**Mini Project Report on**

**“CRYPTOCURRENCY TRACKER APP”**

Submitted to

**Visvesvaraya Technological University**

Jnana Sangama, Belagavi In partial fulfillment of the requirements for the award of Degree in

Bachelor of Engineering in Computer Science and Engineering

**Name of student USN**

**Vishant Patil 2BU19CS056**

**Swapnil Chimanapagol 2BU20CS408**

Under the guidance of

**Asst. Prof. Shaheen Mujawar**

Computer Science and Engg. Department

S.G.B.I.T. Belagavi



Department of Computer Science and Engineering

**Accredited by NBA**

S.S EDUCATION TRUST

**S. G. BALEKUNDRI INSTITUTE OF TECHNOLOGY**

***An ISO 21001:2018 certified institution***

Shivabasavanagar, Belagavi-10, Karnataka.

**2021 – 2022**

**S. G. BALEKUNDRI INSTITUTE OF TECHNOLOGY**

***An ISO 21001:2018 certified institution***

Shivabasavanagar, Belagavi-10, Karnataka.



Department of Computer Science and Engineering

**Accredited by NBA**

**Mobile Application Development (18CSMP68)**

**CERTIFICATE**

Certified that the Mini Project Work entitled **“CRYPTOCURRENCY TRACKER APP”**is a bonafide work carried out by **Mr.Vishant Patil (2BU19CS056)**, **Mr.Swapnil Chimanapagol (2BU20CS408)** in partial fulfillment for the award of Degree of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belagavi, during the year 2021-2022. The Mini Project report has been approved as it satisfies the academic requirements in respect of Mini Project work prescribed for the said Bachelor of Engineering Degree.

Guide HOD

Asst. Prof. Shaheen Mujawar Dr.B.S.Halkarnimath

Principal

Dr. B. R. Patagundi

viva

Name of the examiners: Signature with date

1.

2.

**ABSTRACT**

The topic that we have selected is called “Cryptocurrency Tracker” which we have developed using Android Studio. It is an App that shows the values of top cryptocurrencies in real time. Every time when the App is clicked, the values get updated. The software is recognized by the app with the help of “Androidmanifest.xml” file. The app uses a java language which does not affect any content of the App whenever it is used..

Cryptocurrency is an encrypted, peer-to-peer network for facilitating digital barter, a technology developed eight years ago. Bitcoin, the first and most popular cryptocurrency, is paving the way as a disruptive technology to long standing and unchanged financial payment systems that have been in place for many decades. While cryptocurrencies are not likely to replace traditional fiat currency, they could change the way Internet-connected global markets interact with each other, clearing away barriers surrounding normative national currencies and exchange rates. Technology advances at a rapid rate, and the success of a given technology is almost solely dictated by the market upon which it seeks to improve. Cryptocurrencies may revolutionize digital trade markets by creating a free flowing trading system without fees.

The App gives live updates of all the cryptocurrency from all over the internet without going to their official website. This app collects all data from the official website and shows it in one app so that it is easier for people and doesn't waste their valuable time.

The idea of tracking cryptocurrency prices and improving profitability of individuals as they do not have to go browsing for prices on different websites and learn about market ups and downs which makes people more motivated in the cryptocurrency field.

**ACKNOWLEDGEMENT**

It is our proud privilege and duty to acknowledge the kind help and guidance received from several persons in preparation of this report. It would not have been possible to prepare this report in this form without their valuable help, cooperation and guidance.

First and foremost, we wish to record our sincere gratitude to Management of this college and to our beloved Professor, **Dr.B.R.Patagundi,** Principal, S. G. Balekundri Institute of Technology, Belagavi for his constant support and encouragement in preparation of this report and for making available library and laboratory facilities needed to prepare this report.

Our sincere thanks are also due to **Dr. B. S. Halakarnimath,** Head, Department of Computer Science and Engineering in S.G.B.I.T. for the valuable suggestions and guidance through the period of preparation of this report.

We express our sincere gratitude to our beloved guide, **Asst. Prof. Shaheen Mujawar** in the Dept. of Computer Science and Engineering, S.G.B.I.T., Belagavi for guiding us in investigations for this project and in carrying out mini project work. Our numerous discussions with her were extremely helpful. We hold her in esteem for guidance, encouragement and inspiration received from him/her.

The course on MOBILE APPLICATION DEVELOPMENT was very helpful to us in giving the necessary background information and inspiration in choosing this topic for the project. The classroom lectures of Mrs. SHAHEEN MUJAVAR were very much knowledgeable, informative and interesting. Our sincere thanks to Mrs. SHAHEEN MUJAVAR for the inspiring lectures and for help in choosing the topic of this project work.

Last but not the least, I wish to thank my family members for financing my studies in this college as well as for constantly encouraging me to learn engineering. Their personal sacrifice in providing me this opportunity to learn engineering is gratefully acknowledged.

**Belgaum**

**Date: (Name of the students)**

**Vishant Patil**

**Swapnil Chimanapagol**

**CONTENTS**

Certificate

Abstract i

Acknowledgement ii

List of Figures iii

List of Tables ix

Chapter 1: INTRODUCTION

1.1 Motivation ……………………………………………………………………….6

1.2 Proposed System …………………………………………………...................... 6

1.3 Related Work ……………………………………………………………………6

Chapter 2: REQUIREMENTS ANALYSIS AND SYSTEM SPECIFICATION

2.1 Software Requirements Specification ………… ………………………………..7

2.2 Hardware Requirement Specification……………………………………………7

Chapter 3: SYSTEM DESIGN

3.1 Detailed Design ……………………………………………………................... 8

3.2 Application Design…………………………………………………….............. .9

3.3 User Interface Design …………………………………………………………..10

3.4 Database Design ………………………………………………………………. 12

3.6 Entity Relationship Diagram ………………………………………...................13

Chapter 4: IMPLEMENTATION AND TESTING

4.1 Introduction to Programming Languages, IDE’s, Tools & Technologies ………14

4.2 Security and Permissions in Android …………………………………………. .19

4.3 Test Plan and Test Activities ………………………………………………........25

Chapter 5: RESULTS AND DISCUSSIONS

5.1 User Interface Representation …………………………………………………...22

Chapter 6: FUTURE WORK ………………………………………………………………..26

CONCLUSION…………………………………………………………………26 Bibliography ………………………………………………………..26

**LIST OF FIGURES**

**Figure no. Particulars Page No**

3.1. Data Flow Diagram 0 9

3.2 Application design 10

3.2.1 The Layout Editor ER Diagram 10

3.2.2 Button in layout Editor 11

4.2.3.1 The Project Files in Android View

4.2.3.2 The Android Studio main window 14

4.2.3.3 An inline variable value 16

4.2.3.4 Your project's (1) instrumented tests and

(2) local JVM tests are visible in either the

Project view (left) or Android view (right) 18

5.1.1 Welcome Page 27

5.1.2 Register Page 27

5.1.3 Login 28

5.1.4 My Profile 29

5.1.5 Edit Profile 30

5.1.6 Edit Email 31

5.1.7 Change Password 32

5.1.8 Delete Account 33

5.1.9 User Profile 34

5.1.10 Home 34

5.1.11 Login History 35

5.1.12 After Logout 35

**INTRODUCTION**

Crypto - is a short term for “Cryptography” and Currency refers to “The money currently in use”. Cryptocurrency is an electronic money, a computer technology which is used for hiding information or identities of an individual. In recent years, Cryptocurrencies have attracted a variety of interest from investors, entrepreneurs, regulators and the overall public. There are public discussions of cryptocurrencies worldwide because of frequent changes in their prices from time to time. Lot of people are investing in different cryptocurrencies these days and the return of investment definitely makes it worth it. Cryptocurrencies and behavioral financial and economics.

We can say that we have successfully developed a cryptocurrency tracker that displays the value of top cryptocurrencies in real time. There are lots of crypto currency out there and tracking all of it is hectic with this extension we can have tabular form at one place making it easier to track prices. As the data is tabular and consist of all time high, low and also real time data it is easier to plot graphs and deduce the future trends and also theories about the market. We can test our hypothesis with models and some machine learning code. We can build algorithms to predict the future trends, whether it will go up or down. crypto currency is in everyone's mind and many have started investing and trading in crypto market with this extension it will be easier to track prices and hence improve profitability of individual as they do not have to go browsing for prices in different websites

**1.1Motivation:**

The idea of tracking cryptocurrency prices and improving profitability of individuals as they do not have to go browsing for prices on different websites and learn about market ups and downs which makes people more motivated in the cryptocurrency field.

**1.2Proposed System:**

There are numerous Apps in the markets now to track & keep a record of our crypto. Applications like Wazirx, CoinDex,CoinSwitch Kuber,Unocoin, Bitbns ,Buy Coin and so forth. An extensive part of these applications screens our crypto related adjustments and learns the market up and down by that..

**1.3 Related Work:**

The mobile applications that are available in the market are very useful to the smartphone users and make their life easy. The cryptocurrency tracker is also one of those applications which do not waste valuable time. As there are many similar applications available today we added some innovative features to make our application unique, easy to use and efficient.

**Chapter2**

**REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION**

**2.1 Software Requirements**

* + - Android Operating System on the Smartphone.
    - Android studio platform.
    - The device should have internet connectivity.

**2.2 Hardware Requirements**

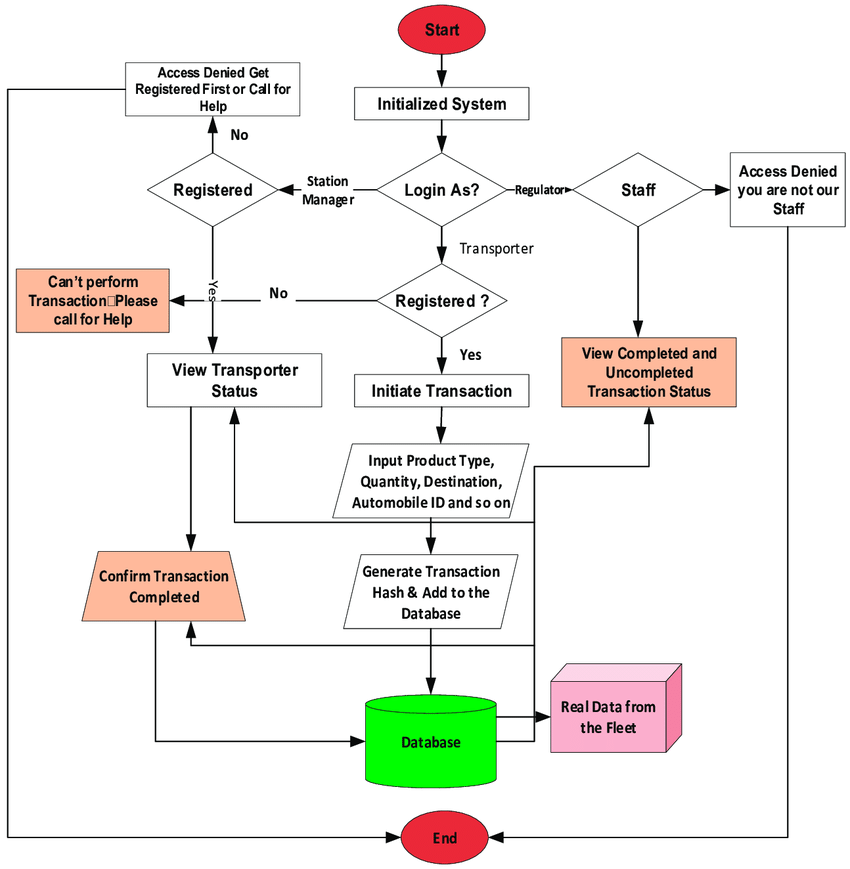
* + - 8GB RAM
    - 500GB Hard Disk
    - I5 Processor

**Chapter 3**

**SYSTEM DESIGN**

**3.1 Detailed design**

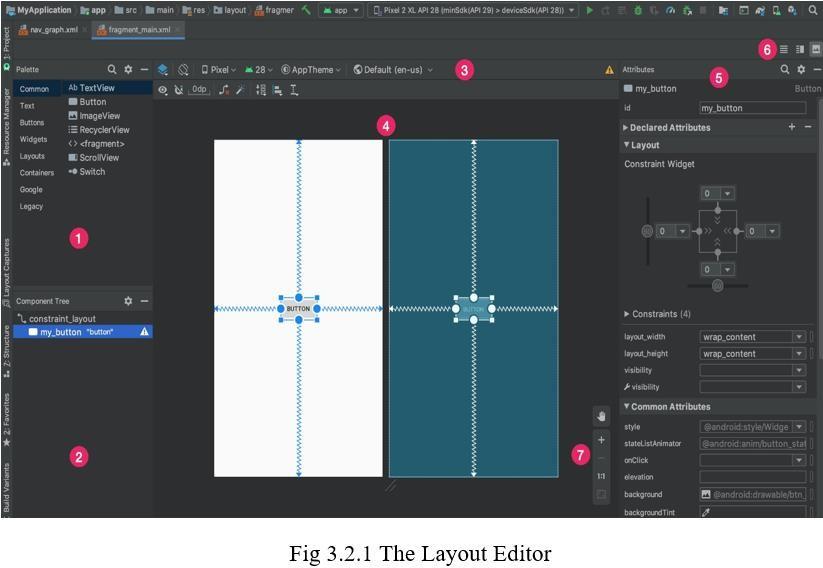
**flow diagram:**



**3.2Application design**

This application like most of the applications will have user login screen and option for registration. The user must register in this application when he/she is using for first time. However, the user who is already registered can login to the application using his/her login credentials that are created by the user at the time of registration.

## 3.2.1Introduction to the Layout Editor



* + - 1. Palette: Contains various views and view groups that you can drag into your layout.
      2. Component Tree: Shows the hierarchy of components in your layout.
      3. Toolbar: Click these buttons to configure your layout appearance in the editor and change layout attributes.
      4. Design editor: Edit your layout in Design view, Blueprint view, or both.
      5. Attributes: Controls for the selected view's attributes.
      6. View mode: View your layout in either Code , Design , or Split  modes. Split mode shows both the Code and Design windows at the same time.
      7. Zoom and pan controls: Control the preview size and position within the editor.
      8. When you open an XML layout file, the design editor opens by default, as shown in figure 1. To edit the layout XML in the text editor, click the Code  button in the top-right corner of the window. Note that the Palette, Component Tree, and Attributes windows are not available while editing your layout in Code view.

## Change the Preview Appearance

The buttons in the top row of the design editor enable you to configure the appearance of your layout in the editor. Corresponding to the numbers in figure 2, the buttons available are as follows:



1.Design and blueprint: Select how you'd like to view your layout in the editor. Choose Design to see a rendered preview of your layout. Choose Blueprint to see only outlines for each view. Choose Design + Blueprint to see both views side-by-side. You can also press B to cycle through these view types.

2.Screen orientation and layout variants: Choose between landscape and portrait screen orientation, or choose other screen modes for which your app provides alternative layouts, such as night mode. This menu also contains commands for creating a new layout variant. You can also press O to change orientation

3.Device type and size: Select the device type (phone/tablet, Android TV, or Wear OS) and screen configuration (size and density). You can select from several pre-configured device types and your own AVD definitions, or you can create a new AVD by selecting Add Device Definition from the list. You can resize the device size by dragging the bottom-right corner of the layout. You can also press D to cycle through the device list.

4.API version: Select the version of Android on which to preview your layout.

5.App theme: Select which UI theme to apply to the preview. Note that this works only for supported layout styles, so many themes in this list result in an error.

6.Language: Select the language to show for your UI strings. This list displays only the languages available in your string resources. If you'd like to edit your translations, click Edit Translations from the drop-down menu. For more information on working with translations, see Localize the UI with Translations Editor.

**3.1.4 Create a new Layout**

When adding a new layout for your app, first create a default layout file in your project's default layout/ directory so that it applies to all device configurations. Once you have a default layout, you can create layout variations for specific device configurations, such as for large screens.

You can create a new layout in one of the following ways:

Use Android Studio’s main menu

1. In the Project window, click the module in which you want to add a layout.

2. In the main menu, select File > New > XML > Layout XML File.

3. In the dialog that appears, provide the file name, the root layout tag, and the source set in which the layout belongs.

4. Click Finish to create the layout.

Use the Projected View

1. Choose the Project view from within the Project window.

2. Right-click the layout directory where you'd like to add the layout.

3. In the context menu that appears, click New > Layout Resource File. Use the Android view

1. Choose the Android view from within the Project window.

2. Right-click the layout folder.

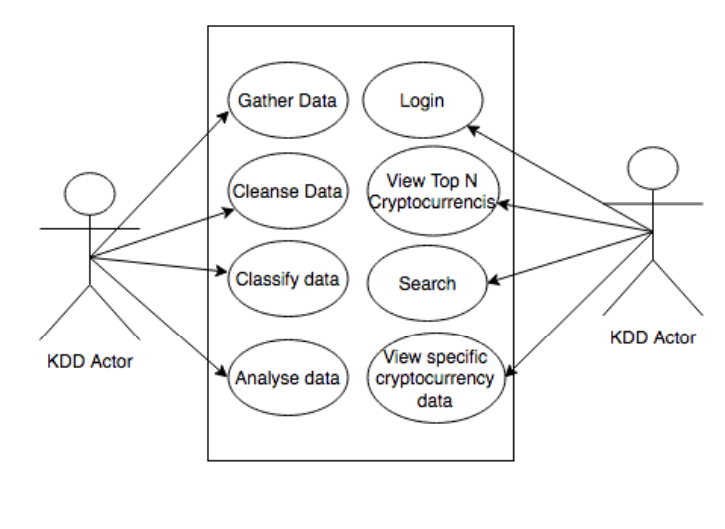
3. In the context menu that appears, select New > Layout Resource File.

Use the Resource Manager

1. In the Resource Manager, select the Layout tab.

2. Click the + button, and then click Layout Resource File.

**3.3 Use Case diagram:**



**Chapter 4**

**Implementation And Testing**

**4.1 Introduction to Programming languages ,IDES,Tools and technologies**

**4.1.1 Java programming languages**

Java is a general-purpose, class-based, object-oriented programming language designed for having lesser implementation dependencies. It is a computing platform for application development. Java is fast, secure, and reliable, therefore. It is widely used for developing Java applications in laptops, data centers, game consoles, scientific supercomputers, cell phones, etc.

Java Platform is a collection of programs that help programmers to develop and run Java programming applications efficiently. It includes an execution engine, a compiler, and a set of libraries in it. It is a set of computer software and specifications. James Gosling developed the Java platform at Sun Microsystems, and the Oracle Corporation later acquired it

**Here are some important Java applications:**

* It is used for developing Android Apps
* Helps you to create Enterprise Software
* Wide range of Mobile java Applications
* Scientific Computing Applications

**History of java programming language**

Here are the important landmarks of java programming language

* The Java language was initially called OAK.
* Originally, it was developed for handling portable devices and set-top boxes. Oak was a massive failure.
* In 1995, Sun changed the name to "Java" and modified the language to take advantage of the burgeoning www (World Wide Web) development business.
* Later, in 2009, Oracle Corporation acquired Sun Microsystems and took ownership of three key Sun software assets: Java, MySQL, and Solaris.

**4.1.2 Java Development Kit (JDK)**

JDK is a software development environment used for making applets and Java applications. The full form of JDK is Java Development Kit. Java developers can use it on Windows, macOS, Solaris, and Linux. JDK helps them to code and run Java programs. It is possible to install more than one JDK version on the same computer.

**Here are the main reasons for using JDK:**

JDK contains tools required to write Java programs and JRE to execute them. It includes a compiler, Java application launcher, Applet viewer, etc.

Compiler converts code written in Java into byte code.

Java application launcher opens a JRE, loads the necessary class, and executes its main method.

**4.1.3 Android System**

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA . On top of IntelliJ's powerful code editor and developer tools, Android Studio offers even more features that enhance your productivity when building Android apps, such as:

• Gradle-based build support

• Android-specific refactoring and quick fixes

• Lint tools to catch performance, usability, version compatibility and other problems

• Pro Guard integration and app-signing capabilities

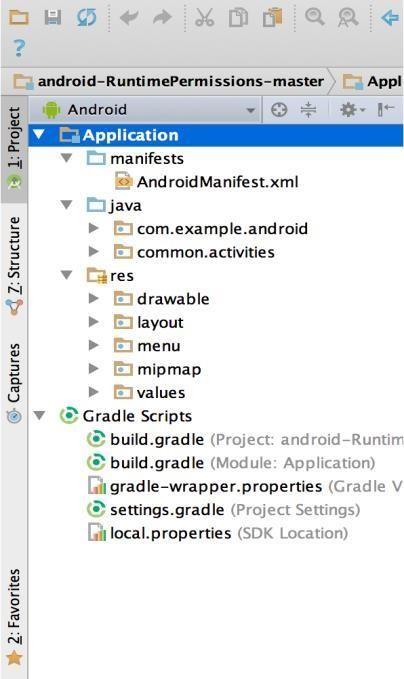
• Template-based wizards to create common Android designs and components

• A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations[18]

• Support for building Android Wear apps

• Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine[19]

• Android Virtual Device (Emulator) to run and debug apps in the Android studio.

**Project Structure** 

**Fig 4.1.3.1 The Project Files in Android View**

Each project in Android Studio contains one or more modules with source code files and resource files. Types of modules include:

* Android app modules
* Library modules
* Google App Engine modules

By default, Android Studio displays your project files in the Android project view, as shown in figure 1. This view is organized by modules to provide quick access to your project's key source files.

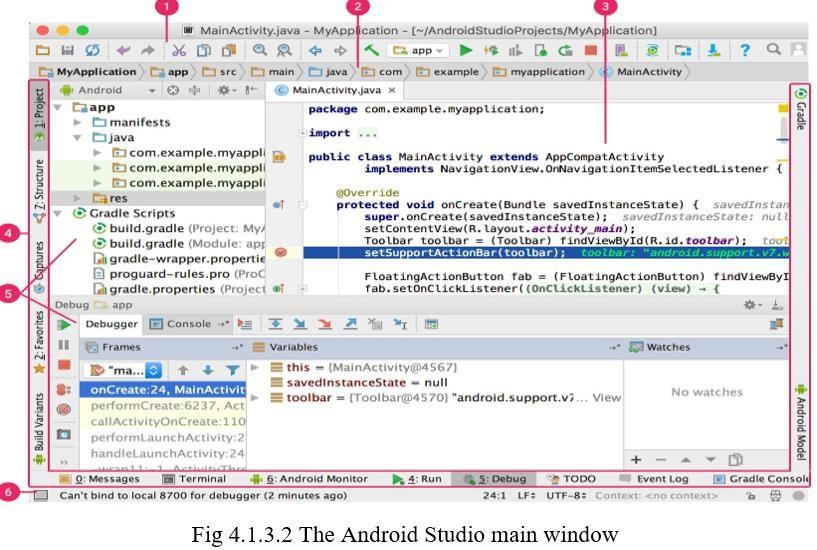
All the build files are visible at the top level under Gradle Scripts and each app module contains the following folders:

* manifests: Contains the AndroidManifest.xml file.
* java: Contains the Java source code files, including JUnit test code.
* res: Contains all non-code resources, such as XML layouts, UI strings, and bitmap images.

The Android project structure on disk differs from this flattened representation. To see the actual file structure of the project, select Project from the Project dropdown (in figure 1, it's showing as Android).

You can also customize the view of the project files to focus on specific aspects of your app development. For example, selecting the Problems view of your project displays links to the source files containing any recognized coding and syntax errors, such as a missing XML element closing tag in a layout file.

**The User Interface**



**The Android Studio main window is made up of several logical areas identified in figure:**

1. The toolbar lets you carry out a wide range of actions, including running your app and launching Android tools.

2. The navigation bar helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the Project window.

3. The editor window is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.

4. The tool window bar runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.

5. The tool windows give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.

6. The status bar displays the status of your project and the IDE itself, as well as any warnings or messages.

7. The toolbar lets you carry out a wide range of actions, including running your app and launching Android tools.

8. The navigation bar helps you navigate through your project and open files for editing. It provides a more compact view of the structure visible in the Project window.

9. The editor window is where you create and modify code. Depending on the current file type, the editor can change. For example, when viewing a layout file, the editor displays the Layout Editor.

10. The tool window bar runs around the outside of the IDE window and contains the buttons that allow you to expand or collapse individual tool windows.

11. The tool windows give you access to specific tasks like project management, search, version control, and more. You can expand them and collapse them.

12. The status bar displays the status of your project and the IDE itself, as well as any warnings or messages.

You can organize the main window to give yourself more screen space by hiding or moving toolbars and tool windows. You can also use keyboard shortcuts to access most IDE features.

At any time, you can search across your source code, databases, actions, elements of the user interface, and so on, by double-pressing the Shift key, or clicking the magnifying glass in the upper right-hand corner of the Android Studio window. This can be very useful if, for example, you are trying to locate a particular IDE action that you have forgotten how to trigger.

**Gradle Build System**

Android Studio uses Gradle as the foundation of the build system, with more Android-specific capabilities provided by the Android plugin for Gradle. This build system runs as an integrated tool from the Android Studio menu, and independently from the command line. You can use the features of the build system to do the following:

Customize, configure, and extend the build process.

Create multiple APKs for your app, with different features using the same project and modules. Reuse code and resources across source sets.

By employing the flexibility of Gradle, you can achieve all of this without modifying your app's core

source files. Android Studio build files are named build gradle. They are plain text files that use Groovy syntax to configure the build with elements provided by the Android plugin for Gradle. Each project has one top-level build file for the entire project and separate module-level build files for each module. When you import an existing project, Android Studio automatically generates the necessary build files.

**Debug and profile tools**

Android Studio assists you in debugging and improving the performance of your code, including inline debugging and performance analysis tools.

**Inline Debugging**

Use inline debugging to enhance your code walk-throughs in the debugger view with inline verification of references, expressions, and variable values. Inline debug information includes:

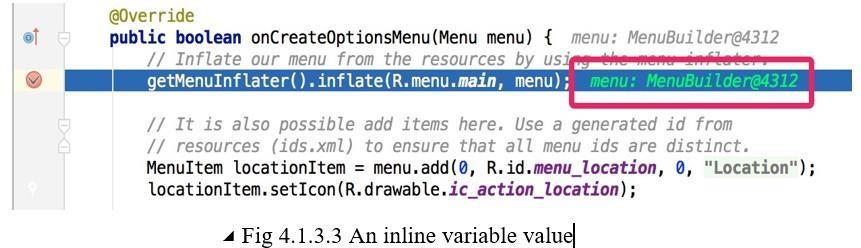
• Inline variable values

• Referring objects that reference a selected object

• Method return values

• Lambda and operator expressions

• Tooltip values



To enable inline debugging, in the Debug window, click Settings and select the checkbox for Show **Values Inline.**

Performance Profiles

Android Studio provides performance profilers so you can more easily track your app’s memory and CPU usage, find deallocated objects, locate memory leaks, optimize graphics performance, and analyze network requests.

With your app running on a device or emulator, open the **Android Profiler tab**. Annotations in Android Studio

Android Studio supports annotations for variables, parameters, and return values to help you catch bugs, such as null pointer exceptions and resource type conflicts. The Android SDK Manager packages the Support- Annotations library in the Android Support Repository for use with Android Studio. Android Studio validates the configured annotations during code inspection.

**4.2 Test Plan and Test Activities**

Android Studio is designed to make testing simple. With just a few clicks, you can set up a JUnit test that runs on the local JVM or an instrumented test that runs on a device. Of course, you can also extend your test capabilities by integrating test frameworks such as Mockito to test Android API calls in your local unit tests, and Espresso or UI Automator to exercise user interaction in your instrumented tests. You can generate Espresso tests automatically using Espresso Test Recorder.

Test types and location

The location of your test code depends on the type of test you are writing. Android Studio provides source code directories (source sets), for the following two types of tests:

Local unit tests

Located at module-name/src/test/java/.

These are tests that run on your machine's local Java Virtual Machine (JVM). Use these tests to minimize execution time when your tests have no Android framework dependencies or when you can mock the Android framework dependencies.

At runtime, these tests are executed against a modified version of android.jar where all final modifiers have been stripped off. This lets you use popular mocking libraries, like Mockito.

**Instrumented tests**

Located at module-name/src/androidTest/java/.

These are tests that run on a hardware device or emulator. These tests have access to Instrumentation APIs, give you access to information such as the Context of the app you are testing, and let you control the app under test from your test code. Use these tests when writing integration and functional UI tests to automate user interaction, or when your tests have Android dependencies that mock objects cannot satisfy.

Because instrumented tests are built into an APK (separate from your app APK), they must have their own AndroidManifest.xml file. However, Gradle automatically generates this file during the build so it is not visible in your project source set. You can add your own manifest file if necessary, such as to specify a different value for `minSdkVersion` or register run listeners just for your tests. When building your app, Gradle merges multiple manifest files into one manifest.

The Gradle build interprets these test source sets in the same manner as it does for your project's app source sets, which allows you to create tests based on build variants.

When you create a new project or add an app module, Android Studio creates the test source sets listed above and includes an example test file in each. You can see them in the Project window as shown in figure

**Instrumented tests Local unit tests**

Located at module-name/src/test/java/.

These are tests that run on your machine's local Java Virtual Machine (JVM). Use these tests to minimize execution time when your tests have no Android framework dependencies or when you can mock the Android framework dependencies.

At runtime, these tests are executed against a modified version of android.jar where all final modifiers have been stripped off. This lets you use popular mocking libraries, like Mockito.

Located at module-name/src/androidTest/java/.

These are tests that run on a hardware device or emulator. These tests have access to Instrumentation APIs, give you access to information such as the Context of the app you are testing, and let you control the app under test from your test code. Use these tests when writing integration and functional UI tests to automate user interaction, or when your tests have Android dependencies that mock objects cannot satisfy.

Because instrumented tests are built into an APK (separate from your app APK), they must have their own AndroidManifest.xml file. However, Gradle automatically generates this file during the build so it is not visible in your project source set. You can add your own manifest file if necessary, such as to specify

a different value for `minSdkVersion` or register run listeners just for your tests. When building your

app, Gradle merges multiple manifest files into one manifest.

The Gradle build interprets these test source sets in the same manner as it does for your project's app source sets, which allows you to create tests based on build variants.

When you create a new project or add an app module, Android Studio creates the test source sets listed above and includes an example test file in each. You can see them in the Project window as shown in figure

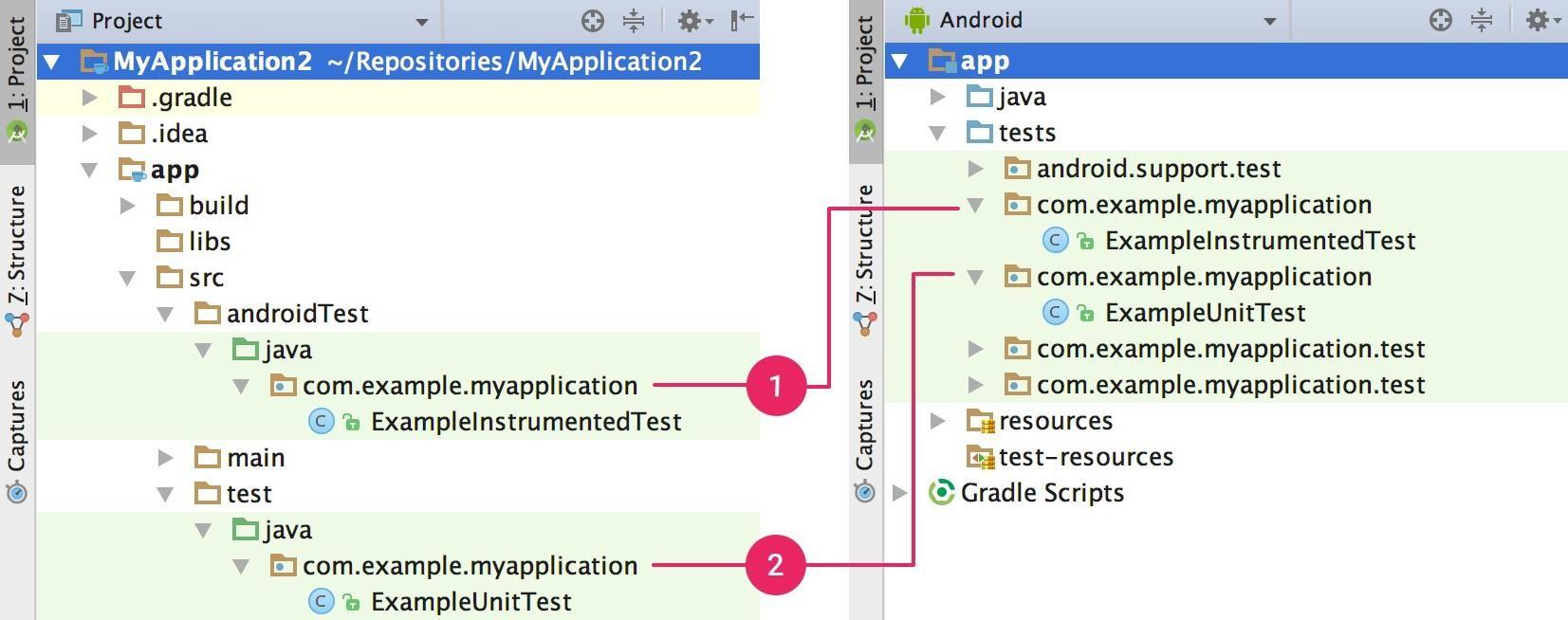


Figure4.2.1. Your project's (1) instrumented tests and (2) local JVM tests are visible in either the Project view (left)or Android view (right)

**Add a new test**

To create either a local unit test or an instrumented test, you can create a new test for a specific class or method by following these steps:

1. Open the Java file containing the code you want to test.

2. Click the class or method you want to test, then press Ctrl+Shift+T (⇧⌘T).

3. In the menu that appears, click Create New Test.

4. In the Create Test dialog, edit any fields and select any methods to generate, and then click OK.

5. In the Choose Destination Directory dialog, click the source set corresponding to the type of test you want to create: androidTest for an instrumented test or test for a local unit test. Then click OK.

Alternatively, you can create a generic Java file in the appropriate test source set as follows:

1.In the Project window on the left, click the drop-down menu and select the Project view.

2.Expand the appropriate module folder and the nested src folder. To add a local unit test, expand the test folder and the nested java folder; to add an instrumented test, expand the androidTest folder and the nested java folder.

3.Right-click on the Java package directory and select New > Java Class.

4.Name the file and then click OK. Run a test

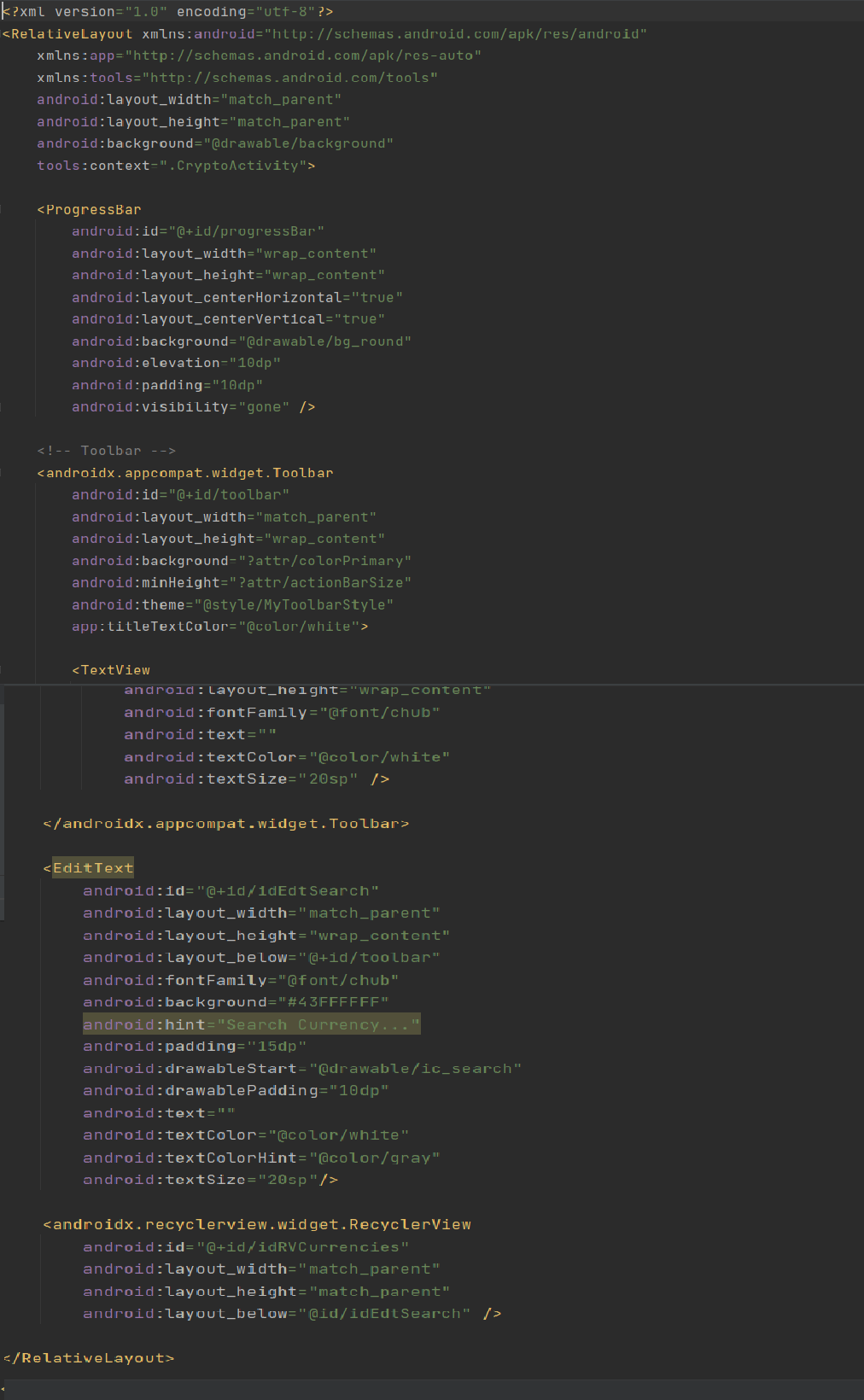
**To run a test, proceed as follows:**

* Be sure your project is synchronized with Gradle by clicking Sync Project in the toolbar.
* Run your test in one of the following ways:
* In the Project window, right-click a test and click Run .
* To run all tests, right-click on the test directory and click Run tests .
* In the Code Editor, right-click a class or method in the test file and click Run to test all methods in the class.

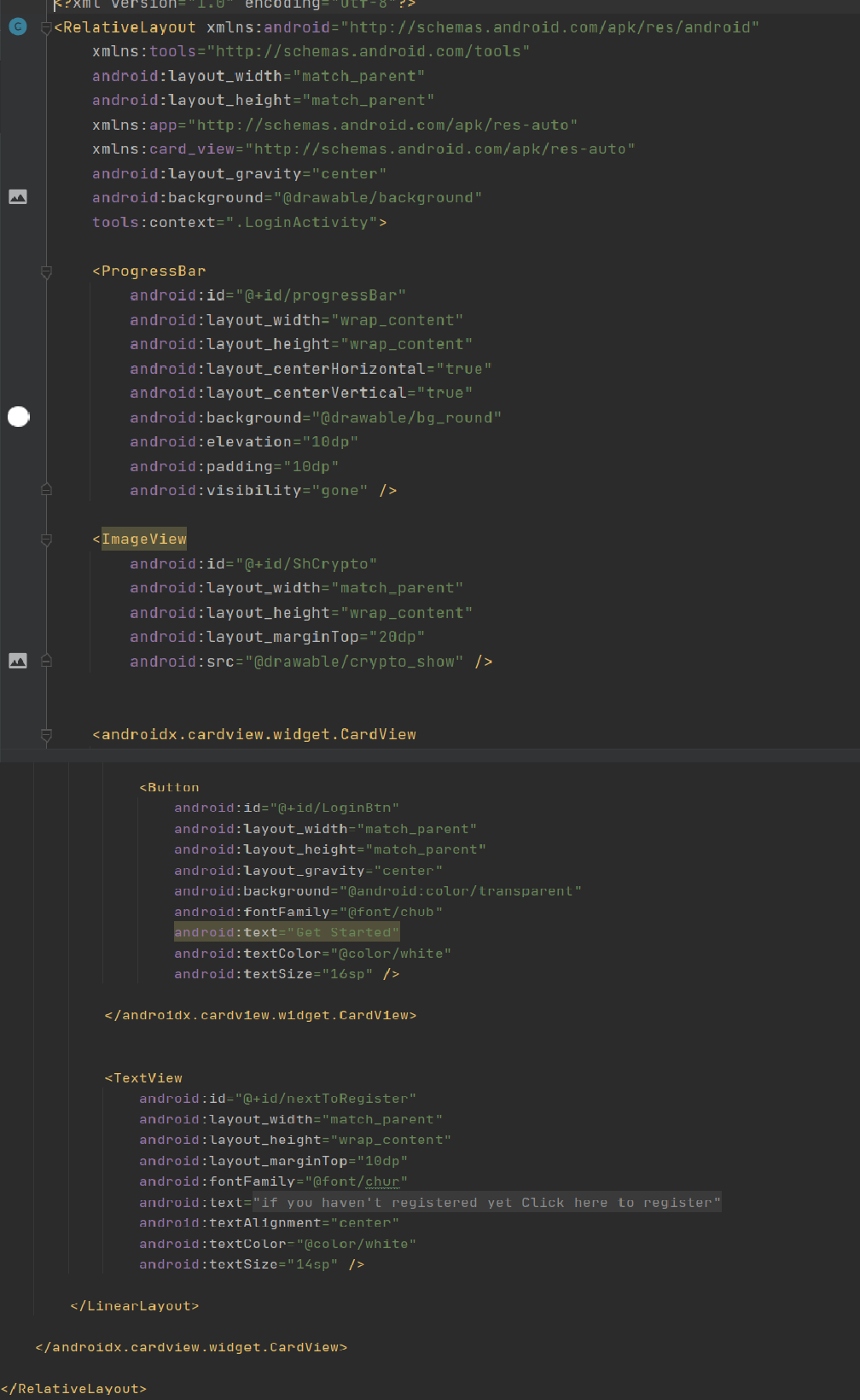
By default, your test runs using Android Studio's default run configuration. If you'd like to change some run settings such as the instrumentation runner and deployment options, you can edit the run configuration in the

**XML Code**

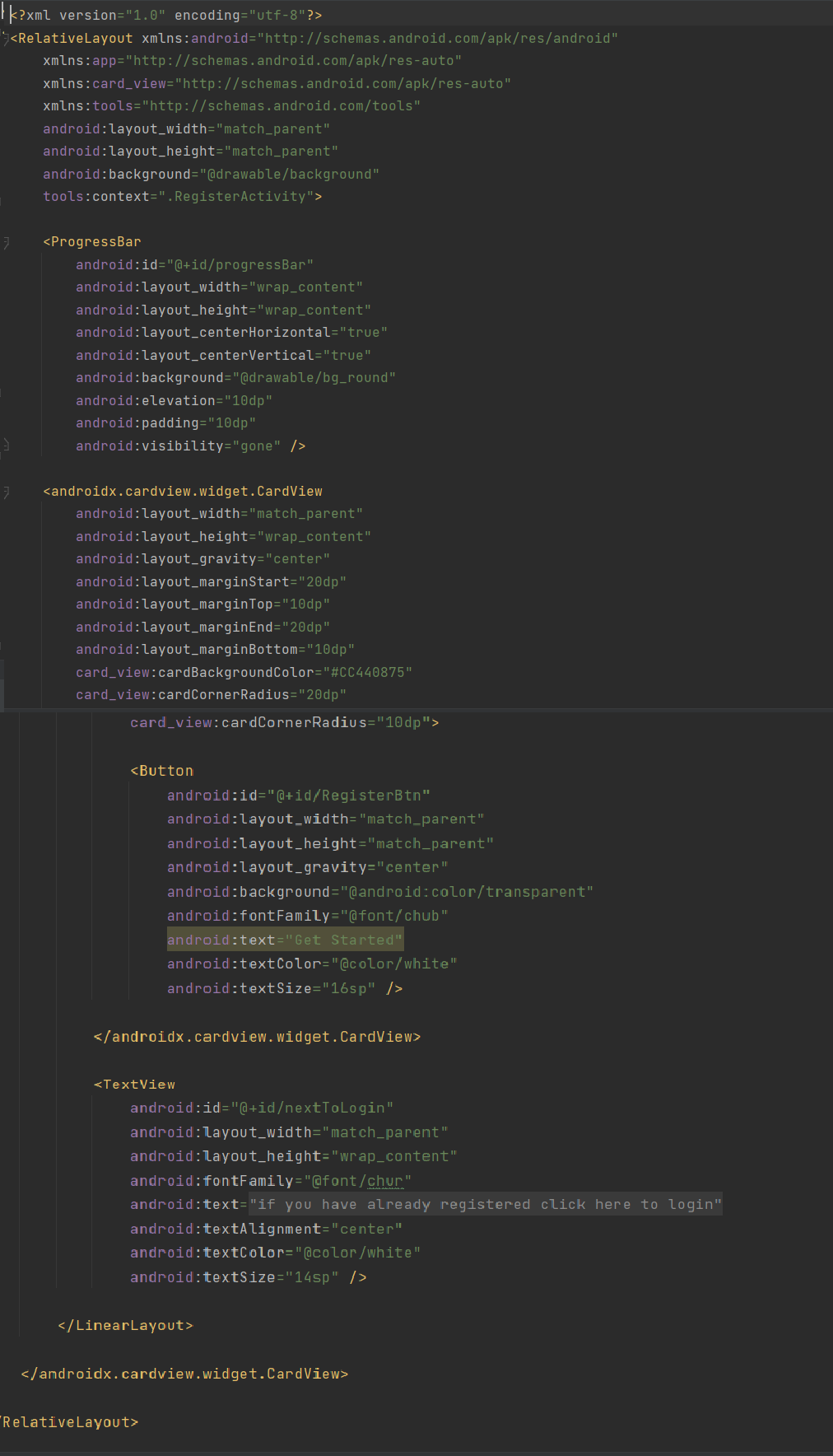
**Crypto.xml**



**Login.xml**

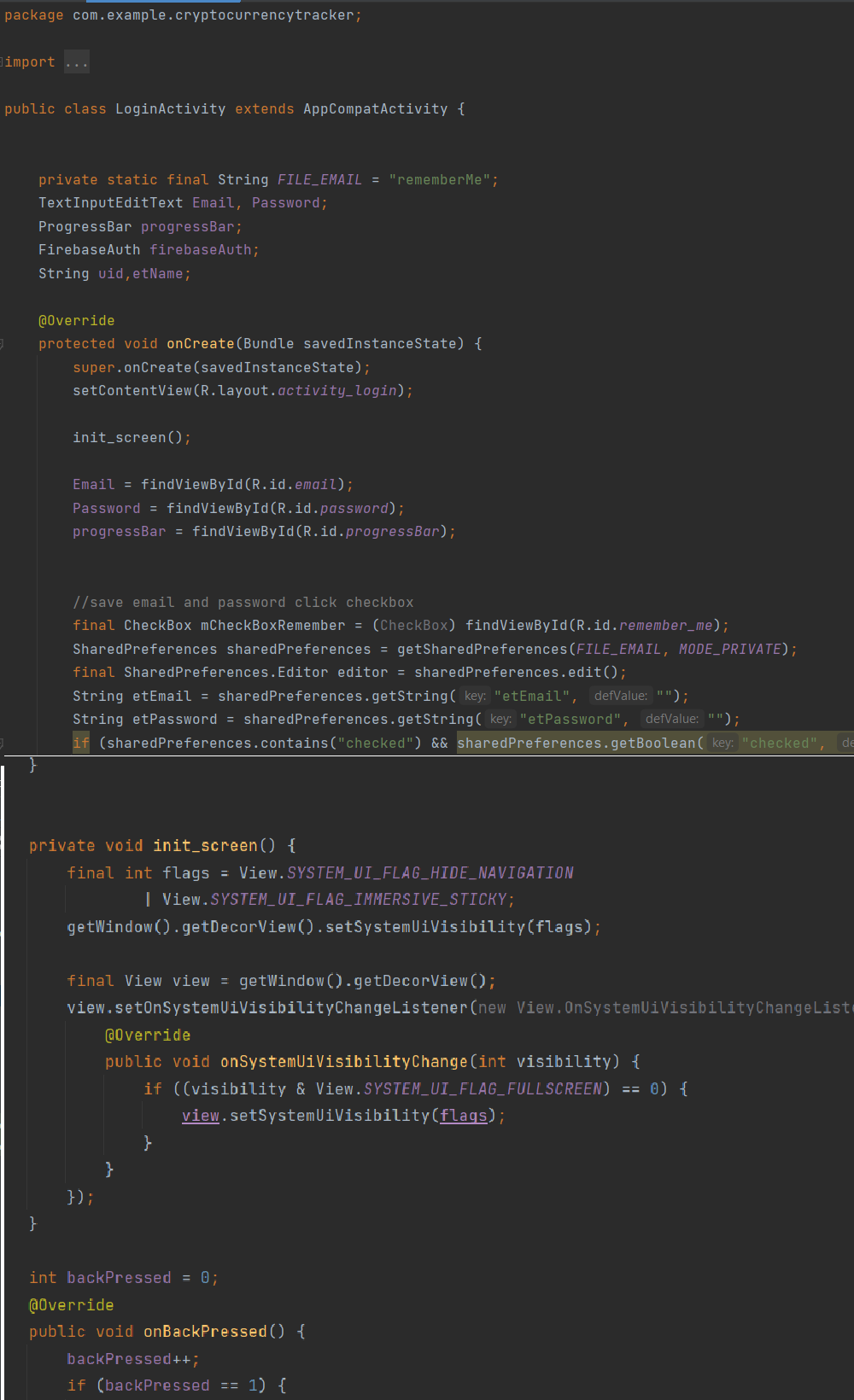


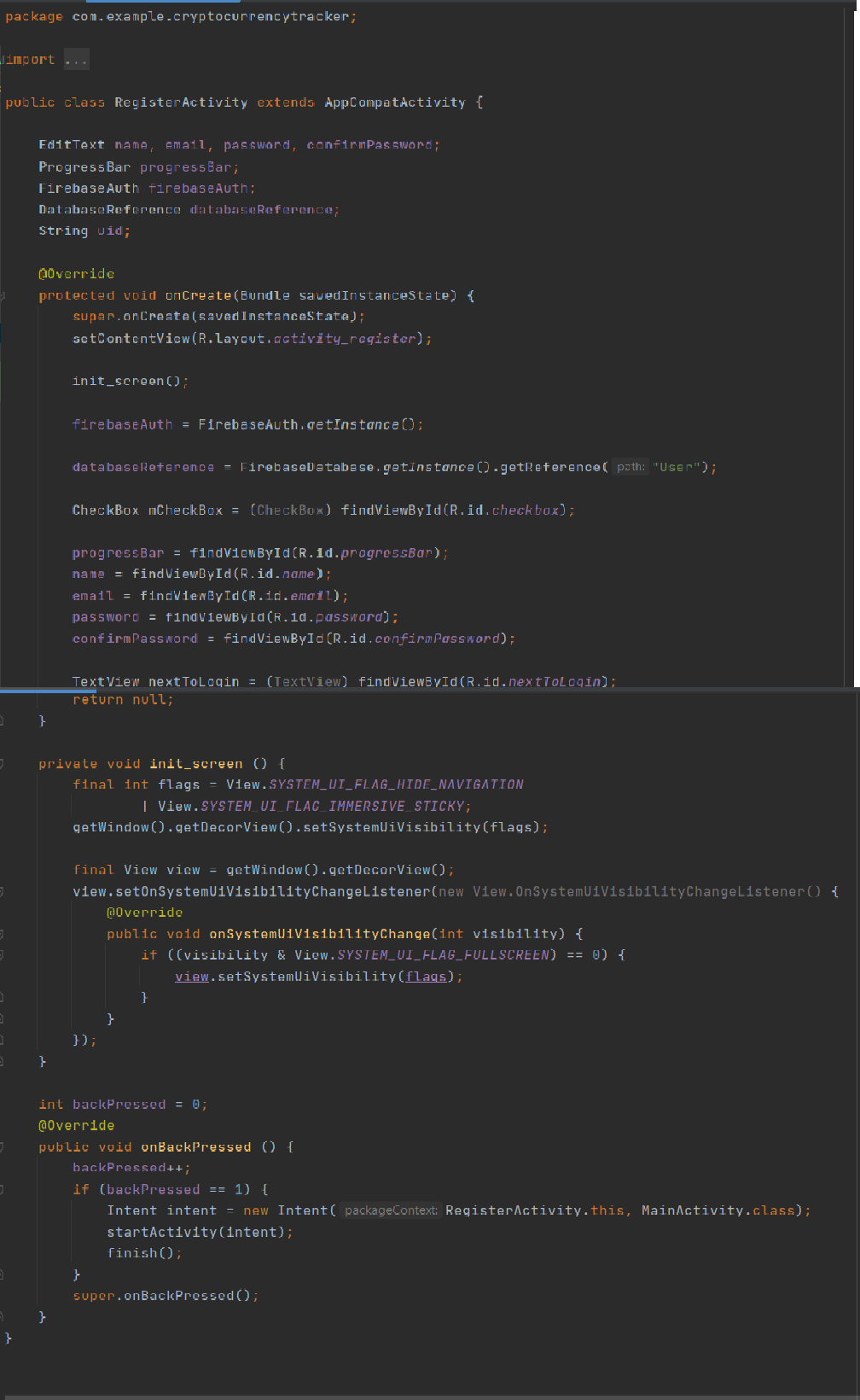
**Register.xml**



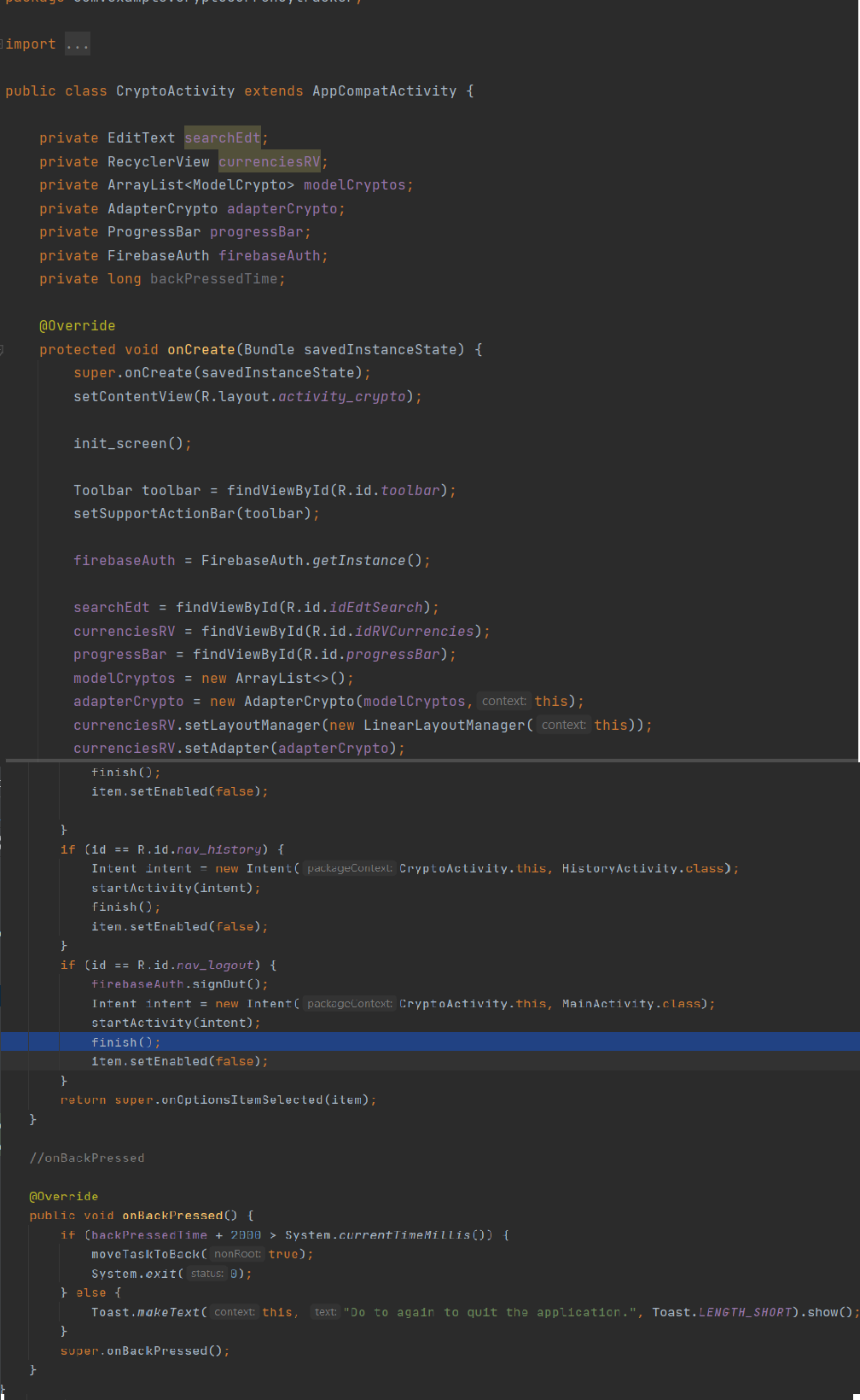
**JAVA Code**

**Login.java**



**Register.java **

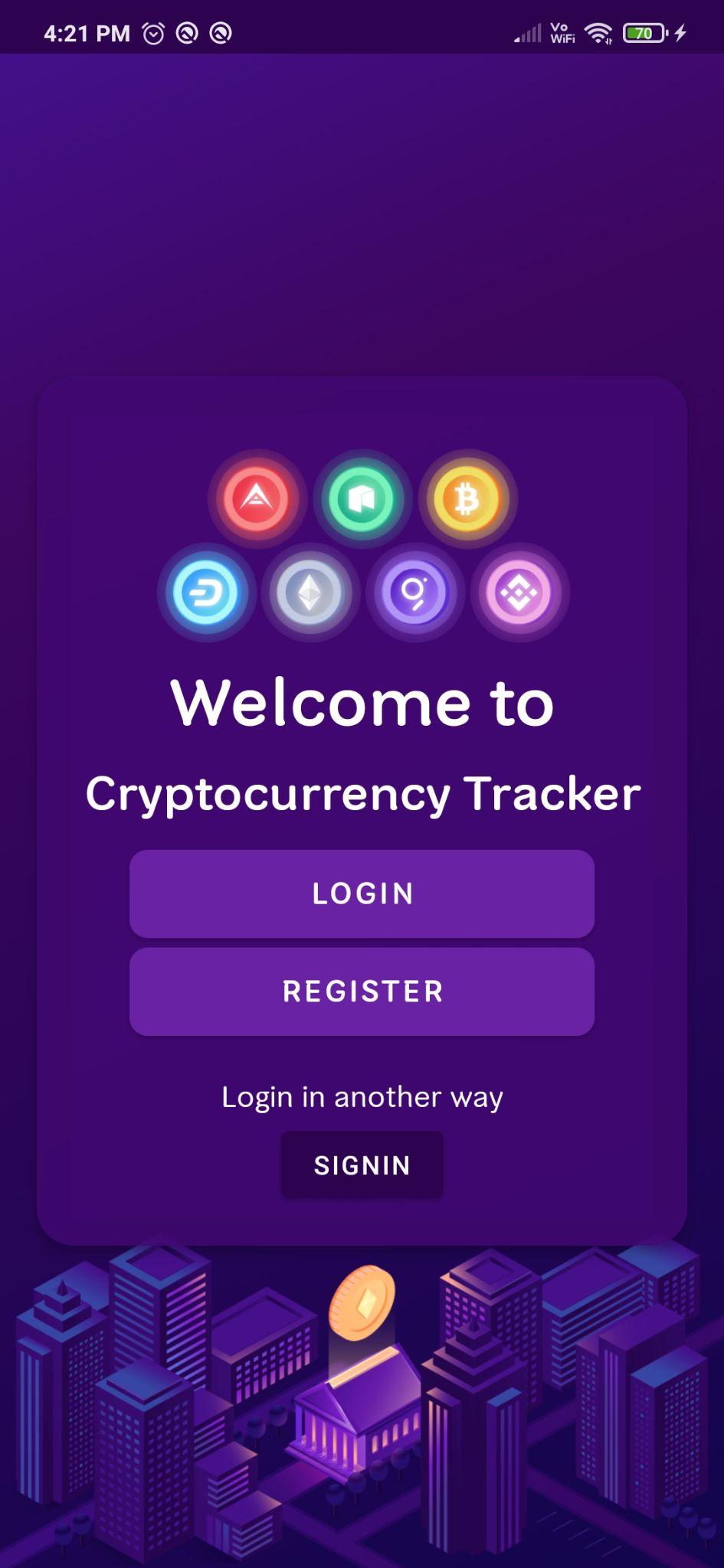
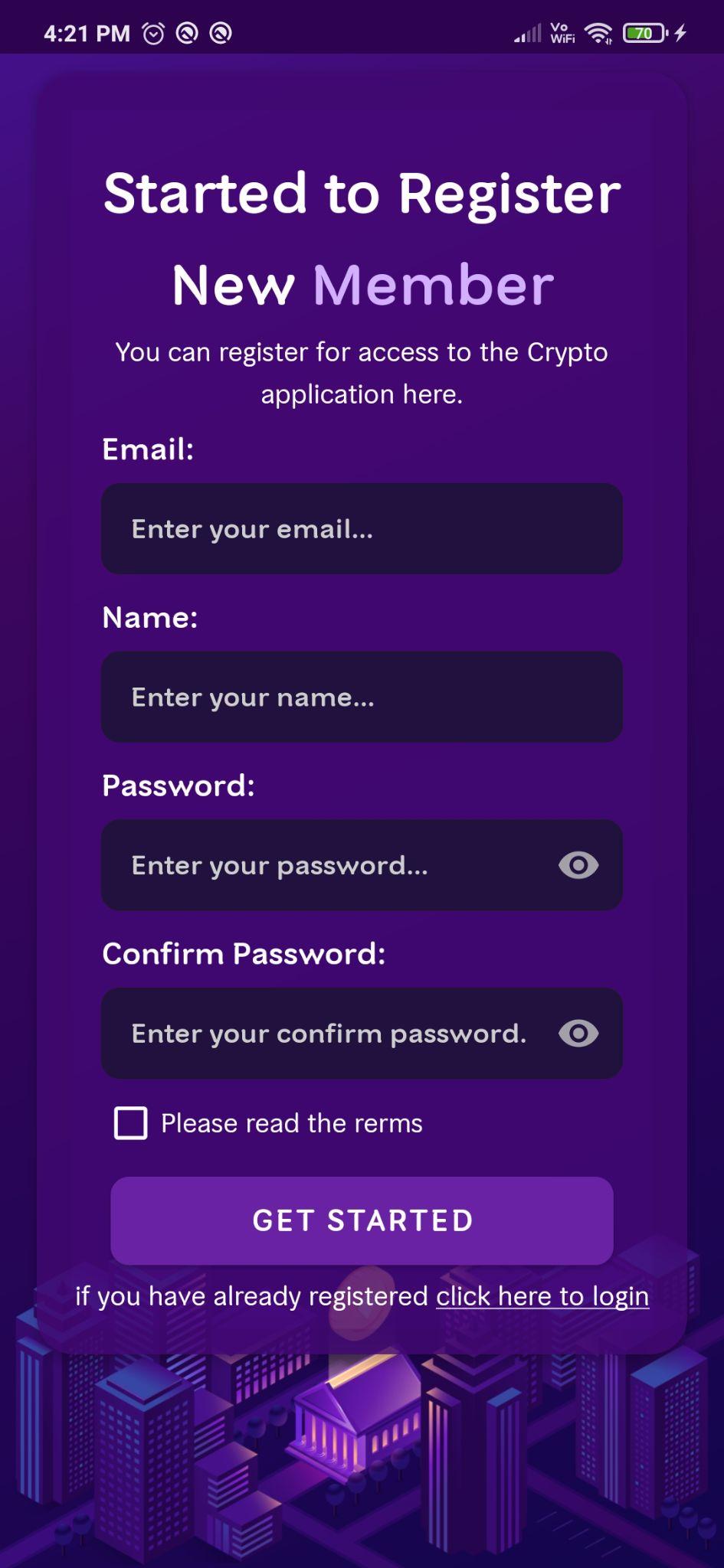
**Crypto.java**

****

**Chapter 5**

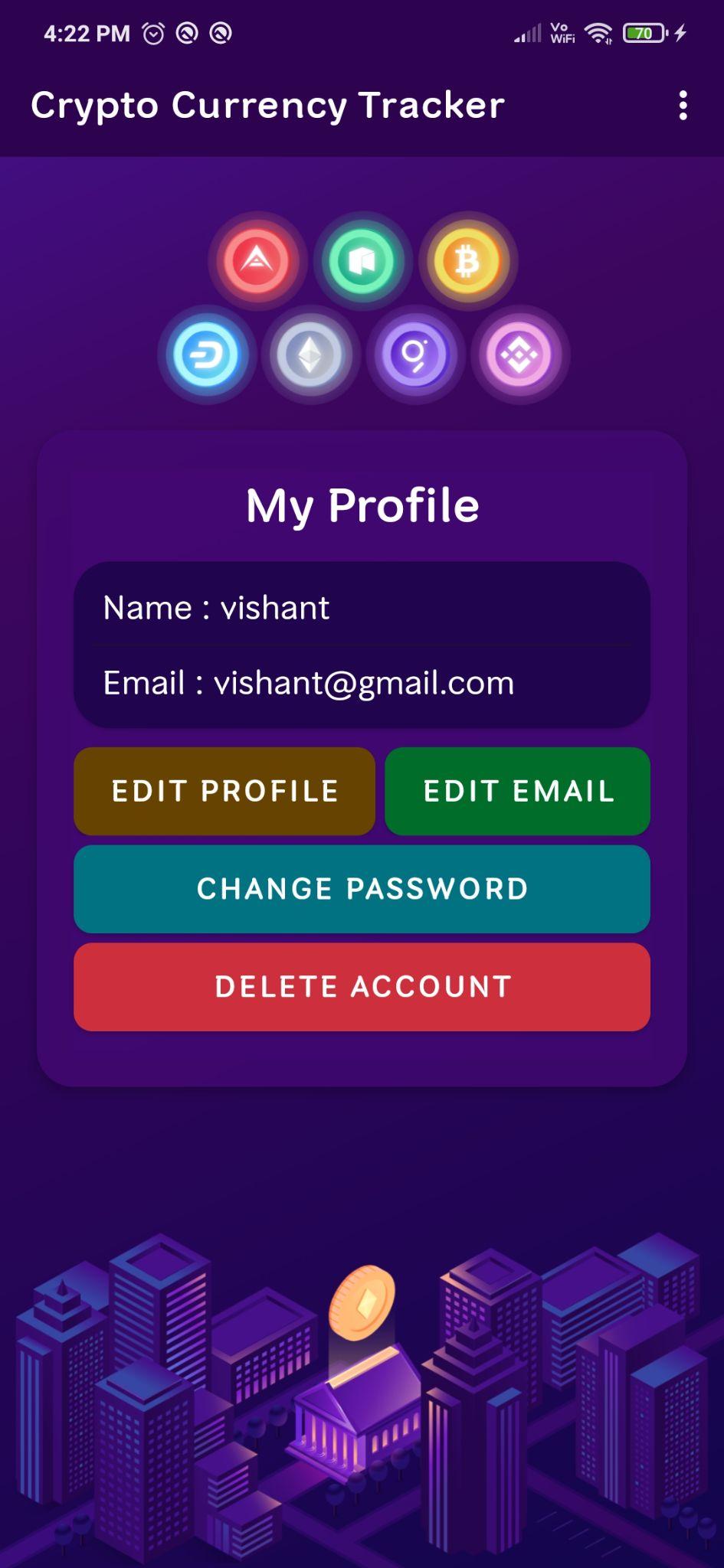
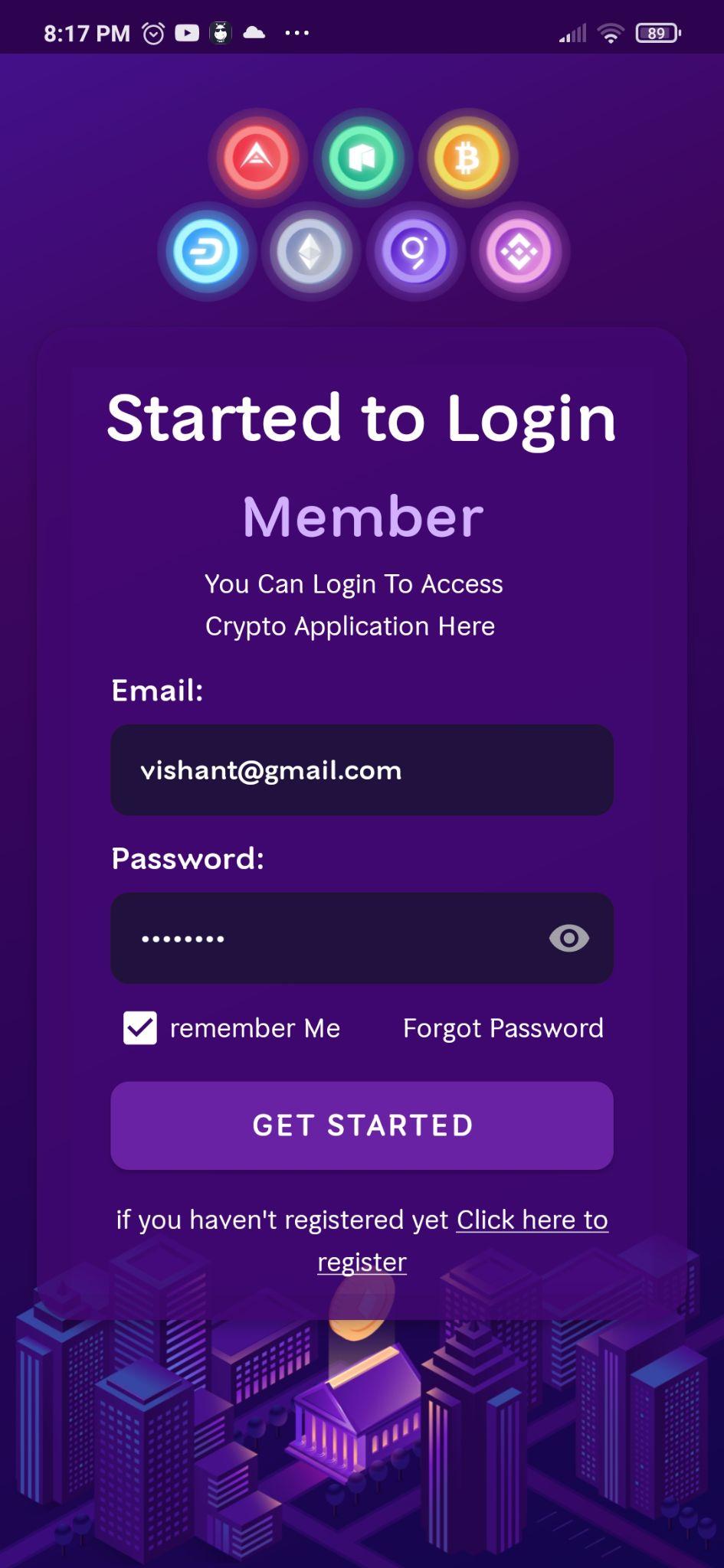
**RESULTS AND DISCUSSIONS**

**5.1Screenshots:**

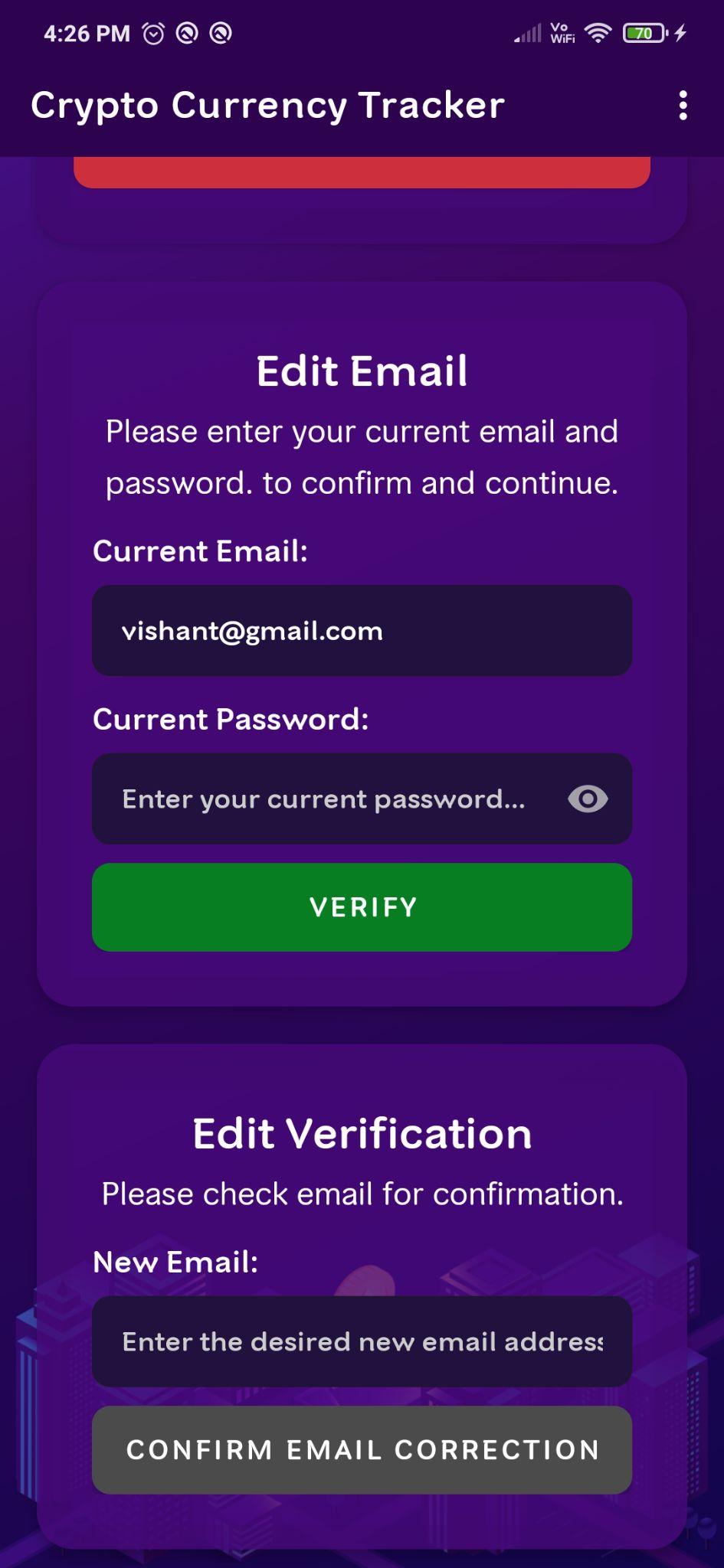
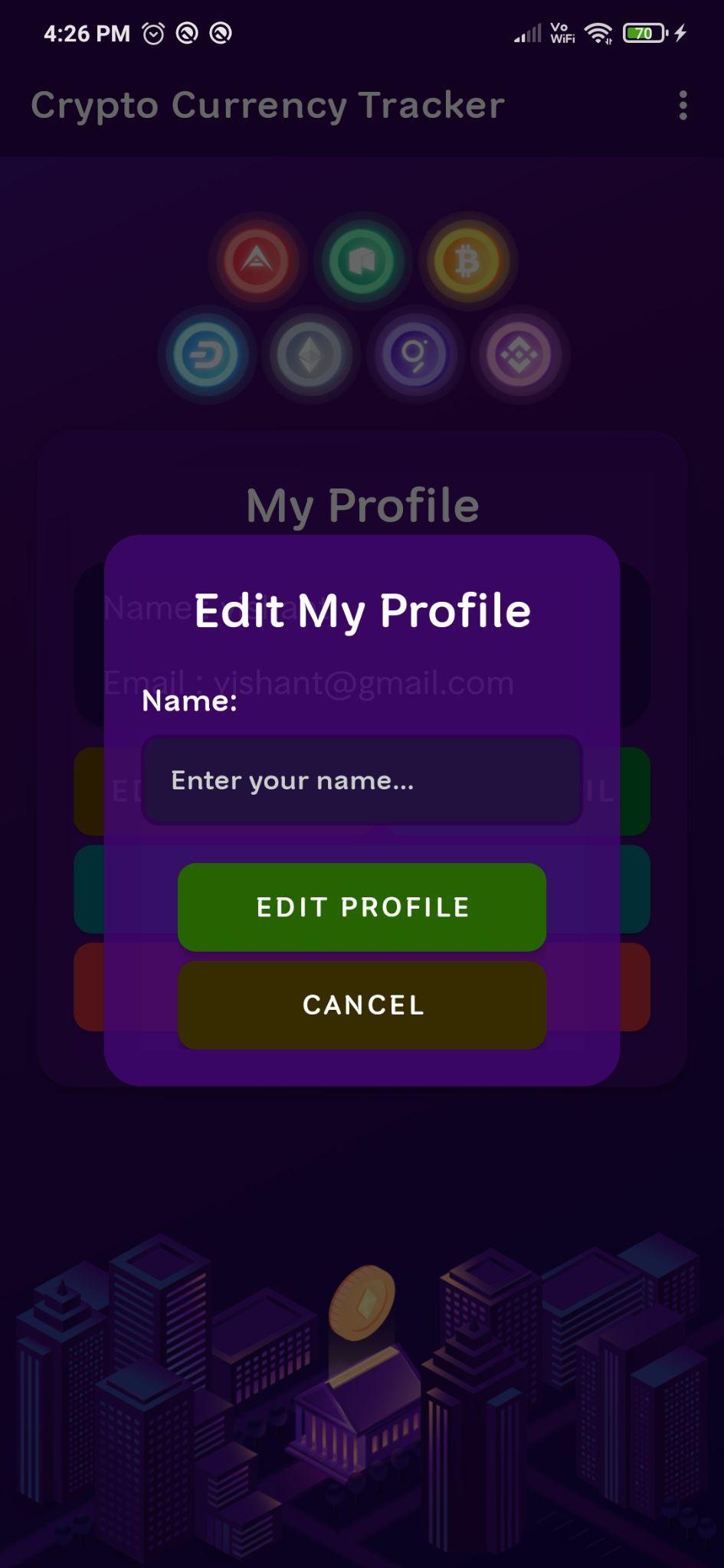
**Welcome Page**  **Register Page**

Figure 5.1.1 Figure 5.1.2



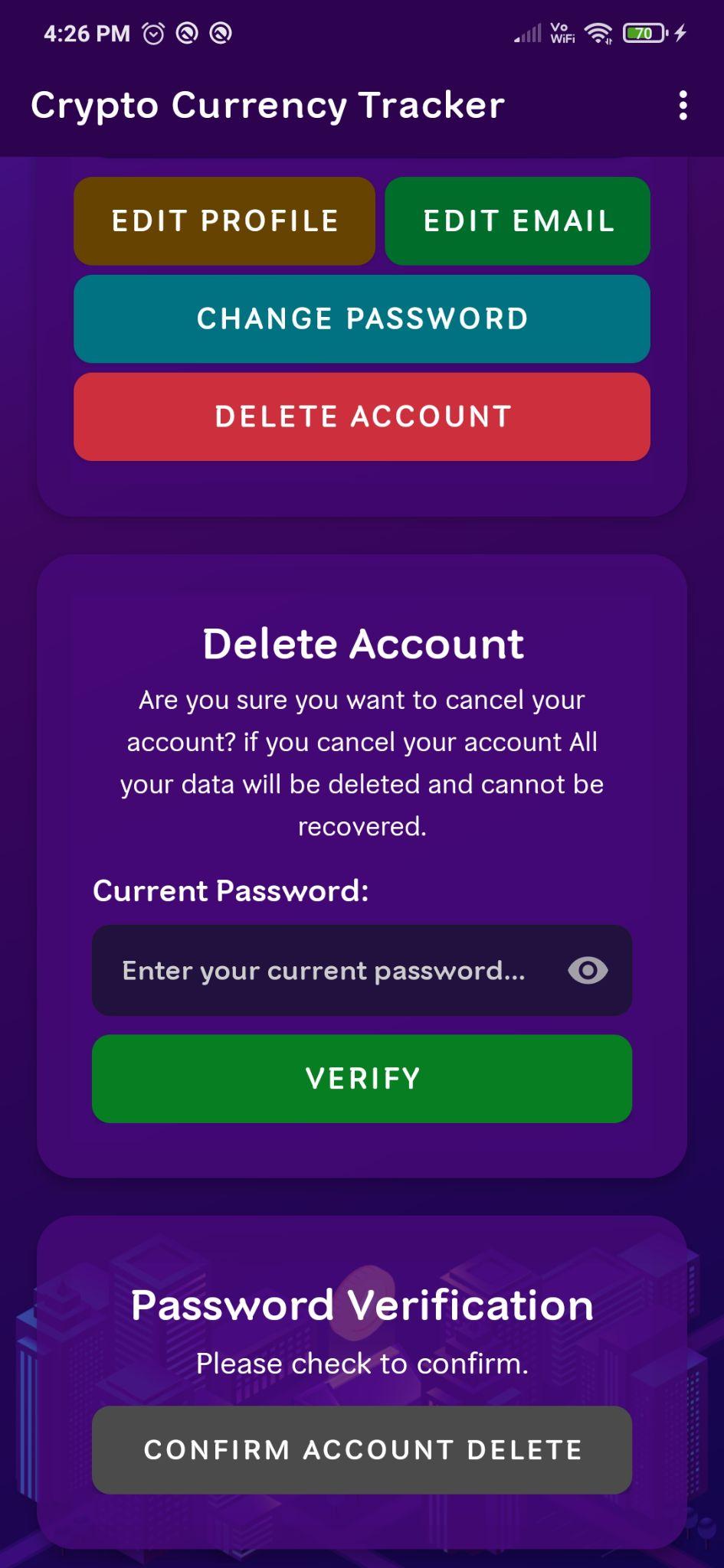
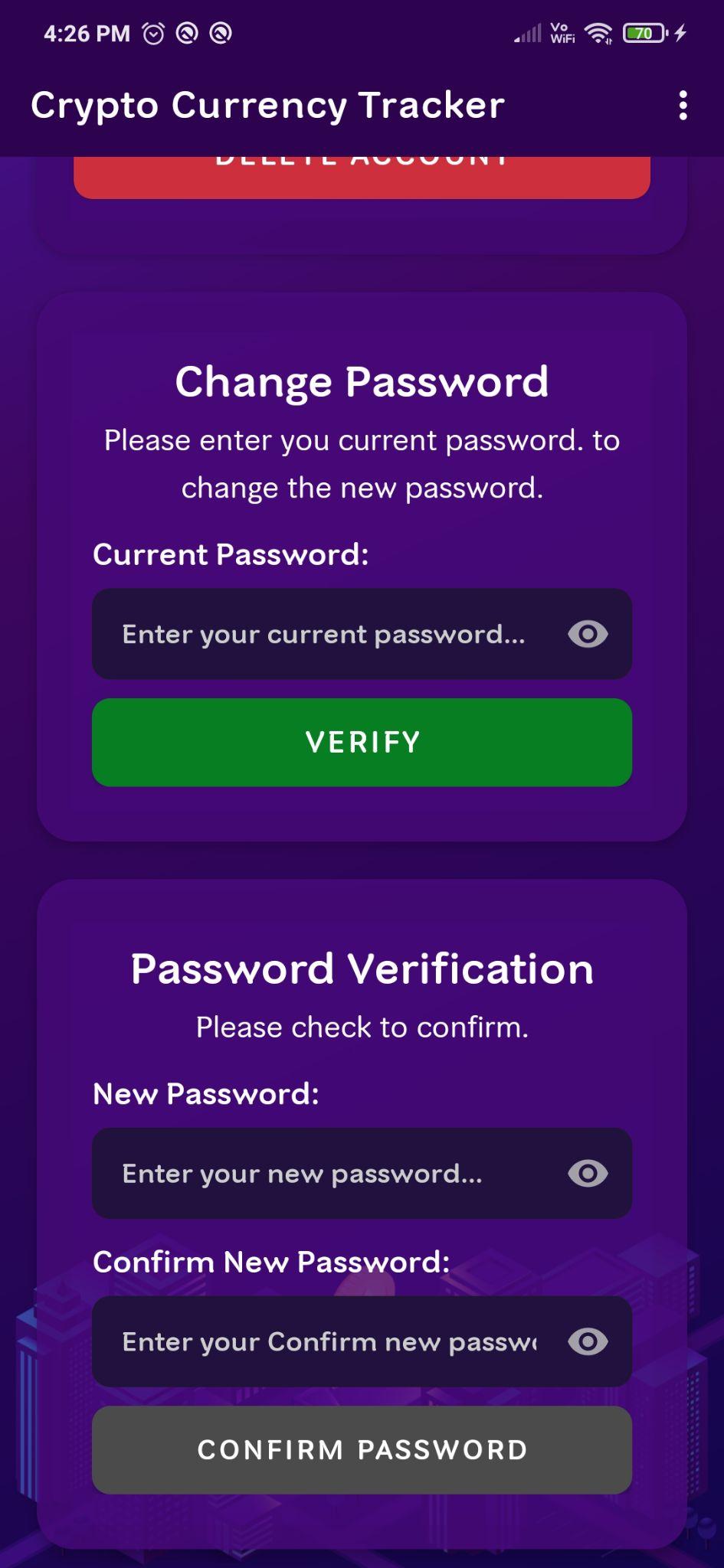
**Login**  **My Profile**

Figure 5.1.3 Figure 5.1.4



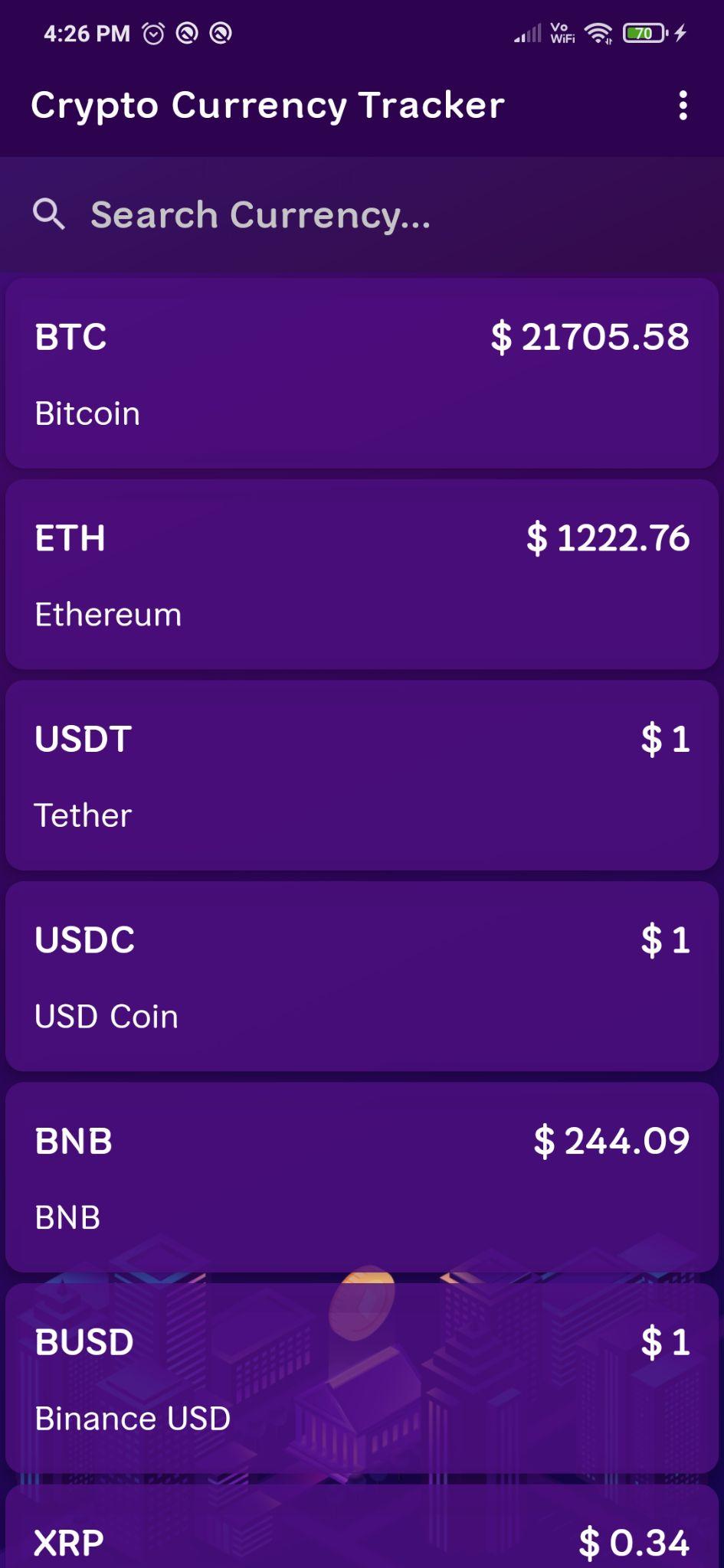
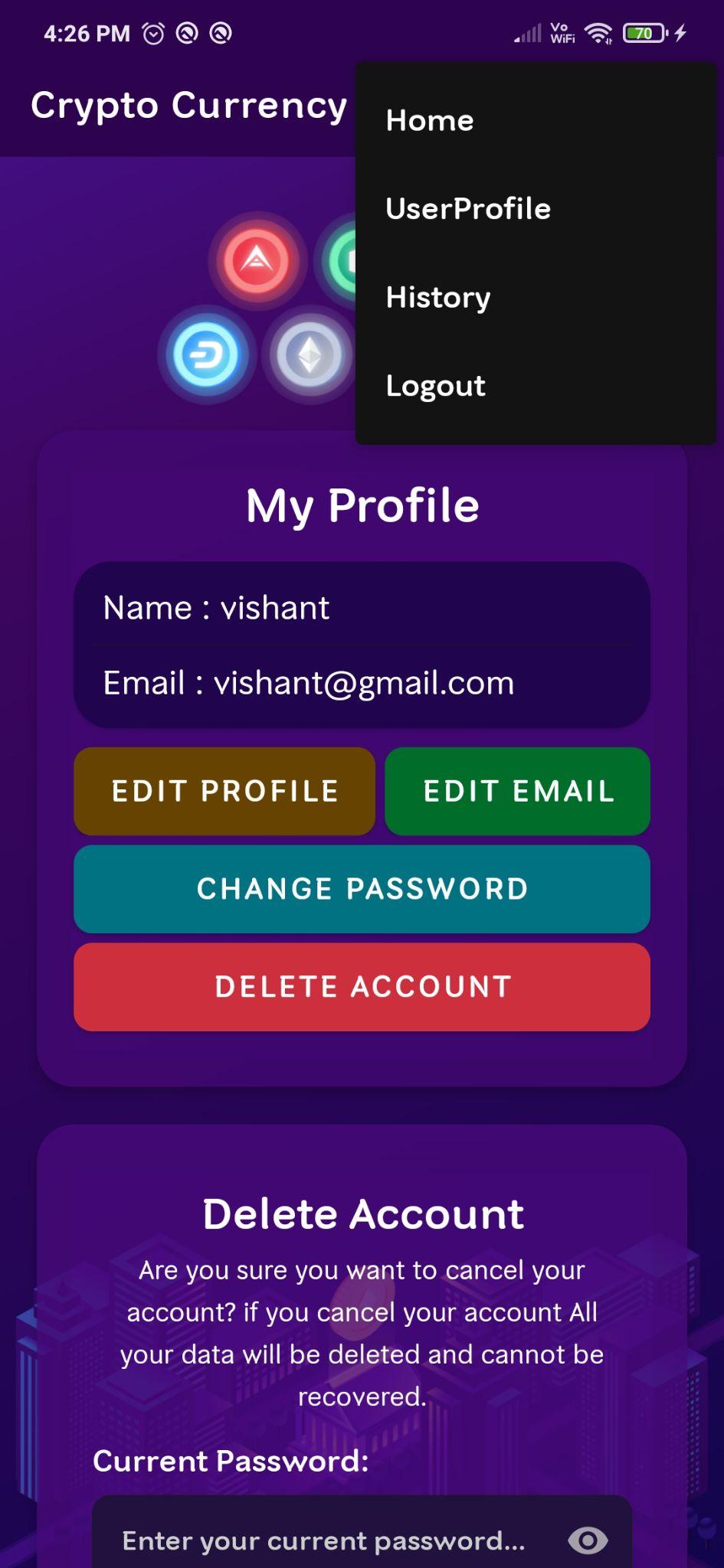
**Edit Profile** **Edit Email**

Figure 5.1.5 Figure 5.1.6



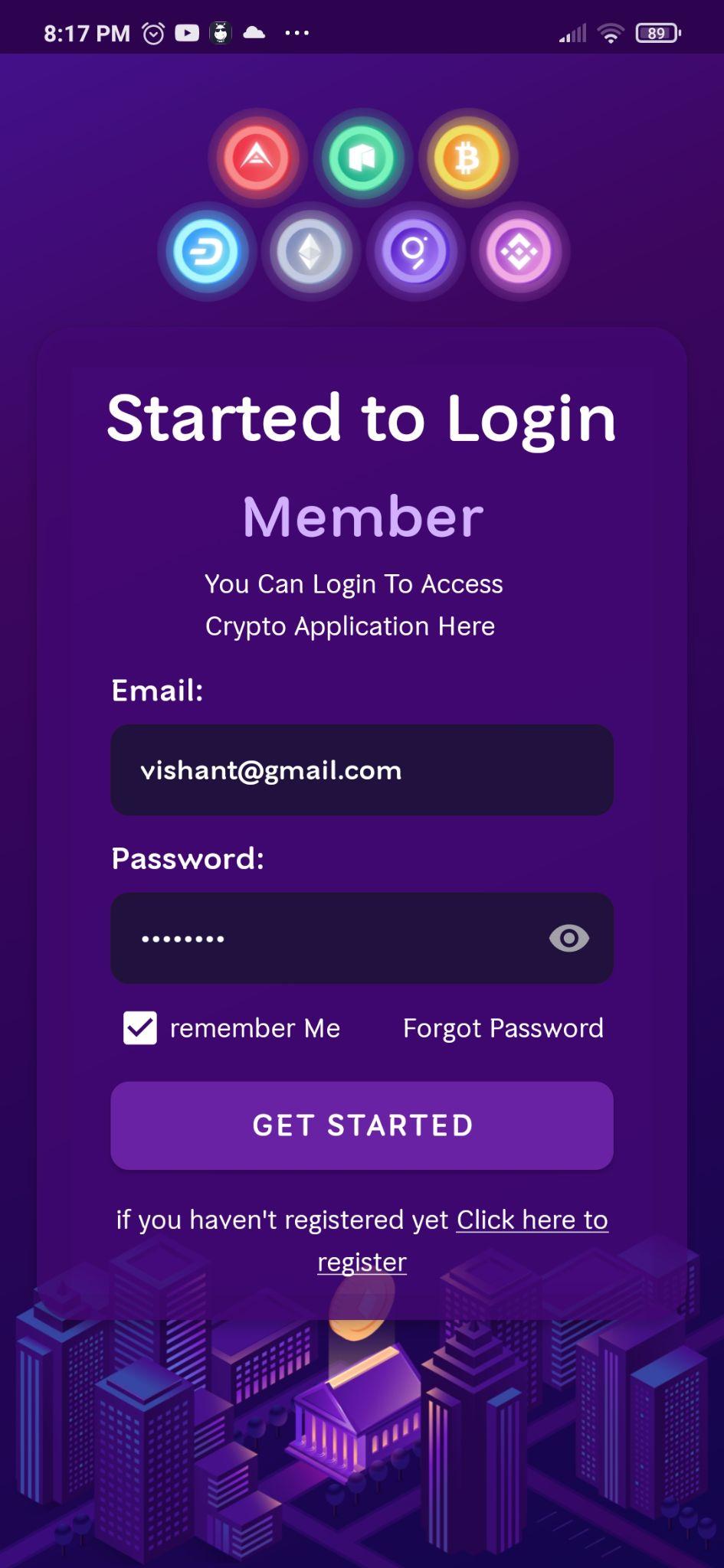
**Change Password**  **Delete Account**

Figure 5.1.7 Figure 5.1.8



**UserProfile** **Home**

Figure 5.1.9 Figure 5.1.10



**Login History**  **After Logout**

Figure 5.1.11 Figure 5.1.12

**Chapter6**

**FUTURE WORK**

Mobile domain on the other side is growing very fast and with less resources (Human Resources), Future Mobile Application Testing is directly proportional to the growth of Mobile Development and research. The Future Enhancements, the application can be allowed to support in all the android versions. History can be set to view all the details even, if the particular detail is deleted from the database.

**Conclusion**

We can finally conclude that in this project we have successfully developed a cryptocurrency tracker that displays the value of top cryptocurrencies in real time. There are lots of crypto currency out there and tracking all of it is hectic with this extension we can have tabular form at one place making it easier to track prices. As the data is tabular and consist of all time high, low and also real time data it is easier to plot graphs and deduce the future trends and also theories about the market. We can test our hypothesis with models and some machine learning code. We can build algorithms to predict the future trends, whether it will go up or down. crypto currency is in everyone's mind and many have started investing and trading in crypto market with this extension it will be easier to track prices and hence improve profitability of individual as they do not have to go browsing for prices in different websites

**Bibliography**

**Websites**

* + - [www.wikipedia.org](http://www.wikipedia.org/)
    - [https://developer.android.com](https://developer.android.com/)
    - [https://www.github.com](https://www.github.com/)