Android Development Using Java and Kotlin

PREPARED BY
Dhruv Patel (2018BTECHEEE003)

FACULTY GUIDE
Dr. Jagdish Prasad Sharma



Department of Electrical and Electronics Engineering
Institute of Engineering and Technology (IET)
JK Lakshmipat University Jaipur

May 2022

Android Development Using Java and Kotlin

Practice School-2

Submitted in partial fulfilment of the requirements

for the degree of

Bachelor of Technology in Electrical and Electronics Engineering

By:
Dhruv Patel (2018BTECHEEE003)

Faculty Guide:
Dr. Jagdish Prasad Sharma



Department of Electrical and Electronics Engineering
Institute of Engineering and Technology (IET)
JK Lakshmipat University Jaipur

May 2022

CERTIFICATE

This is to certify that the Practice School-2 project work entitled "**Android development using java and kotlin**" submitted by **Dhruv Patel (2018BTECHEEE003)**, towards the partial fulfilment of the requirements for the degree of **Bachelor of Technology in Electrical and Electronics Engineering** of JK Lakshmipat University Jaipur is the record of work carried out by them under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for Practice School-II examination.

Dr. Jagdish Prasad Sharma
Assistant Professor,
Department of Electrical and Electronics Engineering
Institute of Engineering & Technology (IET)
JK Lakshmipat University Jaipur

Dr. Devika Kataria
Associate Professor & Head,
Department of Electrical and Electronics Engineering
Institute of Engineering & Technology (IET)
JK Lakshmipat University Jaipur

Date of Submission 27/05/2022

for PS-2 Coordinator

Prof. Dr. – Sanjay Goel
Director,
Institute of Engineering & Technology
JK Lakshmipat University Jaipur

ACKNOWLEDGEMENTS

I wish to express my gratitude to the CEO of the organisation Mr. Jeet Kumar who spared his time from his busy schedule to conduct our learn and grow induction program and provide us with invaluable teachings of the professional world and provide us a platform where we can showcase our skills.

My heartfelt thanks to Ms. Rupali Mittal, Sr. Software Engineer at Intimetec Visionsoft Pvt Ltd for her support in providing me the guidance and mentorship during my training. Thanking all my friends and the trainees in the organisation who have helped me carrying out this work directly or indirectly without whom completion of this training and project work was not possible.

I would also like to thank sincerely to Prof. Dr Devika Kataria, Head of Electrical and Electronics Department and Dr Sanjay Goel, Director, Institute of Engineering and Technology, JK Lakshmipat University for giving us a platform to carry out the learnings.

I also thank my faculty supervisor Dr. Jagdish Prasad Sharma for guiding me throughout the PS-2 process and keeping me on track with everything.

Sincerely yours,
Dhruv Patel (2018BTECHEEE003)

ABSTRACT

Android is the operating system based on linux kernel and other open source softwares. Android OS is basically used for touch screen devices such as tablets and mobile phones but its application is increasing to smart watches and TV systems also. Earlier Java programming language was used to code android applications but now kotlin has replaced Java as the programming language for android devices and google has declared kotlin as the preferred language for android development. This is because Kotlin has much simpler syntax and it is faster to code as compared to Java. Kotlin combines features of object-oriented programming and functional programming whereas Java only supports object-oriented programming. The purpose of this training is to get trained in android development using Java and Kotlin programming languages. the training is executed in segments where in each segment certain topics have to be covered and it follows up with an assignment using those concepts. The assignments is reviewed by the industry mentor and given comments have to be implemented as soon as possible. During the review those topics are also checked on theoretical level with proper reasoning by asking questions from the code of the assignment.

List of Figures

Figure 2.1 Home Screen (Music Player) 18
Figure 2.2 Player screen (Music Player) 18
Figure 2.3 Home Page(Quiz app) 20
Figure 2.4 Question Page (Quiz app) 20
Figure 2.5 Question Page (Quiz app) 21
Figure 2.6 Answers page(Quiz app) 21
Figure 2.7 Score Page (Quiz app) 22
Figure 2.8 Home Page (Playstore) 24
Figure 2.9 Home Page (Quotes App) 26
Figure 5.1 Register Page (Shopping app) 33
Figure 5.2 Login Page (Shopping app) 33
Figure 5.3 Home Page(Shopping app) 34
Figure 5.4 Product list (Shopping app) 34
Figure 5.5 Product Details Page (Shopping app) 35
Figure 5.6 Cart (Shopping app) 35
Figure 5.7 Purchase Screen (Shopping app) 36

List of Flowcharts

Flow Chart 3.1 Training Timeline 27

List of Acronyms

- OS: Operating System
- POC: Proof of Concept
- DB: Database
- UI: User Interface
- MVVM: Model View View Model
- MVC: Model View Controller
- MVP: Model View Presenter
- IDE: Integrated Development Environment
- SCRUM: Systematic Customer Resolution Unraveling Meeting
- DAO: Data Access Objects
- DVM: Dalvik Virtual Machine

Contents

CERTIFICATE III

ACKNOWLEDGEMENTS IV

ABSTRACT V

LIST OF FIGURES VI

LIST OF FLOWCHARTS VII

LIST OF ACRONYMS VII

CHAPTER 1 : ABOUT THE COMPANY 10

1.1 COMPANY NAME 10
1.2 WHAT DOES THE COMPANY DO: 10

CHAPTER 2 : TRAINING WORKFLOW 11

2.1 TRAINING BUNDLE 1 11
2.1.1 Summary *11*
2.1.2 Environmental Setup *11*
2.1.3 Java and Kotlin *11*
2.1.4 OOPs *11*
2.2 TRAINING BUNDLE 2 11
2.2.1 Summary *11*
These are the topics which I learned during the training bundle 2. *12*
2.2.2 Android Project Structure *12*
2.2.3 Android and Android Architecture *12*
2.2.4 Intents and their uses *14*
2.3 TRAINING BUNDLE 3 15
2.3.1 Summary *15*
2.3.2 Constraint Layout *15*
2.3.3 Shared Preferences *15*
2.3.4 Broadcast receiver and custom receiver *15*
2.4 TRAINING BUNDLE 4 16
2.4.1 Summary *16*
2.4.2 Android Service *16*
2.4.3 Android Storage *17*
2.4.4 Room DB *17*
2.4.5 Assignment: Develop a Music Player Application *18*
2.5 TRAINING BUNDLE 5 19
2.5.1 Summary *19*
2.5.2 Fragment and Fragment Lifecycle *19*
2.5.3 Final application size was 10.22 MB *20*
2.6 TRAINING BUNDLE 6 22
2.6.1 Summary *22*
2.6.2 Android Recycler View *22*
2.6.3 Unit Testing *23*
2.6.4 Assignment: Develop a layout using above concepts to implement layout of google play store. *23*
2.7 TRAINING BUNDLE 7 24
2.7.1 Final Project *24*
2.8 TRAINING BUNDLE 8 24
2.8.1 Summary *24*

2.8.2 Coroutines 25
2.8.3 Kotlin Scope Functions 25
2.8.4 JetPack Components 25
2.8.5 Project using Jetpack Components 26

CHAPTER 3 : TIMELINE 27

CHAPTER 4 : KEY TECHNOLOGIES 29

4.1 SUMMARY 29
4.2 ANDROID STUDIO (ANDROID IDE) 29
4.3 FIREBASE 29
4.4 BITBUCKET 30
4.5 SCRUM FRAMEWORK 30
4.5.1 What is Agile? 30
4.5.2 What is Scrum framework? 31
4.5.3 What is sprint? 31
4.5.4 What is product Backlog? 31
4.5.5 What is sprint backlog? 31

CHAPTER 5 : TRAINING FINAL PROJECT (SHOPPING APPLICATION) 32

5.1 SUMMARY 32
5.2 FEATURES TO BE INCLUDED 32
5.3 DEVELOPMENT BUILD 32

CHAPTER 6 : ASSIGNED PROJECT 37

6.1 SUMMARY 37
6.2 DETAILS 37

CHAPTER 7 : RESULT AND CONCLUSION 39

REFERRENCES 40

ANNEXURE 41

ACTIVITY FILES 41
Cart Activity 41
Display All Category Activity 42
Home Page Activity 44
Login Activity 46
Product Details Activity 48
Product List Activity 51
Purchase Screen Activity 53
Register Activity 55
ADAPTER FILES 57
Cart Adapter 57
Categories Adapter 59
Product List Adapter 61
Purshcase Screen Adapter 64
MODEL 65
Categories Model 65
Product Model 66
ROOM DB 67
Cart Database 67
Cart Product entity files 67
Product Dao File 68
CONSTANT FILE 69

CHAPTER 1 : About the Company

1.1 Company Name

Intimetec Visionsoft Pvt Ltd

1.2 What does the company do:

Intimetec is software development company which started with a vision of creating a world class organization built on trust, transparency, integrity and leadership with a commitment to employees, customers, stakeholders and the community. The company works in a variety of fields such as Business Intelligence and Reporting, Cybersecurity, Data Science, DevOps, Legacy Software System Modernization, Mobile Application Development, Quality Assurance, Test Automation, UI/UX Design and Front End Development, Web Development.

In Time Tec provides software development services through four distinct models: a blend of onshore and offshore resources, onshore-only resource teams, offshore-only resource teams, and near-shore teams (offshore teams working within a specific time zone). The team size, delivery model, and technical domain are based on our partner's needs and it's the vested partnership between In Time Tec and our clients that create a powerful context in which we can deliver.

Company also provides software as a service products and one of the major selling product is Cartos Suite.

Cartos Suite is a collection of cloud-based Managed Print Service (MPS) applications that help MPS sales teams, and consultants discover existing print fleets, design future fleets, analyze Total Cost of Ownership (TCO), and easily find and compare groups of devices. This is accomplished through Cartos Suite Fleet Design, TCO, Asset Advisor, and Device Data Service.

CHAPTER 2 : Training workflow

2.1 Training Bundle 1

2.1.1 Summary

These are the topics which were covered during the Training Bundle 1.

1. Environment setup: Android setup with android SDK
2. Java, Kotlin Fundamentals
3. OOPS Concepts
4. Control Statements, Object class, Inheritance, Polymorphism, Abstraction, Encapsulation

2.1.2 Environmental Setup

For this android studio is used which is the most preferred setup used for developing android applications.

2.1.3 Java and Kotlin

Java is an object oriented, class based, secured and concurrent programming language. It was developed by sun microsystems in the year 1995.

Platform

Any hardware device environment where a program is run is known as a platform. Java has runtime environments JRE it is called a platform.

2.1.4 *OOPs*

Object-oriented programming (OOP) is a programming approach in which classes of objects are identified and closely related to the methods (functions) with which they are associated. It also incorporates concepts like as attribute and method inheritance. It's a technique for storing data alongside the procedures needed to process it that's based on a mathematical discipline called "abstract data types." OOP has the potential to raise the level of abstraction in programming.

2.2 Training Bundle 2

2.2.1 Summary

These are the topics which I learned during the training bundle 2.

1. Android Project Structure
2. Create a new application and run it on the emulator as well as physical device.
3. Introduction to Android, Android Architecture.
4. Activity and activity life cycle
5. Intents and their usage: implicit and explicit intents and Start Activity for result.
6. Design layouts using linear layout, frame layout and relative layout.

2.2.2 Android Project Structure

The android project structure incorporates many files which are automatically generated when a new project is created on the android studio.

- Manifest Folder
    - AndroidManifext.xml file

This file contains the details about permissions the application uses such as access to internal storage, internet access etc. This file also mentions the use of services by the application such as broadcast receiver, contains information about android version, metadata. It acts as intermediary between android OS and application.

- Java Folder
    - Files related to application logic which are codded in java and Koltin.
- Resource Folder
    - Contains .xml files related to application UI.
- Gradle Scripts
    - Contains files related to application build.

2.2.3 Android and Android Architecture

The android architecture contains variety of components which collectively support the proper functioning of any android device. Android software is made up of open source Linux Kernel. Open source means that anyone can have access to the android OS source code and make their own modifications, this includes a collection of C/C++ libraries that are accessed via application framework services.

There are many components which comprise while running an android application but while linux kernel is responsible for providing the operating systems core functionality to smartphones, while the Dalvik Virtual Machine (DVM) provides a framework for running the android applications.

The main of components of android architecture are:

1. Applications

Applications are the top layer of Android architecture. This layer will only be used to install pre-installed programmes like home, contacts, camera, and gallery, as well as third-party apps like chat apps, games, and others.

To run within the Android run time, it makes use of the application framework's classes and services.

2. Application framework

A variety of important classes for developing Android apps are included in the Application Framework. It provides a general wrapper for hardware access, which aids in the management of the user interface with application resources. In general, it gives capabilities that enable us to build a class and make that class useful for application development.

It contains a variety of services, such as activity manager, notification manager, view system, package manager, and so on, all of which are useful in the creation of our application to meet the needs.

3. Android Runtime

One of the major aspects of android is the Android Runtime Environment. It includes fundamental libraries as well as the Dalvik Virtual Machine (DVM). Like Java Virtual Machine (JVM) is responsible for building and running java code. Android has DVM to power its applications. DVM is a register based virtual machine and specially designed and improved for android to make sure that android devices run multiple apps efficiently. The memory management and threading depends on the linul kernel layer. The core libraries enable the android device to implement the applications using Java and Kotlin code by the means of programming.

4. Platform Libraries

This includes C/C++ libraries such as Graphics, Open GL, surface manager and Media etc. These libraries provide support to run applications.

- The media library allows user to play and record music and video file formats.
- The display subsystem's access is managed by the surface manager.

- SGL and OpenGL are libraries used for 2D and 3D computer graphics on android devices. These libraries are both cross language and cross platform APIs.
- FreeType provides font support and SQLite provides database support to android devices.
- Web-Kit is a library used for providing web browser functionality to view content and page loading. This acts as default web browser engine.
- SSL (Secure Sockets Layer) is a technology that makes sure that communication between a web server and web browser is encrypted.

5. Linux Kernel

Linux kernel is the blood pumping organ of the android architecture. All the available drivers in android device such as display drivers, camera drivers, audio drivers, Bluetooth drivers and memory drivers etc as managed by the Linux kernel.

The Linux Kernel acts as a layer with abstraction concepts between hardware and software components of android architecture. The Linux Kernel is mainly responsitble for memory, devices and power etc management. The features of Linux kernel are as follows:

1. Security: The security between the programme and the system is handled by the Linux kernel.
2. Memory Management: the freedom to devlop and run our apps through the efficient handling of memory management is done by Linux Kernal.
3. Process Management: this incorporates for smart work like allocating memory to processes that need them. Therefore managing memory well is also a function of Linux Kernel.
4. Network Stack: The network communication is efficiently handled by Linux Kernel.
5. Driver Model: Hardware devlopers for android devices make sure that they build drivers in Linux build to support them on android OS. It ensures that driver applications work properly.

2.2.4 Intents and their uses

Intents are basically messaging objects which are used to request an action from different activity.

There are two types of intents

1. Explicit Intent
2. Implicit Intent

Explicit intents are used to when we know which application component is going to full fill the purpose.

Implicit intents are used when we don't know which application component is going to full fill the purpose therefore, we specify a general action which needs to be performed. This will allow any application capable of handling that action to be available for that specific task.
For example, when we share a contact on phone it shows all the available applications to which the contact can be shared in a specific format.

2.3 Training Bundle 3

2.3.1 Summary

1. Design layout using constraint layout
2. Android Shared preferences (AndroidX security)
3. Broadcast receiver and custom receivers

2.3.2 Constraint Layout

It is a View Group which allows to place and adjust the widgets in a flexible way. It allows features such as drag and drop of widgets and layout elements which increases the speed and efficiency which creating UI layout.

2.3.3 Shared Preferences

This is a type of data storing method that is available in android. If the data is relatively small, then it can be stored in a file in the form of key and value pairs. Each value would have a key assigned to it. There are simple methods to read and write these key value pairs.

2.3.4 Broadcast receiver and custom receiver

Broadcast Receivers are simple devices that respond to broadcast messages sent by other apps or the system itself. In other cases, these communications are referred to as events or intentions. For example, if an application wants another application know that some data has been changed or downloaded to the device and is now available for other applications to use, So the broadcaste receiver in other apps will intercept this communication and will respond as per the codded logic.

2.4 Training Bundle 4

2.4.1 Summary

1. Android service
2. Android Storage (Internal and external)
3. Room DB

2.4.2 Android Service

A service is a component of an application that can do long-running tasks in the background. It doesn't have a graphical user interface. A service that has been started may continue to operate for some time after the user has switched to another application. A component can also bind to a service in order to communicate with it and even do interprocess communication (IPC). A service, for example, can run in the background and manage network transactions, play music, do file I/O, and connect with a content provider.

1. Foreground Service

A forground service is a service which is visible to the user and he can interact with it. A forground service, for example, would be used in audio apps such as amazon music where when a user plays a music and then minimizes the app the music controls appear in the notification section and mucis keeps playing. This is foreground service, even if the user is not interacting with the application the service continue to execute.

When using a foreground service, its is must to show a notification so that users are aware that the service is active. Unless the service is stopped or removed from the foreground, this notification cannot be disregarded.

2. Background Service

A background service is a type of service which is not visible to the user and continues to run in background when the app minimizes. This type of service would be used for example when a application is storing its backup.

3. Bind Service

Bind service is a type of service where the service is only linked to the activity of an application. The service will close as soon as that activity closes. A activity is anything which appears on a screen of a

mobile application at a particular time. To call bind service the application component calls bindService on a service, it bind to it after that. A bind service have a client- service interface which allows the service to interact with the server and send necessary request as per the requirement of the application. Multiple components in a application can bind to the same service but service will discontinue after all the components are terminated or unbinded.

2.4.3 Android Storage

Android employs a file system that is similar to other platforms' disk-based file systems. You have various options for saving your app data on the system:

- App-specific storage: Keep files that are only used by your app in dedicated directories within an internal storage volume or in separate dedicated directories within external storage. Use internal storage folders to preserve sensitive data that shouldn't be accessed by other apps.
- Store files that your app wishes to share with other apps, such as media, documents, and other files, in shared storage.
- Preferences: Key-value pairs are used to store private, rudimentary data.
- Databases: The Room persistence library can be used to store structured data in a private database.

2.4.4 Room DB

Apps that deal with large volumes of structured data can considerably benefit from storing that data locally. The most typical use case is to cache relevant portions of data so that the user may still explore that information while the device is not connected to the internet.

The Room persistence library acts as an abstraction layer over SQLite, allowing for fluent database access while still utilising SQLite's full functionality. Room, in particular, offers the following advantages:

- SQL queries are checked at compile time.
- Convenience annotations that reduce boilerplate code that is repetitious and error-prone.
- Database migration paths have been simplified.

2.4.5 *Assignment: Develop a Music Player Application*

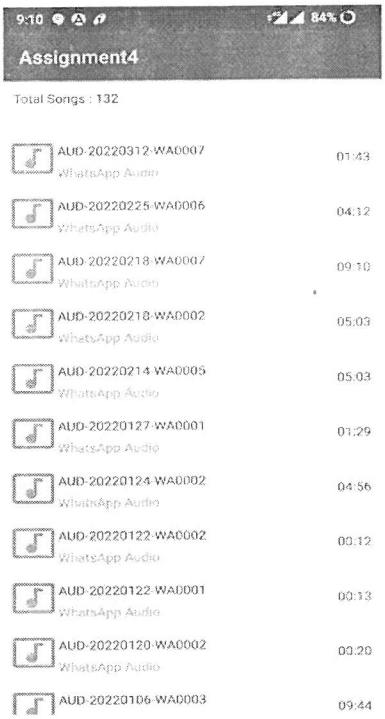


Figure 2.1 Home Screen
(Music Player)



Figure 2.2 Player screen
(Music Player)

The figure 2.1 shows the list of audio files, this window comes right after the app is opened. When an audio file is clicked, it will move to the second screen which is shown in figure 2.2, this is the player screen where user will get options to play, pause, move to next song, and move to previous song.

2.5 Training Bundle 5

2.5.1 Summary

1. Fragment and Fragment lifecycle and navigation components (navigation graph)
2. Content Provider
3. Firebase DB
4. Assignment: Develop a Quiz Application.

2.5.2 Fragment and Fragment Lifecycle

A Fragment is a reusable part of your app's user interface. A fragment has its own layout and lifetime, as well as the ability to handle its own input events. Fragments cannot exist without the assistance of an activity or another fragment. The view hierarchy of the fragment merges with, or is attached to, the view hierarchy of the host.

Modularity

By dividing the UI into separate sections, fragments add modularity and reusability to your activity's UI. Activities are a great area to integrate global components like a navigation drawer throughout your app's user interface. Fragments, on the other hand, are excellent for defining and managing the UI of a single screen or piece of a screen.

Take, for example, an app that adjusts to different screen sizes. The app should include a static navigation drawer and a grid-layout list on larger devices. The app should have a bottom navigation bar and a list with a linear style on smaller screens. It might be difficult to keep track of all of these different activity variations. This approach can be made more manageable by separating navigation components from content. The activity is then in charge of presenting the appropriate navigation UI, while the fragment is in charge of displaying the correct list layout.

2.5.3 Final application size was 10.22 MB

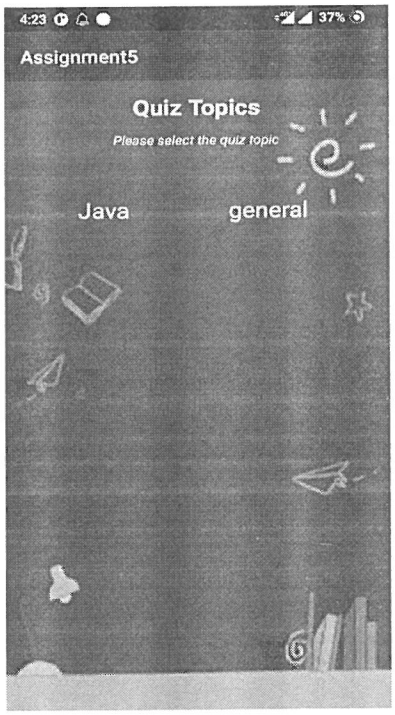


Figure 2.3 Home Page(Quiz app)

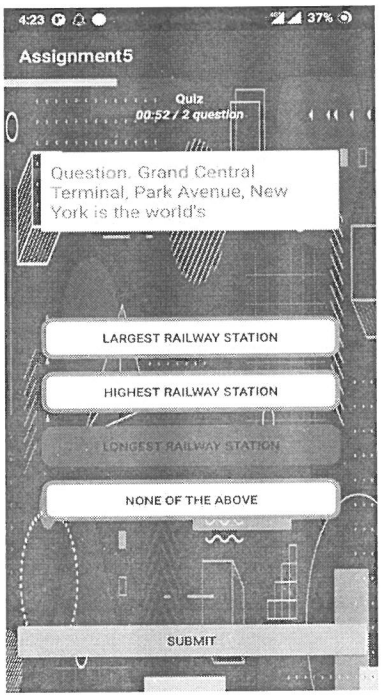


Figure 2.4 Question Page (Quiz app)

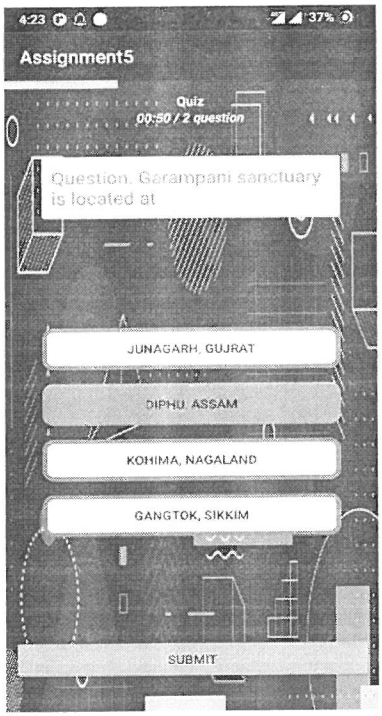


Figure 2.5 Question Page (Quiz app)

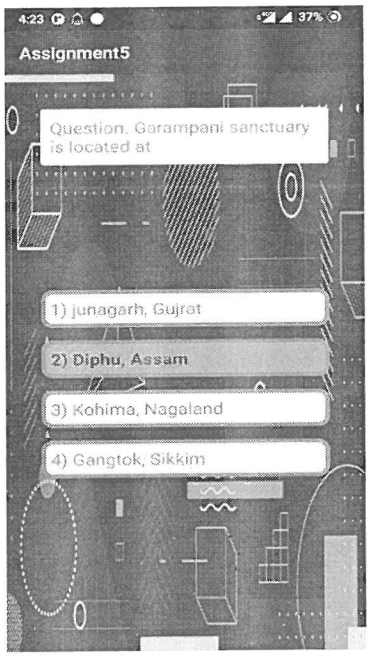


Figure 2.6 Answers page(Quiz app)

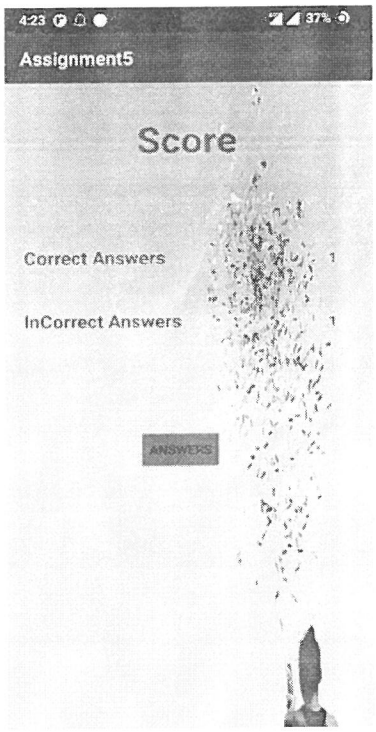


Figure 2.7 Score Page (Quiz app)

Figure 2.3 shows the home screen of the application where user can select the quiz topics after that the user is redirected to the questions page and question changes once it is answered. User can slide left and right to view answered questions. After the quiz is complete and submit button is clicked the Figure 2.7 shows the score screen which provides an option to view answers. After answers button is clicked the answers are shown as they appear in Figure 2.5

2.6 Training Bundle 6

2.6.1 Summary

1. Android Recycler View
2. REST API interaction
3. Unit testing

2.6.2 Android Recycler View

RecyclerView makes it simple to display vast amounts of data. You provide the data and define how each item should appear, and the RecyclerView framework builds the elements as needed.

RecyclerView, as the name implies, recycles those individual elements. RecyclerView does not erase an item's view when it scrolls off the screen. RecyclerView, on the other hand, reuses the view for fresh items that have scrolled across the screen. This reuse greatly increases performance, as well as the responsiveness and power consumption of your programme.

Note: RecyclerView is not just the name of the class, but also the name of the library.

2.6.3 Unit Testing

Unit testing is process of testing each component of application separately. This is the very initial step in fortware testing. The code is written in such a way that each individual function can be tested sperately this ensures single responsibility role and also that each component of application is working perfectly.

Therefore a Unit can be described as a single testable component of a software application which is tested during the software development stage.

The goal is to make sure that individual piece of code is working properly. The Uit testing approach is usually done by developers who test the application using white box testing.

When the programme is complete and given to the Test Engineer, he or she will begin unit testing or component testing by individually examining each component of the module or module of the application.

2.6.4 Assignment: Develop a layout using above concepts to implement layout of google play store.

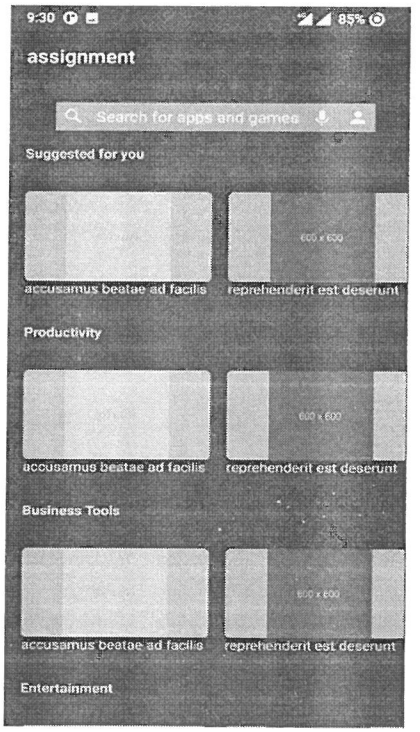


Figure 2.8 Home Page (Playstore)

2.7 Training Bundle 7

2.7.1 Final Project

- (Mock) User login screen
- (Mock) User SignUp Screen
- Products list screen along with products filter
- (Mock)Products details Screen along with add to cart features
- Cart Screen

2.8 Training Bundle 8

2.8.1 Summary

1. Coroutines
2. Kotlin Scope functions
3. Jetpack Components
    a. Live Data
    b. View Model

- MVVM
- MVC
- MVP

c. Data Binding

d. View Binding

2.8.2 Coroutines

Programming that is asynchronous or non-blocking is an important aspect of the development process. When developing server-side, desktop, or mobile applications, it's critical to provide an experience that's not just seamless from the user's point of view, but also scalable when necessary.

By providing coroutine support at the language level and outsourcing most of the functionality to libraries, Kotlin solves this problem in a flexible way.

Coroutines not only enable asynchronous programming, but they also offer up a slew of other possibilities, such as concurrency and actors.

2.8.3 Kotlin Scope Functions

Several functions in the Kotlin standard library have the sole goal of executing a block of code within the context of an object. When you run a lambda expression-based function on an object, a temporary scope is created. You can access the object without knowing its name in this scope. Scope functions are the name for such functions. Let, run, with, apply, and also are the five options.

These functions all do the same thing: they run a block of code on an object. What's different is how this object gets available inside the block and what the expression's result is.

2.8.4 JetPack Components

Jetpack is an Android library collection that incorporates best practises and provides backwards compatibility in your Android apps.

The Jetpack app architecture guide gives you an overview of best practises and recommended architecture to think about when you design your Android app.

The sections that follow explain how to get started with Jetpack components

2.8.5 *Project using Jetpack Components*

In this project I made a quotes app which has next and previous button functions. As app opens it will display a quote and clicking on next and previous button it will change the quote.

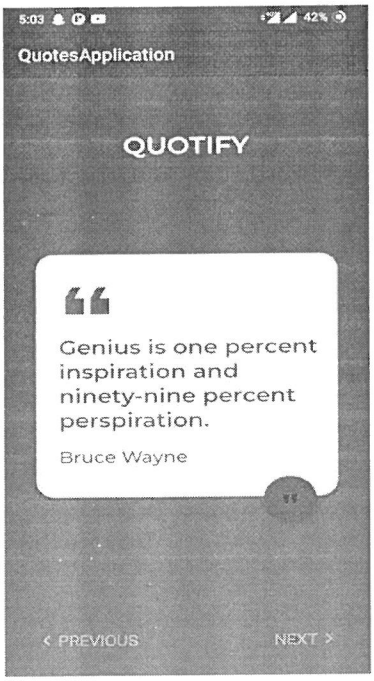
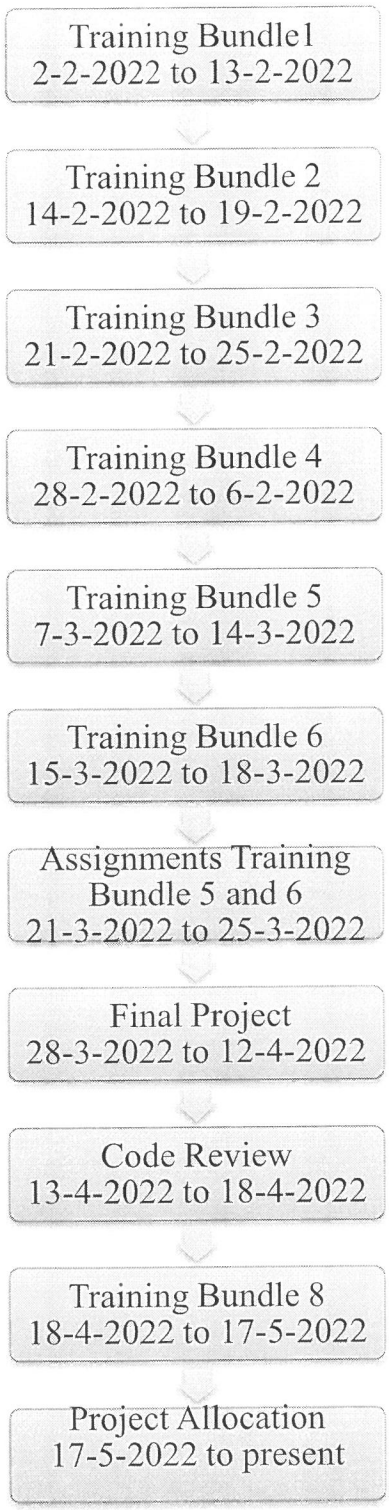


Figure 2.9 Home Page (Quotes App)

CHAPTER 3 : Timeline



Flow Chart 3.1 Training Timeline

The chapter two explains the topics that were covered during my training in the form of bundles but the exact time period of these bundles is not fixed therefore, chapter three gives the duration of time spent on each training bundle and other activities that happened during the training period. The code reviews are ususlly supposed to happen at the end of every training bundle but because the industry mentors were busy it got extended to the date as specified in the flow chart.

Usually a training bundle is just supposed to consume a week but in some training bundles due to company activities and level of difficulty the time period has been extended beyond a week by few days.

The asignments from training bundle one to training bundle three were reviewed and necessary changes were implemented by be at the end of each training period.

During the code review period all the assignments from training bundle four to training bundle seven were reviewed and necessary changes were implemented by me in the code.

CHAPTER 4 : Key Technologies

4.1 Summary

This Chapter will summarise all the technologies which are used in the training period. The explanation will be given in the terms of features, advantages and technical configuration.

4.2 Android Studio (Android IDE)

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA.IntelliJ IDEA is Java based software. Android Studio offers features that enhance your productivity when building Android apps, such as:

- Fast coding and iteration
- Inbuild Emulator to run apps virtually on PC itself
- Auto Suggest features which improves self confidence while coding.
- Wide range of testing tools
- Integrated support for Firebase database
- Layout editor with drag and drop support
- Advanced product development system which involves dependency management, library incorporation and other customizations supports.

4.3 Firebase

Firebase is a Backend service provided by google. There are variety of tools and services in firebase which help users develop application and grow user base. The Firebase is build on infrastructure provided by google. It is a NoSQL database and stores data in the Json format documents in firestore Database.

Key Features:

1. Authentication

It supports multiple authentication methods such as using mobile numbers, passwords, google account, facebook account, twitter account etc. The firebase SDK also allows multiple integration of one or more sign-in options.

2. Realtime database

In realtime database the data is available in realtime and even when the app is offline data is available and can be access using realtime database.

3. Hosting

It provides web hosting using cloud services. For a web app the hosting is fast and content is stored in the content delivery packages world wide.

4. Test lab

The application is tested on google data centers both virtually and on physical devices.

5. Notifications

Using firebase notification facility can be used without any extra haszel of coding. Users can start developing using firebase for free with spark plan.

4.4 BitBucket

Like Git which is a repository management solution. BitBucket is also a git repository solution for professional developers. It allows collaboration on code base and can show the development of an application using visual tools. Other features of Bitbucked include:

- Restricted access to source code to project limited people.
- Workflow control.
- Pull request with commenting inline feature for faster collaboration during code review.
- Jira integration for development tracking.
- REST API control to build festures needed by the team if they are already not available on the market place.

4.5 SCRUM Framework

4.5.1 What is Agile?

Agile is a project management and software development technique aimed to complete the project in a iterative manner. This helps to deliver the product faster. Intstead of releasing the product in one big launch, the process aims to deliver the product in small parts with increasing feature. Requirements, and plans are evaluated continuously so any update in existing version is becomes easy and implementable. Agile involves constant collaboration of all the stackholders involved in the project at every stage.

4.5.2 What is Scrum framework?

Scrum is agile framework that is created for team to work together. Scrum describes a set of tools to be used in meetings and roles that work in synchronization to help teams structure and manage their work.

4.5.3 What is sprint?

A sprint is a short time frame where a scrum team works to complete a set amount of work.

4.5.4 What is product Backlog?

A product backlog in Scrum is a list of features containing brief description about all the set of functionality and UI desired in the product.

4.5.5 What is sprint backlog?

The sprint backlog is the list of tasks identified by the

CHAPTER 5 : Training Final Project (Shopping Application)

5.1 Summary

The final project of the training was intended to check the concepts of a trainee which he had learned over the training period. The task was to develop a shopping application with the features mentioned in below section 4.2. The final project was coded in Java programming language and it includes android concepts of Room Database for developing cart feature, Firebase for authentication and data storage, Search Query to implement product filter in search bar, some android dependencies such as Picasso for Image loading. The apps APK size is 29.26 MB.

The Room DB Code in annexure 7.4 mentions the ProductDao file which is use to store functions (methods) such as insert, query etc.

5.2 Features to be included

- (Mock) User login screen
- (Mock) User SignUp Screen
- Products list screen along with products filter
- (Mock)Products details Screen along with add to cart features
- Cart Screen

5.3 Development Build

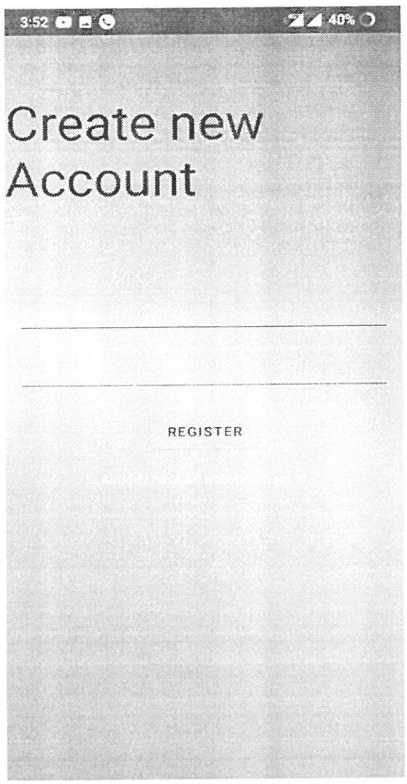


Figure 5.1 Register Page (Shopping app)

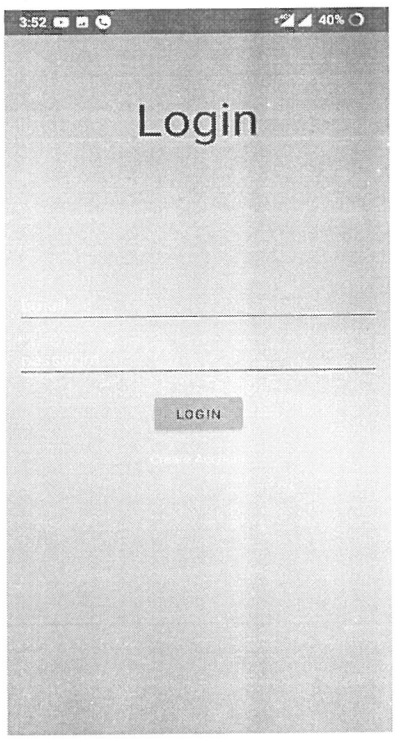


Figure 5.2 Login Page (Shopping app)

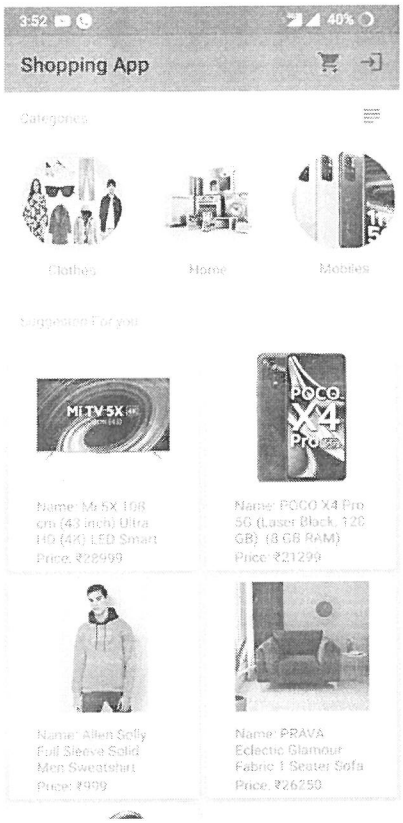


Figure 5.3 Home Page(Shopping app)

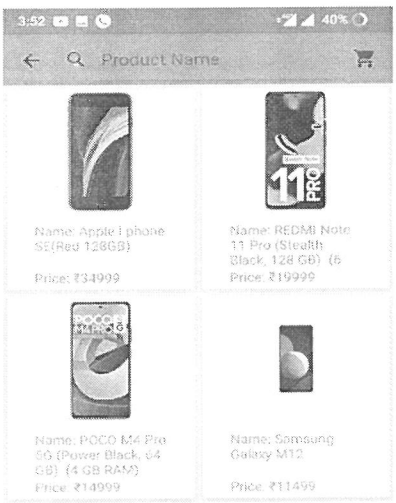


Figure 5.4 Product list (Shopping app)

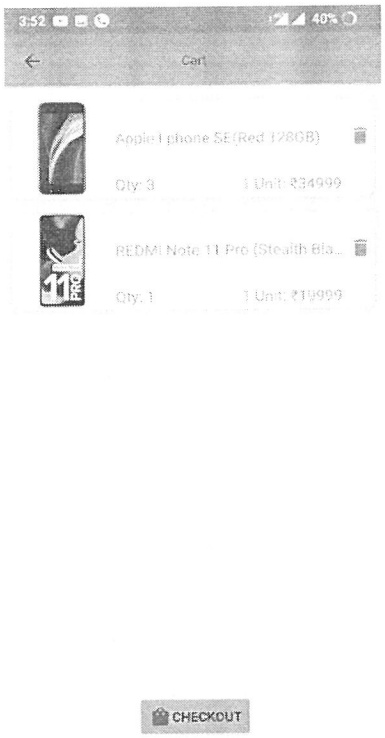


Figure 5.5 Product Details Page (Shopping app)



Figure 5.6 Cart (Shopping app)

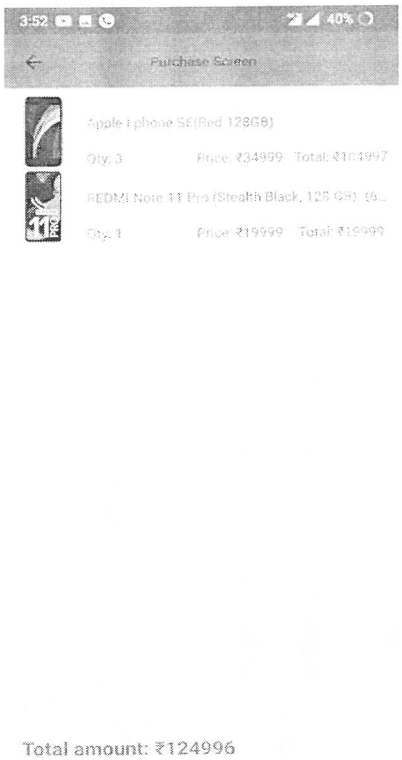


Figure 5.7 Purchase Screen (Shopping app)

CHAPTER 6 : Assigned Project

6.1 Summary

After completion of training I have been assigned to a live project on the designation of the trainee engineer till the total completion of my training time becomes six months. After that I will be promoted to the designation of Jr. Software engineer in the project. Currently as per my role I have been assigned to read and understand the code base of the project and identify the problems related to the UI and improvement of code but these problems are to be reported to the Sr. Software engineer on the project. As I am relatively new to the project I am not allowed to make changes to the live code base. So my major responsibility is to verify the proof of concept (POC) in the live project.
POC is a task assigned by the Sr. software engineer on the project. POC is a hypothetical concept of how things can be implemented in the live project. So my main task is to verify the POC ideas by creating a code separate from the live project.

6.2 Details

The project is mobile application development using Kotlin programming language. The mobile application is owned by the St. Lukes. St. Lukes is a not for profit health system organisation. St. Lukes is dedicated to improve the health of people in their region. The organisation is based in Boise, Idaho, USA. They have many hospitals owned by them in the region and the purpose of app is to connect citizens to the health services provided by the St. Luke's Organisation. Through their app they provide contact information of doctors which are directly associated with St. Luke's organisation in their own clinics and otherwise. They also provide the background information about the doctors such as their biography, education and certification, address and contact information. Users can locate the medical faciltiy owned by the St. Luke's through the app itself.

For the development of the application the SCRUM framework is used which involves daily internal SCRUM meetings in morning with development team and evening SCRUM meeting with the stakeholder such as authorities from St. Luke's organisation in the evening. Every week a sprint planning is conducted where a sprint backlog is created which involves the task to be implemented during the sprint period.

All the other information about the application such as their code base and POC concepts are confidential and are not allowed by the organisation to be shared outside the company premises. So, I am unable to share those details. Also the UI of application in not yet released on playstore so I am also unable to share the latest UI of the application as well.

CHAPTER 7 : Result and Conclusion

During my PS-2 I have been training on android development using Java and Kotlin. During this time I have learned many concepts which are incorporated in a live application that we use everyday. These learnings have given me an understanding about the thought process that is required while building a mobile application. The programs run by the company such as Learn and Code and my thorough review process during my training period has made me realise the correct practice of coding productively. So that the efficiency of a project increases.

When a new person is assigned on the project and he has to go through the complete code to understand the application logic and it consumes a lot of time and energy. Correct coding practice ensures that new member in team can understand the code easily, inaddition it makes the review process and futures modifications in the project easier.

Referrences

1. Android - Broadcast Receivers. (2022). Retrieved 26 May 2022, from https://www.tutorialspoint.com/android/android_broadcast_receivers.htm

2. Save key-value data | Android Developers. (2022). Retrieved 26 May 2022, from https://developer.android.com/training/data-storage/shared-preferences

3. DataBindingUtil | Android Developers. (2022). Retrieved 26 May 2022, from https://developer.android.com/reference/android/databinding/DataBindingUtil.html#inflate(android.view.LayoutInflater,%20int,%20android.view.ViewGroup,%20boolean)

4. Java Tutorial | Learn Java Programming - javatpoint. (2022). Retrieved 26 May 2022, from https://www.javatpoint.com/java-tutorial

5. Kotlin Tutorial - javatpoint. (2022). Retrieved 26 May 2022, from https://www.javatpoint.com/kotlin-tutorial

6. RecyclerView | Android Developers. (2022). Retrieved 26 May 2022, from https://developer.android.com/reference/androidx/recyclerview/widget/RecyclerView

7. Android Jetpack | Android Developers. (2022). Retrieved 26 May 2022, from https://developer.android.com/jetpack

8. Android - Content Providers. (2022). Retrieved 26 May 2022, from https://www.tutorialspoint.com/android/android_content_providers.htm

9. Android - Broadcast Receivers. (2022). Retrieved 26 May 2022, from https://www.tutorialspoint.com/android/android_broadcast_receivers.htm

10. Shared Preferences in Android with Example - GeeksforGeeks. (2019). Retrieved 26 May 2022, from https://www.geeksforgeeks.org/shared-preferences-in-android-with-examples/

11. What is Intent in Android? - GeeksforGeeks. (2020). Retrieved 26 May 2022, from https://www.geeksforgeeks.org/what-is-intent-in-android/

12. Library, D., & Fragment, D. (2022). Android Tutorial => Databinding in Fragment. Retrieved 26 May 2022, from https://riptutorial.com/android/example/6017/databinding-in-fragment

13. Android - Fragments. (2022). Retrieved 26 May 2022, from https://www.tutorialspoint.com/android/android_fragments.htm

ANNEXURE

This contains the code files of final assignment.

Activity Files

Cart Activity

```
package com.example.assignment.activity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.room.Room;

import com.example.assignment.Adapter.CartAdapter;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.example.assignment.roomdb.CartDatabase;
import com.example.assignment.roomdb.CartProduct;
import com.example.assignment.roomdb.ProductDao;

import java.util.List;

public class CartActivity extends AppCompatActivity implements View.OnClickListener {
    private List<CartProduct> cartProductList;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_cart);
        getRoomData();
        setupCartRecyclerView();
        setOnClickListeners();
    }

    private void setOnClickListeners() {
        ImageView ivBack = findViewById(R.id.ivBackIcon);
```

```
    Button btnPurchase = findViewById(R.id.btnPurchase);
    ivBack.setOnClickListener(this);
    btnPurchase.setOnClickListener(this);
  }

  private void getRoomData() {
    CartDatabase database = Room.databaseBuilder(getApplicationContext(),
        CartDatabase.class, Constant.ROOM_DB_NAME).allowMainThreadQueries().build();
    ProductDao productDao = database.ProductDao();
    cartProductList = productDao.getallProduct();

  }

  private void setupCartRecyclerView() {
    RecyclerView cartRecyclerView = findViewById(R.id.rvCart);
    cartRecyclerView.setLayoutManager(new LinearLayoutManager(this,
        LinearLayoutManager.VERTICAL, false));
    CartAdapter cartAdapter = new CartAdapter(cartProductList, this);
    cartRecyclerView.setAdapter(cartAdapter);
  }

  @Override
  public void onClick(View view) {
    int id = view.getId();
    if (id == R.id.ivBackIcon) {
      this.onBackPressed();
    } else if (id == R.id.btnPurchase) {
      Intent intent = new Intent(this, PurchaseActivity.class);
      startActivity(intent);
    }
  }
}
```

Display All Category Activity

```
package com.example.assignment.activity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.util.Log;
import android.widget.ImageView;
```

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.assignment.Adapter.CategoriesAdaptor;
import com.example.assignment.Model.CategoriesModel;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.ArrayList;

public class DisplayAllCategoryActivity extends AppCompatActivity {
    private CategoriesAdaptor categoriesAdaptor;
    private FirebaseFirestore firebaseFirestore;
    private ArrayList<CategoriesModel> categoriesModelArrayList;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_all_category);
        setupRecyclerView();
        getCategoriesData();
        setBackButton();
    }

    @SuppressLint("NotifyDataSetChanged")
    private void getCategoriesData() {
        firebaseFirestore.collection(Constant.COLLECTION_PATH).addSnapshotListener(((value, error) -> {
            if(error!= null){
                Log.e("data",error.getMessage());
                return;
            }
            assert value != null;
            for (DocumentChange documentChange : value.getDocumentChanges()){
                if (documentChange.getType() == DocumentChange.Type.ADDED){
                    categoriesModelArrayList.add(documentChange.getDocument()
                            .toObject(CategoriesModel.class));
                }
                categoriesAdaptor.notifyDataSetChanged();
            }
```

```
        }));
    }
    private void setBackButton(){
        ImageView backButton = findViewById(R.id.ivBackIcon);
        backButton.setOnClickListener(view -> this.onBackPressed());
    }
    private void setupRecyclerView() {
        RecyclerView allCategoryRecyclerView = findViewById(R.id.rvAllCategory);
        categoriesModelArrayList = new ArrayList<>();
        allCategoryRecyclerView.setHasFixedSize(true);
        allCategoryRecyclerView.setLayoutManager(new GridLayoutManager(this,3,
                RecyclerView.VERTICAL,false));
        firebaseFirestore = FirebaseFirestore.getInstance();
        categoriesAdaptor = new CategoriesAdaptor(this,categoriesModelArrayList);
        allCategoryRecyclerView.setAdapter(categoriesAdaptor);
    }
}
```

Home Page Activity

```
package com.example.assignment.activity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.util.Log;
import android.widget.ImageView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.assignment.Adapter.CategoriesAdaptor;
import com.example.assignment.Model.CategoriesModel;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.ArrayList;

public class DisplayAllCategoryActivity extends AppCompatActivity {
    private CategoriesAdaptor categoriesAdaptor;
    private FirebaseFirestore firebaseFirestore;
```

```
private ArrayList<CategoriesModel> categoriesModelArrayList;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_all_category);
    setupRecyclerView();
    getCategoriesData();
    setBackButton();
}

@SuppressLint("NotifyDataSetChanged")
private void getCategoriesData() {
    firebaseFirestore.collection(Constant.COLLECTION_PATH).addSnapshotListener(((value, error) -> {
        if(error!= null){
            Log.e("data",error.getMessage());
            return;
        }
        assert value != null;
        for (DocumentChange documentChange : value.getDocumentChanges()){
            if (documentChange.getType() == DocumentChange.Type.ADDED){
                categoriesModelArrayList.add(documentChange.getDocument()
                        .toObject(CategoriesModel.class));
            }
            categoriesAdaptor.notifyDataSetChanged();
        }
    }));
}
private void setBackButton(){
    ImageView backButton = findViewById(R.id.ivBackIcon);
    backButton.setOnClickListener(view -> this.onBackPressed());
}
private void setupRecyclerView() {
    RecyclerView allCategoryRecyclerView = findViewById(R.id.rvAllCategory);
    categoriesModelArrayList = new ArrayList<>();
    allCategoryRecyclerView.setHasFixedSize(true);
    allCategoryRecyclerView.setLayoutManager(new GridLayoutManager(this,3,
            RecyclerView.VERTICAL,false));
    firebaseFirestore = FirebaseFirestore.getInstance();
    categoriesAdaptor = new CategoriesAdaptor(this,categoriesModelArrayList);
    allCategoryRecyclerView.setAdapter(categoriesAdaptor);
```

```
    }
}
```

Login Activity

```
package com.example.assignment.activity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.example.assignment.constant.Constant;
import com.example.assignment.R;
import com.google.firebase.auth.FirebaseAuth;

import java.util.Objects;

public class Login extends AppCompatActivity implements View.OnClickListener {
    private EditText etEmail, etPassword;
    private Button btnLogin;
    private ProgressBar progressBar;
    private FirebaseAuth firebaseAuth;
    private TextView tvRegister;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        Objects.requireNonNull(getSupportActionBar()).hide();
        etEmail = findViewById(R.id.etEmail);
        etPassword = findViewById(R.id.etPassword);
        btnLogin = findViewById(R.id.btnLogin);
        firebaseAuth = FirebaseAuth.getInstance();
```

```
    progressBar = findViewById(R.id.progressBar);
    tvRegister = findViewById(R.id.tvCreateAccount);
    setOnclickListeners();
    checkCurrentUser();
}

private void setOnclickListeners() {
    tvRegister.setOnClickListener(this);
    btnLogin.setOnClickListener(this);
}

private void checkCurrentUser() {
    if (firebaseAuth.getCurrentUser() != null) {
        startActivity(new Intent(getApplicationContext(), HomePageActivity.class));
        finish();
    }
}

private void createToast(String message) {
    Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
}

private void login() {
    String email = etEmail.getText().toString().trim();
    String password = etPassword.getText().toString().trim();
    boolean checkAllFields = validateFields(email, password);
    if (checkAllFields) {
        progressBar.setVisibility(View.VISIBLE);
        firebaseAuth.signInWithEmailAndPassword(email, password).addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                createToast(getString(R.string.login_successful));
                startActivity(new Intent(getApplicationContext(), HomePageActivity.class));
            } else {
                createToast(Objects.requireNonNull(task.getException()).getMessage());
                progressBar.setVisibility(View.GONE);
            }
            progressBar.setVisibility(View.GONE);
        });
    }
}

private boolean validateFields(String email, String password) {
```

```
    boolean validation = false;
    if (TextUtils.isEmpty(email)) {
        etEmail.setError(Constant.EMAIL_ERROR_MESSAGE);
    }
    else if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        etEmail.setError(Constant.INVALID_EMAIL_ERROR_MESSAGE);
    }
    if (TextUtils.isEmpty(password)) {
        etPassword.setError(Constant.PASSWORD_ERROR_MESSAGE);
    }
    else if (password.length() < 6) {
        etPassword.setError(Constant.PASSWORD_LENGTH_ERROR);
    } else {
        validation = true;
    }
    return validation;
  }

  @Override
  public void onClick(View view) {
    int id = view.getId();
    if (id == R.id.tvCreateAccount) {
        startActivity(new Intent(getApplicationContext(), Register.class));
    } else if (id == R.id.btnLogin) {
        login();
    }
  }
}
```

Product Details Activity

```
package com.example.assignment.activity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
```

```
import androidx.appcompat.app.AppCompatActivity;
import androidx.room.Room;

import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.example.assignment.roomdb.CartDatabase;
import com.example.assignment.roomdb.CartProduct;
import com.example.assignment.roomdb.ProductDao;
import com.squareup.picasso.Picasso;

public class ProductDetails extends AppCompatActivity implements View.OnClickListener {
    private int productQuantity;
    private TextView tvQuantity;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_product_details);
        Intent intent = getIntent();
        setProductDetails(intent);
        setUpAddToCartButton(intent);
        setOnClickListeners();
        productQuantity = 0;
        tvQuantity = findViewById(R.id.tvQuantity);
        tvQuantity.setText(String.valueOf(productQuantity));
    }

    private void setOnClickListeners() {
        ImageView add = findViewById(R.id.ivAdd);
        ImageView minus = findViewById(R.id.ivMinus);
        ImageView ivBack = findViewById(R.id.ivBackIcon);
        ImageView ivCart = findViewById(R.id.ivCartIcon);
        add.setOnClickListener(this);
        minus.setOnClickListener(this);
        ivBack.setOnClickListener(view -> this.onBackPressed());
        ivCart.setOnClickListener(this);
    }

    @SuppressLint("SetTextI18n")
    private void setProductDetails(Intent intent) {
        String productDetails = intent.getStringExtra(Constant.PRODUCT_DETAILS_KEY);
        TextView productDetail = findViewById(R.id.tvProductDescription);
```

```
    productDetail.setText(productDetails);
    TextView productName = findViewById(R.id.tvProductName);
    productName.setText(intent.getStringExtra(Constant.PRODUCT_NAME_KEY));
    TextView productPrice = findViewById(R.id.tvProductPrice);
    productPrice.setText(getString(R.string.currency_symbol) +
            intent.getIntExtra(Constant.PRODUCT_PRICE_KEY, 0));
    ImageView productImage = findViewById(R.id.tvProductImage);
    Picasso.with(this).load(intent.getStringExtra(Constant.PRODUCT_IMAGE_KEY)).into(productImage);
}

private void setUpAddToCartButton(Intent intent) {
    Button addToCartButton = findViewById(R.id.btnAddToCart);
    CartDatabase database = Room.databaseBuilder(getApplicationContext(),
            CartDatabase.class, Constant.ROOM_DB_NAME).allowMainThreadQueries()
            .fallbackToDestructiveMigration().build();
    ProductDao productDao = database.ProductDao();
    addToCartButton.setOnClickListener(view -> {
        if (productQuantity > 0) {
            productDao.insertRecord(new CartProduct(
                    0,
                    intent.getStringExtra(Constant.PRODUCT_NAME_KEY),
                    intent.getIntExtra(Constant.PRODUCT_PRICE_KEY, 0),
                    productQuantity,
                    intent.getStringExtra(Constant.PRODUCT_IMAGE_KEY)
            ));
            Toast.makeText(this, R.string.item_added_to_cart, Toast.LENGTH_SHORT).show();
        }
    });
}

@Override
public void onClick(View view) {
    int id = view.getId();
    if (id == R.id.ivAdd) {
        tvQuantity.setText(String.valueOf(++productQuantity));
    } else if (id == R.id.ivMinus) {
        --productQuantity;
        productQuantity = Math.max(productQuantity, 0);
        tvQuantity.setText(String.valueOf(productQuantity));
    } else if (id == R.id.ivBackIcon) {
        this.onBackPressed();
    } else if (id == R.id.ivCartIcon) {
```

```
        Intent cart = new Intent(this, CartActivity.class);
        startActivity(cart);
    }
  }
}
```

Product List Activity

```
package com.example.assignment.activity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ImageView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.GridLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.example.assignment.Adapter.ProductListAdapter;
import com.example.assignment.Model.ProductModel;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.google.firebase.firestore.DocumentChange;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.ArrayList;

public class ProductList extends AppCompatActivity implements View.OnClickListener {
    private ProductListAdapter productListAdapter;
    private ArrayList<ProductModel> productModelArrayList;
    private FirebaseFirestore database;
    private String documentID;
    private android.widget.SearchView svProduct;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_product_list);
        Intent intent = getIntent();
```

```
    svProduct = findViewById(R.id.svProductSearch);
    documentID = intent.getStringExtra(Constant.CATEGORY_KEY);
    setProductListRecyclerView();
    getProductData();
    searchQuery();
    setOnclickListeners();
}

private void setOnclickListeners() {
    ImageView ivBack = findViewById(R.id.ivBackIcon);
    ImageView ivCart = findViewById(R.id.ivCartIcon);
    ivBack.setOnClickListener(this);
    ivCart.setOnClickListener(this);
}

@SuppressLint("NotifyDataSetChanged")
private void getProductData() {
    database.collection(Constant.COLLECTION_PATH).document(documentID)
            .collection(Constant.SUB_COLLECTION_PATH).addSnapshotListener(((value, error) -> {
        if (error != null) {
            Log.e("data", error.getMessage());
            return;
        }
        assert value != null;
        for (DocumentChange documentChange : value.getDocumentChanges()) {
            if (documentChange.getType() == DocumentChange.Type.ADDED) {
                productModelArrayList.add(documentChange.getDocument()
                        .toObject(ProductModel.class));
            }
            productListAdapter.notifyDataSetChanged();
        }
    }));
}

private void setProductListRecyclerView() {
    RecyclerView productListRecyclerView = findViewById(R.id.rvProductList);
    productModelArrayList = new ArrayList<>();
    productListRecyclerView.setHasFixedSize(true);
    productListRecyclerView.setLayoutManager(new GridLayoutManager(this, 2,
            RecyclerView.VERTICAL, false));
    database = FirebaseFirestore.getInstance();
    productListAdapter = new ProductListAdapter(this, productModelArrayList);
```

```
        productListRecyclerView.setAdapter(productListAdapter);
    }

    private void searchQuery() {
        svProduct.setOnQueryTextListener(new android.widget.SearchView.OnQueryTextListener() {
            @Override
            public boolean onQueryTextSubmit(String s) {
                return false;
            }

            @Override
            public boolean onQueryTextChange(String newText) {
                productListAdapter.getFilter().filter(newText);
                return false;
            }
        });
    }

    @Override
    public void onClick(View view) {
        int id = view.getId();
        if (id == R.id.ivBackIcon) {
            this.onBackPressed();
        } else if (id == R.id.ivCartIcon) {
            Intent cart = new Intent(this, CartActivity.class);
            startActivity(cart);
        }
    }
}
```

Purchase Screen Activity

```
package com.example.assignment.activity;

import android.os.Bundle;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.room.Room;
```

```
import com.example.assignment.Adapter.PurchaseScreenAdapter;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.example.assignment.roomdb.CartDatabase;
import com.example.assignment.roomdb.CartProduct;
import com.example.assignment.roomdb.ProductDao;

import java.util.List;

public class PurchaseActivity extends AppCompatActivity {
    private RecyclerView purchaseRecyclerView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_purchase_screen);
        purchaseRecyclerView = findViewById(R.id.rvPurchaseScreen);
        getPurchaseData();
        setBackButton();
    }

    private void getPurchaseData() {
        CartDatabase database = Room.databaseBuilder(getApplicationContext(),
                CartDatabase.class, Constant.ROOM_DB_NAME).allowMainThreadQueries().build();
        ProductDao productDao = database.ProductDao();
        purchaseRecyclerView.setLayoutManager(new LinearLayoutManager(this,
                LinearLayoutManager.VERTICAL, false));
        List<CartProduct> cartProductList = productDao.getallProduct();
        PurchaseScreenAdapter purchaseScreenAdapter = new PurchaseScreenAdapter(cartProductList, this);
        purchaseRecyclerView.setAdapter(purchaseScreenAdapter);
        TextView totalPurchasePrice = findViewById(R.id.tvTotalPurchasePrice);
        int sum = 0, i;
        for (i = 0; i < cartProductList.size(); i++) {
            sum = sum + (cartProductList.get(i).getProductPrice() * cartProductList.get(i).getQuantity());
        }
        totalPurchasePrice.setText(getString(R.string.total_amount_tag, String.valueOf(sum)));
    }

    private void setBackButton() {
        ImageView ivBack = findViewById(R.id.ivBackIcon);
        ivBack.setOnClickListener(view -> this.onBackPressed());
```

```
    }
}
```

Register Activity

```
package com.example.assignment.activity;

import android.content.Intent;
import android.os.Bundle;
import android.text.TextUtils;
import android.util.Patterns;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.example.assignment.constant.Constant;
import com.example.assignment.R;
import com.google.firebase.auth.FirebaseAuth;

import java.util.Objects;

public class Register extends AppCompatActivity implements View.OnClickListener {
    private EditText etEmail, etPassword;
    private FirebaseAuth firebaseAuth;
    private ProgressBar progressBar;
    private Button btnRegister;
    private TextView tvLogin;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
        Objects.requireNonNull(getSupportActionBar()).hide();
        etEmail = findViewById(R.id.etRegisterEmail);
        etPassword = findViewById(R.id.etRegisterPassword);
        btnRegister = findViewById(R.id.btnRegister);
        tvLogin = findViewById(R.id.tvAlreadyHaveAccount);
```

```
    firebaseAuth = FirebaseAuth.getInstance();
    progressBar = findViewById(R.id.progressBar);
    checkCurrentUser();
    setOnClickListeners();
  }

  private void setOnClickListeners() {
    tvLogin.setOnClickListener(this);
    btnRegister.setOnClickListener(this);
  }

  private void checkCurrentUser() {
    if (firebaseAuth.getCurrentUser() != null) {
      startActivity(new Intent(getApplicationContext(), Login.class));
      finish();
    }
  }

  private void createToast(String message) {
    Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
  }

  private void registerUser() {
    String email = etEmail.getText().toString().trim();
    String password = etPassword.getText().toString().trim();
    boolean checkAllFields = validateFields(email, password);
    if (checkAllFields) {
      progressBar.setVisibility(View.VISIBLE);
      firebaseAuth.createUserWithEmailAndPassword(email, password)
          .addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
              createToast(getString(R.string.registered_successfully));
            } else {
              createToast(Objects.requireNonNull(task.getException())
                  .getMessage());
            }
            progressBar.setVisibility(View.GONE);
          });
    }
  }

  private boolean validateFields(String email, String password) {
```

```
    boolean validation = false;
    if (TextUtils.isEmpty(email)) {
        etEmail.setError(Constant.EMAIL_ERROR_MESSAGE);
    } else if (!Patterns.EMAIL_ADDRESS.matcher(email).matches()) {
        etEmail.setError(Constant.INVALID_EMAIL_ERROR_MESSAGE);
    }
    if (TextUtils.isEmpty(password)) {
        etPassword.setError(Constant.PASSWORD_ERROR_MESSAGE);
    } else if (password.length() < 6) {
        etPassword.setError(Constant.PASSWORD_LENGTH_ERROR);
    } else {
        validation = true;
    }
    return validation;
  }

  @Override
  public void onClick(View view) {
    int id = view.getId();
    if (id == R.id.tvAlreadyHaveAccount) {
        startActivity(new Intent(getApplicationContext(), Login.class));
    } else if (id == R.id.btnRegister) {
        registerUser();
    }
  }
}
```

Adapter Files

Cart Adapter

```
package com.example.assignment.Adapter;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
```

```
import androidx.recyclerview.widget.RecyclerView;
import androidx.room.Room;

import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.example.assignment.roomdb.CartDatabase;
import com.example.assignment.roomdb.CartProduct;
import com.example.assignment.roomdb.ProductDao;
import com.squareup.picasso.Picasso;

import java.util.List;

public class CartAdapter extends RecyclerView.Adapter<CartAdapter.CartViewHolder> {
    private final List<CartProduct> cartProductList;
    private final Context context;

    public CartAdapter(List<CartProduct> cartProducts, Context context) {
        this.cartProductList = cartProducts;
        this.context = context;
    }

    @NonNull
    @Override
    public CartViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(context)
                .inflate(R.layout.cart_item_layout, parent, false);
        return new CartViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull CartViewHolder holder, int position) {
        CartProduct cartProduct = cartProductList.get(position);
        holder.tvProductName.setText(cartProduct.getProductName());
        holder.tvProductPrice.setText(context.getString(R.string.single_unit,
                String.valueOf(cartProduct.getProductPrice())));
        holder.tvQuantity.setText(context.getString(R.string.quantity_tag,
                String.valueOf(cartProduct.getQuantity())));
        Picasso.with(context).load(cartProduct.getProductImageUrl()).into(holder.ivProductImage);
        holder.ibRemoveItem.setOnClickListener(view -> {
            CartDatabase database = Room.databaseBuilder(context,
                    CartDatabase.class, Constant.ROOM_DB_NAME).allowMainThreadQueries().build();
            ProductDao productDao = database.ProductDao();
```

```
            productDao.deleteById(cartProduct.getProdouctId());
            cartProductList.remove(position);
            notifyItemRemoved(position);
        });
    }

    @Override
    public int getItemCount() {
        return cartProductList.size();
    }

    public static class CartViewHolder extends RecyclerView.ViewHolder {
        TextView tvProductPrice, tvQuantity, tvProductName;
        ImageView ivProductImage;
        ImageButton ibRemoveItem;

        public CartViewHolder(@NonNull View itemView) {
            super(itemView);
            tvProductPrice = itemView.findViewById(R.id.tvPrice);
            tvProductName = itemView.findViewById(R.id.tvProductName);
            tvQuantity = itemView.findViewById(R.id.tvQuantity);
            ivProductImage = itemView.findViewById(R.id.tvProductImage);
            ibRemoveItem = itemView.findViewById(R.id.ibRemoveItem);
        }
    }
}
```

Categories Adapter

```
package com.example.assignment.Adapter;

import android.content.Context;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;
```

```
import com.example.assignment.Model.CategoriesModel;
import com.example.assignment.activity.ProductList;
import com.example.assignment.R;
import com.example.assignment.constant.Constant;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;

public class CategoriesAdaptor extends RecyclerView.Adapter<CategoriesAdaptor.CategoriesViewHolder> {
    private final Context context;
    private final ArrayList<CategoriesModel> categoriesModelArrayList;

    public CategoriesAdaptor(Context context, ArrayList<CategoriesModel> categoriesModelArrayList) {
        this.context = context;
        this.categoriesModelArrayList = categoriesModelArrayList;
    }

    @NonNull
    @Override
    public CategoriesViewHolder onCreateViewHolder(
            @NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(context).inflate(
                R.layout.all_category_item_layout, parent, false);
        return new CategoriesViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull CategoriesViewHolder holder, int position) {
        CategoriesModel categoriesModel = categoriesModelArrayList.get(position);
        Picasso.with(context).load(categoriesModel.getCategoryIconUrl()).into(holder.ivCategoryImage);
        holder.tvCategoryTitle.setText(categoriesModel.getCategoryName());
        holder.ivCategoryImage.setOnClickListener(view -> {
            Intent intent = new Intent(context, ProductList.class);
            intent.putExtra(Constant.CATEGORY_KEY, categoriesModel.getCategoryName());
            context.startActivity(intent);
        });
    }

    @Override
    public int getItemCount() {
        return categoriesModelArrayList.size();
    }
```

```
    public static class CategoriesViewHolder extends RecyclerView.ViewHolder {
        TextView tvCategoryTitle;
        ImageView ivCategoryImage;

        public CategoriesViewHolder(@NonNull View itemView) {
            super(itemView);
            ivCategoryImage = itemView.findViewById(R.id.categoryImageIV);
            tvCategoryTitle = itemView.findViewById(R.id.categoryTitleTV);
        }
    }
}
```

Product List Adapter

```
package com.example.assignment.Adapter;

import android.annotation.SuppressLint;
import android.content.Context;
import android.content.Intent;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Filter;
import android.widget.Filterable;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.assignment.Model.ProductModel;
import com.example.assignment.R;
import com.example.assignment.activity.ProductDetails;
import com.example.assignment.constant.Constant;
import com.squareup.picasso.Picasso;

import java.util.ArrayList;
import java.util.Collection;
import java.util.List;

public class ProductListAdapter extends RecyclerView.Adapter<ProductListAdapter.ProductViewHolder> implements
```

```
Filterable {
    private final Context context;
    private final ArrayList<ProductModel> productModelArrayList;

    public ProductListAdapter(Context context, ArrayList<ProductModel> productModelArrayList) {
        this.context = context;
        this.productModelArrayList = productModelArrayList;
    }

    @NonNull
    @Override
    public ProductViewHolder onCreateViewHolder(
            @NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(context).inflate(
                R.layout.product_list_item_layout, parent, false);
        return new ProductViewHolder(view);
    }

    @SuppressLint("SetTextI18n")
    @Override
    public void onBindViewHolder(@NonNull ProductViewHolder holder, int position) {
        ProductModel productModel = productModelArrayList.get(position);
        Picasso.with(context).load(productModel.getImageUrl()).into(holder.ivProductImage);
        holder.tvProductPrice.setText(context.getString(R.string.price_tag,
                String.valueOf(productModel.getPrice())));
        holder.tvProductName.setText(context.getString(R.string.name_tag) + productModel.getName());
        holder.ivProductImage.setOnClickListener(view -> {
            Intent productDetail = new Intent(context, ProductDetails.class);
            productDetail.putExtra(Constant.PRODUCT_NAME_KEY, productModel.getName());
            productDetail.putExtra(Constant.PRODUCT_PRICE_KEY, productModel.getPrice());
            productDetail.putExtra(Constant.PRODUCT_DETAILS_KEY, productModel.getDescription());
            productDetail.putExtra(Constant.PRODUCT_IMAGE_KEY, productModel.getImageUrl());
            context.startActivity(productDetail);
        });
    }

    @Override
    public int getItemCount() {
        return productModelArrayList.size();
    }

    @Override
```

```
public Filter getFilter() {
    return filter;
}

Filter filter = new Filter() {
    @Override
    protected FilterResults performFiltering(CharSequence charSequence) {
        List<ProductModel> filteredList = new ArrayList<>();
        if (charSequence.toString().isEmpty()) {
            filteredList.addAll(productModelArrayList);
        } else {
            for (ProductModel product : productModelArrayList) {
                if (product.getName().toLowerCase().contains(charSequence.toString().toLowerCase())) {
                    filteredList.add(product);
                }
            }
        }
        FilterResults filterResults = new FilterResults();
        filterResults.values = filteredList;
        return filterResults;
    }

    @SuppressLint("NotifyDataSetChanged")
    @Override
    protected void publishResults(CharSequence charSequence, FilterResults filterResults) {
        productModelArrayList.clear();
        productModelArrayList.addAll((Collection<? extends ProductModel>) filterResults.values);
        notifyDataSetChanged();
    }
};

public static class ProductViewHolder extends RecyclerView.ViewHolder {
    TextView tvProductPrice, tvProductName;
    ImageView ivProductImage;

    public ProductViewHolder(@NonNull View itemView) {
        super(itemView);
        tvProductName = itemView.findViewById(R.id.tvProductItemName);
        tvProductPrice = itemView.findViewById(R.id.tvProductItemPrice);
        ivProductImage = itemView.findViewById(R.id.ivProductItem);
    }
```

```
    }
}
```

Purshcase Screen Adapter

```
package com.example.assignment.Adapter;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.example.assignment.R;
import com.example.assignment.roomdb.CartProduct;
import com.squareup.picasso.Picasso;

import java.util.List;

public class PurchaseScreenAdapter extends RecyclerView.Adapter<PurchaseScreenAdapter.PurchaseViewHolder> {
    private final List<CartProduct> cartProducts;
    private final Context context;

    public PurchaseScreenAdapter(List<CartProduct> cartProducts, Context context) {
        this.cartProducts = cartProducts;
        this.context = context;
    }

    @NonNull
    @Override
    public PurchaseViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(parent.getContext())
                .inflate(R.layout.purchase_screen_item_layout, parent, false);
        return new PurchaseViewHolder(view);
    }


    @Override
```

```
    public void onBindViewHolder(@NonNull PurchaseViewHolder holder, int position) {
        CartProduct cartProduct = cartProducts.get(position);
        holder.tvProductName.setText(cartProduct.getProductName());
        holder.tvProductPrice.setText(context.getString(R.string.price_tag,
                String.valueOf(cartProduct.getProductPrice())));
        holder.tvQuantity.setText(context.getString(R.string.quantity_tag,
                String.valueOf(cartProduct.getQuantity())));
        Picasso.with(context).load(cartProduct.getProductImageUrl()).into(holder.ivProductImage);
        int sum = 0;
        sum = (sum + (cartProduct.productPrice * cartProduct.getQuantity()));
        holder.tvTotalPrice.setText(context.getString(R.string.total_amount_tag, String.valueOf(sum)));
    }

    @Override
    public int getItemCount() {
        return cartProducts.size();
    }

    public static class PurchaseViewHolder extends RecyclerView.ViewHolder {
        TextView tvProductPrice, tvQuantity, tvProductName, tvTotalPrice;
        ImageView ivProductImage;

        public PurchaseViewHolder(@NonNull View itemView) {
            super(itemView);
            tvProductPrice = itemView.findViewById(R.id.tvPrice);
            tvProductName = itemView.findViewById(R.id.tvProductName);
            tvQuantity = itemView.findViewById(R.id.tvQuantity);
            ivProductImage = itemView.findViewById(R.id.ivProductImage);
            tvTotalPrice = itemView.findViewById(R.id.tvTotalPrice);
        }
    }
}
```

Model

Categories Model

```
package com.example.assignment.Model;

public class CategoriesModel {
    String categoryIconUrl;
    String categoryName;
```

```
    public CategoriesModel() {
    }

    public String getCategoryIconUrl() {
        return categoryIconUrl;
    }

    public String getCategoryName() {
        return categoryName;
    }
}
```

Product Model

```
package com.example.assignment.Model;

public class ProductModel {
    String name;
    int price;
    String imageUrl;
    String description;

    public ProductModel() {
    }

    public String getName() {
        return name;
    }

    public int getPrice() {
        return price;
    }

    public String getImageUrl() {
        return imageUrl;
    }

    public String getDescription() {
        return description;
    }
}
```

Room DB

Cart Database

```
package com.example.assignment.roomdb;

import androidx.room.Database;
import androidx.room.RoomDatabase;

@Database(entities = {CartProduct.class}, version = 3)
public abstract class CartDatabase extends RoomDatabase {
    public abstract ProductDao ProductDao();
}
```

Cart Product entity files

```
package com.example.assignment.roomdb;

import androidx.room.ColumnInfo;
import androidx.room.Entity;
import androidx.room.PrimaryKey;

@Entity
public class CartProduct {
    @PrimaryKey(autoGenerate = true)
    public int prodouctId;

    @ColumnInfo(name = "productName")
    public String productName;

    @ColumnInfo(name = "productPrice")
    public int productPrice;

    @ColumnInfo(name = "quantity")
    public int quantity;

    @ColumnInfo(name = "imageUrl")
    public String productImageUrl;

    public CartProduct(int prodouctId, String productName, int productPrice, int quantity,String productImageUrl) {
        this.prodouctId = prodouctId;
```

```
        this.productName = productName;
        this.productPrice = productPrice;
        this.quantity = quantity;
        this.productImageUrl = productImageUrl;
    }

    public int getProdouctId() {
        return prodouctId;
    }

    public String getProductName() {
        return productName;
    }

    public String getProductImageUrl() {
        return productImageUrl;
    }

    public int getProductPrice() {
        return productPrice;
    }

    public int getQuantity() {
        return quantity;
    }
}
```

Product Dao File

```
package com.example.assignment.roomdb;

import androidx.room.Dao;
import androidx.room.Insert;
import androidx.room.Query;

import java.util.List;

@Dao
public interface ProductDao {
    @Insert
    void insertRecord(CartProduct product);
```

```
    @Query("SELECT*FROM cartproduct")
    List<CartProduct> getallProduct();

    @Query("DELETE FROM CartProduct WHERE prodouctId = :id")
    void deleteById(int id);
}
```

Constant File

```
package com.example.assignment.constant;

public class Constant {
    public static String EMAIL_ERROR_MESSAGE = "Email is required";
    public static String COLLECTION_PATH = "categories";
    public static String CATEGORY_KEY = "category";
    public static String SUB_COLLECTION_PATH = "product list";
    public static String ROOM_DB_NAME = "cart";
    public static String SUGGESTED_FOR_YOU_COLLECTION_PATH ="suggested for you";
    public static String INVALID_EMAIL_ERROR_MESSAGE = "Invalid type email";
    public static String PASSWORD_ERROR_MESSAGE = "Password is required";
    public static String PASSWORD_LENGTH_ERROR = "Password must be greater or equal to 6 characters";
    public static String PRODUCT_NAME_KEY= "productName";
    public static String PRODUCT_PRICE_KEY= "productPrice";
    public static String PRODUCT_DETAILS_KEY= "productDetails";
    public static String PRODUCT_IMAGE_KEY= "productImage";
}
```

Plot No. 1/2, Kanakpura Industrial Area, Kanakpura, Sirsi Road, Jaipur-302034
Site No.271, Sri Ganesha Complex, Hosur Main Road, BTM 1st Stage, Bangalore- 560068.

Date: 26th May'22

TO WHOMSOEVER IT MAY CONCERN

Employee Name: Dhruv Patel

Employee ID: ITTV/EMP/2671
Address: Plot no. 1 / 2, Kanakpura Industrial Area, Kanakpura. Sirsi Road, Jaipur-302034.

This is to certify that **Mr. Dhruv Patel** is pursuing internship followed by employment with **In Time Tec Visionsoft Pvt. Ltd** from **1st Feb'22** at our **Jaipur** office located at **Plot no. 1/2, Kanakpura Industrial Area, Kanakpura. Sirsi Road, Jaipur-302034.**
During his Internship period, he is entitled to a stipend of Rs. 15,000/- per month, other details are valid as per the offer letter.

For InTimeTec Visionsoft Pvt. Ltd.
Neha. Dubey
Authorised Signatory

Regards,

Neha Dubey
Sr. HR Associate
In Time Tec Visionsoft (p) Ltd.

Confidential

This is a Computer Generated Document. No Signature Required.