

PROFESSIONAL TRAINING REPORT
at
Sathyabama Institute of Science and
Technology
(Deemed to be University)

Submitted in partial fulfillment of the requirements for the
award of Bachelor of Technology Degree in Information
Technology

By

VISHWA.L
REG. NO. 40120108



DEPARTMENT OF INFORMATION TECHNOLOGY

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND
TECHNOLOGY

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

APRIL 2023



SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
(Established under Section 3 of UGC Act, 1956)
JEPPIAAR NAGAR, RAJIV GANDHI SALAI
CHENNAI- 600119
www.sathyabama.ac.in

SCHOOL OF COMPUTING

BONA FIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of
VISHWA.L (Reg. No: 40120108) who carried out the project
entitled "**STOCK MARKET(XVII) APPLICATION USING FLUTTER**"
under my supervision from January 2023 to May 2023.

Internal Guide

Dr. R. Subhashini, M.E., PhD.

Head of the Department

Dr. R. Subhashini, M.E., Ph.D.,

Submitted for Viva voce Examination held on 29/04/23

Internal Examiner

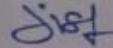
External Examiner

DECLARATION

I, VISHWA.L hereby declare that the project report entitled **STOCK MARKET(XVII) APPLICATION USING FLUTTER** done by me under the guidance of **Dr.Subhashini** is submitted in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Information Technology.

DATE: 29/04/23

PLACE: CHENNAI



SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management of SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D**, Dean, School of Computing and **Dr. R. Subhashini M.E., Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.R.Subhashini, M.E., Ph.D.**, for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

ABSTRACT

This response provides an overview of what a stock market group buy and sell app developed using Flutter might entail. It describes several key components of the app, including authentication, a dashboard, group buy and sell feature, notifications, portfolio management, and payment gateway integration. The response emphasizes the importance of designing the app with an intuitive and user-friendly interface to enhance the user experience. Overall, the response serves as an abstract of the main points covered in a report for a stock market group buy and sell app developed using Flutter.

TABLE OF CONTENTS

CHAPTER No.	TITLE	PAGE No
	ABSTRACT	I-II
	LIST OF ABBREVIATIONS	iv
	LIST OF FIGURES	v
1	INTRODUCTION	1
	1.1 Features of flutter	2
	1.1.1 Advantages of flutter	2
	1.1.2 Disadvantage of flutter	3
	1.2 Dart programming	3
	1.2.1 Advantages of dart	3
	1.2.2 Disadvantages of dart	4
	1.3 Introduction to widgets	5
	1.3.1 Gestures	6
	1.3.2 Widget bulid visualization	6
	1.4 Installation in windows	9
	1.5 Installation in macos	13
2	AIM AND SCOPE OF THE PRESENT INVESTIGATION	15
	2.1 AIM	15
	2.2 SCOPE	15

3	METHODOLOGY AND ALGORITHMS USED	16
3.1	System requirements	16
3.1.1	windows	16
3.1.2	Mac	16
3.1.3	Linux	16
3.1.4	Chrome os	17
3.2	METHODS AND ALGORITHMS USED	17
3.3	Flow chart	18
3.4	Components and widgets used	19
3.4.1	Components	20
3.4.2	Widgets	21
3.4.2.1	In main.dart	22
3.4.2.2	In home_page.dart	22
4	RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS	23
4.1	Results	23
4.1.1	Splash Screen intro section	23
4.1.2	Login screen	24
4.1.3	Signup Screen	25
4.1.4	After login it will return to home page	26

4.1.5	2 routes in home page	
4.1.6	Home	27
4.1.7	Community	28
4.2	Selecting of a Stocks	29
4.3	Check the Stock prices & Graph	
4.4	Buying Stocks	
4.5	Select Payment Gateway	
4.6	Thankyou Page	
5	SUMMARY AND CONCLUSION	30
5.1	Summary	30
5.2	Conclusion	30
	APPENDIX	35
	REFERENCES	55

LIST OF ABBREVIATIONS

ABBREVIATION	EXPANSION
HTML	HYPERTEXT MARKUP LANGUAGE
UI	USER INTERFACE
OS	OPERATING SYSTEM
IDE	INTEGRATED DEVELOPMENT ENVIRONMENT
AOT	AHEAD OF TIME
JIT	JUST IN TIME
RAM	RANDOM ACCESS MEMORY

CHAPTER – 1

INTRODUCTION

In general, developing a mobile application is a complex and challenging task. There are many frameworks available to develop a mobile application. Android provides a native framework based on Java language and iOS provides a native framework based on Objective-C / Swift language.

However, to develop an application supporting both the OSs, we need to code in two different languages using two different frameworks. To help overcome this complexity, there exists mobile frameworks supporting both OS. These frameworks range from simple HTML based hybrid mobile application framework (which uses HTML for User Interface and JavaScript for application logic) to complex language specific framework (which do the heavy lifting of converting code to native code). Irrespective of their simplicity or complexity, these frameworks always have many disadvantages, one of the main drawback being their slow performance.

In this scenario, Flutter - a simple and high performance framework based on Dart language, provides high performance by rendering the UI directly in the operating system's canvas rather than through native framework.

Flutter also offers many ready to use widgets (UI) to create a modern application. These widgets are optimized for mobile environment and designing the application using widgets is as simple as designing HTML.

To be specific, Flutter application is itself a widget. Flutter widgets also supports animations and gestures. The application logic is based on reactive programming. Widget may optionally have a state. By changing the state of the widget, Flutter will automatically (reactive programming) compare the widget's state (old and new) and render the widget with only the necessary changes instead of re-rendering the whole widget.

1. FEATURES OF FLUTTER

Flutter framework offers the following features to developers –

- Modern and reactive framework.
- Uses Dart programming language and it is very easy to learn.
- Fast development.
- Beautiful and fluid user interfaces.
- Huge widget catalog.
- Runs same UI for multiple platforms.
- High performance application

1. Advantages of flutter

Flutter comes with beautiful and customizable widgets for high performance and outstanding mobile application. It fulfils all the custom needs and requirements. Besides these, Flutter offers many more advantages as mentioned below –

- Dart has a large repository of software packages which lets you to extend the capabilities of your application.
- Developers need to write just a single code base for both applications (both Android and iOS platforms). Flutter may to be extended to other platform as well in the future.
- Flutter needs lesser testing. Because of its single code base, it is sufficient if we write automated tests once for both the platforms.
- Flutter's simplicity makes it a good candidate for fast development. Its customization capability and extendibility makes it even more powerful.
- With Flutter, developers has full control over the widgets and its layout.
- Flutter offers great developer tools, with amazing hot reload.

2. Disadvantages of flutter

Despite its many advantages, flutter has the following drawbacks in it –

- Since it is coded in Dart language, a developer needs to learn new language (though it is easy to learn).
- Modern framework tries to separate logic and UI as much as possible but, in Flutter, user interface and logic is intermixed. We can overcome this using smart coding and using high level module to separate user interface and logic.
- Flutter is yet another framework to create mobile application. Developers are having a hard time in choosing the right development tools in hugely populated segment.

2. DART PROGRAMMING

Dart is an open-source general-purpose programming language. It is originally developed by Google. Dart is an object-oriented language with C-style syntax. It supports programming concepts like interfaces, classes, unlike other programming languages Dart doesn't support arrays. Dart collections can be used to replicate data structures such as arrays, generics, and optional typing.

Dart language supports the following data types –

- Numbers – It is used to represent numeric literals - Integer and Double.
- Strings – It represents a sequence of characters. String values are specified in either single or double quotes.
- Booleans – Dart uses the bool keyword to represent Boolean values - true and false.
- Lists and Maps – It is used to represent a collection of objects.

1. Advantages of dart

- The first advantage is that it is easy to learn. Any JavaScript programmer can quickly relearn how to write code in Dart. To do this, they only need to familiarize themselves with the basic principles of this programming language.
- The second thing is about its availability of documentation. Since Google is developing the interpreter for Dart, all the features of the language are described in detail. This allows you to quickly get answers to almost any questions that may arise during the training process, or directly while writing code.
- The third advantage is its high performance. Programs written in Dart tend to run faster than programs created in JavaScript.
- Dart is very stable and it can be used to build production quality real-time applications. It is an object-oriented programming language with support for inheritance, interfaces and optional typing features.
- It uses AOT and JIT compilation - Dart has the unique capability to handle both Ahead of time and Just in time compiling. In AOT, the Dart code can be directly converted into native machine code. While in the mode of JIT, it can be compiled for exceptionally fast development cycles and game-changing workflow.
- If you want to start writing your first Dart program without any installation or configuration, there is DartPad for you. The DartPad interface is plain and simple. Write your code and click 'Run' to execute the code. DartPad introduces users to the world of Dart programming. The support for libraries in DartPad is restricted to the basic level.

2. Disadvantages of dart

- The first drawback is that Dart has very limited resources online and it's hard to find solutions to problems. The reason for it is lack of a larger and cohesive developer communities, which will support you.
- Dart is fairly new to the programmers and rarely used in the market. So if you will look for a job on this position, it will be harder to find a company with a vacancy of Dart programmer, because it hasn't yet spread widely.
- Dart is currently in development. That's exciting, and it's not something "wrong" with Dart, but if you start developing in Dart now, there's a certain chance that the API will change, or that things won't be documented fully or correctly, and the amount of information on the web is less than, for example, the amount you can find about jQuery.
- The absence of native support. For a beautiful demonstration of dart's capabilities, developers must bring it to users' browsers. It turns out that to achieve at least 80% success, Dart developers must provide native support for it in all popular browsers.
- To sum up, Dart is an interesting programming language with features to facilitate Web, mobile and command line apps. Its major advantages are its stability and ease of learning. In the case of Web applications, the combination with AngularDart makes it a very powerful tool. Generally, we think that despite its novelty, this programming language is worth your attention.

1.3 INTRODUCTION TO WIDGETS

The core concept of the Flutter framework is In Flutter, Everything is a widget. Widgets are basically user interface components used to create the user interface of the application.

In Flutter, the application is itself a widget. The application is the top-level widget and its UI is built using one or more children (widgets), which again build using its children widgets. This composability feature helps us to create a user interface of any complexity

Here the following points are worth noting –

- MyApp is the user-created widget and it is built using the Flutter native widget, MaterialApp.
- MaterialApp has a home property to specify the user interface of the home page, which is again a user-created widget, MyHomePage.
- MyHomePage is built using another Flutter native widget, Scaffold
- Scaffold has two properties - body and appBar
- body is used to specify its main user interface and appBar is used to specify its header user interface
- Header UI is built using Flutter native widget, AppBar and Body UI is built using Center widget.
- The Center widget has a property, Child, which refers to the actual content and it is built using Text widget.

1. Gestures

Flutter widgets support interaction through a special widget, GestureDetector. GestureDetector is an invisible widget having the ability to capture user interactions such as tapping, dragging, etc., of its child widget. Many native widgets of Flutter support interaction through the use of GestureDetector. We can also incorporate interactive features into the existing widget by composing it with the GestureDetector widget. We will learn the gestures separately in the upcoming chapters.

2. Widget build visualization

In Flutter, widgets can be grouped into multiple categories based on their features, as listed below –

- Platform specific widgets
- Layout widgets
- State maintenance widgets
- Platform independent / basic widgets

Some of the most used material widgets are as follows –

- Scaffold
- AppBar
- BottomNavigationBar
- TabBar
- TabBarView
- ListTile
- RaisedButton(Text button)
- FloatingActionButton(Elevate button)
- FlatButton(Text button)
- IconButton
- DropdownButton
- PopupMenuButton
- ButtonBar
- TextField
- Checkbox
- Radio
- Switch
- Slider
- Date & Time Pickers

- SimpleDialog
- AlertDialog

Some of the most used Cupertino widgets are as follows

- CupertinoButton
- CupertinoPicker
- CupertinoDatePicker
- CupertinoTimerPicker
- CupertinoNavigationBar
- CupertinoTabBar
- CupertinoTabScaffold
- CupertinoTabView
- CupertinoTextField
- CupertinoDialog
- CupertinoDialogAction
- CupertinoFullscreenDialogTransition
- CupertinoPageScaffold
- CupertinoPageTransition
- CupertinoActionSheet
- CupertinoActivityIndicator
- CupertinoAlertDialog
- CupertinoPopupSurface
- CupertinoSlider

The most important properties of the Text widget are as follows

- maxLines, int – Maximum number of lines to show
- overflow, TextOverFlow – Specify how visual overflow is handled using TextOverFlow class

- style, TextStyle – Specify the style of the string using TextStyle class
- textAlign, TextAlign – Alignment of the text like right, left, justify, etc., using TextAlign class
- textDirection, TextDirection – Direction of text to flow, either left-to-right or right- to-left

4. INSTALLATION IN WINDOWS

In this section, let us see how to install Flutter SDK and its requirement in a windows system.

Step 1 - Go to URL, <https://flutter.dev/docs/get-started/install/windows> and download the latest Flutter SDK.

Step 2 - Unzip the zip archive in a folder, like C:\flutter\

Step 3 - Go to URL, <https://developer.android.com/studio#downloads> and download the latest version.

Step 4 - Make sure your PC have jdk.

Step 5 - Update the system path to include flutter bin directory

- From the Start search bar, enter 'env' and select Edit environment variables for your account.
- Under User variables check if there is an entry called Path:

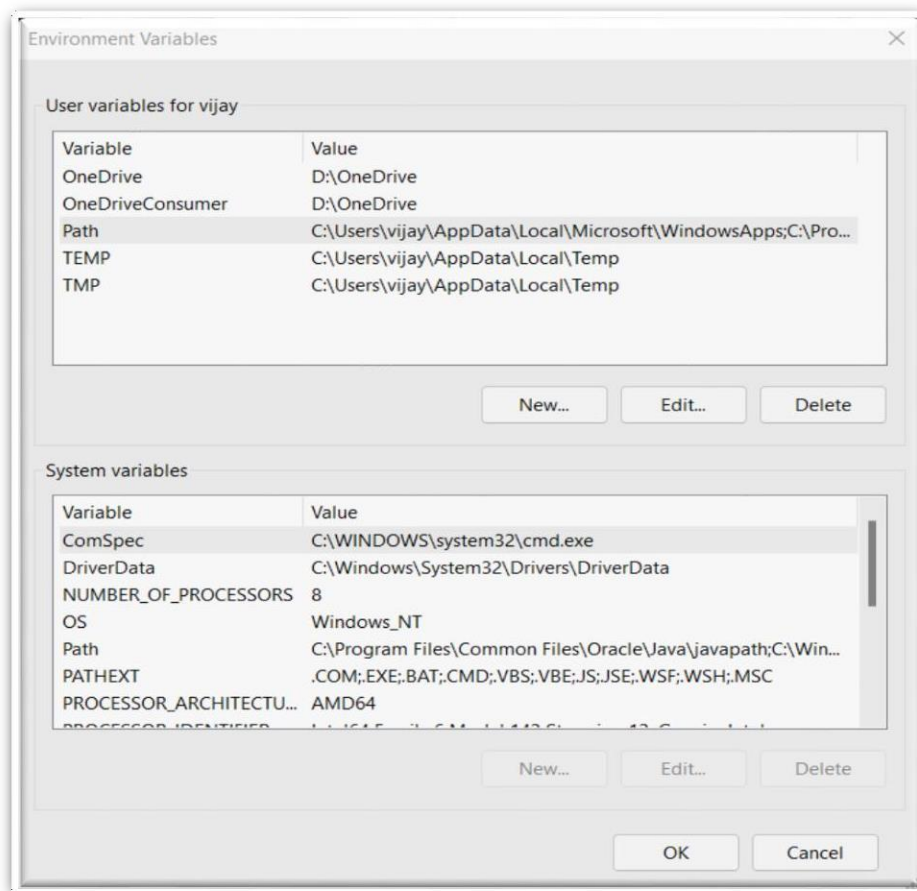


Fig 1.1 Environment variable

Then edit the path:

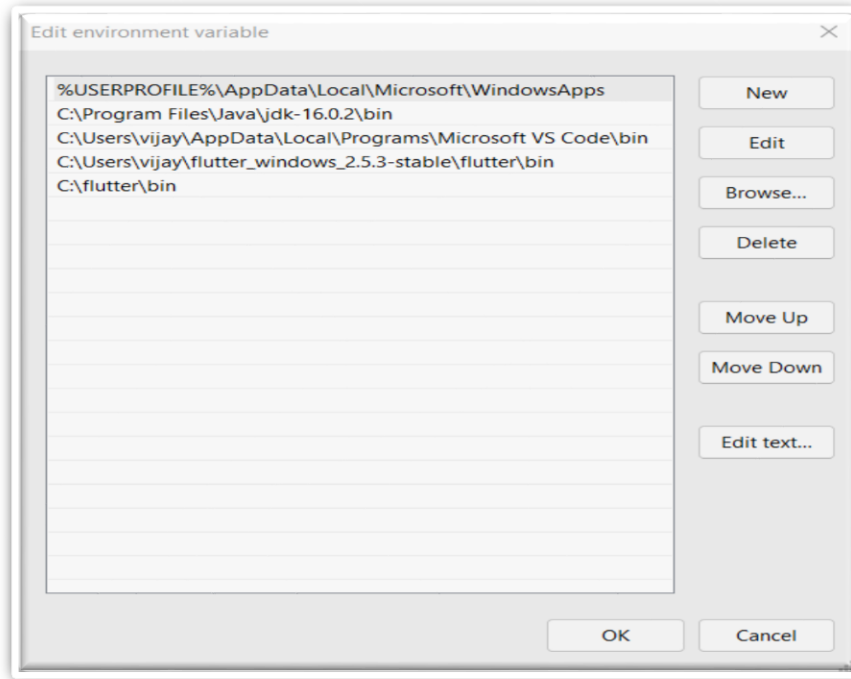
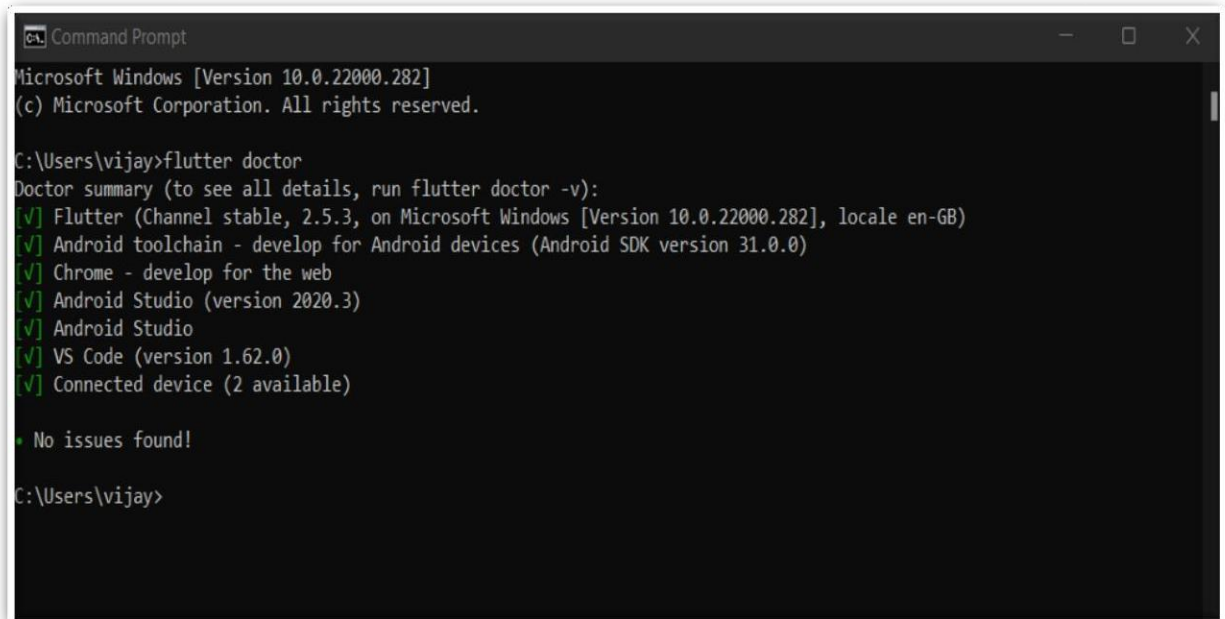


Fig 1.2 Edit environment variable

Step 4 - Flutter provides a tool, “flutter doctor” to check that all the requirement of flutter development is met, running the above command will analyze the system and show its report as shown below



```
Command Prompt
Microsoft Windows [Version 10.0.22000.282]
(c) Microsoft Corporation. All rights reserved.

C:\Users\vijay>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 2.5.3, on Microsoft Windows [Version 10.0.22000.282], locale en-GB)
[✓] Android toolchain - develop for Android devices (Android SDK version 31.0.0)
[✓] Chrome - develop for the web
[✓] Android Studio (version 2020.3)
[✓] Android Studio
[✓] VS Code (version 1.62.0)
[✓] Connected device (2 available)

• No issues found!

C:\Users\vijay>
```

Fig 1.3 Command prompt

Step 5 - Install the latest Android SDK, if reported by flutter doctor

Step 6 - Install the latest Android Studio, if reported by flutter doctor

Step 7 - Start an android emulator or connect a real android device to the system.

Step 8 - Install Flutter and Dart plugin for Visual Studio or Android Studio. It provides startup template to create new Flutter application, an option to run and debug Flutter application, etc.,

- Open Visual Studio / Android Studio.
- Click File → Settings → Plugins.
- Select the Flutter plugin and click Install.
- Click Yes when prompted to install the Dart plugin.
- Restart Visual Studio

1.5 INSTALLATION IN MACOS

To install Flutter on MacOS, you will have to follow the following steps –

Step 1 – Go to URL, <https://flutter.dev/docs/get-started/install/macos> and download latest Flutter SDK. As of April 2019, the version is 1.2.1 and the file is flutter_macos_v1.2.1-stable.zip.

Step 2 – Unzip the zip archive in a folder, say /path/to/flutter

Step 3 – Update the system path to include flutter bin directory (in ~/.bashrc file).

```
> export PATH = "$PATH:/path/to/flutter/bin"
```

Fig 1.4 Export path

Step 4 – Enable the updated path in the current session using below command and then verify it as well.

```
source ~/.bashrc  
source $HOME/.bash_profile  
echo $PATH
```

Fig 1.5 Source

Flutter provides a tool, flutter doctor to check that all the requirement of flutter development is met. It is similar to the Windows counterpart.

Step 5 – Install latest XCode, if reported by flutter doctor

Step 6 – Install latest Android SDK, if reported by flutter doctor

Step 7 – Install latest Android Studio, if reported by flutter doctor

Step 8 – Start an android emulator or connect a real android device to the system to develop android application.

Step 9 – Open iOS simulator or connect a real iPhone device to the system to develop iOS application.

Step 10 – Install Flutter and Dart plugin for Android Studio. It provides the startup template to create a new Flutter application, option to run and debug Flutter application in the Android studio itself, etc.,

- Open Android Studio(VS code)
- Click Preferences → Plugins
- Select the Flutter plugin and click Install
- Click Yes when prompted to install the Dart plugin.
- Restart Android studio.(VS code)
- Android studio

CHAPTER – 2

AIM AND SCOPE OF STOCK MARKET

1. AIM

To build a XVII application using Flutter in Visual Studio IDE..In The aim of a stock market group buy and sell app is to provide a platform for individuals to invest in stocks as a group. The app would facilitate the process of group decision-making for buying and selling stocks, providing users with a collaborative approach to investing.

2. SCOPE

The scope of a stock market group buy and sell app developed using Flutter would include the following features and functionalities:

- 1.User authentication: The app would allow users to create accounts and log in using email/password or social media credentials.
- 2.Dashboard: The app would provide users with a dashboard that displays current stock prices, company news, and charts displaying past stock performance.
- 3.Group buy and sell feature: Users would be able to join existing groups or create their own, with each group having a set amount of money that they could use to buy stocks. Members of the group would be able to vote on which stocks to purchase and the app would execute trades automatically.
- 4.Notifications: The app would send notifications to users when a group has made a trade or when a stock reaches a certain price point.
- 5.Payment gateway integration: The app would provide a secure way for users to transfer funds into the group and receive payments when the group sells the stock.

CHAPTER – 3

METHODOLOGY AND ALGORITHMS USED

1. SYSTEM REQUIREMENTS

The cool thing about using Flutter for creating cross-platform native mobile apps is the fact that you can build those on almost any OS.

Here are some System Requirements for Android Studio which is needed for running an Android simulator.

1. Windows

- Microsoft® Windows® 7/8/10 64-bit
- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

2. Mac

- Mac® OS X® 10.10 Yosemite or higher, up to 10.14 macOS Mojave
- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

3. Linux

- GNOME or KDE desktop - Tested on gLinux based on Debian.
- 64-bit distribution capable of running 32-bit applications
- GNU C Library glibc 2.19 or later

- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended 500 MB for IDE + 1.5 GB for Android SDK and emulator system image
- 1280 x 800 minimum screen resolution

4. Chrome OS

- 8 GB RAM or more recommended
- 4 GB of available disk space minimum
- 1280 x 800 minimum screen resolution
- Intel i5 or higher U series or higher recommended

3.2 METHODOLOGY AND ALGORITHMS USED

Using Flutter a free and open-source mobile UI framework created by google a Fes application project is build using a Dart language.

1. Starting up

- Open and setup Visual Studio Code or Android Studio Code according to your convince and install the Flutter and Dart plugins/extensions in order to create a new flutter project
- A default basic flutter application will created and the necessary files and package are included by VS code its as default.

2. Including required dependencies at pubspec.yml

- Add `animated_splash_screen: ^1.1.0`
For create and work with SplashScreen page.
- Add `flutter_slidable: ^0.5.7`
A Flutter implementation of slidable list item with directionl slide actions

that can be dismissed.

- Add assets and paste the relative path of the images folder which will be having the images necessary for our project.

- flutter_launcher_name: “^0.0.1”

A command-line tool which simplifies the task of updating your flutter app's launcher name. If you use this package, you don't need modify AndroidManifest.xml (for android) or info.plist(ios), by adding,

Flutter_launcher_name:

Name: “XVII”

After setting up the configuration, Run the package.

- firebase_core: ^0.5.2+1

After adding dependencies run this command

flutter pub add firebase_core

A flutter plugin to use the Firebase Core API, which enables connecting to multiple Firebase apps.

- cloud_firestore: ^0.14.3+1

After adding it run this command

Flutter pub add cloud_firestore

A flutter plugin to use the Cloud Firestore API

- lottie: ^1.4.3

After adding it run this command

Flutter pub add lottie

A flutter plugin to use the lottie animation.

cupertino_icons: ^1.0.2

google_nav_bar: ^5.0.6

pine_icons: ^2.0.1

page_transition: ^2.0.9

firebase_auth: ^4.1.0

- Google_maps_flutter:^2.2.1
- Permission_handler:^10.0.2
- Location:^4.4.0
- Flutter_toast:^8.0.9
- Firebase_database:^10.0.3
- Image_picker:^0.8.6
- Image_cropper:^3.0.0
- Get:^4.6.5
- Google_fonts:^3.0.1
- Path:^1.8.2
- Geolocator:^9.0.2
- Connectivity:^0.3.2

1. Components

Creating a Flutter project named XVII Application, in our application Splash Screen will be the first screen after splash screen it navigates to the Home Page. Home consists of an appBar, bottomNavigationBar with two icons (Home, Community) on pressed it navigates accordingly.

- android - Auto generated source code to create android application
- assets - contains image folder for the splash screen
- build - When you run a flutter project, it builds depending on which emulator or device it is running on, doing a Gradle or XCode build using the folders inside it. In short, those folders are entire apps which set the stage for the flutter code to run
- ios - Auto generated source code to create ios application
- lib - Main folder containing Dart code written using flutter framework
- lib/main.dart - Entry point of the Flutter application
- lib/Screens - contains dart files for
 - Login
 - Signup
 - Community
 - Stocks
- lib/widgets - contains dart files for
 - MessageTileWidget
 - GroupTileWidget
 - GreenIntroSignup
 - GreenIntroWidget
 - CompletedListWidget
- lib/services - contains dart files for
 - Authservice
 - Databaservice
- lib/api - contains dart file for firebase
- gitignore - Git version control file
- metadata - auto generated by the flutter tools
- packages - auto generated to track the flutter packages
- pubspec.yaml - Used by Pub, Flutter package manager

- pubspec.lock - Auto generated by the Flutter package manager, Pub
- README.md - Project description file written in Markdown format.

3.4.2 Widgets

Stateless and statefull widgets:

To make our app look more like a todo app, let's show some tasks. You may have noticed that our simple app above is a `StatelessWidget`. This means that it cannot be changed dynamically. For our todo app, this is no good, because todo items get added and removed all the time. However, a `StatelessWidget` can have children which are dynamic, which are `StatefulWidget`s. Let's break out our stateful functionality the todo list container from the overall app container.

To make a stateful widget, we need two classes – one for the widget itself, and a second which creates the state. This setup allows the state to be easily saved, and enables the use of features such as hot reloading.

- Widgets are high level objects used to describe any part of an application. It can be but is not limited to UI elements such as buttons ; layout alignment, padding; data a theme, configurations
- Widgets can be everything. In flutter, almost everything you'll code will be inside a Widget. Even things like "redux reducers" are widgets in flutter.
- The core concept of the Flutter framework is In Flutter, Everything is a widget.
Widgets
- are basically user interface components used to create the user interface of the application.

- In Flutter, the application is itself a widget. The application is the top-level widget and its UI is built using one or more children widgets, which again build using its children
- widgets. This composability feature helps us to create a user interface of any complexity.

1. In main.dart

- runApp() Function takes the given Widget and makes it to the widget tree in main.dart the widget tree consists of a MaterialApp widget which describes theme, primarySwatch and scaffoldBackgroundColor.

2. In home_page.dart

- Scaffold() - implements the basic material design visual layout structure. This class provides APIs for showing drawers and bottom sheets.
- AppBar() - AppBar and Scaffold widgets, and from material.dart, the app is starting to look a bit more Material. For example, the app bar has a shadow and the title text inherits the correct styling automatically. AppBar widget lets you pass in widgets for the leading widget, and the actions of the title widget.
- BottomNavigationBar class - A material widget that's displayed at the bottom of an app for selecting among a small number of views, typically between three and five.

The bottom navigation bar consists of multiple items in the form of text labels, icons, or both, laid out on top of a piece of material. It provides quick navigation may be a better fit.

A bottom navigation bar is usually used in conjunction with a scaffold, where it is provided as the Scaffold.bottomNavigationBar argument.

- FloatingActionButton class - A material design floating action button. A floating action button is a circular icon button that hovers over content to promote a primary action in the application. Floating action buttons are most commonly used in the Scaffold.

onPressed callbacks to AddTodoDialogWidget()

CHAPTER – 4

RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

1. RESULTS

1. Splash Screen intro section

A splash screen is a graphical control element consisting of a window containing an image, a logo, and the current version of the software. A splash screen can appear while a game or program is launching.

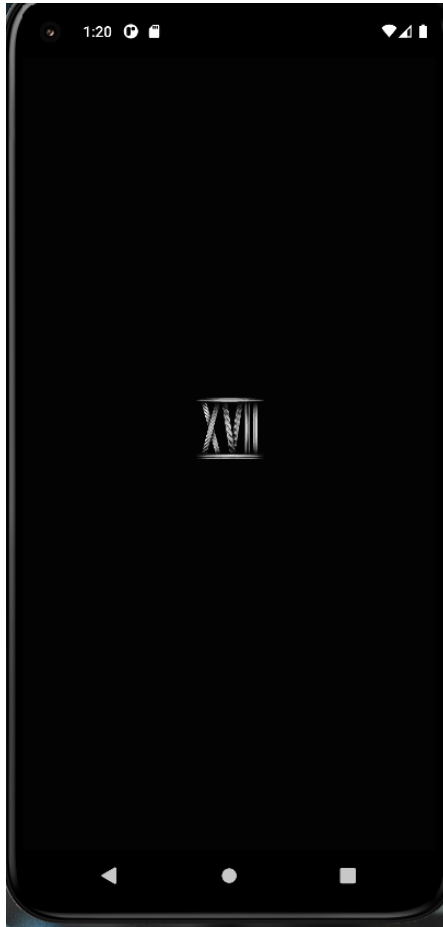


Fig 4.1 Splash screen

4.1.2. Homepage intro section

This consists of AppBar, BottomNavigationBar Items with label 'home' "community".

In Home page list of Stock is available you have to select anyone, And it navigate to chart page .

Search, Filter options are available in home page to check the stocks.

Chart page has analysis of that stock with graphs(ups and down).

For buying & selling stocks or shares first you need a demat account .

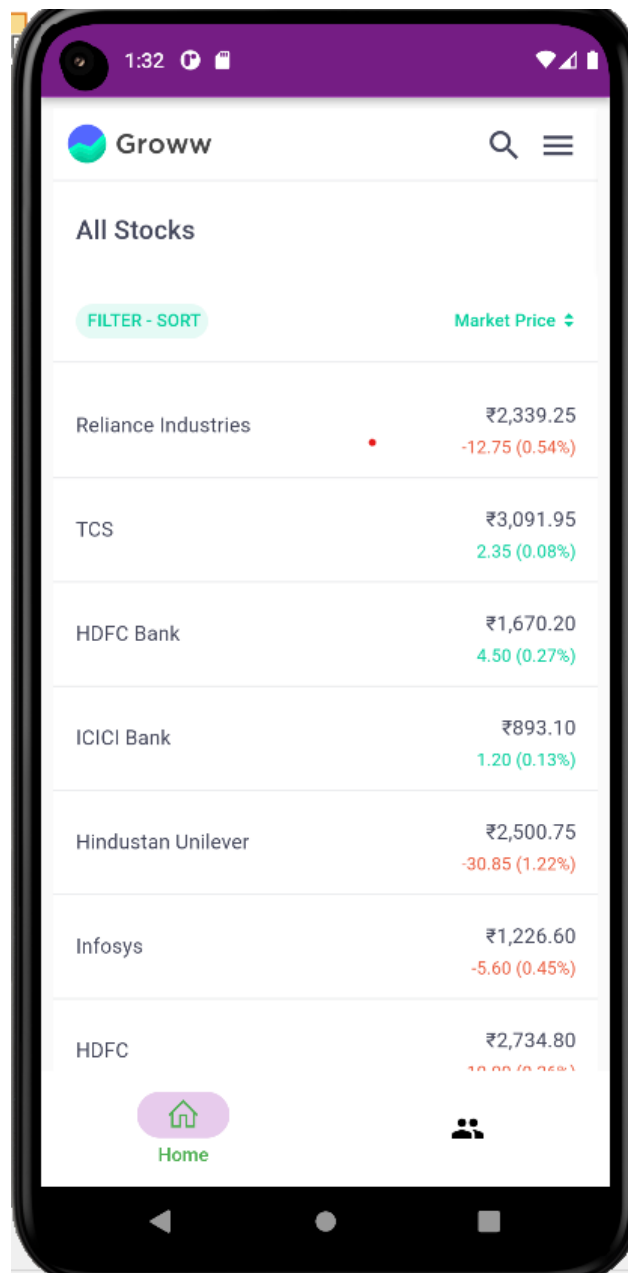
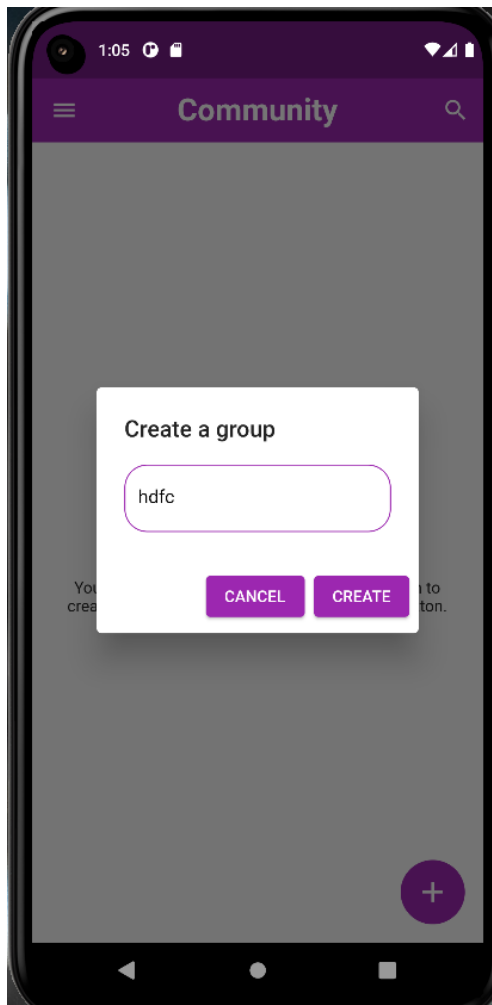


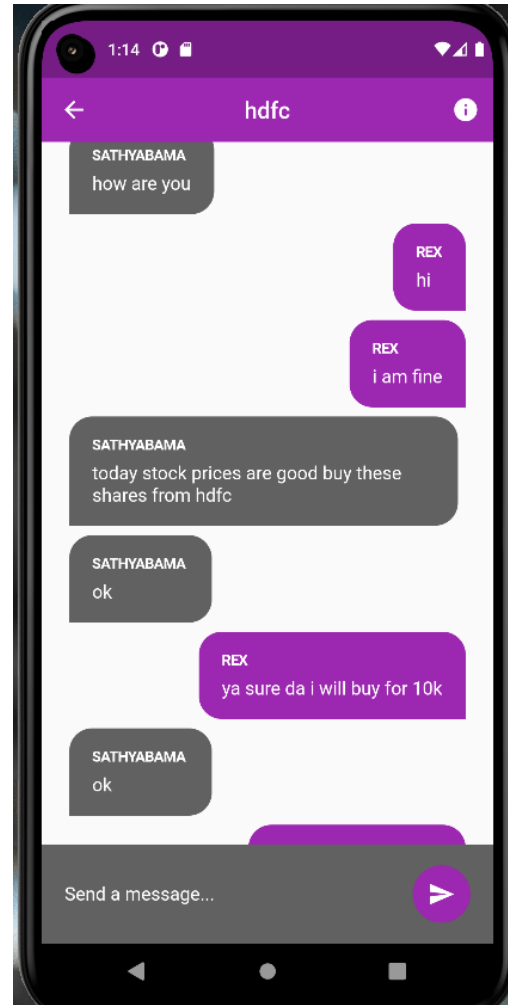
Fig 4.2 Home screen

4.1.3. Add community name intro in community section

This screen pops up when we press the add icon(+) floating action button on the community page, Title area takes the input of the community name which we going to assign for the community.



Create Community



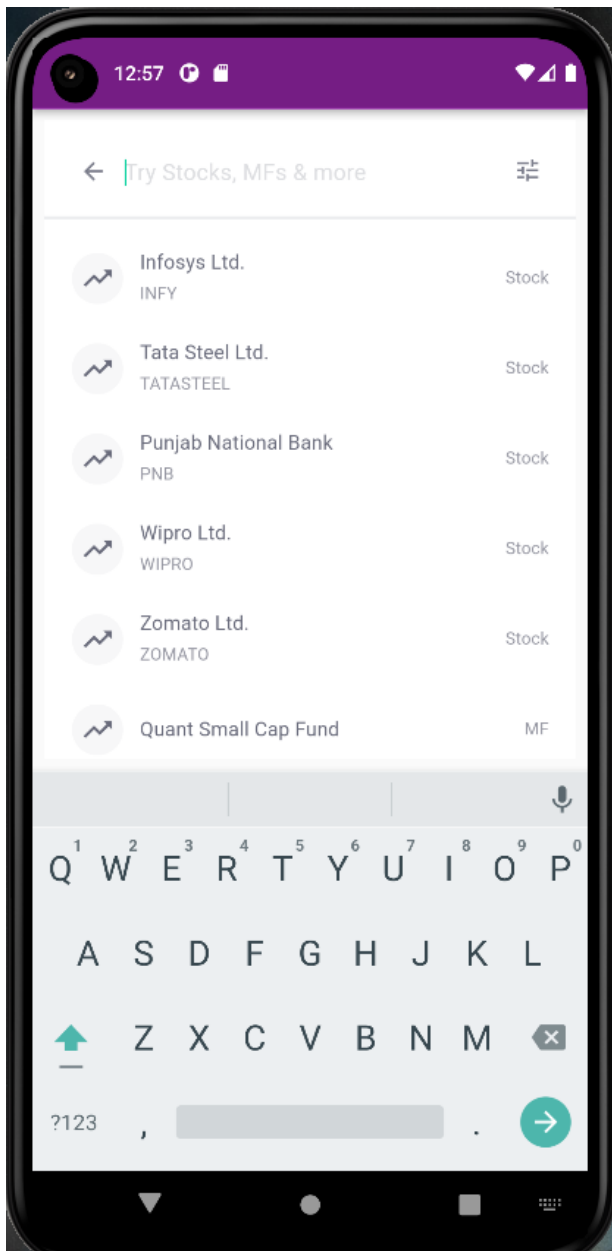
Group page

Home page (Search,Filter,Charts)

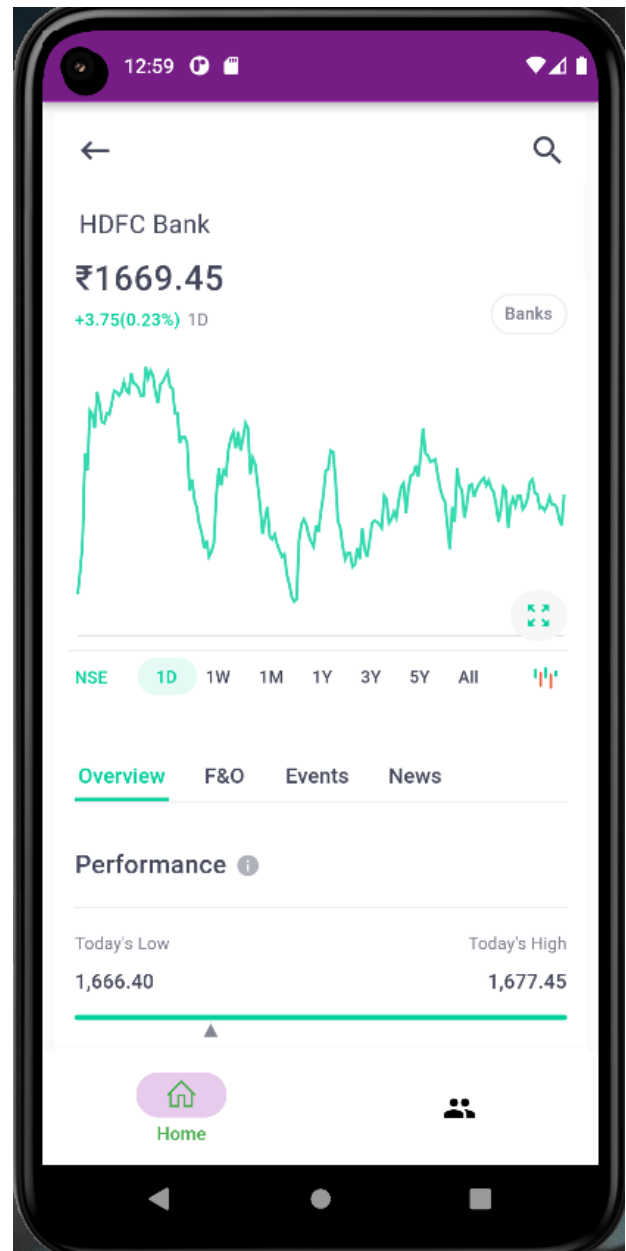
Search the share name(stock) using search bar.

Filter option is there for your user convenience to check the stocks .

Selecting any of the stocks then it will show the graph.



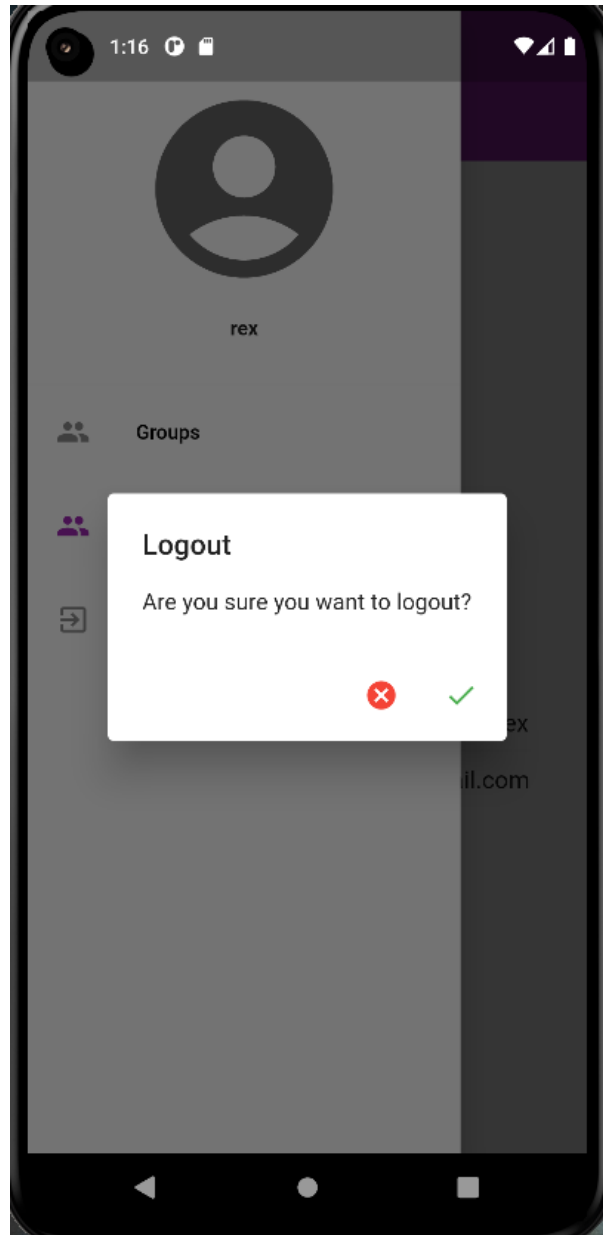
Selected scrap



variety scrap

Logout :

Clicking logout in the navigation bar it will show one alert dialog for logout .
Then click Logout it will logout from your User Id.



Logout

Admin :

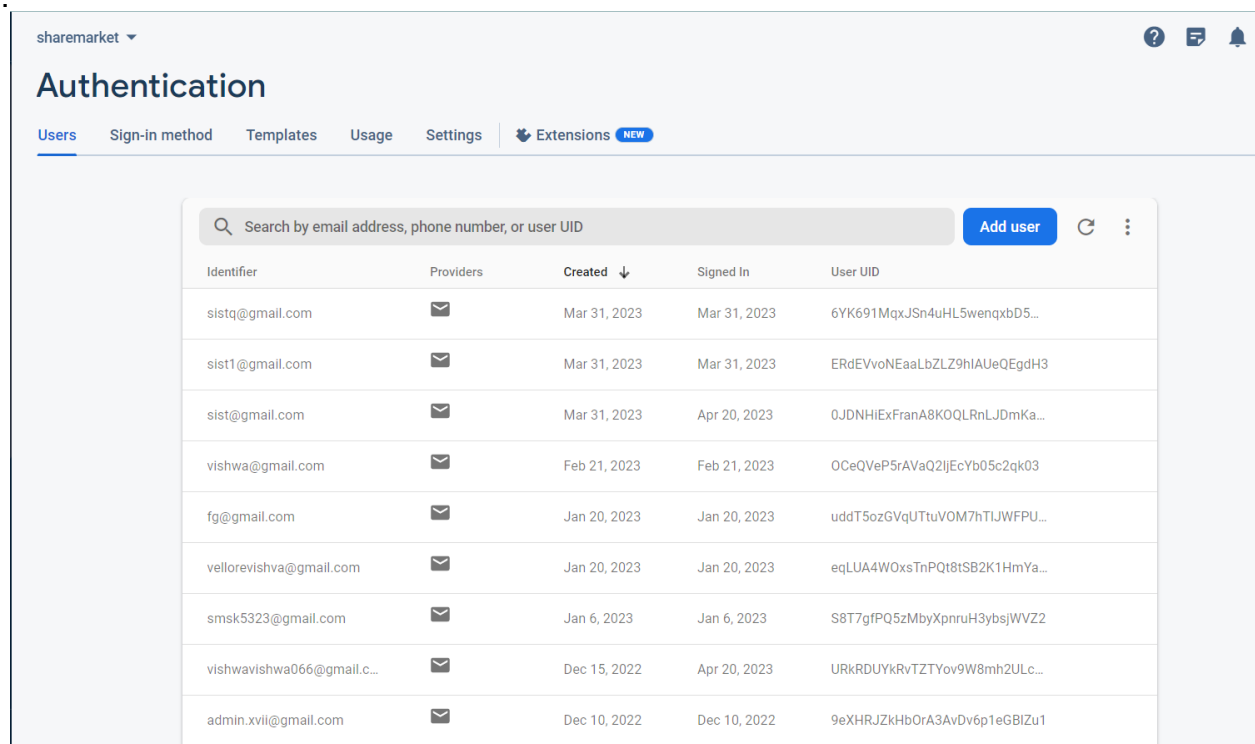
1. User management: The dashboard would allow the admin to manage users, including creating, updating, and deleting user accounts. The admin could also view user activity, such as which groups they are part of and their portfolio performance.

2. Group management: The dashboard would allow the admin to manage groups, including creating, updating, and deleting groups. The admin could also view group activity, such as which users are part of each group and the group's portfolio performance.

3. Stock management: The dashboard would allow the admin to manage the list of stocks available for purchase within the app. The admin could add, edit, or delete stocks and view their current prices and performance.

4. Notification management: The dashboard would allow the admin to manage notifications sent to users, including creating, editing, and deleting notifications. The admin could also view the history of notifications sent to users.

5. Payment management: The dashboard would allow the admin to manage payments made by users, including tracking funds transferred into the app and disbursing funds to users when the group sells stocks.



The screenshot shows the 'Authentication' section of a dashboard titled 'sharemarket'. It features a navigation bar with links for 'Users', 'Sign-in method', 'Templates', 'Usage', 'Settings', and 'Extensions' (marked as 'NEW'). Below the navigation bar is a search bar with the placeholder text 'Search by email address, phone number, or user UID' and an 'Add user' button. The main content area displays a table of users with the following columns: Identifier, Providers, Created, Signed In, and User UID. The table contains 10 rows of user data.

Identifier	Providers	Created	Signed In	User UID
sistq@gmail.com		Mar 31, 2023	Mar 31, 2023	6YK691MqxJSn4uHL5wenqxbD5...
sist1@gmail.com		Mar 31, 2023	Mar 31, 2023	ERdEVvoNEaaLbZLZ9hiAUeQEgdH3
sist@gmail.com		Mar 31, 2023	Apr 20, 2023	0JDNHIExFranA8K0QLRnLJDmKa...
vishwa@gmail.com		Feb 21, 2023	Feb 21, 2023	OCeQVeP5rAVaQ2lJcYb05c2qk03
fg@gmail.com		Jan 20, 2023	Jan 20, 2023	uddT5ozGVqUTtuVOM7hTlJWFPU...
vellorevishva@gmail.com		Jan 20, 2023	Jan 20, 2023	eqLUA4WOxsTnPQt8tSB2K1HmYa...
smsk5323@gmail.com		Jan 6, 2023	Jan 6, 2023	S8T7gfPQ5zMbyXpnruH3ybsjWVZ2
vishwavishwa066@gmail.c...		Dec 15, 2022	Apr 20, 2023	URkRDUYkRvTZTYov9W8mh2ULc...
admin.xvli@gmail.com		Dec 10, 2022	Dec 10, 2022	9eXHRJZkHbOrA3AvDv6p1eGBIZu1

Admin authentication

sharemarket

Cloud Firestore

?

XV

groups

3P7yhowmVz1vqxMV19T0

messages

More in Google Cloud

groups

+ Add document

33rYI9csnL0rNR2F2I71

3P7yhowmVz1vqxMV19T0

3cxdj1TlI49ZmpHS0h4s

C9iYCumD6ut9kAyp5rV

WqGZ9KTD0T0BbiggbKtI

fIuFgtTKqiY7TFghFqM9

3P7yhowmVz1vqxMV19T0

+ Start collection

messages

+ Add field

admin: "URkRDUYkRvTZTYov9W8mh2UL

groupIcon: ""

groupId: "3P7yhowmVz1vqxMV19T0"

groupName: "hdfc"

members

0 "URkRDUYkRvTZTYov9W8mh2UL

1 "0JDNHiExFranA8KOQLRnLJDmK

recentMessage: "sure"

messages

+ Add document

4e3jUrsPbyfgeMW5baon

7iW9V0kjTTCeWg1kF2Dw

JGNtPff0aSefCb0dnrEM

JLJresJ7luV36MaIHL9c

Ju4NiQn6g2bXsJE8FvQq

Nq1ZvVPBL2rjr4K4NZTR

07DDfNGBcTTt1TJ3bzjG

SBMIsE1A7hNxq1UWgrj1

VI8gCPCRjUwVKeob7PV

Wsty1BVaRmMNMW201ygx

raadGAQj5KyzaYLSz76U

s5sRmiOzodZy88MGVpMW

uENHyxpHaSDiIEN3YkFt

y0BXKhp7RUymqnUa3PeH

Database location: nam5

Admin for managing messages

2. PERFORMANCE ANALYSIS

- The performance of the application is very efficient on latest android Os version 12
- The performance of the application is very efficient on latest mac version macos 12 monterey
- Internet connection is recommended
- Minimum RAM required is 2gb
- Due to efficient build strategy the app size is comparatively lower so the performance will be better
- Completely responsive

CHAPTER – 5

SUMMARY AND CONCLUSION

1. SUMMARY

- This project is done based on flutter frame work using dart programming language with the help of Visual Studio Code IDE and mobile emulator / real time device .
- A stock market app is a form of shares where people use application to buy and sell their shares.
- It helps users to sell&buy a sharesthrough their community.
- Our app helps us become user friendly.
- A use of firebase_core provides access to the firebase configuration and performs initialization of the native Firebase app
- A use of cloud firestore is a NoSQL document database that lets you easily store, sync, and query data for your mobile and web apps - at global scale

2. CONCLUSION

In conclusion, a stock market group buy and sell app developed using Flutter has the potential to provide users with a collaborative and convenient approach to investing in the stock market. The app could include features such as group buy and sell, notifications, portfolio management, payment gateway integration, and an intuitive user interface.

Overall, a stock market group buy and sell app developed using Flutter would aim to enhance the user experience, prioritize security and reliability, and provide users with a collaborative approach to investing in the stock market.

FUTURE WORK

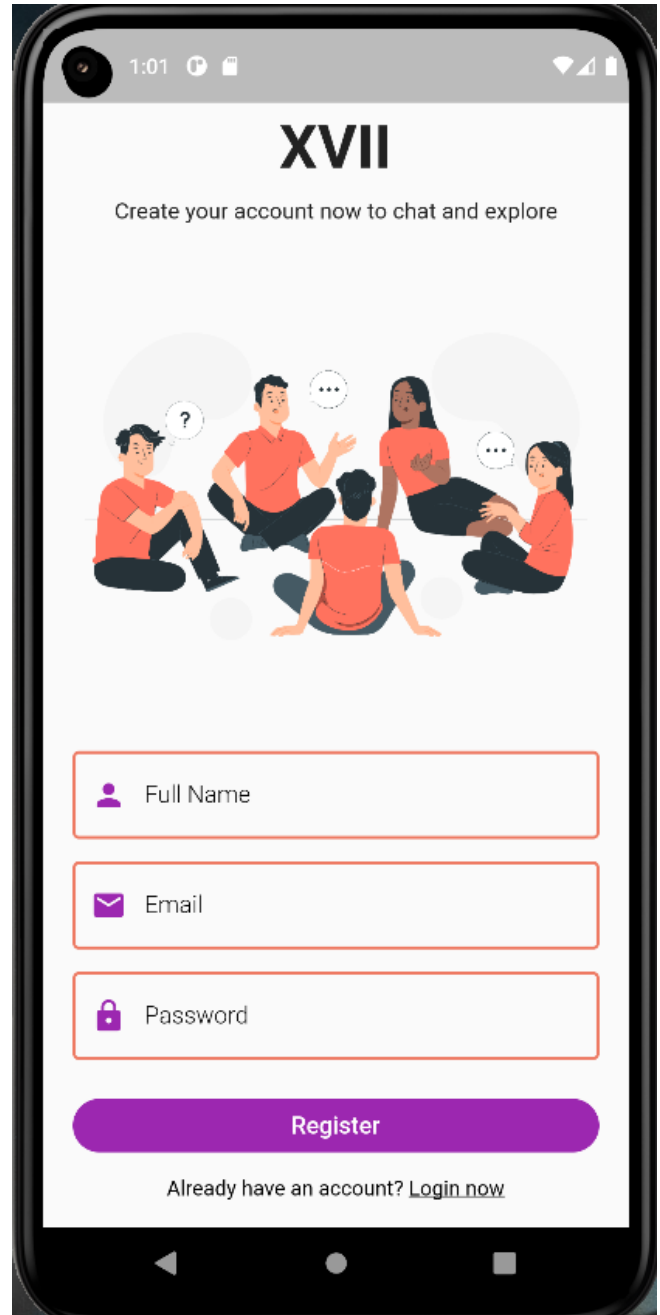
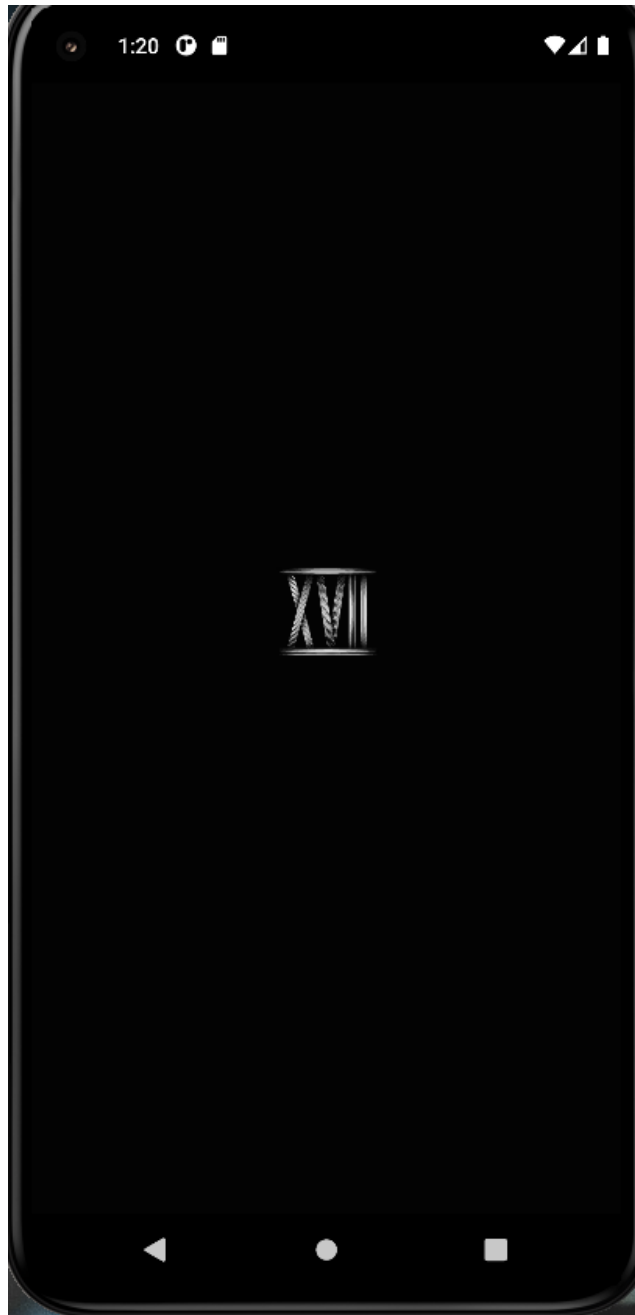
When it comes to mobile app development there are plenty of frameworks and programming languages to choose from. But, in recent years Flutter is gaining popularity because of its numerous benefits. Developers are enjoying working on this platform and the subsequent updates provide them with the flexibility and innovation they crave.

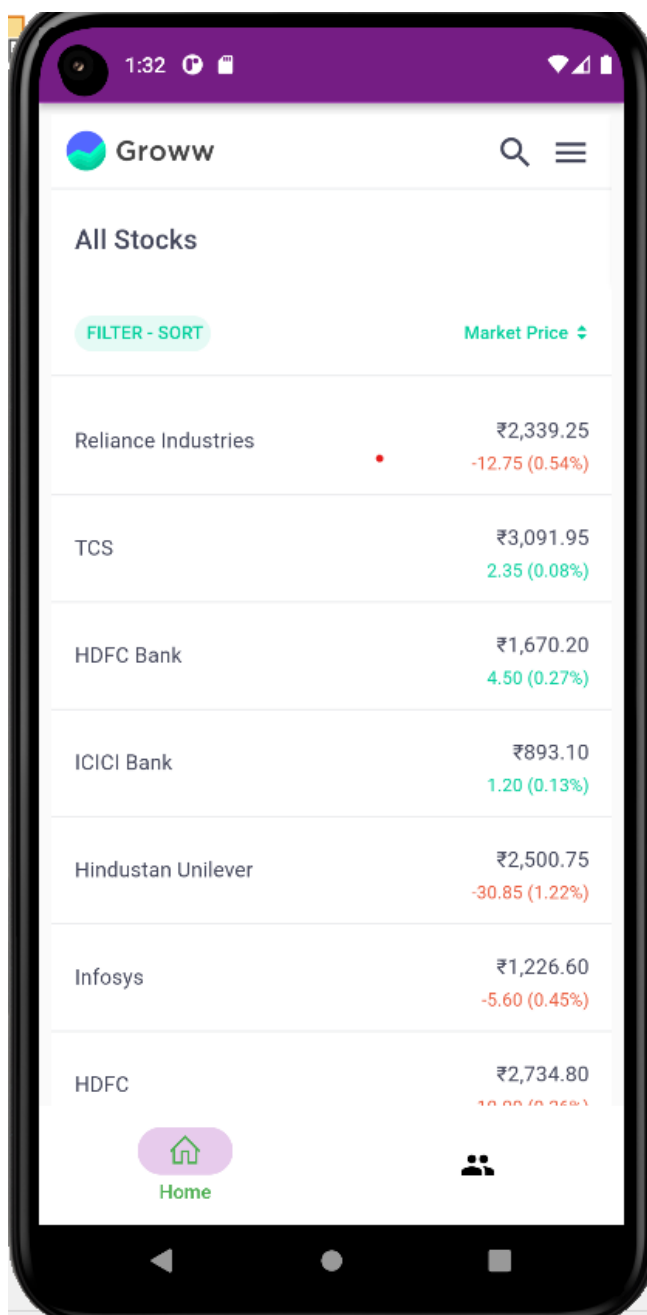
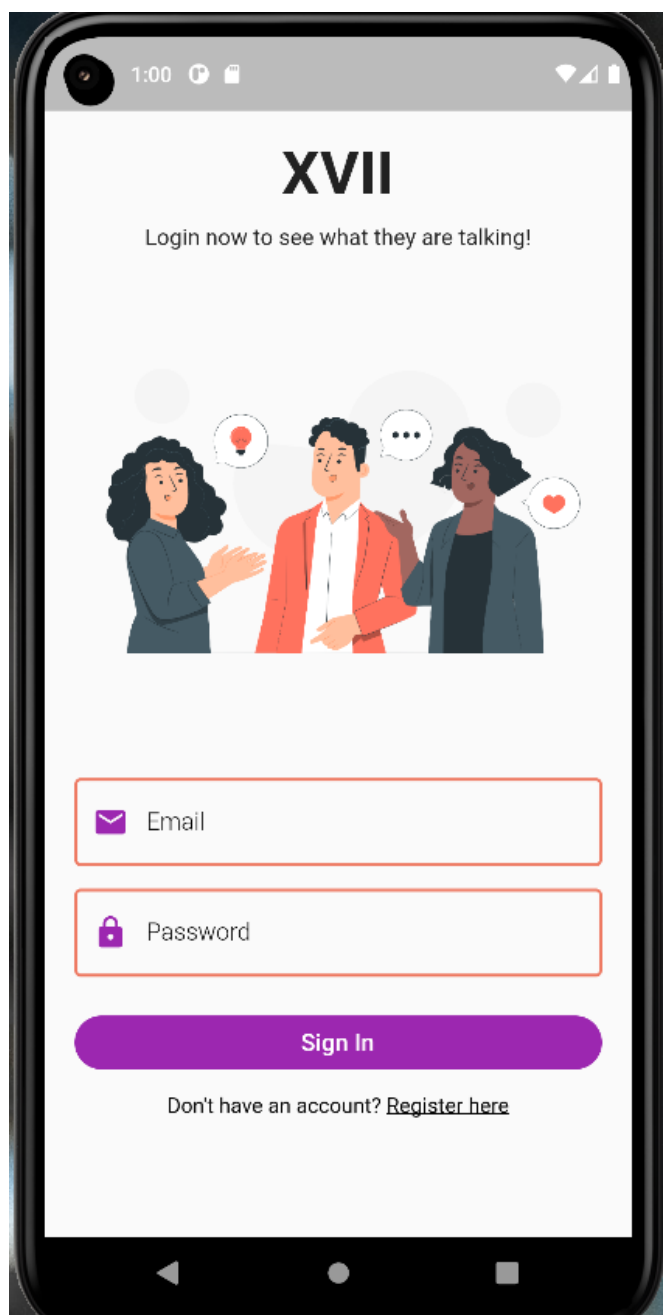
In fact, there has been a gradual growth in the popularity of Flutter SDK (software development kit) as about 42% of Statista survey respondents chose it as their preferred platform in 2021. While it has surely become the most used programming language for mobile app development, the release of Flutter 3 on May 11, 2022, is expected to further solidify this effect.

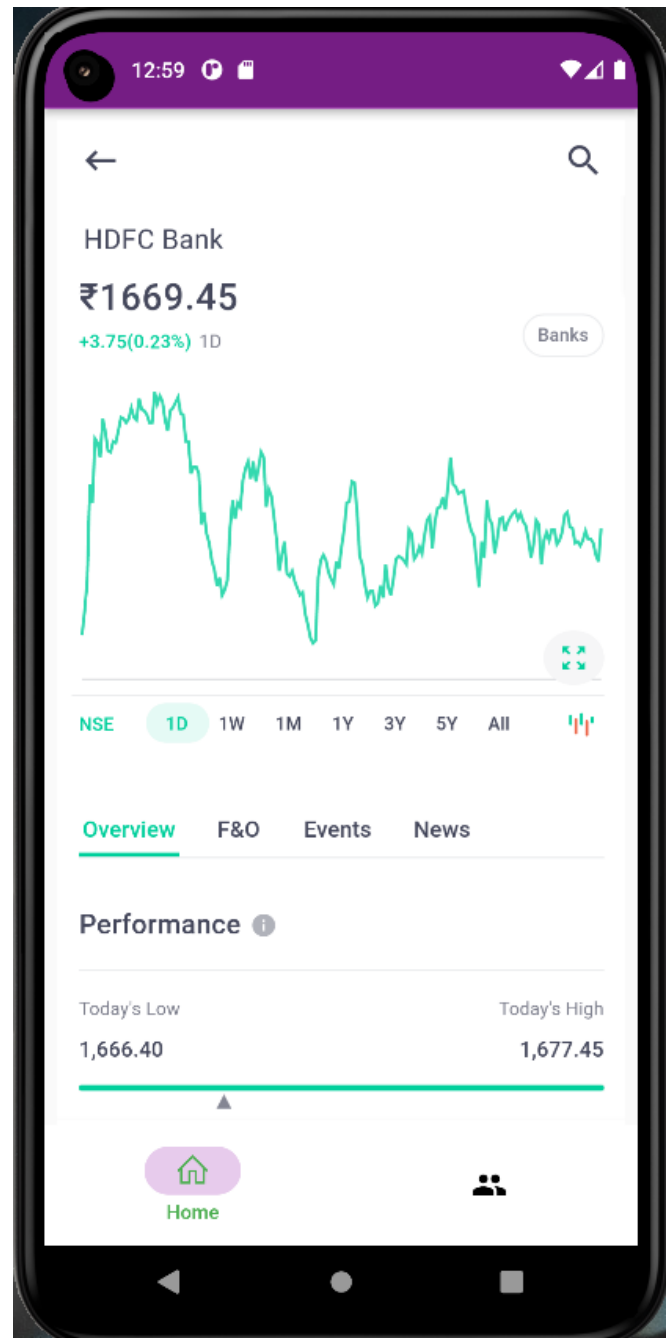
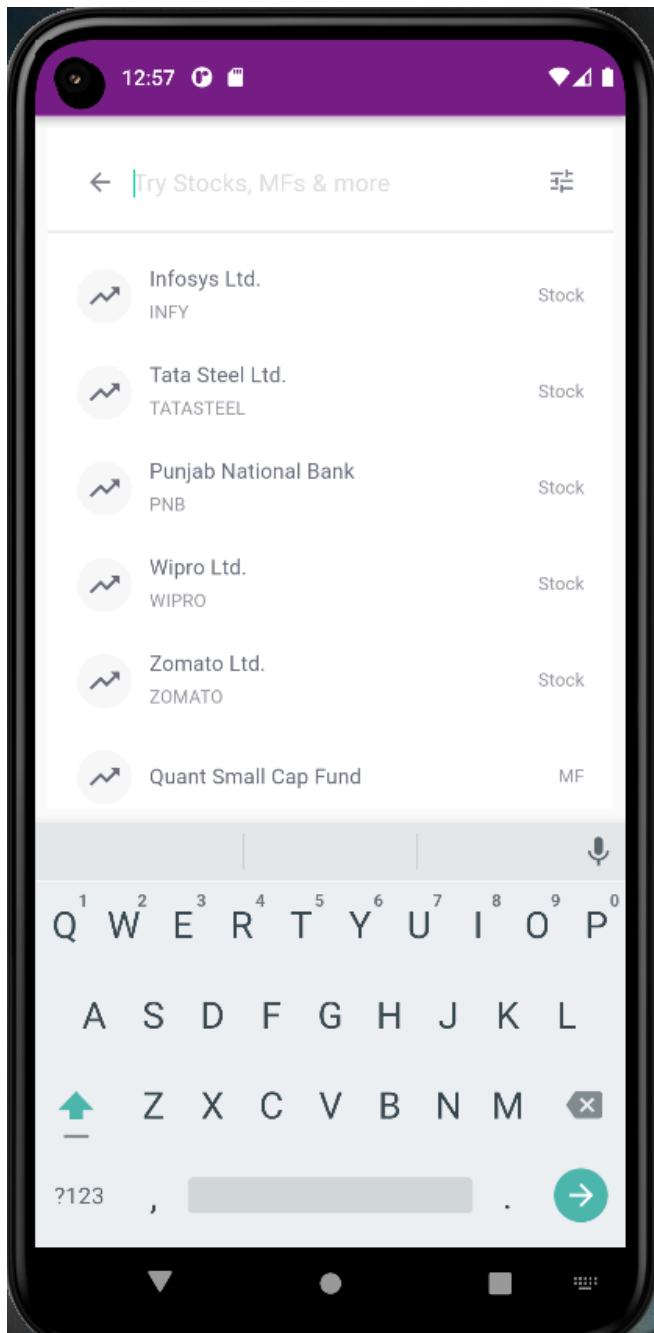
Flutter is easy to learn and implement as it is dependent on Dart programming language. Dart is simple and easy to learn by all programmers compared to other programming languages. Due to various Flutter 3 features like a single code base, support for multiple platforms, etc., Flutter stands out.

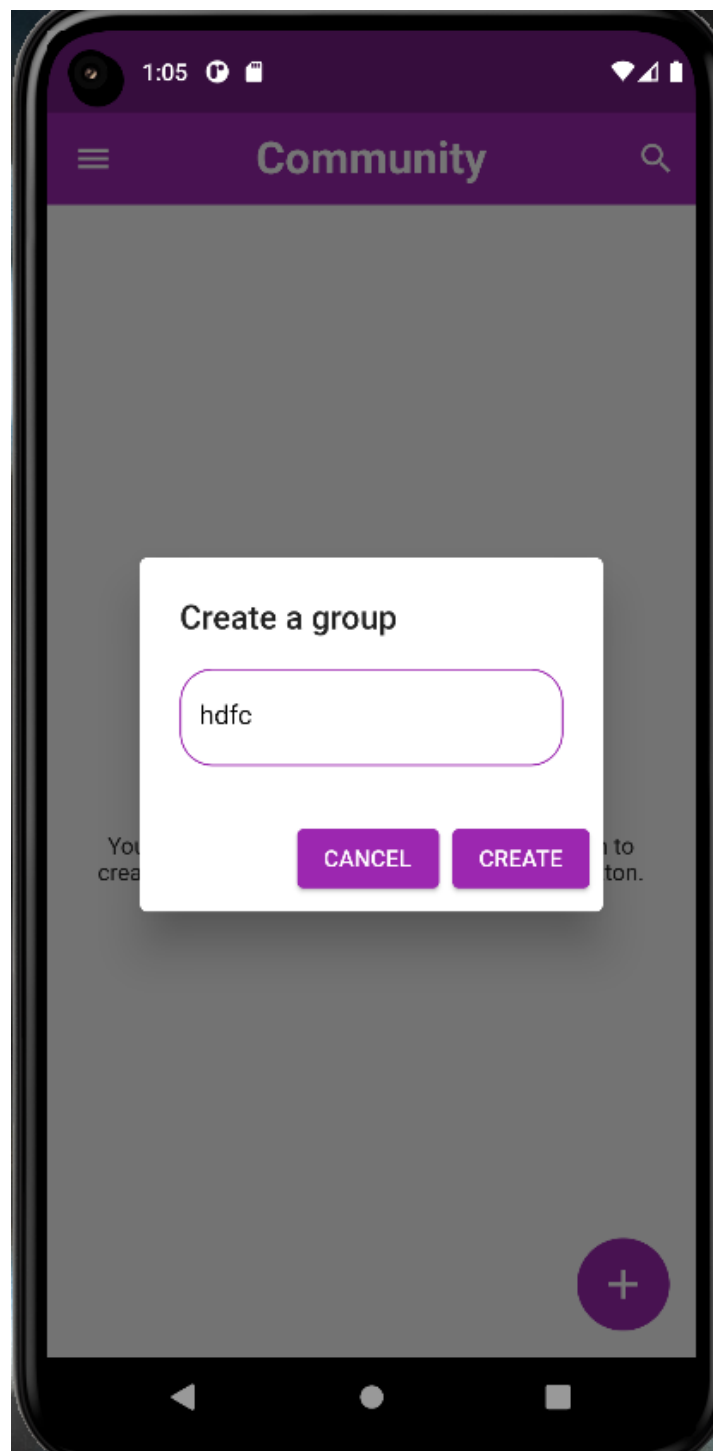
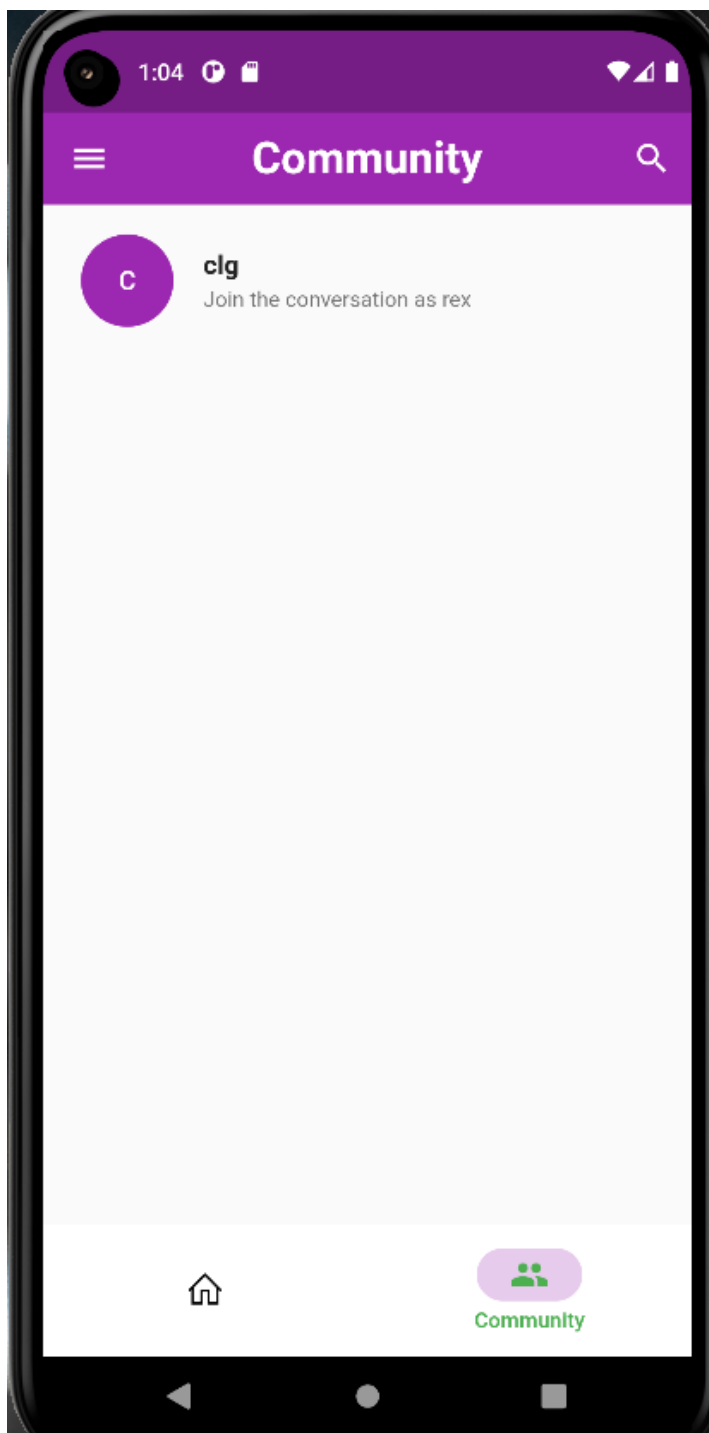
APPENDIX

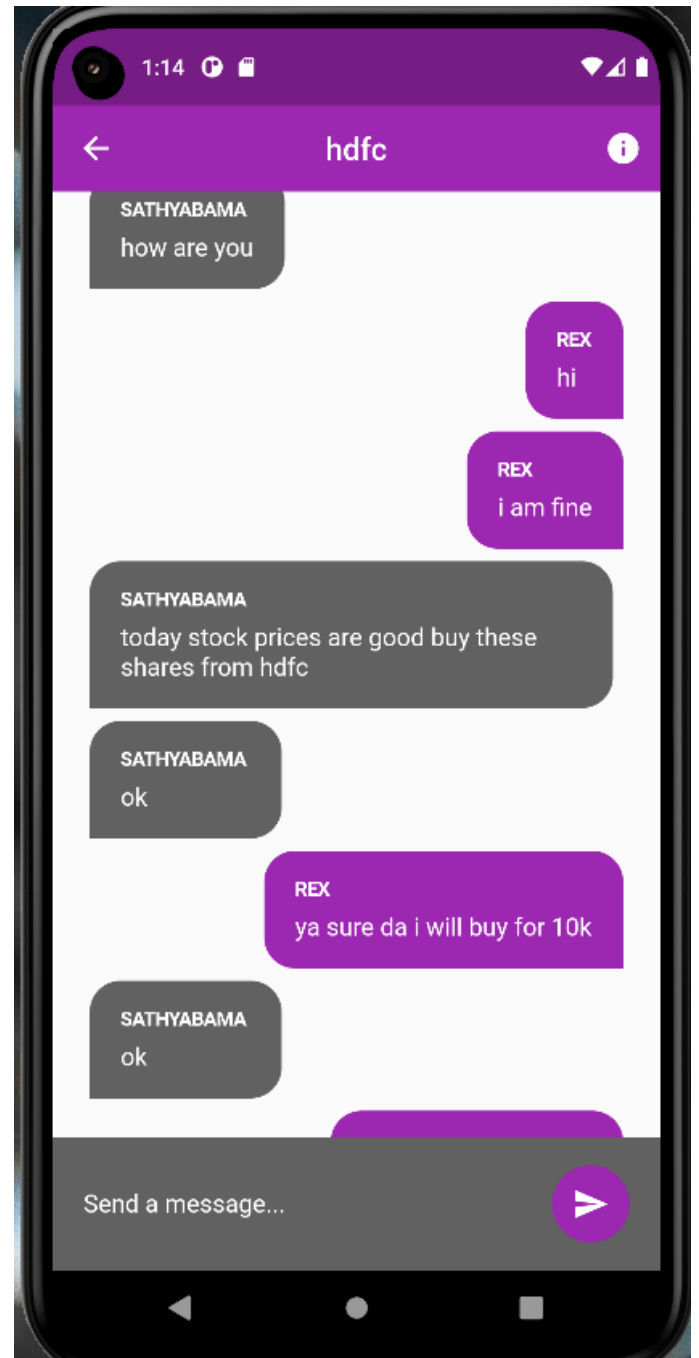
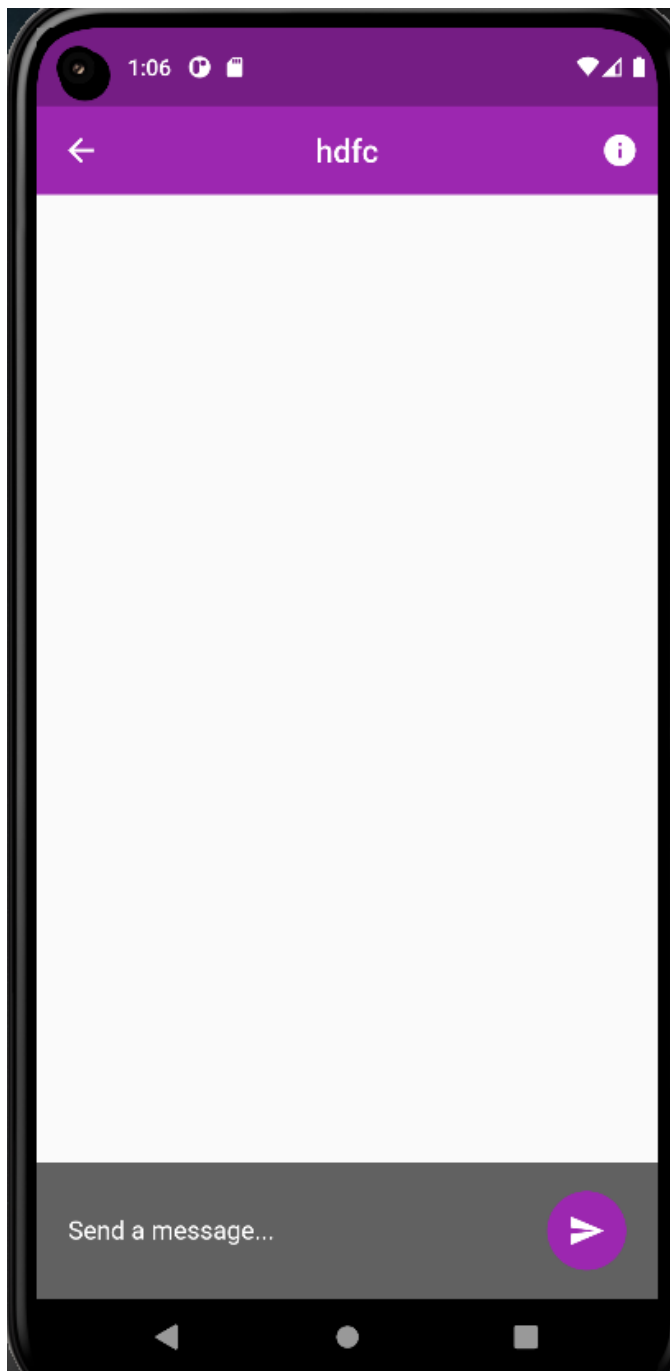
A. SCREENSHOTS

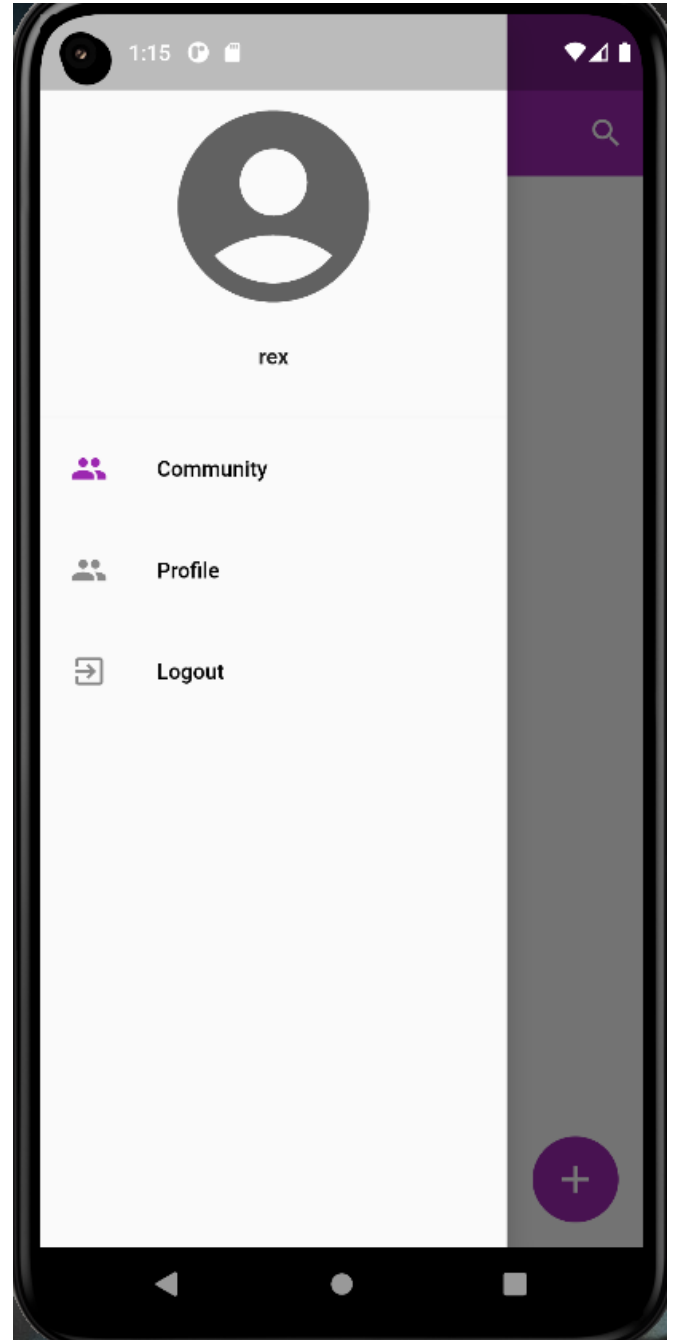
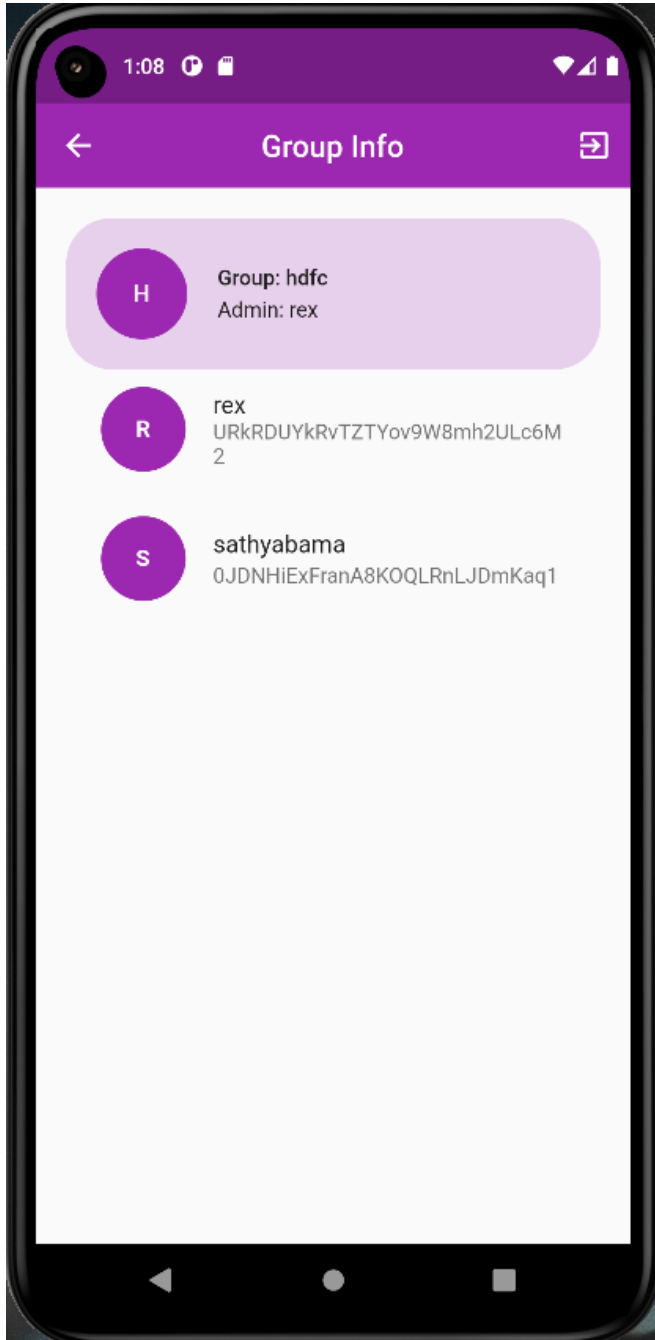


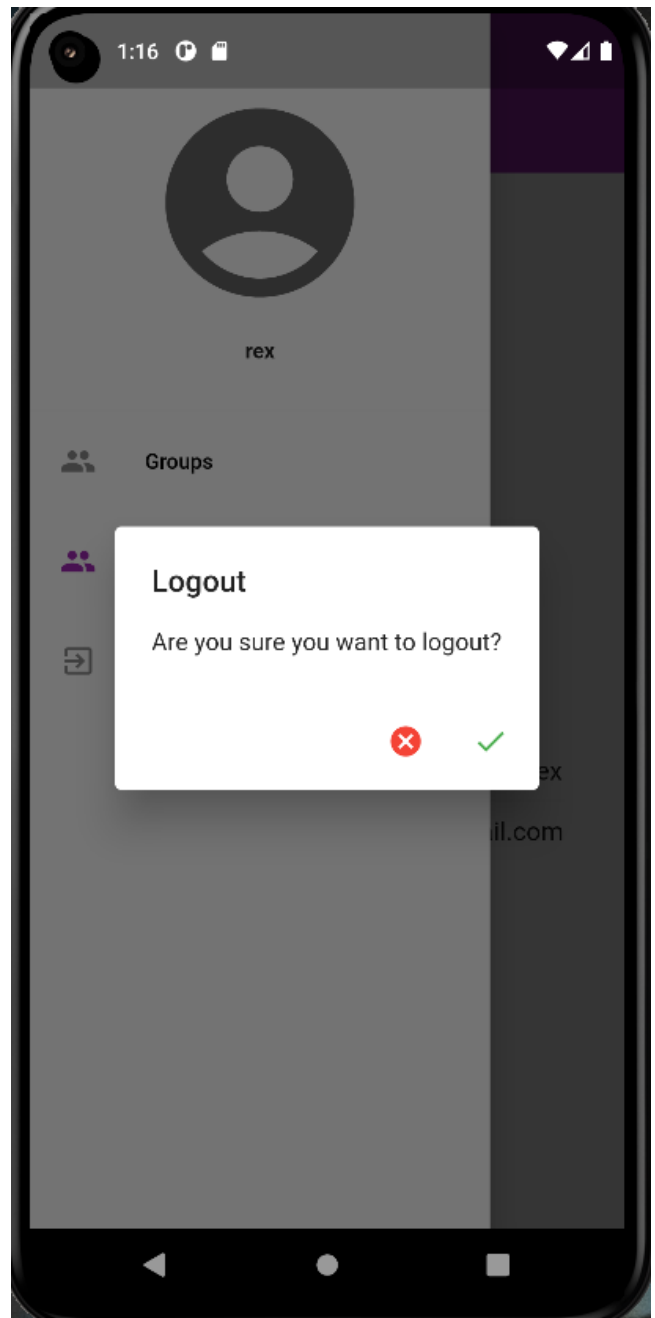












REFERENCES

- <https://flutter.dev/docs/reference/tutorials>
- <https://pub.dev/flutter/packages>
- <https://flutter.dev/docs/development/ui/advanced/splash-screen>
- <https://www.section.io/engineering-education/how-to-build-a-flutter-todo-app/>
- <https://daily-dev-tips.com/posts/build-a-todo-list-app-with-flutter/>
- <https://mobikul.com/how-to-make-the-to-do-list-in-the-flutter/>
- <https://itsallwidgets.com/to-do-list>