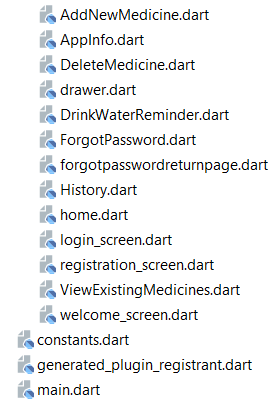
# Criterion C: Development

Virtualis Nutrix is a multi-platform reminder application for my client who finds it difficult to use the inbuilt reminder system in her respective mobile phone. This program has been developed in Flutter[[1]](#footnote-1) using Dart as the primary language. The product has been tested and works on both Android and iOS devices. The screenshots presented in this document are from an Android device. The program structure below displays all the pages in the software.



Program Structure (Source: Android Studio)

**Model-View-Controller architecture (MVC)**

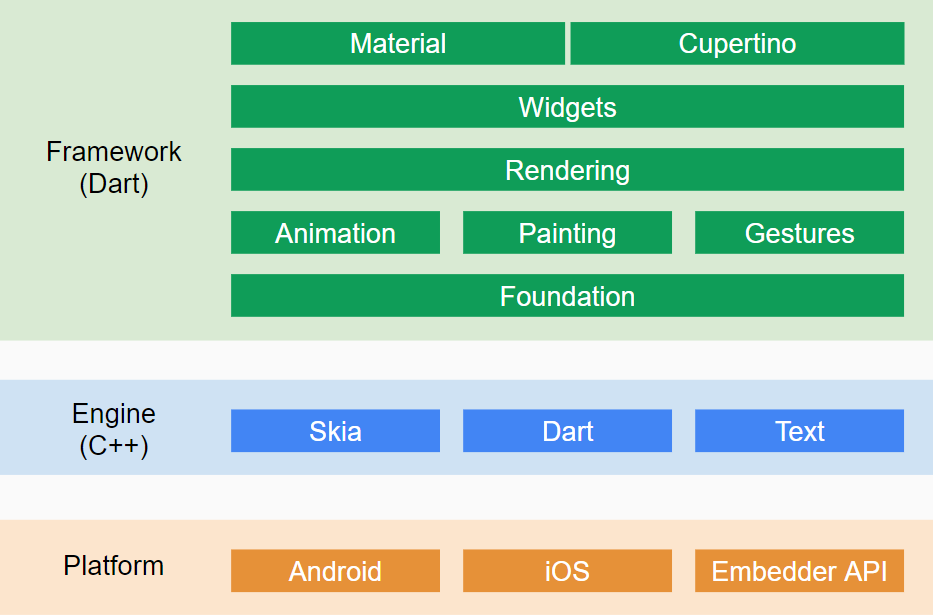
Flutter uses MVC architecture. The Model-View-Controller (MVC) is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller[[2]](#footnote-2). In the App, the data model has been developed using Firebase and the View and Controller are implemented using Flutter classes.



MVC Structure Diagram (Source: <https://developer.mozilla.org/en-US/docs/Glossary/MVC/model-view-controller-light-blue.png>)

**Layers**

Layers are an extremely important concept of the Flutter framework. Layers are arranged in multiple categories based on the complexity and are arranged in the top to down approach. The topmost layer is the UI of the application. The second layer contains all the flutter widgets, and the layers go down to rendering, gestures etc. as shown in the image below.



**Layers in flutter app development (Source:** <https://uniandes-se4ma.gitlab.io/books/assets/flutter-stack.png>**)**

**Database Authentication**

The database authentication contains information about all the users registered for the app. Each user has a specific UID which is used in the application to display information about the user.

Graphical user interface, text, application, email

Description automatically generatedDatabase Authentication

Users email address

Unique user UID

**Data Model**

The data model contains 3 tables with an authentication database. The following are screenshots from the Firebase database.

**Graphical user interface, text, application, email

Description automatically generated**

‘History’ data table

Graphical user interface, text, application, email

Description automatically generated

‘Reminders’ data table

**Graphical user interface, text, application

Description automatically generated**

‘Users’ data table

**View and Controllers Objects**

**Each class used for every screen in the application has this implemented in it. The class contains individual methods to define the UI and controller methods as well associated with every user action on the screen. The controller methods to retrieve the information call the firebase database to fetch a filtered set of documents from the Firebase Database and are delivered as a QuerySnapshot object. This object contains data as an Associative Array. Associative array is a data structure which stores information as an array of name-value pairs**[[3]](#footnote-3)**. One can filter or fetch elements of such arrays using keywords. Below is the screenshot of the view and controller objects for the Registration Screen class.**

A picture containing graphical user interface

Description automatically generated

Outline of the structure of the RegistrationScreen class (Source: Screenshot from Android Studio)

**The Navigation System**

Each page is given a specific id to make it simpler to travel between different pages. The initialRoute defines the page, which is displayed on launching the app, which in this case, is the WelcomeScreen. These routes are defined in the main.dart file which contains the MyApp class.

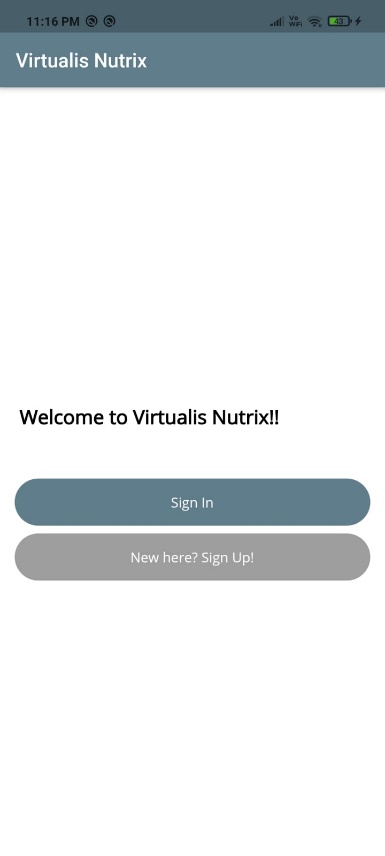
**Text

Description automatically generated**

The MyApp class and the Navigation Routes (Source: Screenshot from Android Studio)

**User Authentication**

1. ***Welcome Page***

**

Screenshot of the welcome page

A stateful widget in flutter the one that changes when a user interacts with it. Hence, the welcome page is a stateful widget as it is used to send users to different pages.

Graphical user interface, text

Description automatically generated

The WelcomeScreen class (Source: Screenshot from Android Studio)

The ‘Welcome Page’ recognizes the type of user navigate to the appropriate authentication page using the Navigator method. The Navigator.pushNamed method is used to send users to different pages.

**

Navigation Functions (Source: Screenshot from Android Studio)

1. *Sign up page*

*Graphical user interface, text

Description automatically generated\*

Screenshot of the Sign-Up page

The Sign-Up page is used to add a new user’s information to the database. A TextField is used for inputting the users’ details. The email is obtained from the following Container. All of the user’s details are obtained by the same manner.



Example of a TextField used for inputting the users email address (Source: Screenshot from Android Studio)

When the form is validated, the createUserWithEmailAndPassword() function is used to add the user’s email and password to the Firebase Authentication database. After registering, the user is sent to the sign in page.

****

Adding the user to the database (Source: Screenshot from Android Studio)

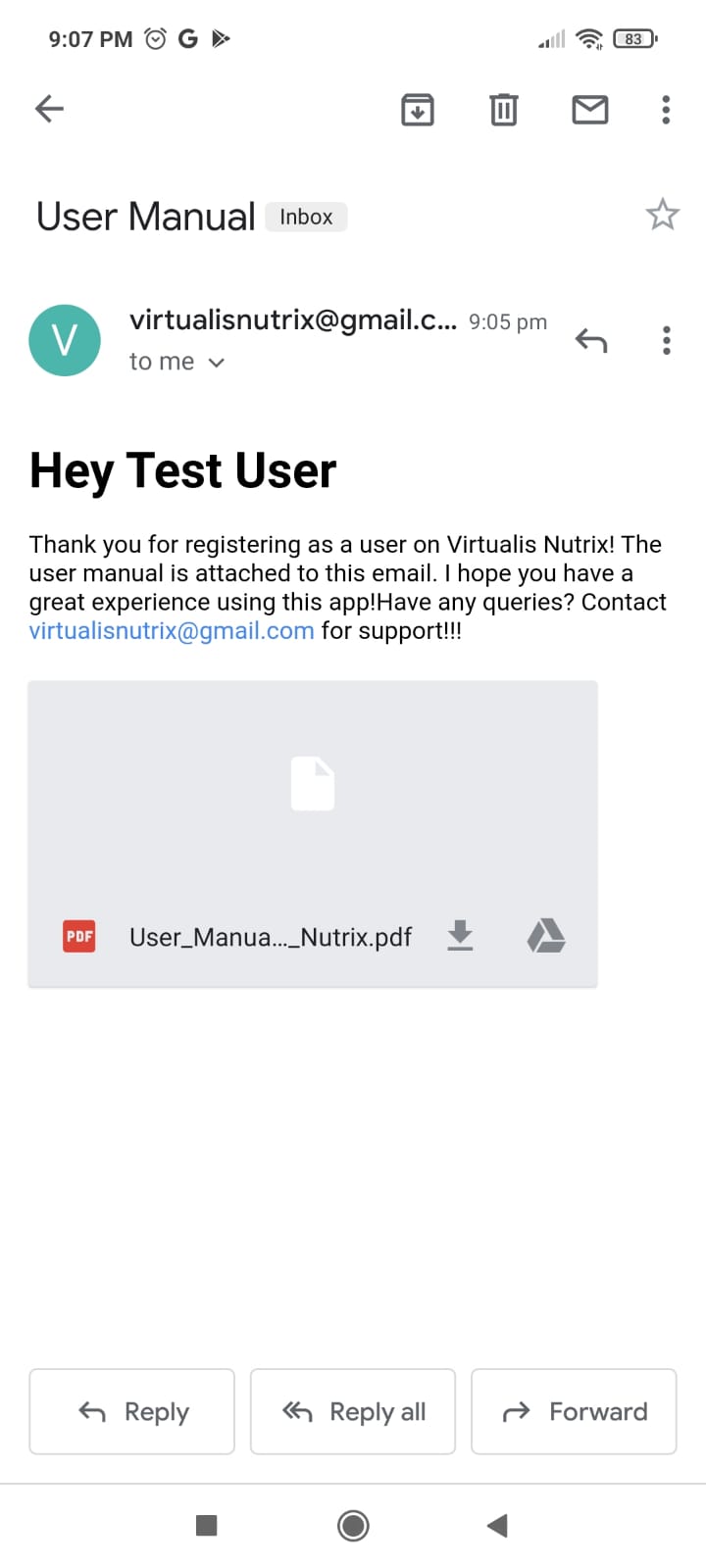
A validator is used in order to display errors if the user enters the wrong email or number. Below is a screenshot for the validator for the mobile number.

Graphical user interface, text, application

Description automatically generated

Phone number validator (Source: Screenshot from Android Studio)

When a user successfully registers on the app, the following email is sent to their email address along with the User Manual (Appendix C1) attached as an attachment for the app.



User manual attached to the email sent to the newly registered user

Email sent to the user after creating an account

The user manual is a PDF document containing information on how to use all the features of the app. To send the email, the sendMail function is called. The gmail Simple Mail Transfer Protocol (SMTP) server is used as a route to send the email.

Text

Description automatically generated with medium confidence

sendMail function to send an email to a user (Source: Screenshot from Android Studio)

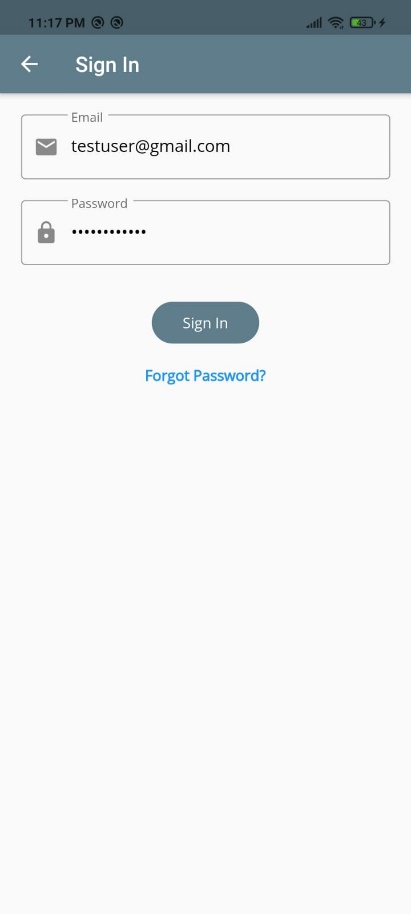
In case an error occurs and the email is not sent to the registered user, the error message is displayed.

Text

Description automatically generated

Error message in case email is not sent to the newly registered user(Source: Screenshot from Android Studio)

1. *Sign In page*

****

Screenshot of the Sign-In page

When the Sign In button is pressed, the data from the TextField is validated through the signInWithEmailAndPassword() method imported from the Firebase Authentication. If the authentication is successful, the email is used to customize the pages based on the users profile.

**Graphical user interface, text, application

Description automatically generated**Signing-In method (Source: Screenshot from Android Studio)

1. *Forgot password page*

**Graphical user interface, application

Description automatically generated Graphical user interface, text, application, chat or text message

Description automatically generated**

Password Reset Page (Source: Screenshot from Android Studio)

This page is used to reset the user’s password. The user has to enter their email into the TextField. When the user clicks on the ‘Send reset email’ button, the PasswordReset function which uses the sendPasswordResetEmail() function is called.

Graphical user interface, text

Description automatically generated

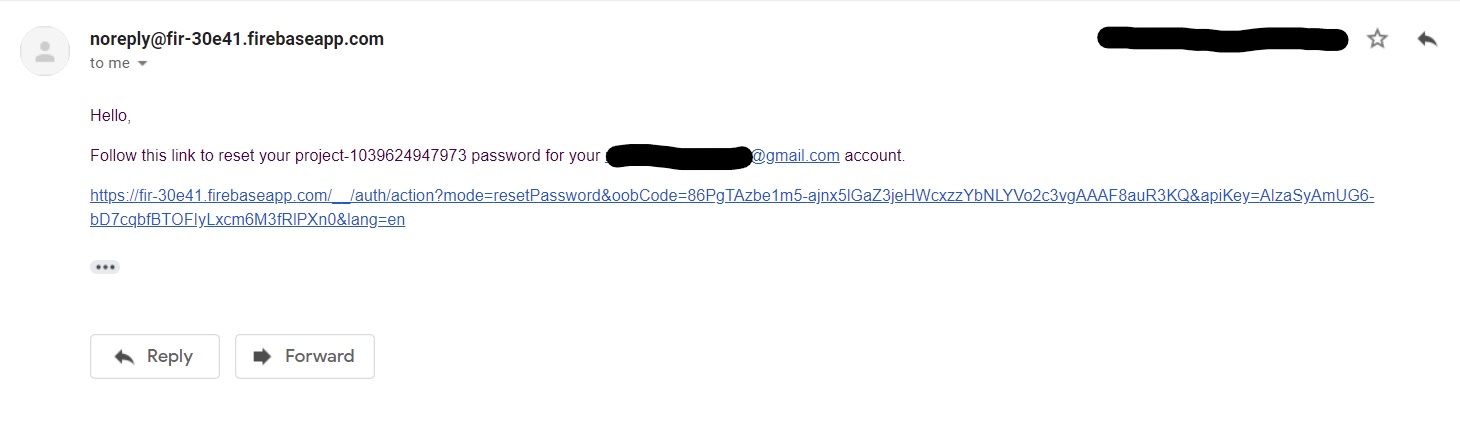
ForgotPassword Class (Source: Screenshot from Android Studio)

Graphical user interface, text, application, email

Description automatically generated

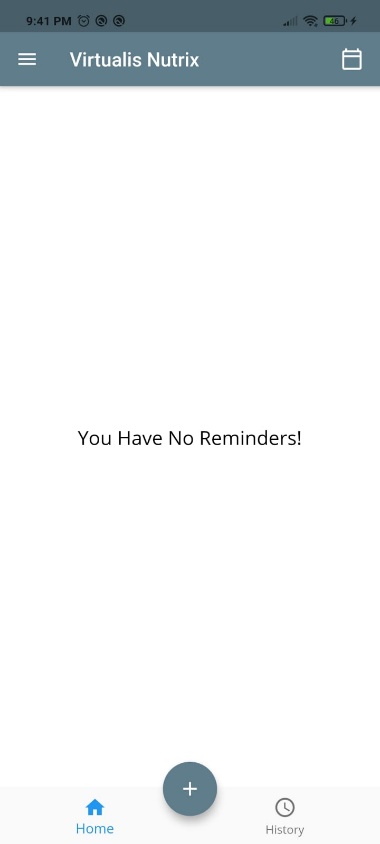
PasswordReset() function (Source: Screenshot from Android Studio)

An email is sent to the user which includes a link the user has to click on in order to reset their password. Once the e-mail is successfully sent, the user is returned to the log in screen. The user can enter their credentials after resetting the password and successfully log into the app.



Reset Password Email (Source: Screenshot from Gmail)

**Homepage**

****

Home Screen (Source: Screenshot from Android Studio)

This is the first page the user sees when they log into the app. All the medicines which have been added by the user are displayed here. The page uses QuerySnapshot to import details from the database and a Card widget to display the information in simple yet efficient manner. The “+” button on the bottom can be used to add a reminder.

**Graphical user interface, text, application

Description automatically generated**

**Query Snapshot being used to display data from the database in the app** (Source: Screenshot from Android Studio)

**Add New Reminder**

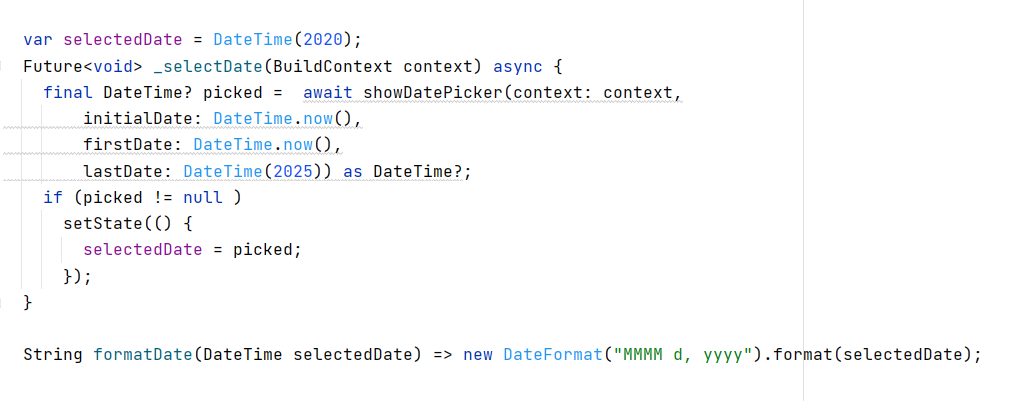
**Graphical user interface, text, application

Description automatically generated Graphical user interface, text, application

Description automatically generated**

Adding a new reminder

This page is used to add a new reminder. While importing the reminders specified date and time, the DateTime function is called.



DateTime function (Source: Screenshot from Android Studio)

Once the user inputs the information, it is added to the specified database along with the user’s information. The user is then navigated back to the home page.

**Add to Calendar**

An IconButton is used to create a calendar entry for the ride to provide notifications at the time of the ride. The DateTime.parse() function is used to change the datatype of the date. On pressing the calendar icon on the home screen, the reminders added by the user are added to the inbuilt calendar app of their phone as shown below:

Add to calendar button

The calendar app of the user’s phone opens with the name and time pre entered in the form

**Graphical user interface, text, application

Description automatically generated** A screenshot of a computer keyboard

Description automatically generated with medium confidence

Adding the reminder to the inbuilt phone reminder app

Graphical user interface, text, application

Description automatically generated

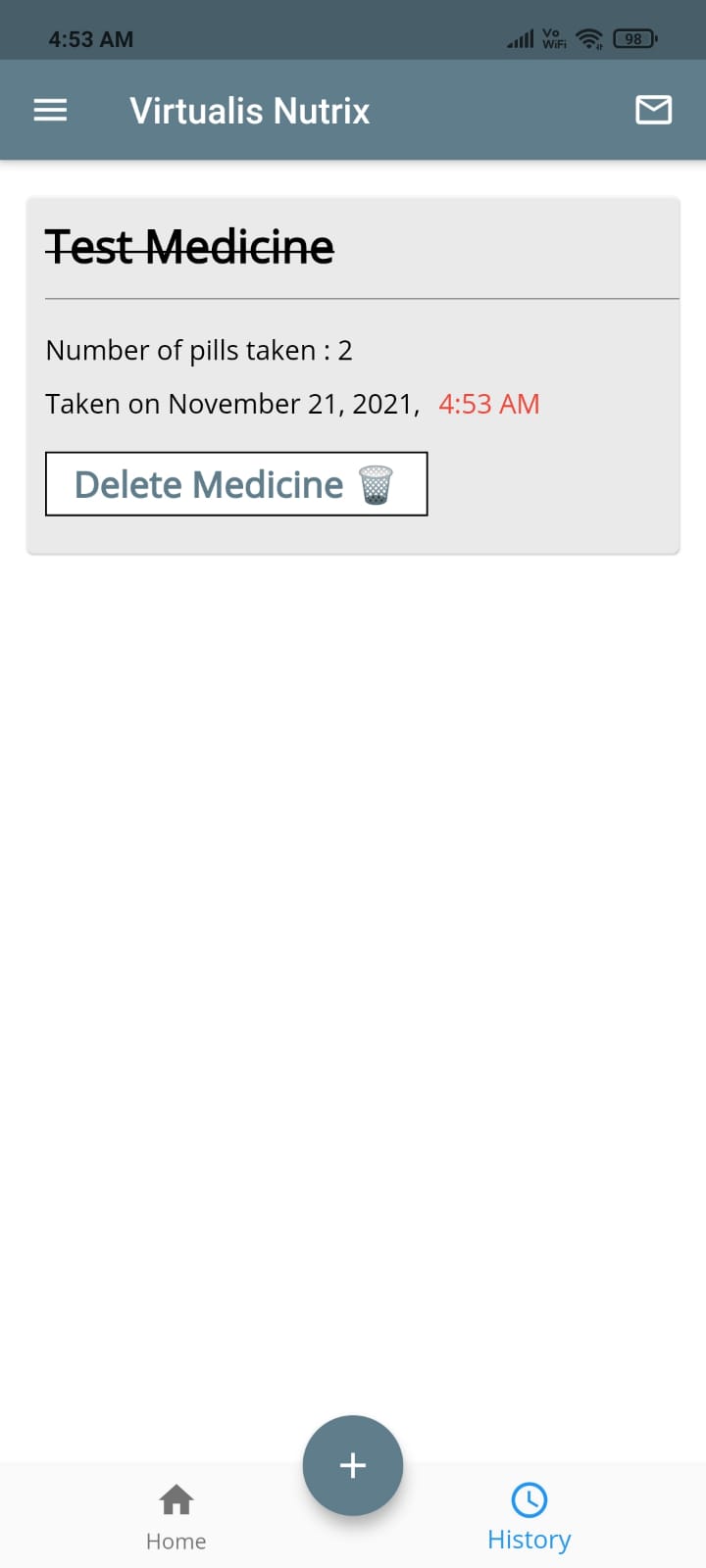
IconButton used to add the reminder to the calendar (Source: Screenshot from Android Studio)

**Medicine History Page**



History Page

Upon clicking the time icon in the bottom navigation bar, the user is sent to the history page of the app. All medicines taken by the user are displayed here. Upon clicking the ‘Medicine Taken’ option on the card of the home screen, the reminder is shifted here.



The taken medicines name comes with a strike on it

History page with Test Medicine Reminder

The medicine names here are displayed with a ‘-‘ on the name in order to not make the user confused between the home page and the history page. The time taken of the medicine is the time the user clicks the ‘Taken’ button on the home screen. This is displayed in red to avoid confusion for the user.

If there are no medicines taken and the database is empty, a message is displayed saying “You have not taken any medicines.”

Text

Description automatically generated with low confidence

Message displayed when no reminders have been taken

Text

Description automatically generated

Snippet of the history page code (Source: Screenshot from Android Studio)

If the user presses the email icon on the top right of the history page, an email is sent to the logged in user’s email address with all the data on this page. To launch the email activity, the mailer library is imported and the sendHistoryMail function is used. A snippet of the function can be seen below.

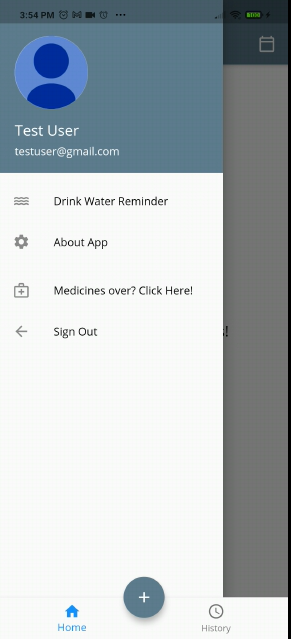
Text

Description automatically generated

Snippet of the sendHistoryMail function (Source: Screenshot from Android Studio)

**Hamburger menu**

Graphical user interface, text, application

Description automatically generated 

Menu Icon



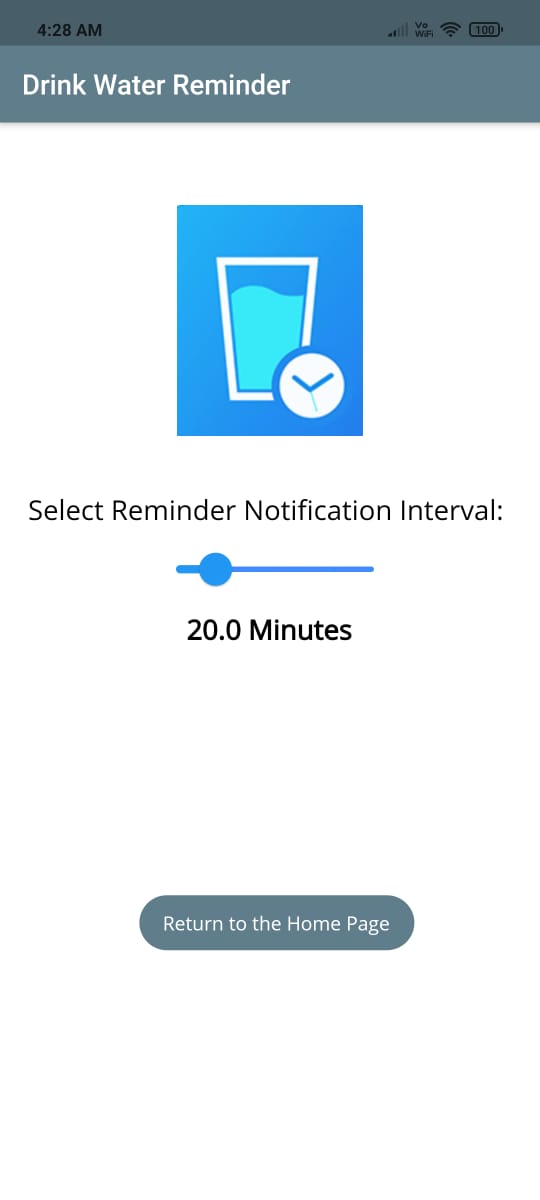
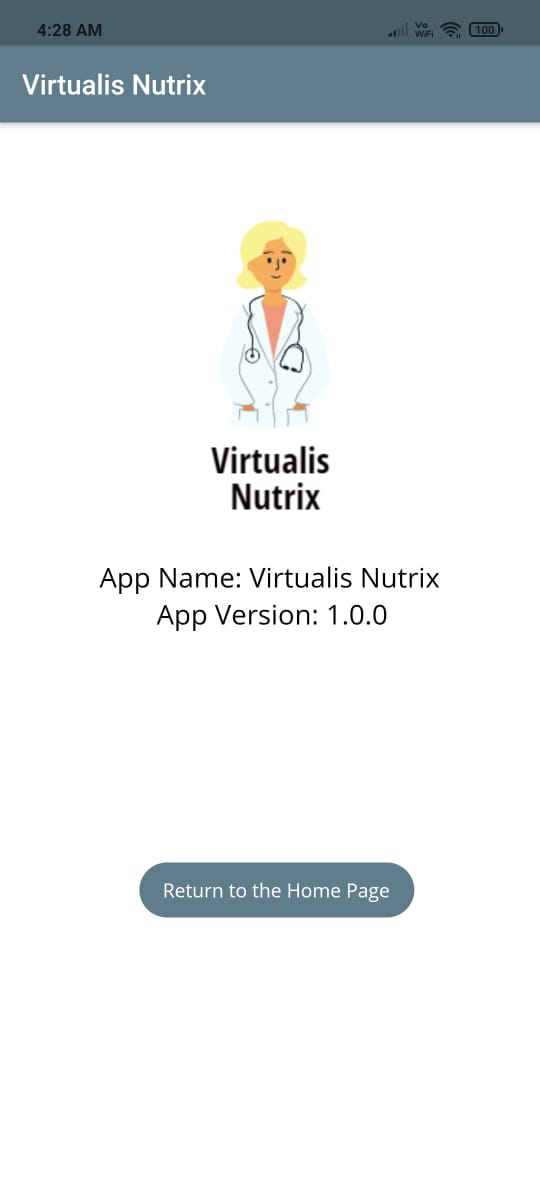
(a) Hamburger Menu to confirm the user (b) The UI once the user opens the menu

On the top left side of the homepage, there is a menu icon. On pressing it, the hamburger menu is displayed. The user can also access features like Drink Water Reminder, Finding the closest medical store, Information about the app, and sign out here.

Graphical user interface, text, application

Description automatically generated

The code to display the users name (Source: Screenshot from Android Studio)

** **

1. Drink Water Reminder Settings (b) Information about the app

The Drink Water Settings allows the user to modify the interval time between notifications with the use of a slider. Upon selecting a specific time, the user can easily navigate back to the homepage with the help of a button. The slider uses the SliderTheme class.

****

The SliderTheme class (Source: Screenshot from Android Studio)

**Google Maps Integration (Finding closest medicine store)**

In case a user wants to buy a medicine, they can access the ‘Medicines Over? Click here’ button and google maps will be opened with the location of the closest medical store to the user. This is done using the MapUtils class, which uses the launchurl function.

Graphical user interface, text, application, email

Description automatically generated

MapUtils class (Source: Screenshot from Android Studio)

**Sign Out**

On clicking sign out, the user is logged out of the application and database and gets sent to the welcome screen.

**Graphical user interface, text, application

Description automatically generated**

Sign-out function (Source: Screenshot from Android Studio)

**Meal Notifications**

**My client had specifically asked me for meal notifications. To implement these, I incorporated the cloud messaging beta feature on Firebase. The notifications for all 3 meals of the day can be seen on the database in the picture below.**

**Graphical user interface, text, application, email

Description automatically generated**

**Meal Notifications (Source: Screenshot from Firebase Database)**

**Dependencies**

A dependency is another package that your application needs in order to work[[4]](#footnote-4). Dependencies are the biggest advantage in flutter as it allows many other features to be incorporated in the app. The following dependencies have been used in this project:

Text, letter

Description automatically generated

Dependencies for the project (Source: Screenshot from Android Studio)

Word count: 1350

1. "Flutter - Build Apps For Any Screen". Flutter.Dev, 2021, https://flutter.dev/. [↑](#footnote-ref-1)
2. "MVC Framework - Introduction". Tutorialspoint.Com, 2021, https://www.tutorialspoint.com/mvc\_framework/mvc\_framework\_introduction.htm#:~:text=The%20Model%2DView%2DController%20(,the%20view%2C%20and%20the%20controller.&text=MVC%20is%20one%20of%20the,create%20scalable%20and%20extensible%20projects. [↑](#footnote-ref-2)
3. "Associative Arrays In PHP - Geeksforgeeks". Geeksforgeeks, 2018, https://www.geeksforgeeks.org/associative-arrays-in-php/. Accessed 3 Mar 2022. [↑](#footnote-ref-3)
4. "Package Dependencies". Dart.Dev, 2022, https://dart.dev/tools/pub/dependencies#:~:text=Dependencies%20are%20one%20of%20the,handles%20transitive%20dependencies%20for%20you. [↑](#footnote-ref-4)