Take Reading 23-25

Readings Page 25-29

Edit and Delete Readings 30-32

Reports 33-40

Alerts 41-44

### Application Design

The application design section will include design documents for the project. Each document is essential in the development of the application. Every design document must be presented to and signed off by the stakeholders. Feedback is necessary to iterate on each design document. Included below are the class diagram and the wireframes.

## Class Diagram

The class diagram below represents the Java classes and how they interact with each other. The classes include variables with types and names. The classes interact with the Room database to create, read, update, and delete blood sugar readings. The Reading Entity class is the core of the application. This class interacts with all other classes in the application.

Diagram

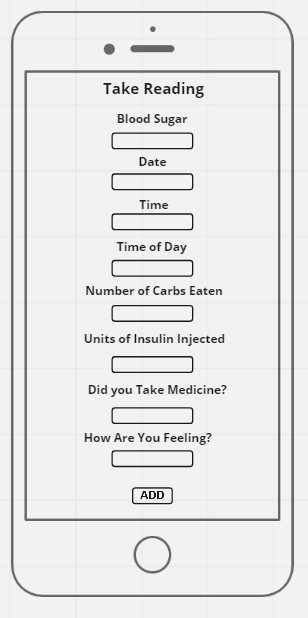
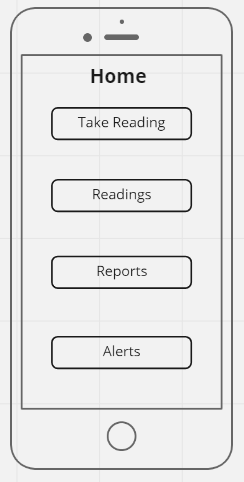
Description automatically generated

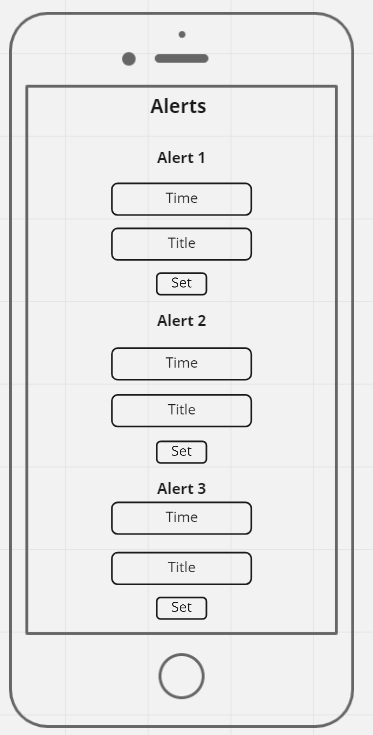
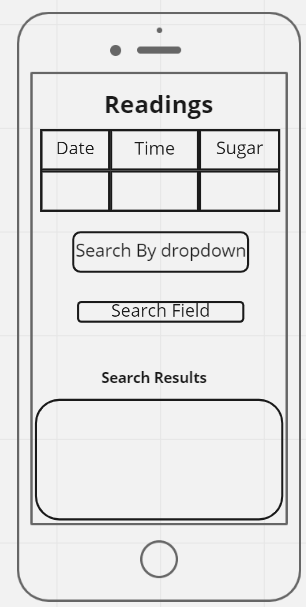
UI Design

## Wireframes

Wireframes are low-fidelity mockups of the final application. It provides a visual flow of the application. The wireframe displays all screens of the application. Wireframes are the basis of prototypes. The wireframes are presented to the stakeholders of the company. Feedback from the stakeholders is used to iterate on the wireframes. The final wireframes must be signed off by the stakeholders. Once the wireframes have been signed off the creation of the prototype can begin.

Diagram

Description automatically generated



# Unit Test Plan

## Introduction

### Purpose

Unit testing is essential when developing professional applications. Unit tests validate that the application functions properly. Each test splits the application into small units of code. This allows for more organized and focused testing. Unit tests work well to test obscure edge cases in applications. Unit tests specifically test the code of the application.

### Overview

The unit tests below verify the database of the application works correctly. The database stores the information entered into the application user interface. The database is queried on every screen of the app except the login and alarms screen. When readings are entered, edited, or deleted the application interacts with the database. When readings are searched or used to generate reports the application interacts with the database. The entire “testDatabase” class creates a database and an instance of the “readingDAO” class. It then tests if a reading can be added, accessed, and deleted correctly.

The function “createDb” creates a Room database and an instance of the “readingDAO” class which is used to interface with the database. The “@Before” annotation above the

## Test Plan

### Items

Three items are required to complete the tests. The first is a test database. The second is an instance of the “readingDAO” class. The third is a “readingEntity” which is a reading that is stored in the database.

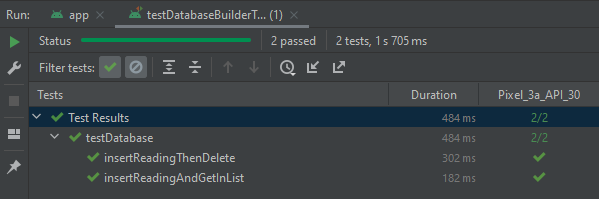
### Features

The function “created” creates a Room database and an instance of the “readingDAO” class which is used to interface with the database. The “@Before” annotation above the “createDb” function tells the IDE to run that piece of code before anything else in the tests. The “closeDb” function closes the database to prevent memory issues after the tests have been run. The “@After” annotation tells the IDE to run that piece of code after all the tests have been run.

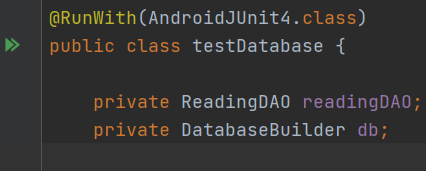
The “insertReadingAndGetInList” test creates a “readingEntity” and assigns values to each parameter in the constructor. Then the reading is inserted into the database. A list is then created and is assigned all the readings in the database. Only one reading should be in the database. Finally, “assertEquals” tests if the reading in the list is the same as the reading we created at the beginning of the test.

The “insertReadingThenDelete” test creates a “readingEntity” and assigns values to each parameter in the constructor. Then the reading is inserted into the database. A list is then created and is assigned all the readings in the database. Only one reading should be in the database. The first “assertEquals” tests if the number of readings in the list is one. The reading is then deleted from the database. Now there should be no readings in the database. The second “assertEquals” tests if the number of readings in the database is zero.

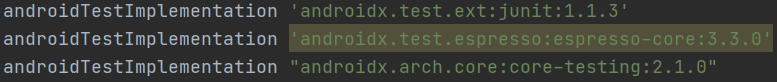
### Deliverables

The tests will produce a pass or fail test result in Android Studio as shown below.

### Tasks

The attached project file must be opened in Android Studio before running the tests. The test class is in the java folder. Then it is under the “com.example.capstoneclaytondixon.Database” package. The file name is “testDatabase”. Click on the green arrow next to the name of the class in the file to run all the tests in the file. The application will then be built, and the tests will be run. 

### Needs

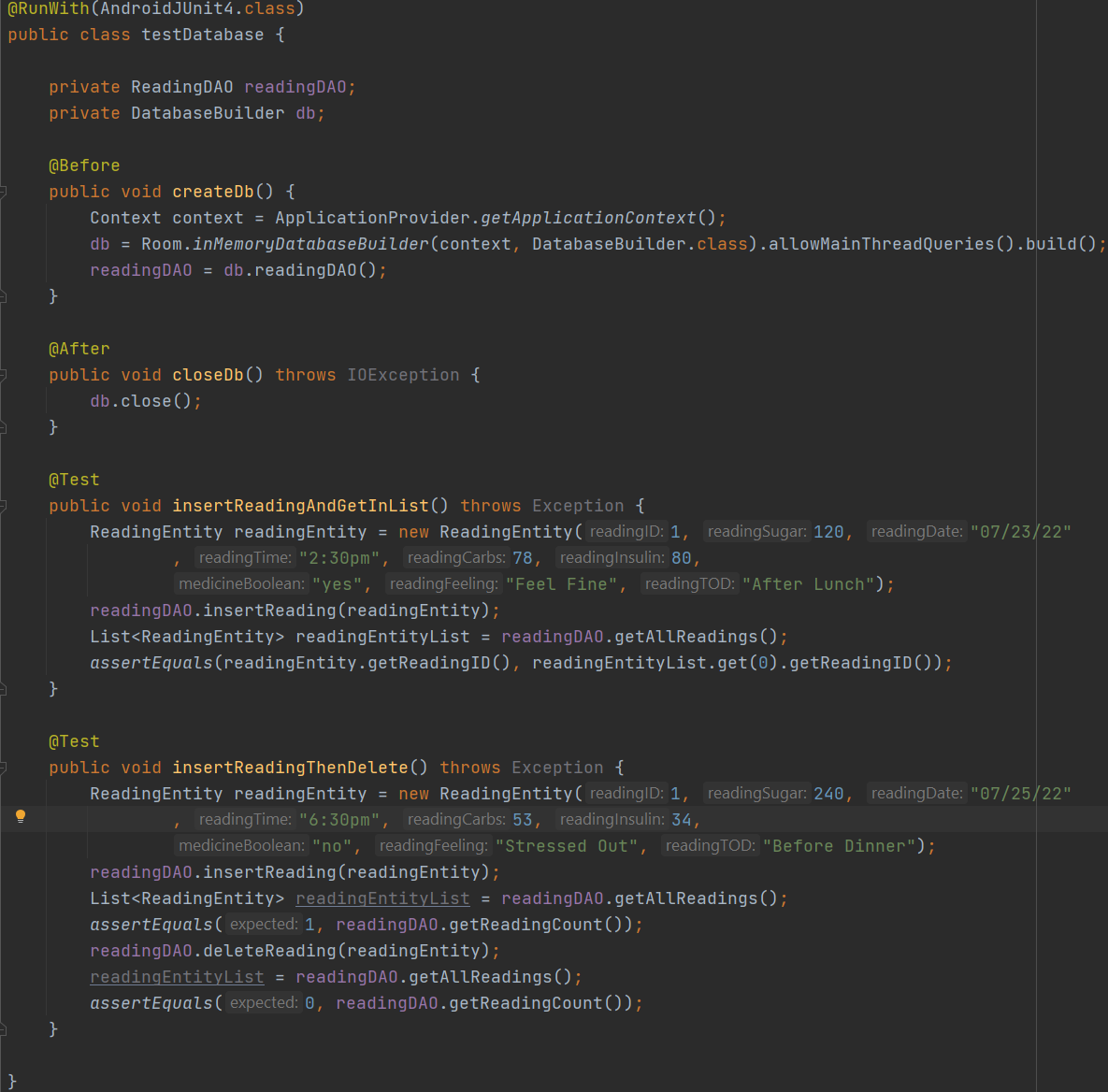
* Android Studio Version 2020.3.0.0 or above
* Room Version 2.3.0
* testInstrumentationRunner 'androidx.test.runner.AndroidJUnitRunner' added in the build.gradle file under android inside the defaultConfig section
* These dependencies need to be added to the dependencies section in the build.gradle file
* The project code files

### Pass/Fail Criteria

The tests are successful if every test passes and is indicated by a green check mark. This means the desired outcome was achieved for each test.

### Specifications

The code for the entire test class is included below. The class name is “testDatabase”. The test class is in the java folder. Then it is under the “com.example.capstoneclaytondixon.Database” package.

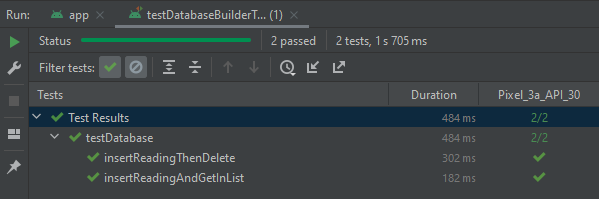


### Procedures

* Open Diabetix project in Android Studio.
* Navigate to the “com.example.capstoneclaytondixon.Database package”
* Create a Java test class
* Add required imports to said class
* Add a function to create the database before the tests have been run
* Add a function to close the database after all the tests have been run
* Add a function to test if a reading can be added and accessed successfully from the database
* Add a function to test if a reading can be added and deleted successfully from the database
* Run the tests in the class
* If the tests fail resolve the issues
* If the tests pass you are done

### Results

A screenshot containing the results of the unit tests is included below.



### Source Code

All source code for the application is in a zip file named “Capstone Clayton Dixon Code.zip”

### Link to Live Version

I did not upload my app to the Google Play Store. There is no link to the live version. Please use the zip file named “Capstone Clayton Dixon Code.zip” to run the application in Android Studio. A test account has already been created to log in.

* Username: username
* Password: password

# Maintenance Guide

### Installation and Using the Application

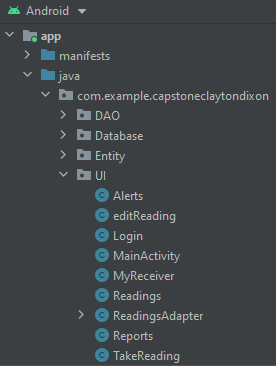
The information below provides details on how to set up and run the application for maintenance purposes. This is intended for developers and not end users.

### Requirements

* Android Studio 2020.3.0.0 or later
* Android Virtual Device with Android Version 11
* Android Java SDK 32

### Installation

* Download the project files to your computer.
* Extract the “Capstone Clayton Dixon Code.zip” file to the desired location on your computer.
* Open Android Studio and go to File in the top left. Click on File and then Open…
* This will bring up a file explorer. Go to the location you extracted the zip file to.
* Select the folder titled “CapstoneClaytonDixon”.
* This should open the project in Android Studio.
* Now you must wait for Android Studio to download all the dependencies for the project.
* Navigate to the “Login.java” file. I attached a screenshot below to show where the file is.



* The default test username is “username”, and the default password is “password”. Altering the login information to your preferred details is simple. Change the userList.add(“username”) to the desired login username. Change the passList.add(“password”) to the desired login password. See the screenshot below for more context.



* You should finally be able to run the application. Click the green arrow in the top right of Android Studio to start the application.

# User Guide

### Introduction

The user guide assists new users of the Diabetix application. The guide provides a walkthrough of every screen and function of the application. Instructions and screenshots for logging in, adding a reading, editing or deleting a reading, searching readings, and generating reports will be outlined below. Pressing the back arrow at any time in the application will take you to the previous page. This includes taking you from the Main Screen to the Login Screen.

### Installation

Installation for general use will be different than installation for maintenance. The application would be hosted on Google Play Store in a real project. Installation on the Play Store is simple. Search for the application name which is Diabetix. Then click install on the app page.

### Login Credentials

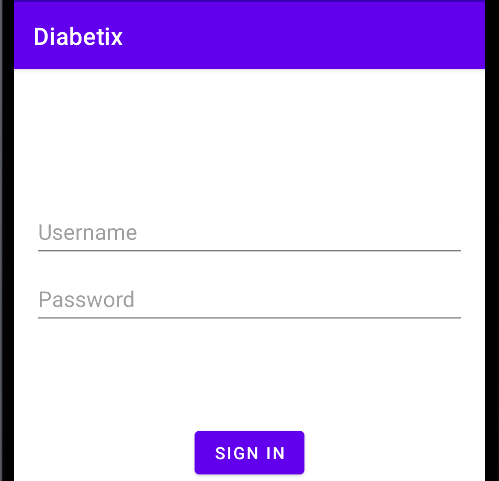
The login credentials for the application would be tied to your Google account. The username and password are the same as the Google account you are logged into. For this demonstration, I will be logging in with the test account information.

### Login

1. Launch the application from your android device app list. The icon of the application is a blue graduation hat with a green background.



1. Once that app has loaded you will be at a login screen.
2. Log in to the application with your Google login information.
3. Click Sign In when you have typed in your username and password.



### Main Screen

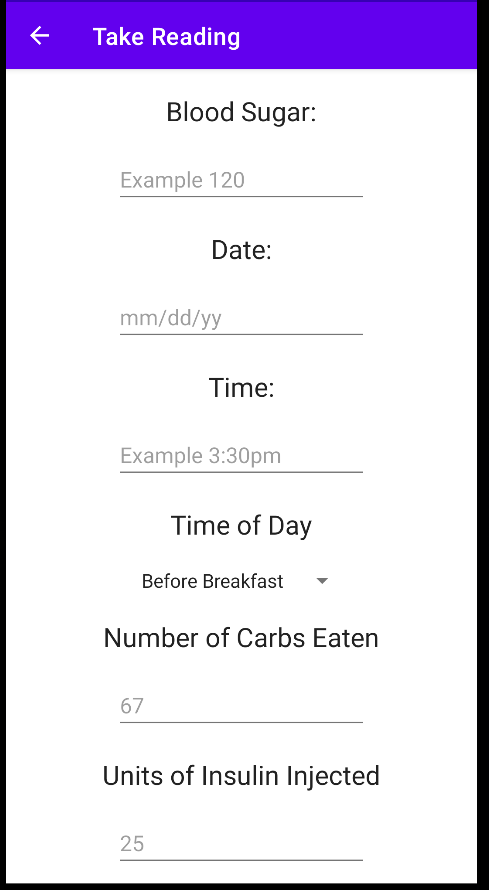
## After logging in you will be brought to the Main Screen.

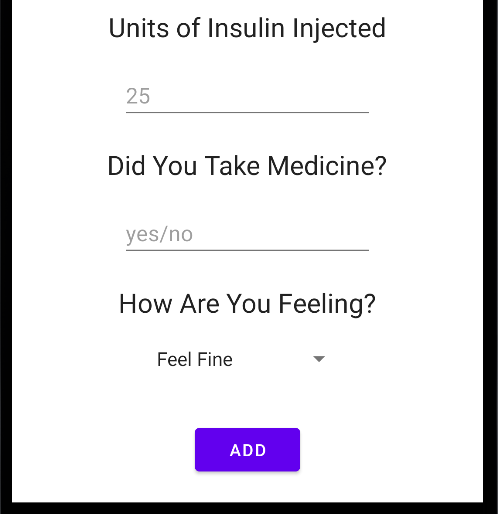
1. Each of the four buttons takes you to a different section of the application.
2. The Take Reading button brings you to the reading entry page.
3. The Readings button takes you to the readings page where you can view all the readings entered and search for readings.
4. The Reports button takes you to the reports page where you can generate various reports on the entered readings. Pressing the back arrow in the top left takes you back to the login page.



### Take Reading

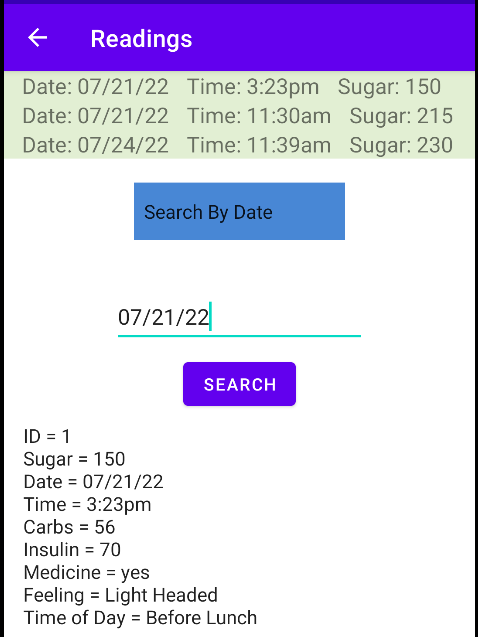
1. Clicking on the Take Reading button from the Main Screen takes you to the Take Reading page. Here you can insert blood sugar readings.
2. There are several fields on this page and each one needs to be filled out. Fill each field with your desired information.
3. The blood sugar field accepts numbers.
4. The date field accepts alphanumerical characters.
5. The time of day field is a drop-down. Select the relevant option.
6. The number of carbs eaten field accepts numbers.
7. The units of insulin injected field accepts numbers.
8. The did you take medicine field is either yes or no.
9. The how are you feeling field is a drop-down. Select the relevant option. Click the Add button once all fields have been filled. The application will then automatically take you to the Readings page. Pressing the back arrow in the top left at any time will cancel this action.



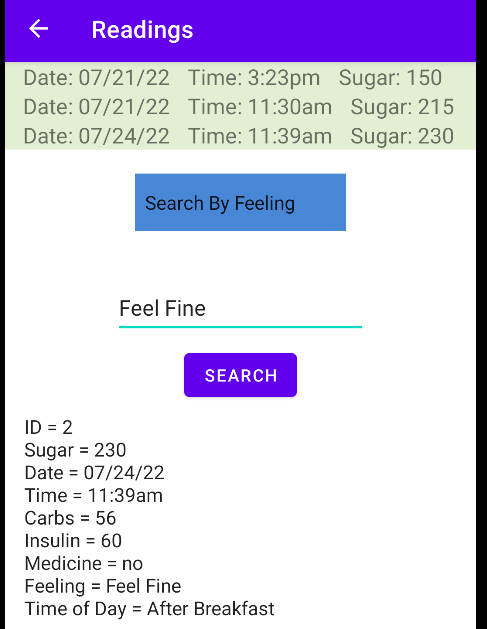


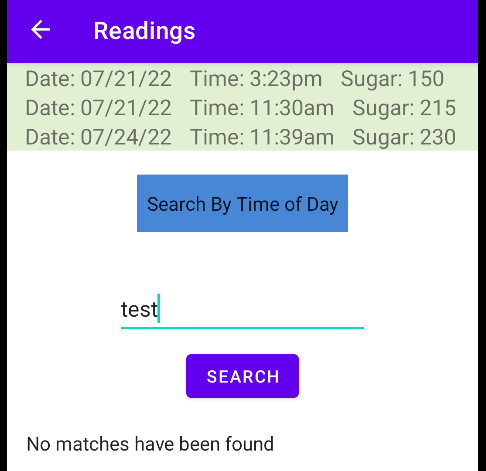
### Readings Page

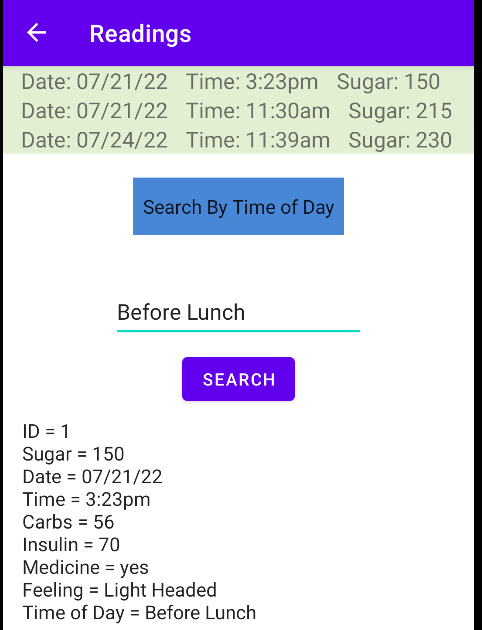
1. The readings page has multiple sections. The top section will display the already entered readings.
2. The Search By Date drop-down lets you select how you want to search the readings. You have three options to choose from. The options are “Search By Date”, “Search By Feeling”, and “Search By Time of Day”.
3. The field below the drop-down is the search bar. Enter the keyword you want to search for.
4. The Search By Date option allows you to type in a date with the format of “mm/dd/yy”. Once you have typed the date into the field, click on the search button. This will display the readings that were inserted with the date specified in the search field. There are two readings with the specified date. You can use your mouse or finger to scroll up and down to show all search results.



1. The Search By Feeling option allows you to search for readings by feeling. The options are Feel Fine, Stressed Out, Light Headed, Don’t Feel Well, and Other. Choose one of these options and type it into the search field. Click the search button. This will display the readings that were inserted with the feeling specified in the search field. You can use your mouse or finger to scroll up and down to show all search results.

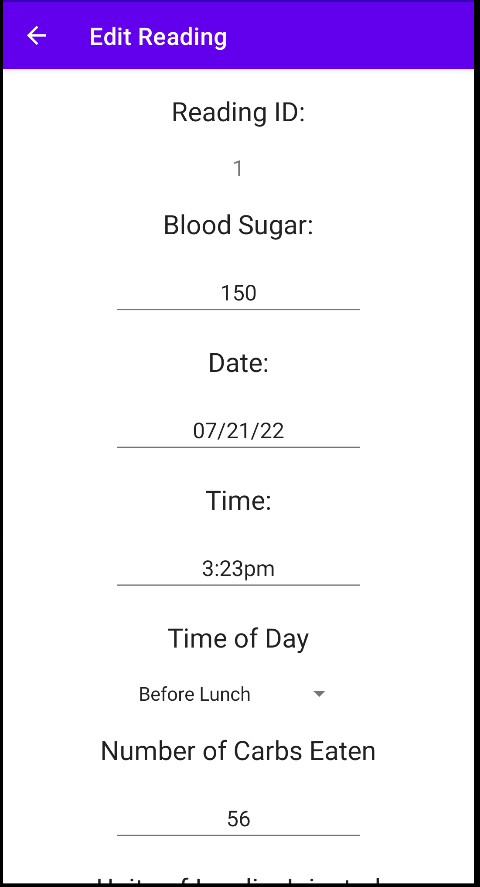


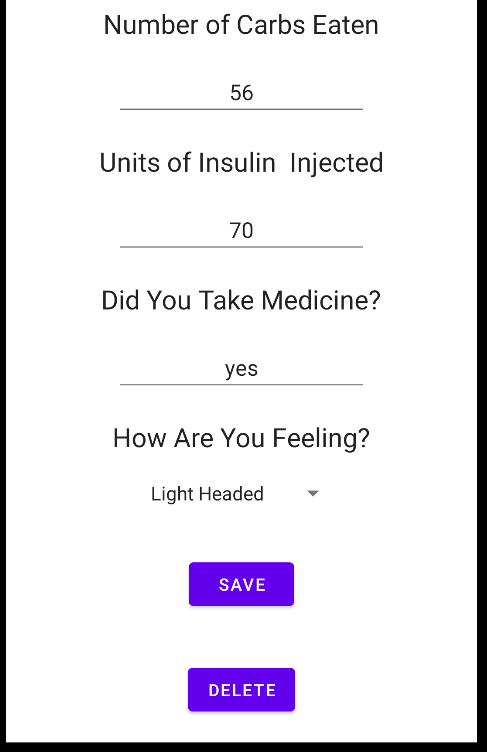
1. The Search By Time of Day option allows you to search for readings by the time of day. The options are Before Breakfast, After Breakfast, Before Lunch, After Lunch, Before Dinner, After Dinner, No Meal, Other/Snack, and After Exercise. Choose one of these options and type it into the search field. Click the search button. This will display the readings that were inserted with the time of day specified in the search field. You can use your mouse or finger to scroll up and down to show all search results.
2. Using any option and typing in a keyword that does not exist will return “No matches have been found.” 



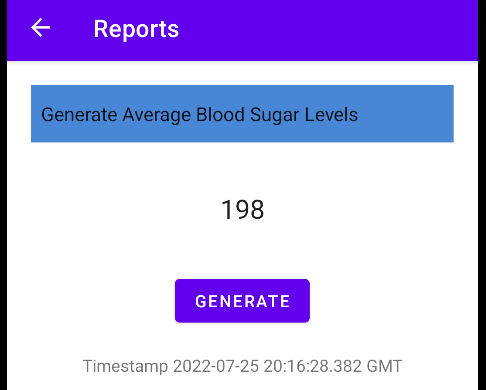
### Edit and Delete Readings

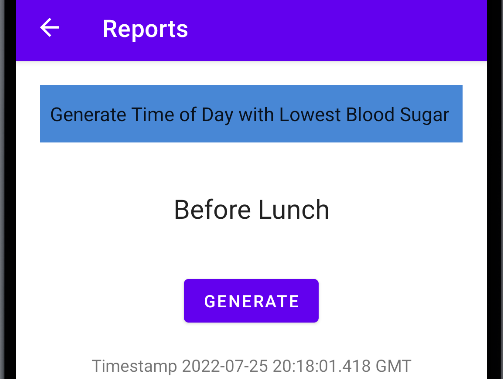
1. Clicking on any readings in the top section of the page will take you to the readings edit page for that reading.
2. There are several fields on this page and each one needs to be filled out. Fill each field with your desired information.
3. The blood sugar field accepts numbers.
4. The date field accepts alphanumerical characters.
5. The time of day field is a drop-down. Select the relevant option.
6. The number of carbs eaten field accepts numbers.
7. The units of insulin injected field accepts numbers.
8. The did you take medicine field is either yes or no.
9. The how are you feeling field is a drop-down. Select the relevant option. Click the Save button to save the changes to the reading. Click the Delete button to delete the reading.



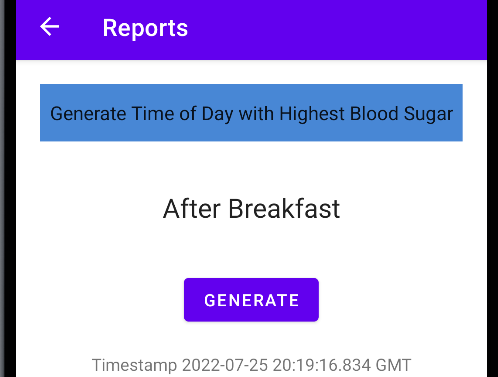


### Reports

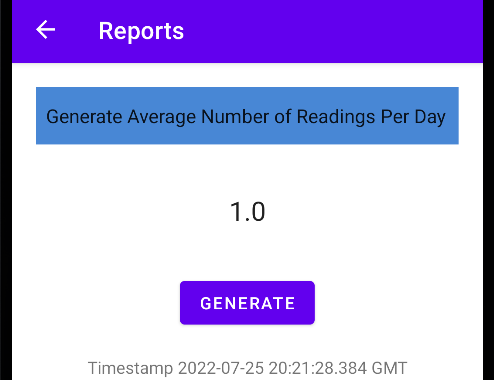
1. Clicking on the Reports button on the Main Screen takes you to the Reports page. There are five elements to the page. The first element is at the top. Generate Average Blood Sugar Levels is a drop-down to select what report you want to run. There are six total options. They are “Generate Average Blood Sugar Levels”, “Generate Time of Day with Lowest Blood Sugar”, “Generate Time of Day with Highest Blood Sugar”, “Generate Average Number of Readings Per Day”, “Generate Readings Under 201”, and “Generate Readings Over 200”.
2. The element below the drop-down is the results field. This is where the result of the report will be output.
3. The element below the results field is the generate button. This generates the report you have selected in the drop-down above.
4. The element below the generate button is the timestamp the report was generated.
5. The final element is at the bottom. This is a list of all the readings used for each report. You can use your mouse or finger to scroll up and down to show all readings used.
6. The Generate Average Blood Sugar Levels option takes all the entered readings and finds the average blood sugar.
7. The Generate Time of Day with Lowest Blood Sugar option takes all the entered readings and finds the time of day with the lowest blood sugar reading.



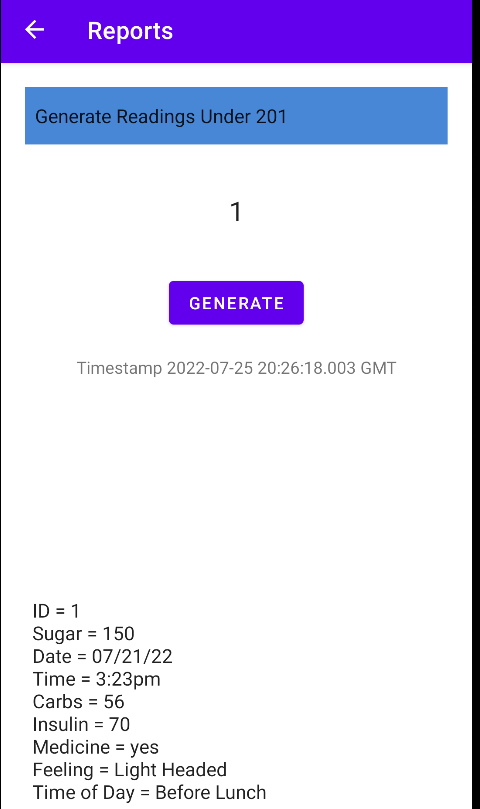
1. The Generate Time of Day with Highest Blood Sugar option takes all the entered readings and finds the time of day with the highest blood sugar reading.



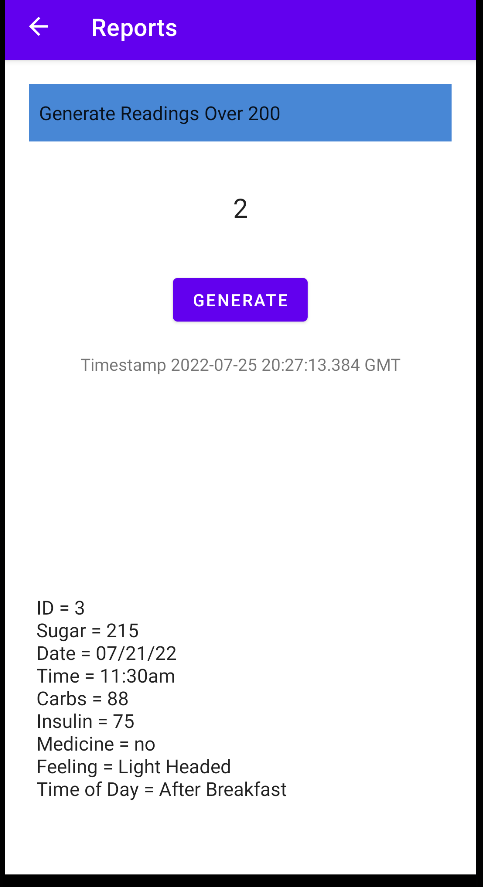
1. The Generate Average Number of Readings Per Day option takes all the entered readings and adds them together. The readings are then divided by the number of days between the earliest reading and the latest reading.

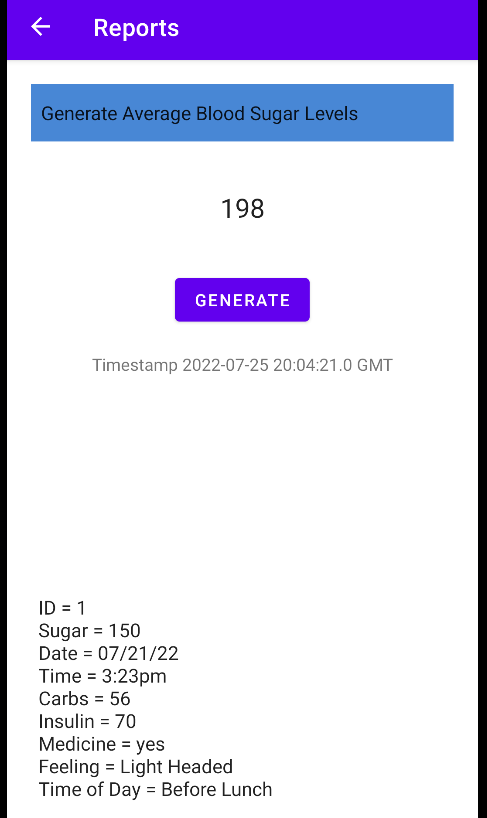


1. The Generate Readings Under 201 option displays the number of readings under 201 in the results field and lists them below. You can use your mouse or finger to scroll up and down to show all readings.



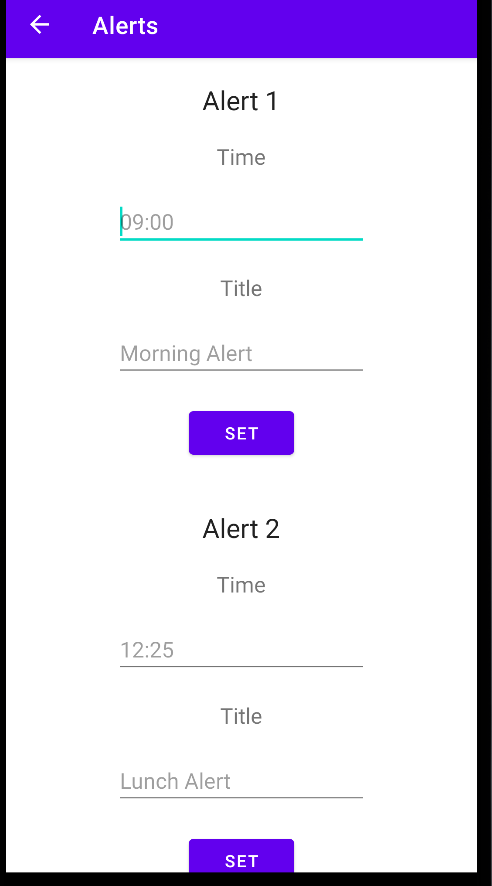
1. The Generate Readings Over 200 option displays the number of readings over 200 in the results field and lists them below. You can use your mouse or finger to scroll up and down to show all readings.



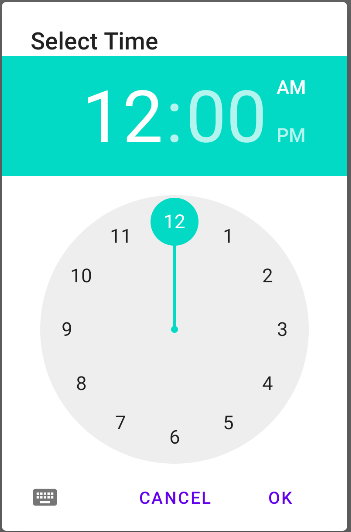


### Alerts

1. Clicking on the Alerts button on the Main Screen takes you to the Alert page.
2. There are three alerts you can set concurrently. You can set only one if you want. You can scroll down to the other alerts with your finger or mouse.



1. There are three elements to each Alert section. The time field can be clicked twice to bring up a clock to select the time. Select the hour then the minute. You can set it to AM or PM. Once you are finished click the OK button and the time field will be populated with the time you selected.



1. The contents of the title field will be the information displayed on the alarm notification.
2. Once both fields are filled click the set button to schedule the alarm.
3. Below is what the notification will look like.

