Logo, company name

Description automatically generated

**Mobile Application Development**

**Class: BSCS 9A**

**End Semester Project Report**

**SuSha – The Shoe Shop**

**Submitted by:**

**Aisha Tariq - 289885**

**Sumaiyyah Tahir – 284482**

**Table of Contents**

[Storyline 4](#_heading=)

[Business Idea 4](#_heading=)

[Target Audience 4](#_heading=)

[Competitor Analysis 4](#_heading=)

[Prominent Features 4](#_heading=)

[Flutter Concepts Applied 5](#_heading=)

[Complex UI using combination of widgets 5](#_heading=)

[User interactivity and navigation 5](#_heading=)

[Models and views separation 7](#_heading=)

[Asynchronous Programming 8](#_heading=)

[Enhanced UX - animation, responsiveness, and login status check 8](#_heading=)

[Firebase connectivity 9](#_heading=)

[Custom Packages: 10](#_heading=)

[Additional Flutter Concepts Not Taught in Class 13](#_heading=)

[Google Maps API integration and Permission handling 13](#_heading=)

[Unit Testing and Widget Testing 13](#_heading=)

[Creating and Using Custom Packages 13](#_heading=)

[User Interface and the Workflow 15](#_heading=h.vy77swxpft4j)

**List of Figures**

[Figure 1: Views 4](#_heading=h.30j0zll)

[Figure 2: Models 4](#_heading=h.1fob9te)

[Figure 3: Asynchronous Programming 5](#_heading=h.3znysh7)

[Figure 4: Hero Animation 5](#_heading=h.2et92p0)

[Figure 5: Firebase Initialization 5](#_heading=h.tyjcwt)

[Figure 6: Firebase Project 6](#_heading=h.3dy6vkm)

[Figure 7: Authentication files 6](#_heading=h.1t3h5sf)

[Figure 8: Firebase Authentication Backend 7](#_heading=h.4d34og8)

[Figure 9: Firestore Files 7](#_heading=h.2s8eyo1)

[Figure 10: Firestore Backend 8](#_heading=h.17dp8vu)

[Figure 11: Auth Tests 8](#_heading=h.3rdcrjn)

[Figure 12: Widget Testing 9](#_heading=h.26in1rg)

# Storyline

Every year, millions of pairs of shoes are produced worldwide, but many of them end up in landfills or tucked away in closets, never to be worn again. This is where SuSha comes in. The app offers a platform for users to resell their shoes and give them a new life. The app also provides a convenient and cost-effective way for buyers to purchase gently used shoes, reducing the environmental impact of overproduction.

## Business Idea

SuSha's business idea is to create a marketplace for shoe enthusiasts in Pakistan. By allowing users to resell their shoes through the platform's owners, SuSha creates a convenient way to sell shoes and make some money. At the same time, buyers can find affordable shoes that are still in good condition. The platform charges a small fee for every sale made through the app, generating revenue for the business.

## Target Audience

1. Shoe enthusiasts in Pakistan who are looking for affordable, gently used shoes.
2. Young adults and students who want to stay on-trend on a budget.
3. Environmentally conscious consumers who want to reduce waste.

## Competitor Analysis

We have identified several competitors in the market that offer similar products. One such example is a website called “Thrifty Pre-loved” that deals in the selling of used clothes, accessories, and shoes. The UI for this website is very cluttered.

However, our product offers benefits that set it apart from the competition. SuSha sells a unique collection of shoes, affordable prices, seamless user experience and appealing user interface.

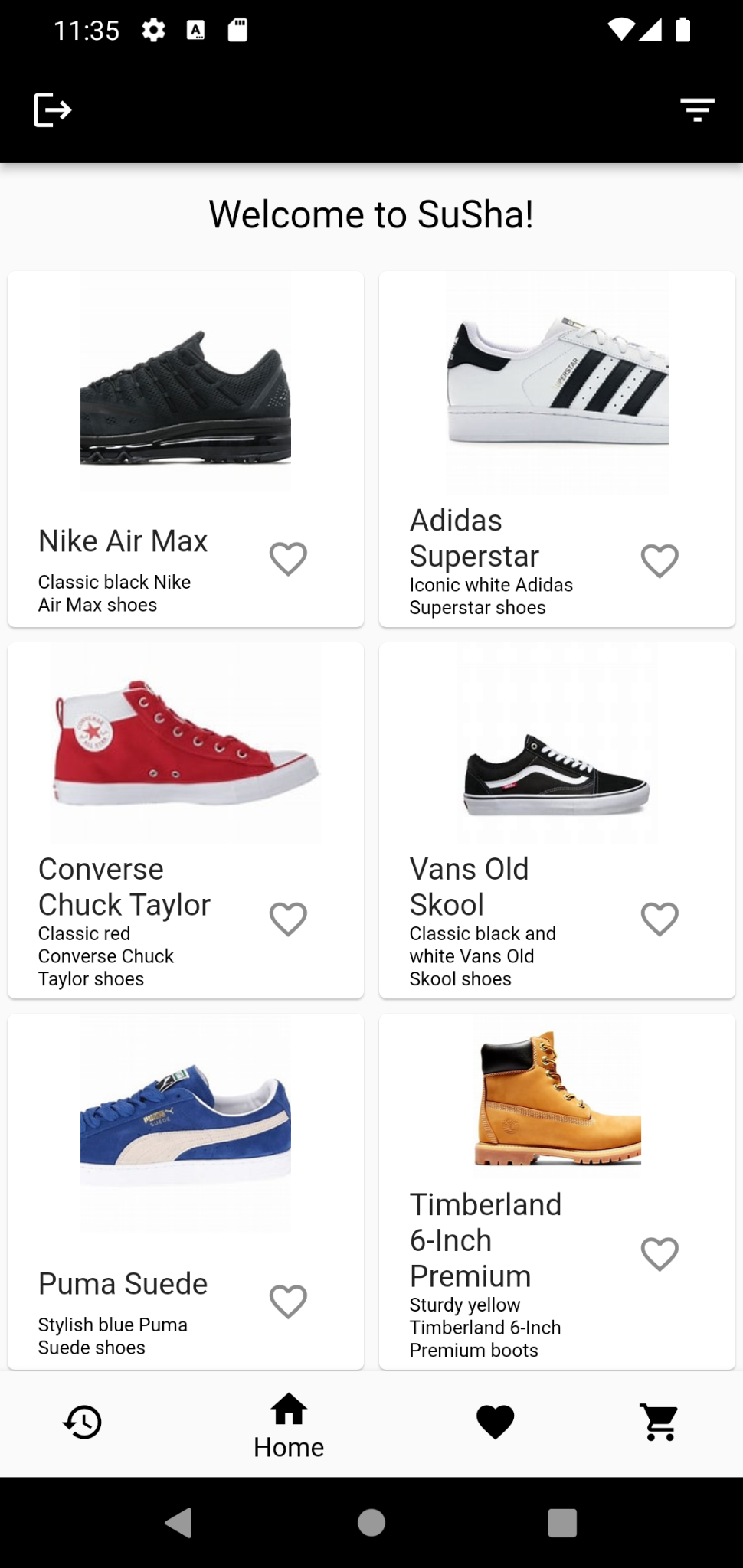
# Prominent Features

1. Minimal, efficient, and user-friendly Interface
2. Browsing through available shoes and placing orders
3. Maps integration for location selection
4. Email verification on sign up
5. Filtering through the available shoes based on color and price
6. Adding shoes to favorites and adding shoes to cart
7. Viewing past orders

# Flutter Concepts Applied

## Complex UI using combination of widgets

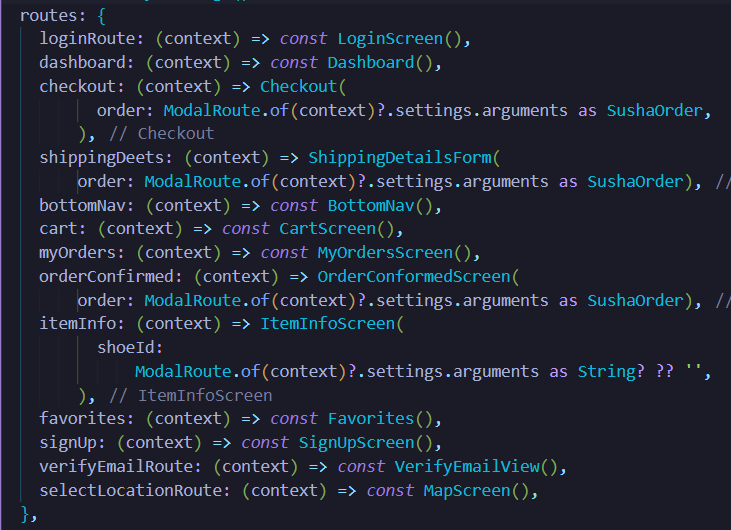
Our screens are made up of a combination of different widgets. The dashboard screen that can be seen in the figure below consists of a GridView widget that consists of a List Widget that returns Cards.The Each Card in turn consists of an Image and ListTile widget.

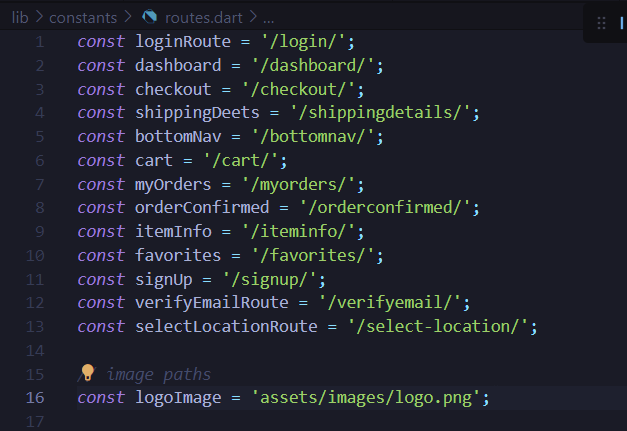


The remaining screens are also laid out similarly with a different combination of widgets.

## User interactivity and navigation

Defined all routes clearly in main and used a separate constants file to improve code organization and readability, and ensure there are no inconsistencies. Also included a bottom navigation bar so the user could easily switch between the different screens.





## Models and views separation

Clear separation of logic and code into separate view and model sections.

**Our Views:**

A screenshot of a computer

Description automatically generated with medium confidence

*Figure 1: Views*

**Our Models in Firestore Package:**

A screenshot of a computer

Description automatically generated with low confidence

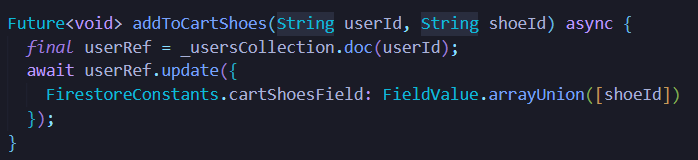
*Figure 2: Models*

## Asynchronous Programming

We used asynchronous programming to access data in the Firestore collections like user, shoes or orders as well as write to these collections.

The UserFirestoreServices, ShoeFirestoreServices and OrderFirestoreServices all use asynchronous methods to provide different services.

For example, the UserFirestoreServices provides an addToCart service so when the user adds an item to their cart, the cartShoes document for the specific user document gets updated to contain the id of the shoes the user added.



*Figure 3: Asynchronous Programming*

## Enhanced UX - animation, responsiveness, and login status check

**Animation**

We included Hero animation between our dashboard and item information screens. The hero (an Image is wrapped with the Hero widget) flies to from the dashboard to the item information screen when the image is tapped. The following code snippet exists in both the screens.

A picture containing text, font, screenshot, line

Description automatically generated

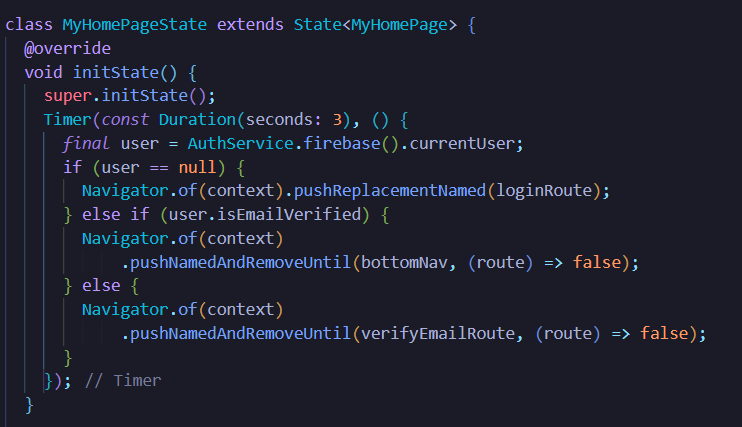
*Figure 4: Hero Animation*

**Responsiveness**

We’ve made the app responsive by making use of Expanded widget and the GridView widget.

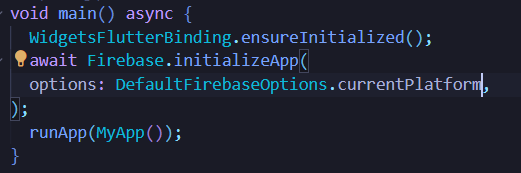
**Check login status**

When the app starts up, it checks if the user is already logged in, or registered but not verified, or not logged in at all and takes the user to the relevant view (Dashboard/ Email Verification/ Login) based on the login and verification status.

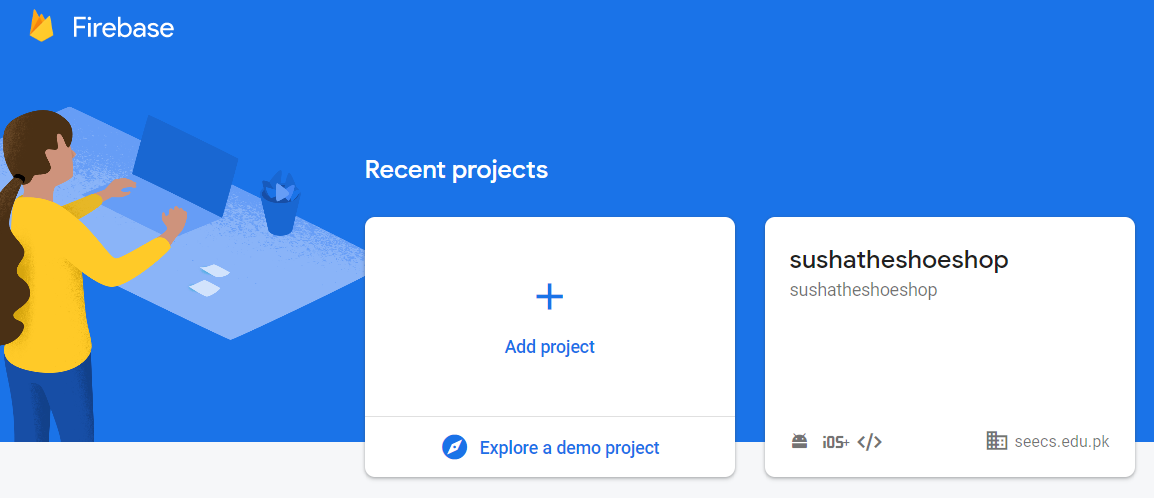


## Firebase connectivity

Firebase is initialized before running the app. All the required Firebase dependencies were included in the pubspec.yaml file as well. The app’s authorization and database is fully functional and connected to firebase.



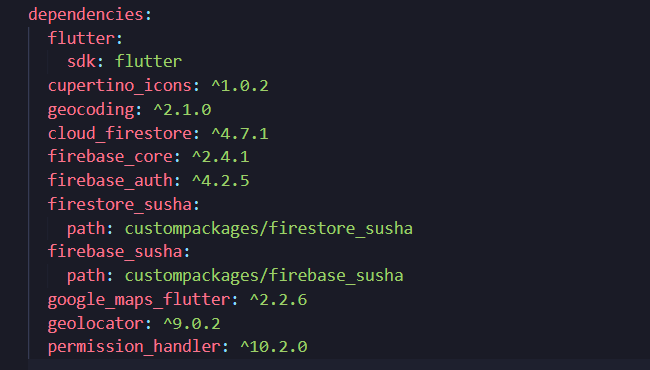
*Figure 5: Firebase Initialization*



*Figure 6: Firebase Project*

## Custom Packages:

We created custom packages using dart package template and included these packages within the root folder of our project. We then defined them in the susha project’s pubspec.yaml file as dependencies so that we could easily access them throughout the code.

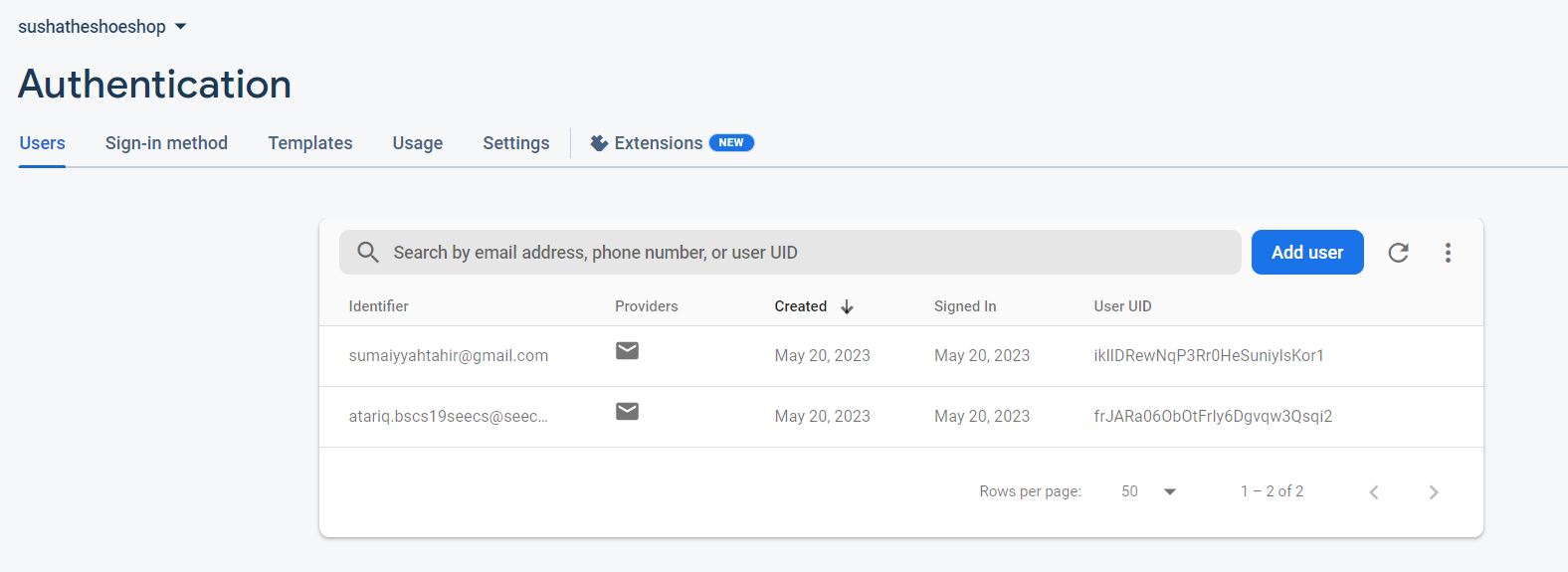


**Firebase\_susha Package:**

Contains functionality for connecting the app to firebase and performing all necessary authorization functions.



*Figure 7: Authentication files*

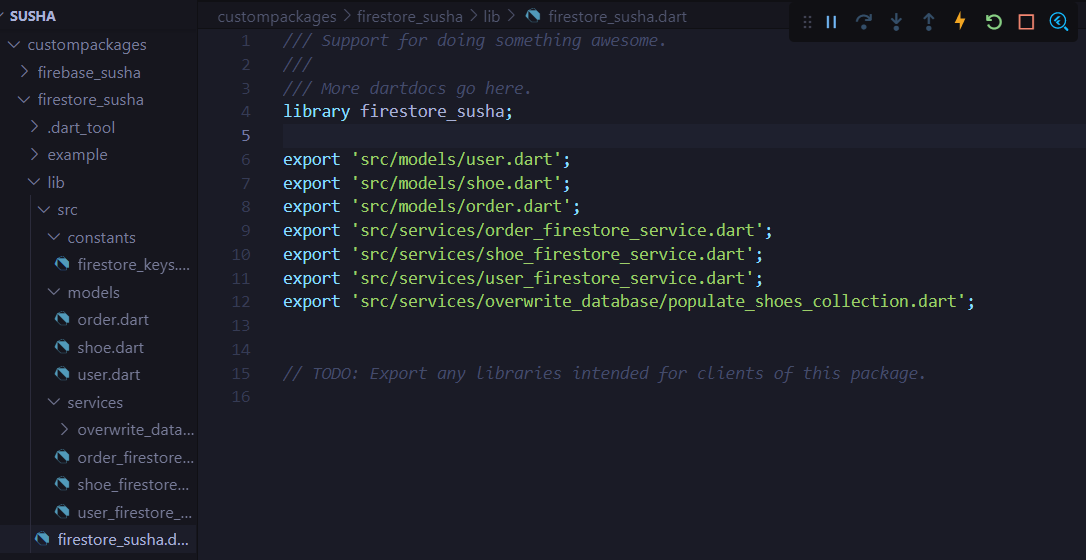


*Figure 8: Firebase Authentication Backend*

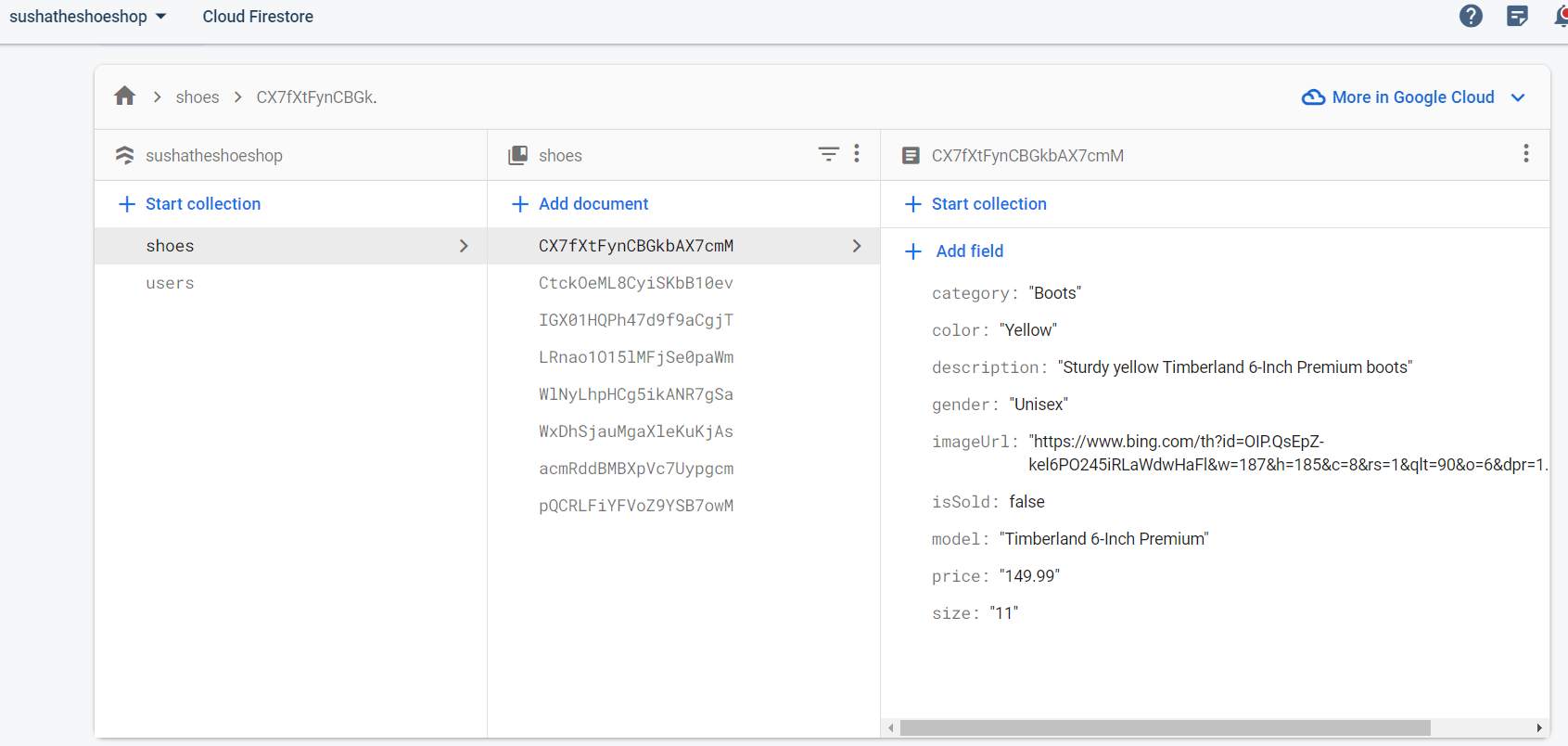
**Firestore\_susha Package:**

Contains functionality for storing, retrieving and updating all 3 kinds of our app’s models to the firestore database. Our models are Shoe, SushaOrder, and User.

Also contains a service for **populating the shoe collection in the database using a local json file** that contains the data of shoe items to be added to the store.

**

*Figure 9: Firestore Files*



*Figure 10: Firestore Backend*

# Additional Flutter Concepts Not Taught in Class

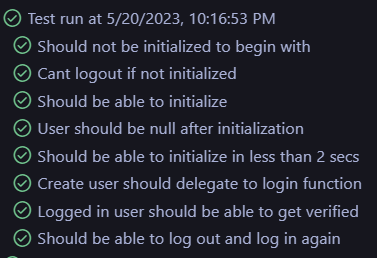
## Google Maps API integration and Permission handling

Integrated google maps API, created a separate view that opens a map and lets user selects a location, and added functionality to automatically extract and populate the user’s address fields in the shipping details form based on the user’s selected coordinates.

Also included user permissions for device location so that before accessing user’s location for the map, the user prompts the user to provide access permissions.

## Unit Testing and Widget Testing

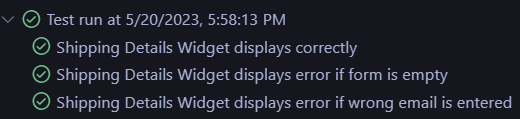
**Auth Tests**



*Figure 11: Auth Tests*

**Widget Tests**

Using the flutter\_test package we tested the ShippingDetailsForm widget to see if it is displayed correctly or if it gave errors as it should be when no input is given or the wrong email is given and as can be seen the tests passed.



*Figure 12: Widget Testing*

## Creating and Using Custom Packages

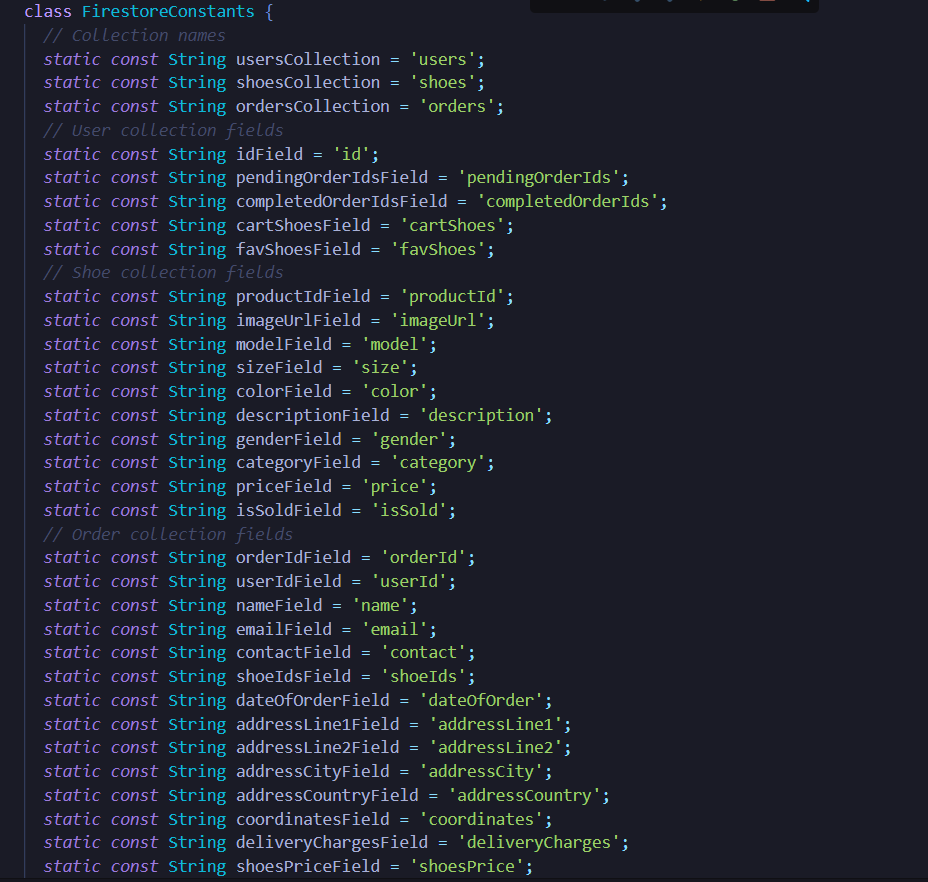
As mentioned before we created custom packages for Firebase Services.

The first package we created was for Firebase Authentication.

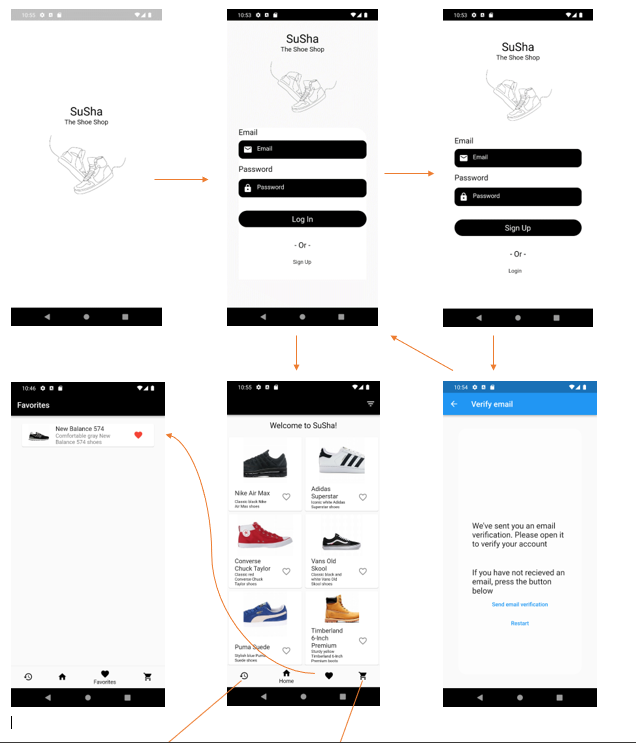
The second package was for Firebase Cloud Firestore.

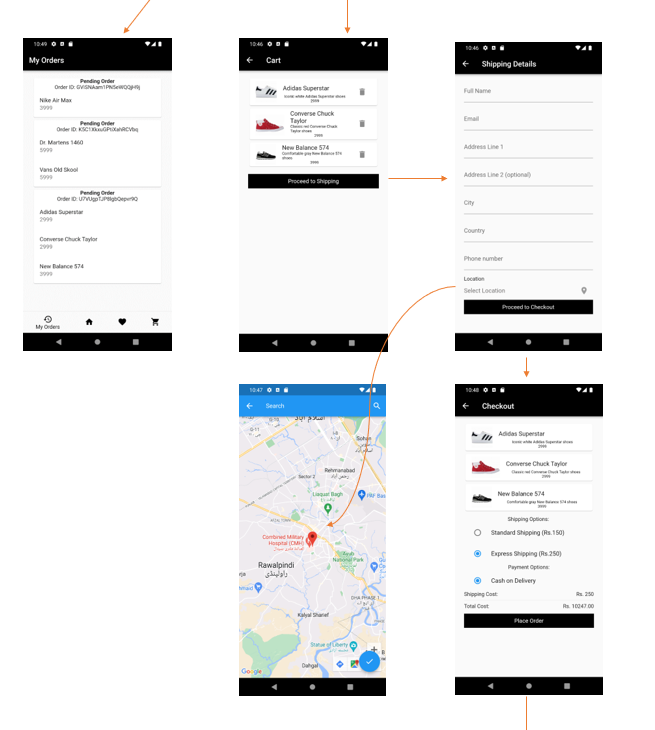
**Clear Organization of Code and Using constants to prevent mistyped keys and strings**

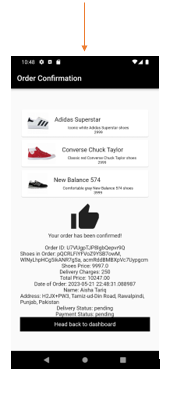
Organized project components efficiently and clearly into separate directories such as assets, views, services, constants etc. Also created separate constant files for routes strings aswell as firestore keys to ensure that the same values were used throughout the codebase.



## User Interface and the Workflow







**To apply filters**

