Q1/7 Match The Following
(4/4 Marks)

Match the terms given in column A to their corresponding description in column B.

Kotlin	Used to add functionality to the UI of an Android app.
Android Studio	Contains tools and uses libraries to create an Android app.
XML	Used to develop the user interface of an Android app.
Android SDK	Contains all the libraries used for Android development.

Q2/7 Multiple Choice Question
(2/2 Marks)

"I am a versatile environment and I have various tools like compiler, editor, file manager, etc. and can also use external resources to provide a user friendly outcome. Who am I?"

ASDK

BXML

[✓ **C**IDE]

DKotlin

Explanation
A IDE is an environment or software application which comprises of various tools like an editor, compiler, package manager, automation tools and debugger.

Q3/7 Multiple Choice Question
(1/1 Marks)

"I carry data which can be displayed or presented over the internet. Who am I?"

ASDK

[✓ **B**XML]

CIDE

DWebsite

Explanation
XML is a markup language which can be used to make the UI (also known as views), and can also carry data over the internet.

Q4/7 Multiple Choice Question
(1/1 Marks)

"I form the functional back-end of any Android application. Who am I?"

AKotlin

BJava

CC/C++

DAll of the above

Explanation
Kotlin, Java and C/++ all three of them can be used to create the functionality of an Android app.

Q5/7 Multiple Choice Question
(2/2 Marks)

What will be the number of the upcoming version of Android after Pie?

A9

B10

C11

D12

Explanation
From the pattern of the alphabets observed in the previous Android versions, we see that the versions are following the A, B, C, D, E, ….and so on alphabet format. Also, the numbers of the versions have been 4 for Kitkat, 5 for Lollipop, and so on. Hence, we can easily say that the next version of Android will have its number as 10 since Pie was Android 9.

Q6/7 Multiple Choice Question
(1/1 Marks)

Given:

The name of the tag is 'Koto'.

Which of these will be its opening tag?

A<Koto/>

B<Koto>

C</Koto>

D/<Koto>

Explanation
The opening tag for any XML tag (here Koto) is the name of the tag written inside angular brackets.

Q7/7 Multiple Choice Question
(1/1 Marks)

Given:

The name of the function is 'lino'.

Choose the correct function definition for this function.

A

fun lino() {

....

....

}

B
lino(){

....

....

}

C
fun lino{

....

....

}

D
fun lino()

....

....

Explanation
The function definition in Kotlin looks like this:

fun function_name(){
....
....
}

Here the function name is 'lino'.
And therefore option 1 is the correct answer here.

Q1/15 Multiple Choice Question
(1/1 Marks)

How can we define and initialise a variable 'a' equal to 4 such that the value of 'a' cannot be altered in later stages of the program?

A

```
var a = 4
```

B

```
var a: Int

a = 4
```

C ✓

```
val a = 4
```

D

```
var a: Int = 4
```

Explanation
In cases where the value of a variable does not change, we use 'val'.

Q2/15 Multiple Choice Question
(2/2 Marks)

Which of these is the correct way to create an object of the class 'Android'?

A ✓

```
var androidObject = Android()
```

B

```
var androidObject = new Android()
```

C

```
var androidObject in Android()
```

D

```
var androidObject = Android
```

Explanation
The syntax to create an object of any class in Kotlin is:

```
var <name_of_object> = <Class_name>()
```

Hence, here for class 'Android', the object would be:

```
var androidObject = Android()
```

Q3-5/15 Reading Comprehension
(8/8 Marks)

Passage

Consider the given array:

```
var letters = arrayListOf(“a”, “b”, “c”, “d”, “e”, “f”, “g”, “h”,
“i”, “j”, “k”, “l”, “m”, “n”, “o”)
```

and answer the questions (3-5) that follow.

Solution

Q3/15
(3/3 Marks)

If we have another array

```
var new = arrayOf("p", "q", "r")
```

then what will be the output for the following code:

```
letters.addAll(new)
```

A
[a, b, c, d, e, f, g, h, i, j, k, l, m, n, o]

[✓ **B**
[a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r]]

C
[p, q, r, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o]

D
[p, q, r]

Explanation
All the members of the new arraylist are appended at the end of the current arraylist.

Q4/15
(2/2 Marks)

After performing the above, what will be the output of the following:

```
println(letters.get(4))
```

A ✓
e

B
f

C
b

D
Runtime Error

Explanation
The index starts from 0 and hence the 5th element is present at the index 4.

Q5/15
(3/3 Marks)

What will be the output of the following function:

```
letters.set(6, "x")
```

A
[a, b, c, d, e, x, g, h, i, j, k, l, m, n, o]

B
[p, q, r, a, b, c, x, e, f, g, h, i, j, k, l, m, n, o]

C ✓
[a, b, c, d, e, f, x, h, i, j, k, l, m, n, o, p, q, r]

D
Runtime error

Explanation
Here the 7th element present at index 6 is replaced by the given string 'x'.

Q6/15 Multiple Choice Question
(2/2 Marks)

What will be the output of the piece of code given below:

```
for(i in 1..20 step 3227 % 5){
    print(i)
}
```

A
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

B
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

C
1 3 5 7 9 11 13 15 17 19

D
2 4 6 8 10 12 14 16 18 20

Explanation
The step gives us the jump for the loop and in this case the step is calculated as 3227%5 which is equal to 2. Hence the step is 2 and we start the iteration from 1.

Q7/15 Multiple Choice Question
(3/3 Marks)

What will be the output for the given code:

```
var sum = 0
var i = 10
while (i != 0) {
    sum += i // sum = sum + i
    --i
}
println(sum)
```

A
1 2 3 4 5 6 7 8 9 10

B
10 9 8 7 6 5 4 3 2 1

C
55

D
50

Explanation
Here the while loop runs from 10 to 1 adding all the elements and putting the value in 'sum'. Finally, after the loop ends the value of 'sum' is printed which will hold the sum of all integers from 10 to 1.

Q8/15 Multiple Choice Question
(3/3 Marks)

What will be the output for the given code:

```
val a = 10
val b = 10
val max = if (a > b) a else b
```

```
when (max) {
    in 1..10 -> print("x is in the range")
    !in 10..20 -> print("x is outside the range")
    else -> print("none of the above")
}
```

A
x is in the range

B
x is outside the range

C
None of the above

D
Error

Explanation
Here the value of max will be 10. Then this value is checked with the various conditions inside the when statement. The first condition is satisfied since 10 lies in 1 to 10 (in keyword considers both the upper and the lower limit). Hence 'x is in the range' is printed.

Q9/15 Multiple Choice Question
(2/2 Marks)

```
1. val a = 10
2. for(i in 1..10) {
3.     val b = i
4. }
5. print (b)
```

The program given above produces compilation error when executed. Which of these changes will rectify the given code to give a successful result?

A
Change line 1 to:

var a = 10

B
Put the print statement in line 5 inside the 'for-loop' block

C
Change line 3 to:

var b = i

D
Put the declaration of variable 'a' inside the 'for-loop'

Explanation

The print statement in the program is trying to print the variable which is outside its scope. Hence, if we put the print statement inside the 'for-loop', the code will run without any errors.

Q10/15 Multiple Choice Question
(4/4 Marks)

Which of these ways will be used to create a data class 'Person' for the code given below, such that the output is:

Koto, 19

Lino, 21

```
fun addNames(): List<Person>{
    return listOf(Person("Koto", 19), Person("Lino", 21))
}

fun main(){
    var names =addNames()
    for (member in names){
        println("${member.name}, ${member.age}")
    }
}
```

A

```
data class Person()
```

B

```
data class Person(val Koto = 19, val Lino = 21)
```

C

```
data class Person(var name, var age)
```

D ✓

```
data class Person(var name: String, var age: Int)
```

Explanation
Here the requirements of the Person class are name and age. Hence the data class will have two parameters i.e. name and age where name is of String type and age being a number is of Integer type.

Q11/15 Multiple Choice Question
(6/6 Marks)

Which of these is the correct way to write the function sendMessageToClient such that all null cases are tested:

```
fun sendMessageToClient(client: Client?, message: String?, mailer: Mailer){
    //write your null tests here
    //Hint: message is sent using this code: mailer.sendMessage(email,
message)
}


class Client (val personalInfo: PersonalInfo?)


class PersonalInfo (val email: String?)interface Mailer {
    fun sendMessage(email: String, message: String)
}
```

A

```
val email = client?.personalInfo?.email


mailer.sendMessage(email, message)
```

B ✓

```
val email = client?.personalInfo?.email


if(email != null && message!= null){
    mailer.sendMessage(email, message)
}
```

C

```
val email = client?.personalInfo?.email


if(email != null){
    mailer.sendMessage(email, message)
}
```

D

```
val email = client?.personalInfo?.email


if(email != null || message!= null){
    mailer.sendMessage(email, message)
}
```

Explanation
Here in order to send the message inside the sendMessageToClient() method, we need an email and a mailer to send the message. In these we see that client and message can be null while mailer is not null. Also, the client contains personal information which in turn contains the email. Hence our email will be:

val email = client?.personalInfo?.email

The variables used above are all nullable as they all have a '?' in the data type and in turn this new email can also be null. Now, to make sure nothing is null we use the if condition on the nullable quantities i.e. email and message. Therefore, the result is:

```
if(email != null && message != null){
    mailer.sendMessage(email, message)
}
```

We do not check the mailer to be null since it was orginally a non null value as there was no '?' after it.

Q12/15 Multiple Choice Question
(3/3 Marks)

Reduce the given function into single line:

```
fun matchPassword(pass: String, confirmPass: String): Boolean {
    if (pass == confirmPass){
        return true
    }
    else {
        return false
    }
}
```

A

```
fun matchPassword(pass, confirmPass): Boolean {
    return pass = confirmPass
}
```

B

```
fun matchPassword(pass: String, confirmPass: String) {
    return pass = confirmPass
}
```

C

```
fun matchPassword(pass: String, confirmPass: String): Boolean {
    return pass == confirmPass
}
```

D

```
fun matchPassword(pass: String, confirmPass: String): Boolean {
    pass == confirmPass
}
```

Explanation
Here we see that both the if body and the else body have a return statement and we only need to check whether pass == confirmPass, so we directly return the checked part. Since the return type of the function is boolean, it will always return a boolean value only. Also the equality operator returns a boolean value itself, so we do not need to explicitly write the return statements for each. Hence the reduced code will be :

return pass == confirmPass

Q13/15 Multiple Choice Question
(3/3 Marks)

Look at the interface given below:

```
interface AndroidStudio{


    fun writeCode()
    fun runApp()
    fun integrateSDK()


}
```

What will be the correct implementation of this interface in the AndroidApp class?

A

```
class AndroidApp implements AndroidStudio {
     override fun writeCode() {
         print("I use this function to write the code")
    }

     override fun runApp() {
         print("I use this function to run the app")
    }

     override fun integrateSDK() {
         print("I use this function to integrate the SDK")
    }

}
```

B

```
class AndroidApp : AndroidStudio {
     override fun writeCode() {
         print("I use this function to write the code")
    }

     override fun runApp() {
         print("I use this function to run the app")
    }

     override fun integrateSDK() {
         print("I use this function to integrate the SDK")
    }

}
```

C

```
class AndroidApp : AndroidStudio {
     fun writeCode() {
         print("I use this function to write the code")
```

```
    }

     fun runApp() {
         print("I use this function to run the app")
    }

     fun integrateSDK() {
         print("I use this function to integrate the SDK")
    }

}
```

D

```
class AndroidApp : AndroidStudio() {
     override fun writeCode() {
         print("I use this function to write the code")
    }

     override fun runApp() {
         print("I use this function to run the app")
    }

     override fun integrateSDK() {
         print("I use this function to integrate the SDK")
    }


}
```

Explanation
The correct implementation of interface AndroidStudio into the class AndroidApp will be:

```
class AndroidApp : AndroidStudio {
      override fun writeCode() {
          print("I use this function to write my code")
     }

      override fun runApp() {
            print("I use this function to run my app")
        }

         override fun integrateSDK() {
            print("I use this function to integrate the SDK")
        }

}
```

Q14/15 Arrange In Order
(0/2 Marks)

Arrange the following in the decreasing order of their visibility.

- :::public
- :::internal
- :::private
- :::protected

Solution
public

internal

protected

private

Explanation
public - Public has the most visibility as it is visible throughout the project.

internal - Internal has the second highest visibility as internal components are visible throughout the module.

protected - Protected components are visible to only the class members and subclasses.

private - Private is only visible to the class members.

Q15/15 Match The Following
(4/4 Marks)

Match the symbols with their correct operations.

Arithmetic
+, -, *, /, %

Logical
&&, ||, !

Assignment
=, +=, -=, *=, /=

Comparison
>, <, >=, <=, ==

Explanation
Arithmetic: +, -, *, /, %

Logical: &&, ||, !

Assignment: =, +=, -=, *=, /=

Comparison: >, <, >=, <=, ==

1. **Which IDE are we using for the development of Android applications?**

   A IntelliJ IDEA
   B Android Studio
   C MS Word
   D Notepad

   Explanation

   For developing Android applications, the primary IDE is Android Studio and we are also going to use the same.

2. **In which directory do we place the images used in the application?**

   A Drawable
   B Mipmap
   C Layout
   D Values

   Explanation

   All the images and design files used in our application will be placed inside the Drawable directory.

3. **I convey all the information about the application to the Android OS and to the Play Store. Who am I?**

   A activity_main.xml
   B build.gradle(Module: app)
   C AndroidManifest.xml
   D com.internshala.activitylifecycle

   Explanation

   AndroidManifest.xml file contains all the information of the app and it describes it to the Android Operating System and also to the Play Store.

4. **Which tag is used to declare an activity in the manifest file?**

   A <manifest>
   B <application>
   C <activity>
   D <action>

Explanation

Inside the manifest file, an activity is declared using the <activity> tag as a child of the <application> tag.

5. In which file is the version of Kotlin declared?

A build.gradle(Module: app)
B build.gradle(Project: ActivityLifecycle)
C AndroidManifest.xml
D Activity_main.xml

Explanation

The version of Kotlin and Android Studio are project level things which are declared inside the build.gradle(Project: ActivityLifecycle) file.

6. What is the API level for Android Pie?

A API 26
B API 27
C API 28
D API 29

Explanation

According to the chart present at the link (click here), the API level for Android Pie is API 28.

1. **How many methods are there in an activity lifecycle?**

   A 5
   B 6
   C 7
   D 8

   Explanation

   There are a total of 7 different lifecycle methods in the activity lifecycle, namely: onCreate(), onStart(), onResume(), onPause(), onStop(), onRestart(), onDestroy().

2. **Which method is initially called when the activity is first presented on the screen?**

   A onCreate()
   B onStart()
   C onResume()
   D onRestart()

   Explanation

   When the activity is first presented on the screen, it gets created, and hence, the onCreate() method is called.

3. **Which method is not called when we press the home button from an activity?**

   A onPause()
   B onStop()
   C onDestroy()
   D None of the above

   Explanation

   When we press the home button, the activity is not destroyed but stopped instead, hence the onDestroy() method is not called.

4. **Which lifecycle method will be called when the phone unexpectedly shuts down?**

   A onPause()
   B onStop()
   C onDestroy()

D None of the above

Explanation

When the device unexpectedly shuts down, the OS calls onDestroy() on everything.

5. **After which lifecycle method, does the activity become ready for user interaction?**

   A onCreate()
   B onStart()
   C onResume()
   D onRestart()

   Explanation

   When the activity is created and started, using the onCreate() and onStart() methods, the activity is just visible to the user but the user cannot interact with it. It is only when the onResume() method is called that the activity becomes ready for user interaction.

6. **Which is the mandatory lifecycle method that always needs to be implemented in our activity?**

   A onCreate()
   B onStart()
   C onResume()
   D onRestart()

   Explanation

   It is mandatory to use the onCreate() method in an activity class.

7. **Where can we find the resource IDs of all the resources used in our project?**

   A Manifest file
   B res directory
   C R file
   D Gradle files

Explanation

Android automatically creates the resource IDs of all the resources used in an Android project. For this, the Android Studio automatically creates an ‘R’ file.

8. What is used to send data from one activity to another?

A String
B Bundle
C Integer
D We cannot send data from one activity to another.

Explanation

Bundles are generally used for passing data between various Android activities. This data can be in the form of a string, integer, boolean, etc.

1. **Which ViewGroup will be preferred if we have three Views aligned in a vertical order?**

   A LinearLayout
   B RelativeLayout
   C ScrollView
   D None of the above

   Explanation
   In order to align the Views vertically in a linear order, we prefer to use the **LinearLayout.**

2. **Which attribute can be used to align the Views in the center horizontally inside a LinearLayout?**

   A android:layout_alignParenCenter
   B android:center
   C android:layout_gravity
   D android:layout_centerInParent

   Explanation
   android:layout_gravity = “center_horizontal” is used to align the views in the center of the screen horizontally.

3. **Which is the preferred scale when giving the size of a text?**

   A dp
   B sp
   C px
   D No scale

   Explanation

   For giving the size of a text, we use the scale **sp** (scalable pixels).

4. **What does dp stand for?**

   A dense pixels
   B density pixels
   C density dependent pixels
   D density independent pixels

Explanation

The full form of **dp** is density independent pixels. This is used so that a particular given size of any View or **ViewGroup** is constant, though the displays may have different pixel density.

5. **Which image format is preferred for storing local images in the drawable directory?**

   A PNG
   B JPG
   C JPEG
   D BMP

   Explanation
   According to the convention, **PNG** is the most preferred image format for storing local images inside the drawable directory.

6. **Which tag will be used to create a View which accepts the user's mobile number?**

   A TextView
   B EditText
   C ImageView
   D Button

   Explanation
   When we need to accept user input from a **View** we need to add an **EditText** tag.

Views can be aligned with respect to each other as well as with respect to the parent using the RelativeLayout.

A True
B False

Explanation

As the name suggests, **RelativeLayout** is used to align the Views in relation to the other Views and **ViewGroups** present.

Which attribute in the RelativeLayout can be used to align Views one below the other?

A android:below
B android:below_layout
C android:layout_below
D android:bottom

Explanation

To align Views one below the other in a **RelativeLayout**, we need to use the **android:layout_below** attribute.

Which tag will be used to display the privacy policy in an app?

A TextView
B EditText
C ImageView
D Button

Explanation

Since the privacy policy will be a single document containing text, we will use a **TextView** to display it.

Which tag will be used to display a company logo?

A TextView
B EditText
C ImageView
D Button

Explanation

Since the company logo will be an image, hence we will use an **ImageView** to display the same.

Which attribute is used to align the Views to the top of the parent in the RelativeLayout?

A android:alignParent
B android:gravity
C android:layout_alignTop
D android:layout_alignParentTop

Explanation

The **android:layout_alignParentTop** attribute is used to align the Views to the top of the parent in the **RelativeLayout**.

Which attribute is used to give the text inside the button?

A android:text
B android:buttonText
C android:textButton
D android:name

Explanation

The attribute **android:text** is used to give the text inside the button.

Layout orientation does not have any effect on the Relative layout.

A True
B False

Explanation

Since all the Views inside the Relative Layout are arranged in relation to one another, the layout orientation does not have any significance in the **RelativeLayout**.

1. **How many children can a ScrollView have?**

   A One
   B Two
   C Three
   D More than three

   Explanation

   A **ScrollView** can only have one child. However, it can have multiple grandchildren.

2. **Which of the following is not a ViewGroup?**

   A LinearLayout
   B RelativeLayout
   C TextView
   D ScrollView

   Explanation

   TextView cannot have any children and hence is a View whereas the others can be used as the root element (parent tag) inside a layout file.
   Although the ScrollView by its name seems like a View, but it is not, since it can have other Views and ViewGroups inside i

3. **What is the maximum pages of scrolling for which a ScrollView can be used?**

   A One page
   B Two pages
   C Three pages
   D Not-defined

   Explanation

   Using the **ScrollView**, we can make the page as long as we want. There is no such limit defined.

4. **Which tag would you use if you want the user to input his/her name?**

   A TextView
   B EditText
   C ImageView
   D Button

   Explanation

EditText is the field where a user can input text and hence we will use the same to get the name of the user.

5. **According to good coding practices, which of these names will you give to the ID of an image of a company's logo?**

   A companyLogo
   B imgImage
   C imgLogo
   [✓ **D imgCompanyLogo**]

   Explanation

   imgCompanyLogo would be the most preferred choice as it gives the details of the View i.e. **ImageView** and also the image inside that View i.e. company's logo.

6. **Do the yellow highlighted lines in the code create an issue in the execution of the app?**

   A Yes
   [✓ **B No**]

   Explanation

   No, as they are just warnings and can be used to make some changes, but never hinder the execution of the code.

7. **Predict the tag of the View which is present to the left of the given View:**

```
<TextView
android:id="@+id/txtEmail6"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:text="natasha@avengers.com"
android:layout_toRightOf="@id/imgNatasha"
android:textSize="18sp"
android:layout_below="@id/txtName6"
android:textColor="#000000"
```

```
android:layout_marginTop="15dp"/>
```

A EditText
B TextView
C ImageView
D Button

Explanation

The View present on the left will be the one to which this **TextView** is aligned right of. Hence we check the attribute **android:layout_toRightOf** and there we have the ID of the required View which is **imgNatasha**. By the given ID we can understand that the required View is an **ImageView**.

1. **Which colour is displayed on the Toolbar of the app?**

   [✓ A colorPrimary]
   B colorPrimaryDark
   C colorAccent
   D None of the above

   Explanation

   The colorPrimary defined inside the colors.xml file is the colour displayed on the Toolbar of the app.

2. **Which of the following is not a valid hex colour code?**

   A #000000
   B #ffffff
   [✓ C 002341#]
   D #ad1457

   Explanation

   The hex codes of the colours contain a '#' symbol followed by 6 characters.

3. **Which of these is the correct way to create a string inside the strings.xml file?**

   A `<string name=my_string></string>`
   B `<string name="my_string">My String</string>`
   C `<string name="my_string" My String</string>`
   D `<string name="my_string">My String<string>`

   Explanation

   In option 1, we haven't added the quotes to the string name and the value is also missing.
   In option 3, there is no closing angle bracket for the opening tag.
   In option 4, the closing tag of <string> does not have the backward slash.

4. **How many launcher activities can be present in an Android app?**

   [✓ A One]
   B Two
   C Three
   D Four

Explanation

An Android app can have only one launcher activity but more than one main activity.

5. **Which attribute in the AndroidManifest.xml is used to give the app name?**

   A android:icon
   B android:name
   C android:theme
   D android:label

   Explanation

   The attribute android:label inside the AndroidManifest.xml file is used to give the app name.

6. **Which of these methods is used to add a click to a Button?**

   A setClickListener { … }
   B setOnClickListener { … }
   C setonclicklistner { … }
   D clickListener { ... }

   Explanation

   In order to add a click listener to a button we add a setOnClickListener { … } method.

7. **The OnClickListener interface is present in which of these classes?**

   A MainActivity
   B AppCompatActivity
   C View
   D Button

   Explanation

   The interface OnClickListener is present inside the View class. Hence when we implement it we use View.OnClickListener inside the MainActivity class.

8. **Look at the code for a button given below:**

```
btnMessage.setOnClickListener()
```

Inside the onClick() method we have:

```
Toast.makeText(this@MainActivity, “I am Zoltar, the
future predictor”, Toast.LENGTH_SHORT).show()
```

What will be the output when we click the given button?

A I am Zoltar, the future predictor
B I am Zoltar
C Compilation error
D App stopped unfortunately

Explanation

This will result in a compilation error since inside the **setOnClickListener** we have not specified the context i.e. ‘this’. Hence, the correct implementation would have been:

```
btnMessage.setOnClickListener(this)
```

1. **Intents create a bridge between two activities.**

   A True [✓]
   B False

   Explanation

   Since an intent declares the direction of movement from one activity to another, it can be thought of as a bridge between two activities.

2. **How do we retrieve text as String from an EditText?**

(Given: The variable name for EditText is editText).

   A editText.txt
   B editText.text
   C editText.text.toString
   D editText.text.toString() [✓]

   Explanation

   In order to retrieve data as String from the **editText**, we use the code: **editText.text.toString()**

3. **Which of the following code snippets will start a new activity using Intents?**

   A
   ```
   val intent = Intent(this@StartActivity,
   DestinationActivity::class.java)
   ```

   B
   ```
   val intent = Intent(this@StartActivity,
   this@DestinationActivity)
   startActivity(intent)
   ```

   C
   ```
   val intent = Intent(this@StartActivity,
   DestinationActivity.class.java)
   startActivity(intent)
   ```

   D [✓]
   ```
   val intent = Intent(this@StartActivity,
   DestinationActivity::class.java)
   startActivity(intent)
   ```

   Explanation

   The correct syntax to open a new activity using intents is:
   val intent = Intent(this@StartActivity, DestinationActivity::class.java)
   startActivity(intent)

4. **How do you pass the data between two activities?**

   A All the data inside the application can be accessed without any exceptions.
   B We need to put the data inside an intent and then pass it.
   C Data from one activity is automatically sent to the next activity.
   D We cannot pass data between two activities.

   Explanation

   In order to pass the data from one activity to another, we put the data inside an intent and then this data is passed when the intent opens up a new activity.

5. **How do we provide context of the current activity, given the name of the current activity is MainActivity?**

   A MainActivity::class.java
   B this@MainActivity.kt
   C this.MainActivity
   D this@MainActivity

   Explanation

   The context of the current activity (MainActivity in this case) is provided by using the ‘this’ keyword. The correct syntax for doing the same is this@MainActivity.

6. **What is the use of the finish() method?**

   A It makes the activity go into paused state.
   B It stops the running activity.
   C It destroys the current activity.
   D None of the above.

   Explanation

   Adding finish() destroys the activity. So, when the user tries to re-open that activity, it is re-created from the start.

7. **Which of these parameters inside the putExtra() method are used to put the data inside the intent?**
   Given: putExtra(“___________”,__________)

A Name of data, Value of data
B Value of data, Name of data
C Name of intent, Value of intent
D Source Activity, Destination activity

Explanation

The parameters inside the putExtra() method in order of their occurrence are: Name of data, Value of data. Hence the given would be: putExtra(“name_of_data”, value_of_data)

1. **Where do we store a small amount of data in an app?**

   A Internal Database
   B File
   C SharedPreferences
   D External Database

   Explanation

   Small amounts of data like a string or a couple of strings or numbers can be saved inside SharedPreferences of an app.

2. **Which mode of shared preferences is used so that it is only accessed by the calling application?**

   A MODE_PRIVATE
   B MODE_APPEND
   C MODE_ENABLE_WRITE_AHEAD_LOGGING
   D MODE_MULTI_PROCESS

   Explanation

   MODE_PRIVATE is used so that the shared preferences file is only accessed by the calling application.

3. **Which window is used to check the reasons for the crash in an application?**

   A Design preview
   B Editor window
   C Gradle
   D Logcat

   Explanation

   All the events including the reasons for the crash happening inside an application are recorded in the Logcat window.

4. **In which form is the data stored inside the shared preferences?**

   A Tables and columns
   B Key-value pairs
   C Indexed values
   D All of the above

Explanation

The data inside the shared preferences is stored in the form of key-value pairs.

5. **Predict the datatype of the values associated with the given keys:**

   training -> Android app development
   organisation -> Internshala
   price -> 1349.00
   isAwesome -> true

   A String, String, Double, Boolean ✓
   B String, Boolean, Int, String
   C Text, Text, Number, String
   D Text, String, Int, Boolean

   Explanation

   In the given options, Text and Number are not any data types. Also, the first value is the name of the training which would be String, organisation name is also a String, price looks like an Integer but has values after decimal i.e. 1349.00, hence it is Double and finally true and false are Boolean data types. Therefore, the correct answer is: String, String, Double, Boolean.

17/ 24
SCORE

71%
PERCENT

No. of questions: 15
Correct answer: 12
Incorrect answer: 3

All　Only incorrect

Q1/15 **Multiple Choice Question** ✓ (1/1 Marks)

In which file do we change the version of Kotlin in our Android app?

A Build.gradle (Module: app)

[✓ B Build.gradle(Project: <project_name>)]

C settings.gradle

D gradle-wrapper.properties

Explanation

The version of Kotlin and Android Studio is defined in the project level build.gradle file and any changes required in the versions are done in these files.

Q2/15 **Multiple Choice Question** ✓ (1/1 Marks)

Can we nest a <TextView/> inside another <TextView/>?

A Yes

[✓ B No]

C Depends on the parent layout

Explanation

<TextView/> is a widget and we cannot nest a widget inside another widget. However, we can nest layouts inside layouts.

Q3/15 **Multiple Choice Question** ✓ (2/2 Marks)

After the login is successful, we want to save the login preference as 'true' in the SharedPreferences. Which of the following code snippets will be used to do this?

Given:

```
val sharedPreferences = getSharedPreferences("pref_name", MODE_PRIVATE)
```

A
```
sharedPreferences.putBoolean("login", true)
```

B
```
sharedPreferences.putBoolean("login", true).apply()
```

C
```
sharedPreferences.edit().putBoolean("login", true).apply()
```
✓

D
```
sharedPreferences.edit().putBoolean("login", true)
```

Explanation

Here the SharedPreference variable is given as sharedPreferences and hence to save the login preference as 'true' we will put a Boolean value as 'true'. After putting the value, save the changes using the apply() method. For this we use the following code snippet:

```
sharedPreferences.edit().putBoolean("login", true).apply()
```

Q4/15 **Multiple Choice Question** ✓ (1/1 Marks)

Which parent layout will you use if you have to create a page which contains more than 10 widgets aligned vertically?

A RelativeLayout

B LinearLayout

C ScrollView ✓

D None of the above

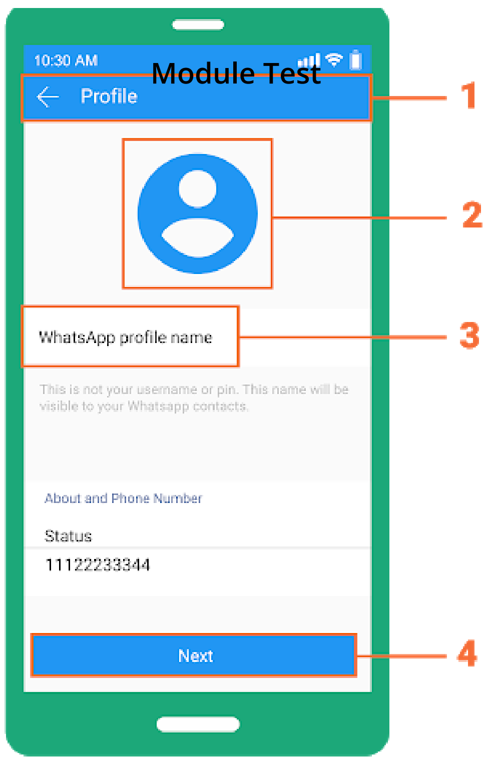
Explanation

Here the number of vertically aligned widgets is a large number (10) and there is a big possibility of the layout exceeding the phone screen. Hence, we will use a ScrollView in such cases.

Q5-8/15 **Reading Comprehension** ✗ (5/7 Marks)

Passage

Refer to the image given below and answer the questions that follow:



Solution

Q5/15 ✗ (0/2 Marks)

Identify the elements 1, 2, 3, 4

A
1. Button
2. ImageView
3. TextView
4. Button

B ✗
1. Toolbar
2. ImageButton
3. TextView
4. Button

C
1. Toolbar
2. ImageView
3. EditText
4. Button

D ✓
1. Toolbar
2. ImageView
3. TextView
4. Button

Explanation

1- This is the top bar of the app, known as the toolbar.

2- Here we see adefault image, which can be treated as a user image. We use an ImageView to display the image.

3- We see an uneditable, static text, and hence the TextView is used to display this.

4- This is a clickable button, hence the Button View is used to make it.

Q6/15 ✓ (2/2 Marks)

Which of these code snippets is used to add the text ('Profile') in the component labelled as 1?

A Add the attribute

```
android:text = "Profile"
```

in the XML file

B Use the method

```
setText("Profile")
```

in the Kotlin file

C Use the code ✓

```
supportActionBar?.title = "Profile"
```

in the Kotlin file

D Use the code

```
toolbar?.title = "Profile"
```

in the Kotlin file

Explanation

Since the given component is a toolbar, we need to add the title to it. The correct way to do it is to use the action bar implementation of it i.e.:

```
supportActionBar?.title = "Profile"
```

Q7/15 ✓ (2/2 Marks)

Which of the following is used to make component 2?

A ✓

```
<ImageView
        android:layout_width="160dp"
        android:layout_height="160dp"
        android:src="@drawable/default_user_image"
        android:layout_marginTop="30dp"
        android:layout_alignCenterHorizontal="true" />
```

B

```
<ImageView
        android:layout_width="160sp"
        android:layout_height="160sp"
        android:src="@drawable/default_user_image"
        android:layout_marginTop="30dp"
        android:layout_alignCenterHorizontal="true" />
```

C

```
<ImageView
        android:layout_width="160dp"
        android:layout_height="160dp"
        android:layout_marginTop="30dp"
        android:layout_alignCenterHorizontal="true" />
```

D

```
<ImageView
        android:layout_width="160dp"
        android:layout_height="160dp"
        android:src="@image/default_user_image"
        android:layout_marginTop="30dp"
        android:layout_alignCenterHorizontal="true" />
```

Explanation

In option 2, the height and width are given in sp, while it needs to be given in dp.

In option 3, the src attribute to give the image is missing.

In option 4, the image in the src attribute is coming from the @image directory, which si not present in the Android Studio.

Hence, the correct answer is option 1.

Q8/15 ✓ (1/1 Marks)

Which of these code snippets will be used to prompt a Toast message "Hello, click" when the user clicks on the component 4?
Given: Component 4 is defined as:

```
val btnNext = findViewById(R.id.btnNext)
```

and the name of the activity is MainActivity.kt

A

```
btnNext?.setClickListener {
                Toast.show(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT)
            }
```

B

```
btnNext?.setOnClickListener {
                Toast.show(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT)
            }
```

C ✓

```
btnNext?.setOnClickListener {
                Toast.makeText(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT).show()
            }
```

D

```
btnNext?.setOnClickListener {
                Toast.makeText(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT)
            }
```

Explanation

Component 4 is a button, and in order to add clicks on a button, we use the setOnClickListener { ... }. For the toast, we use the makeText() method fo the Toast class like this:

```
Toast.makeText(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT).show()
```

Hence, in order to add the toast to the button click, the code used is:

```
btnNext?.setOnClickListener {
    Toast.makeText(this@MainActivity, "Hello, click", Toast.LENGTH_SHORT).show()
}
```

Q9-12/15 Reading Comprehension ✓ (6/6 Marks)

Passage

Let's say we have a login page with two EditTexts (mobile number and password) and a login button.

Given:

```
val etMobileNumber = findViewById(R.id.etLogin)
val etPassword = findViewById(R.id.etPassword)
```

Using the information given above, answer the questions.

Solution

Q9/15 ✓ (1/1 Marks)

How will we send both the login credentials i.e. mobile number and password to the next activity?

A ✓

```
val intent = Intent(this@LoginActivity, MainActivity::class.java)
intent.putExtra("mobileNumber", etMobileNumber.text.toString)
intent.putExtra("password", etPassword.text.toString)
startActivity(intent)
```

B

```
val intent = Intent(this@LoginActivity, MainActivity::class.java)
intent.putExtra("mobileNumber", etMobileNumber)
intent.putExtra("password", etPassword)
startActivity(intent)
```

C

```
val intent = Intent(this@LoginActivity, MainActivity::class.java)
intent.putIntentData("mobileNumber", etMobileNumber.text.toString)
intent.putIntentData("password", etPassword.text.toString)
startActivity(intent)
```

D

```
val intent = Intent(this@LoginActivity, MainActivity::class.java)
intent.putExtra("mobileNumber", etMobileNumber.text.toString)
intent.putExtra("password", etPassword.text.toString)
```

Explanation

In order to send the data to another activity, we use an intent like this:

```
intent.putExtra("nameOfData", valueOfData)
```

The intent objects are created like this:

```
val intent = Intent(this@LoginActivity, MainActivity::class.java)
```

Hence, the correct answer is:

```
val intent = Intent(this@LoginActivity,
MainActivity::class.java)intent.putExtra("mobileNumber",
etMobileNumber.text.toString)intent.putExtra("password",
etPassword.text.toString)startActivity(intent)
```

Q10/15 ✓ (2/2 Marks)

Once we have sent the data, how will we receive it in the new activity?

A
```
val mobileNumber = intent.getString("mobileNumber")
val password = intent.getString("password")
```

B
```
val mobileNumber = Intent().getString("mobileNumber")
val password = Intent().getString("password")
```

C ✓
```
val mobileNumber = intent.getStringExtra("mobileNumber")
val password = intent.getStringExtra("password")
```

D
```
val mobileNumber = Intent().getStringExtra("mobileNumber")
val password = Intent().getStringExtra("password")
```

Explanation

To receive the data sent from an intent, we use the getStringExtra() method like this:

```
intent.getStringExtra("nameOfData")
```

Q11/15 ✓ (1/1 Marks)

After receiving the login credentials in the new activity, how will we store the mobile number in the SharedPreferences?

Given:

```
val sharedPreferences = getSharedPreferences("preferences", MODE_PRIVATE)
```

A
```
sharedPreferences.edit().putString("mobileNumber", mobileNumber)
```

B ✓
```
sharedPreferences.edit().putString("mobileNumber", mobileNumber).apply()
```

C
```
sharedPreferences.putString("mobileNumber", mobileNumber).apply()
```

D
```
sharedPreferences.edit().putText("mobileNumber", mobileNumber).apply()
```

Explanation

To store the received credentials in the sharedPreferences, we use the putString() method and then apply() it to save the changes.

Q12/15 ✓ (2/2 Marks)

After saving the data in the preferences, how will we extract it to use it in the application?

Given:

```
val sharedPreferences = getSharedPreferences("preferences", MODE_PRIVATE)
```

A
```
val mobileNumber = sharedPreferences.edit().getString("mobileNumber").apply()
```

B
```
val mobileNumber = sharedPreferences.getString(mobileNumber)
```

C
```
val mobileNumber = SharedPreferences().getString("mobileNumber")
```

D ✓
```
val mobileNumber = sharedPreferences.getString("mobileNumber")
```

Explanation

In order to retreive a String from the SharedPreferences, we use the getString() method on the sharedPreferences object.

Q13/15 Multiple Choice Question ✗ (0/3 Marks)

If the weightSum of a LinearLayout is 10, what will be the correct division of weights of the 4 child Views present inside it?

A 2.5, 2.5, 2.5, 2.5 ✗

B 2.2, 5.4, 1, 1.4

C 3.5, 3.5, 1.2, 1.8

D All of the above ✓

Explanation

In order to divide the layout according to its weightSum, we need to take care that the sum of the weights of the individual Views should be equal to the weightSum of the parent layout. In this case, the weightSum of the parent layout is 10 while the sum of division of all the options is 10 and hence we can divide this layout in any of the given ways.

Arrange the lifecycle methods according to their order of occurrence.

Given:

The activity was opened, then the user moves to another activity, and then comes back to the first activity after a while.

- onCreate()
- onStart()
- onPause()
- onResume()
- onStop()
- onRestart()

Solution

- onCreate()
- onStart()
- onResume()
- onPause()
- onStop()
- onRestart()

Explanation

When the activity opens and activity for the first time, the onCreate() -> onStart() -> onResume() is called.

When the user moves to the next activity, the onPause() is called and if the user stays there the onStop() is called.

Finally, when the user navigates back to this activity the onRestart() -> onStart() -> onResume() are called.

Q15/15 Fill In The Blank ✓ (1/1 Marks)

The class responsible for creating a bridge between two activities is known as an _______.

Type Your Answer

Intent ✓

Explanation

Intent class of the SDK is used to create a bridge between two activities.

Prev
Text Chapter
(android/screen/7096)

Next
Video
(android/screen/7013)

What is the use of the Asynctask class?

A
It is used to display the progress bar while the data loads.

B
It is used to make network requests to an external server and fetch data.

C
It is used to achieve multithreading so that the expensive tasks are performed at the worker thread, thus preventing the app from ANR errors.

D
It is used to optimise the apps so that extensive RAM is not used by the application, and hence the app runs perfectly fine on low end phones.

Explanation

Asynctask class divides the processes into two thread i.e. the main thread and worker thread. This happens to avoid the application freezing and prevent ANR errors.

Q2/17 Multiple Choice Question
(1/1 Marks)

How is data stored in the SQLite Database?

A
Key-Value pairs

B
Rows-Columns

C
Data Objects

D
Array

Explanation
SQLite is a relational database and hence the data is stored in the form of Rows and Columns.

Q3/17 Multiple Choice Question
(1/1 Marks)

Which directory is used to store the image resources used in the app?

A
res

B
drawable

C
layouts

D

values

Explanation

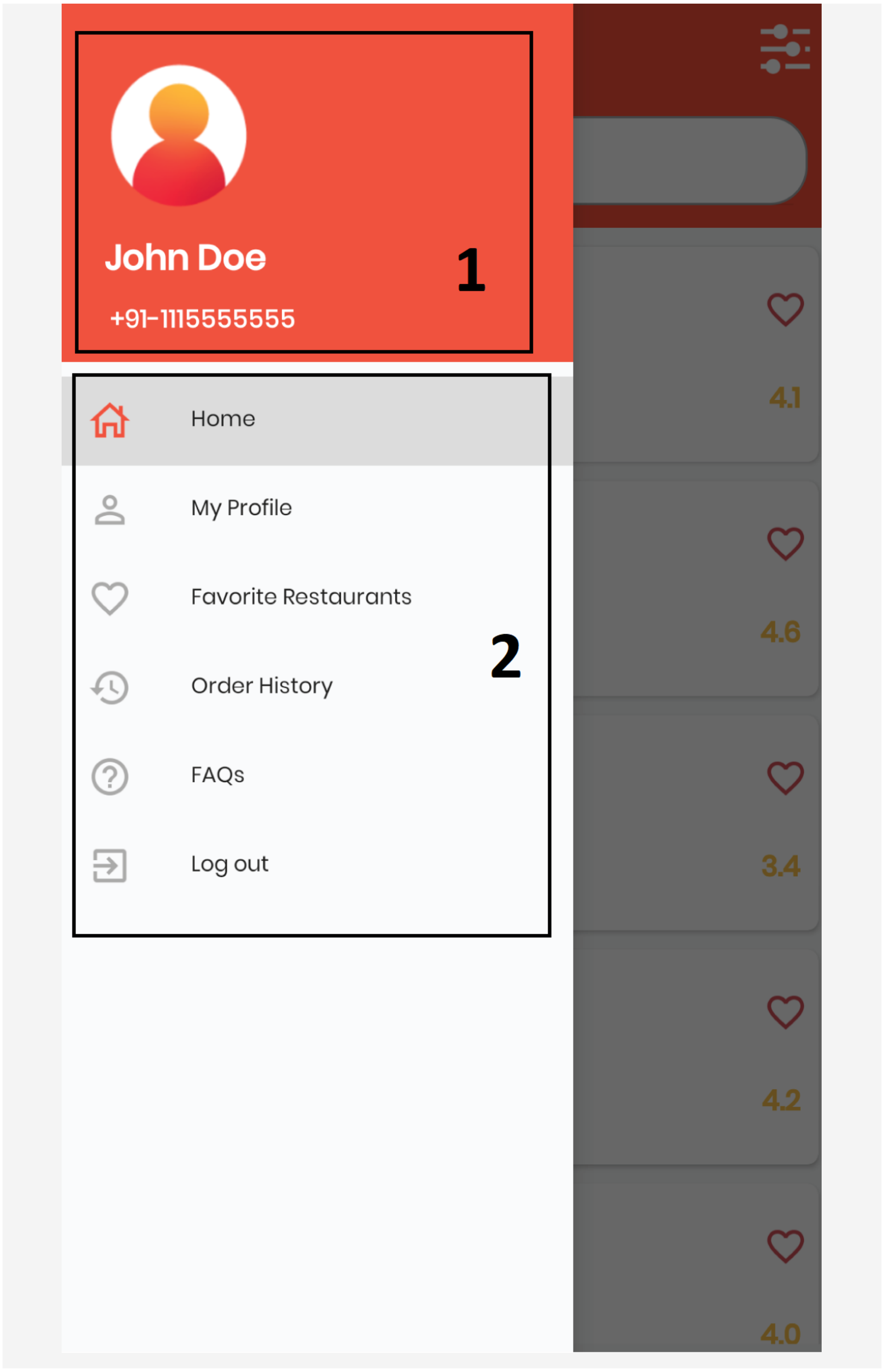
All the images used in the project are stored inside the drawable directory.

Q4-8/17 Reading Comprehension

(14/16 Marks)

Passage

Refer to the image given below and answer the questions that follow:


John Doe
+91-1115555555
1
Home
My Profile
Favorite Restaurants
Order History
FAQs
Log out
2
4.1
4.6
3.4
4.2
4.0

Solution

Q4/17
(2/2 Marks)

Identify the components 1 and 2.

A
1. LinearLayout

2. RecyclerView

B
1. RelativeLayout

2. LinearLayout

C
1. DrawerHeader

2. Drawer Menu

D
1. DrawerHeader

2. RecyclerView

Explanation
Both the components belong to the navigation drawer itself where 1 is drawer header and 2 is the menu options of the navigation drawer.

Q5/17
(0/2 Marks)

Identify the parent tag for the complete view in the image.

A
<fragment/>

B
<android.support.v4.widget.DrawerLayout/>

C
<android.support.design.widget.CoordinatorLayout/>

D
<android.support.design.widget.NavigationView/>

Explanation
The given is a navigation drawer and the tag used to add this drawer is

<android.support.design.widget.NavigationView/>.

Q6/17
(4/4 Marks)

Identify the menu file for the given navigation drawer.

A

```xml
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:checked="true"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:icon="@drawable/action_person"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item
            android:id="@+id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@+id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>

</menu>
```

B

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@id/home"
            android:checked="true"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@id/myProfile"
            android:icon="@drawable/action_person"
            android:title="@string/my_profile" />

        <item
            android:id="@id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item
            android:id="@id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history"/>

        <item
            android:id="@id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>
```

```
</menu>
```

C

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item
            android:id="@+id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@+id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>
```

```
</menu>
```

D

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:checked="true"
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:title="@string/favorites" />

        <item
            android:id="@+id/order_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:title=""@string/faqs"" />

        <item
            android:id="@+id/logout"
            android:title="@string/logout" />
    </group>

</menu>
```

Explanation
In option 2 - The defined IDs lack a '+' sign as the correct way of defining an ID is:

android:id="@+id/home"

In option 3 - The menu item for profile lacks the attribute icon and hence no image would be present in the navigation drawer.

In option 4 - The icon attribute is missing for all the menu items.

Hence, the correct option for this question is option 1.

Q7/17
(4/4 Marks)

Suppose each option of the navigation drawer has a fragment attached to it. How would you open a new fragment?

Given you are at HomeFragment and you need to open ProfileFragment.

A

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(HomeFragment, profileFragment)
```

B

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.open(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

C ✓

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

D

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.commit()
```

Explanation
In order to open a new fragment, initialise the fragment transaction like this:

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
```

Now replace the current frame with the frame of the ProfileFragment like this:

```
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
```

Finally commit the transaction like this:

```
fragmentTransaction.commit()
```

Hence the correct option becomes:

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

Q8/17
(4/4 Marks)

How do we add the component 1 to the navigation drawer?

A
Create a header layout say drawer_header.xml and add the line given below in the navigation view tag:

app:headerLayout="@layout/drawer_header"

B
Add a LinearLayout inside the navigation view and then inside it add the respective elements.

C
Create the complete navigation drawer inside a LinearLayout and then add the components.

D
Add the attributes app:drawer_image and app_drawerTitle in the navigation view tag.

Explanation
In order to add the drawer header just create a separate layout file for header and then add the code given below as attribute to the NavigationView:

app:headerLayout="@layout/drawer_header"

Q9-10/17 Reading Comprehension
(3/5 Marks)

Passage

Consider the table (HIGHSCORES) of items given below which represents the schema of the table in SQLite DB and answer the questions that follow:

Name	Score
Abc	100
Pqr	150
Lmno	120
Xyz	200
Ghi	320

Solution

Q9/17
(0/2 Marks)

Write an SQL query to fetch all the data of the given table.

A
"SELECT * FROM TABLE HIGHSCORES"

B
"SELECT FROM TABLE HIGHSCORES"

C
"SELECT ALL FROM TABLE HIGHSCORES"

D
"SELECT * FROM HIGHSCORES"

Explanation
The SELECT statement is used to select data from a database. The syntax of the query is:

SELECT * FROM table_name

Q10/17
(3/3 Marks)

Create a data model to store the player names and high scores.

A

```
class HighScores(val playerName: String, val score: Int)
```

B

```
data class HighScores(val playerName: String, val score: Int)
```

C

```
data HighScores(val playerName: String, val score: Int)
```

D

```
data class HighScores(val playerName: String, val score: String)
```

Explanation
The data model is created by using the data keyword before declaring the class. Also, here the name is a String while the score is an Integer and hence the data model will become:

```
data class HighScores(val playerName: String, val score: Int)
```

Q11/17 Multiple Choice Question
(1/1 Marks)

Which View is used to display long lists with custom data?

A
<ScrollView/>

B
<RecyclerView>

C
A collection of <RelativeLayout>

D
A collection of <LinearLayout>

Explanation
In order to display a long list with either same number of elements or dynamic number of elements, we use the RecyclerView.

Q12/17 Multiple Choice Question
(3/3 Marks)

Consider an element of a list with an image and title. Create the ViewHolder class for the same.

Given :

id of Image : imgThumbnail
id of title: txtTitle

A

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

B

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder() {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

C

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val imgThumbNail: ImageView= findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = findViewById(R.id.txtTitle)
}
```

D

```
class MyViewHolder(view: View) : ViewHolder(view) {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

Explanation
The ViewHolder class for the given View having an image and a text would be:

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val imgThumbNail: ImageView=
view.findViewById(R.id.imgThumbnail)
    val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

Q13/17 Multiple Choice Question
(2/2 Marks)

Given the JSON named as jsonObject:

```
{
    "id": "1",
    "name": "Internshala Trainings",
    "rating": "4.5",
    "cost_for_one": "1350",
}
```

Pull out the value of cost_for_one using Kotlin.

A

```
val cost = JSON().getString("cost_for_one")
```

B

```
val cost = jsonObject.getValue("cost_for_one")
```

C

```
val cost = jsonObject.getString(cost_for_one)
```

D

```
val cost = jsonObject.getString("cost_for_one")
```

Explanation
Since the required entity is String, we fetch it using the getString() method of the JSON class.

Q14/17 Multiple Choice Question
(2/2 Marks)

Which of these manifest permissions are required for connecting the app to the internet?

A
android.permission.INTERNET_ACCESS

B
android.permission.NETWORK_ACCESS

C
android.permission.INTERNET

D
android.permission.internet.FULL_ACCESS

Explanation
android.permission.INTERNET is the required permission to connect an app to the internet. Without this the app cannot make any connection to the internet.

Q15/17 Multiple Choice Question
(4/4 Marks)

How do you keep the spalsh screen for 2 seconds and then automatically change it to the next screen?

Given:

```
private fun openNewActivity() {
            val intent = Intent(this@SplashActivity,
LoginActivity::class.java)
            startActivity(intent)
            finish()
 }
```

A

```
 Handler.postDelayed({
                openNewActivity()
            }, 2000)
```

B

```
 Handler().postDelayed({
                openNewActivity()
            }, 2000)
```

C

```
 Handler().delay({
                openNewActivity()
            }, 2000)
```

D

```
 Handler().postDelayed({
                openNewActivity()
            }, 2)
```

Explanation
We use the handler to keep the thread for 2 seconds and then start a new activity.

Q16/17 Multiple Choice Question
(2/2 Marks)

Suppose a request is made and the response is sent to the main thread. Soon after it the same request is repeated. How long will it take for the second request to fetch the data?

Given: We are using Volley to send the request.

A

It will take the same amount of time as it took the first time.

B
It will take lesser amount of time for the request to fetch data as this time the items would be picked up from the cache instead of the server.

C
It will take more time as the number of requests on the server have increased and it will take extra additional buffer to send the data.

D
This will depend on the speed of the internet in the device which is used to make the request.

Explanation
The responses from a volley requests are saved in the cache of the device and stays there for some time so that when a new request is made within a few minutes, the same data can be called.

Q17/17 Multiple Choice Question
(2/2 Marks)

What is the benefit of creating a signed APK?

A
The Android system uses the certificate as a means of identifying the author of an application and allowing only the certified users' apps to be installed in the devices.

B
The Android system uses the certificate as a means of identifying the author of an application. In case any explicit content is found in the app, the owner is penalised for it.

C
The Android system uses the certificate as a means of identifying the author of an application and establishing trust relationships between applications.

D
We can upload unsigned apk also as an anonymous person, however in such cases the developers are not identified and the revenue generated by the app is gone to Google.

38/ 42

SCORE

90%

PERCENT

No. of questions: 17
Correct answer: 15
Incorrect answer: 2

All Only incorrect

Q1/17 Multiple Choice Question ✓ (3/3 Marks)

What is the use of the Asynctask class?

A It is used to display the progress bar while the data loads.

B It is used to make network requests to an external server and fetch data.

[✓ C It is used to achieve multithreading so that the expensive tasks are performed at the worker thread, thus preventing the app from ANR errors.]

D It is used to optimise the apps so that extensive RAM is not used by the application, and hence the app runs perfectly fine on low end phones.

Explanation

Asynctask class divides the processes into two thread i.e. the main thread and worker thread. This happens to avoid the application freezing and prevent ANR errors.

Q2/17 Multiple Choice Question ✓ (1/1 Marks)

How is data stored in the SQLite Database?

A Key-Value pairs

B Rows-Columns

C Data Objects

D Array

Explanation

SQLite is a relational database and hence the data is stored in the form of Rows and Columns.

Q3/17 Multiple Choice Question ✓ (1/1 Marks)

Which directory is used to store the image resources used in the app?

A res

[✓ B drawable]

C layouts

D values

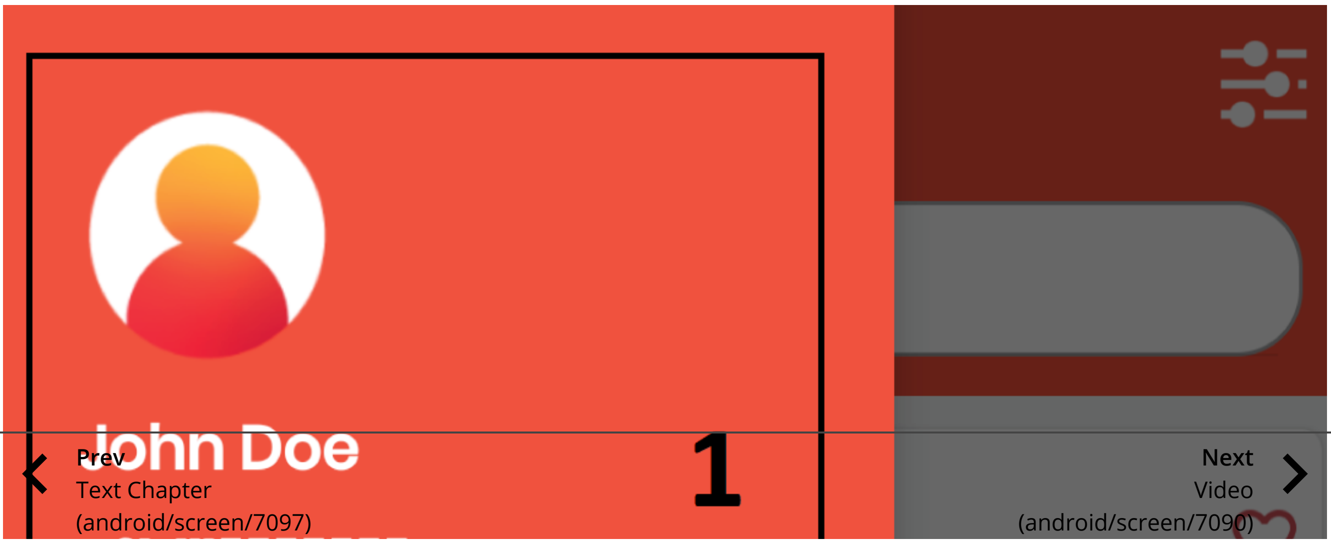
Explanation

All the images used in the project are stored inside the drawable directory.

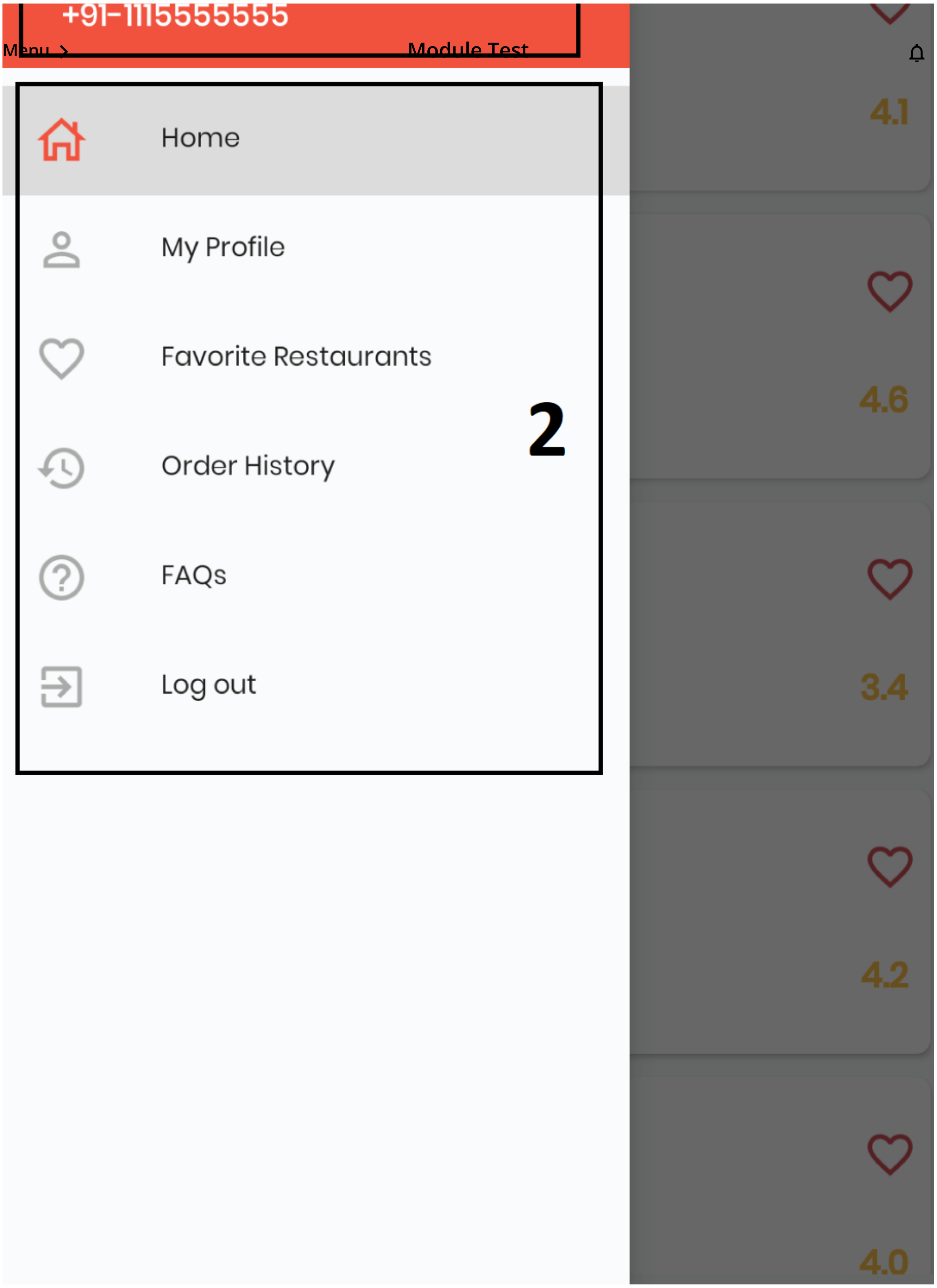
Q4-8/17 Reading Comprehension ✗ (14/16 Marks)

Passage

Refer to the image given below and answer the questions that follow:



Prev Text Chapter (android/screen/7097)
Next Video (android/screen/7090)


+91-1115555555

Menu >

Module Test

Home

My Profile

Favorite Restaurants

Order History

FAQs

Log out

2

4.1

4.6

3.4

4.2

4.0

Solution

Q4/17 ✓ (2/2 Marks)

Identify the components 1 and 2.

A 1. LinearLayout

2. RecyclerView

B 1. RelativeLayout

2. LinearLayout

C 1. DrawerHeader ✓

2. Drawer Menu

D 1. DrawerHeader

2. RecyclerView

Explanation

Both the components belong to the navigation drawer itself where 1 is drawer header and 2 is the menu options of the navigation drawer.

Q5/17 ✗ (0/2 Marks)

Identify the parent tag for the complete view in the image.

A <fragment/>

B <android.support.v4.widget.DrawerLayout/> ✗

C <android.support.design.widget.CoordinatorLayout/>

D <android.support.design.widget.NavigationView/> ✓

Explanation

The given is a navigation drawer and the tag used to add this drawer is

<android.support.design.widget.NavigationView/>.

Q6/17 ✓ (4/4 Marks)

Identify the menu file for the given navigation drawer.

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:checked="true"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:icon="@drawable/action_person"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item
            android:id="@+id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@+id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>

</menu>
```

A ✓

B

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@id/home"
            android:checked="true"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@id/myProfile"
            android:icon="@drawable/action_person"
            android:title="@string/my_profile" />

        <item
            android:id="@id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item
            android:id="@id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history"/>

        <item
```

```
            android:id="@id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>

</menu>
```

C

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:icon="@drawable/action_home"
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:icon="@drawable/action_fav"
            android:title="@string/favorites" />

        <item

            android:id="@+id/order_history"
            android:icon="@drawable/action_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:icon="@drawable/action_faq"
            android:title="@string/faqs" />

        <item
            android:id="@+id/logout"
            android:icon="@drawable/action_logout"
            android:title="@string/logout" />
    </group>

</menu>
```

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">

    <group android:checkableBehavior="single">

        <item
            android:id="@+id/home"
            android:checked="true"
```

D
```
            android:title="@string/home" />

        <item
            android:id="@+id/myProfile"
            android:title="@string/my_profile" />

        <item
            android:id="@+id/favRes"
            android:title="@string/favorites" />

        <item
            android:id="@+id/order_history"
            android:title="@string/order_history" />

        <item
            android:id="@+id/faqs"
            android:title=""@string/faqs"" />

        <item
            android:id="@+id/logout"
            android:title="@string/logout" />
    </group>

</menu>
```

Explanation

In option 2 - The defined IDs lack a '+' sign as the correct way of defining an ID is:

android:id="@+id/home"

In option 3 - The menu item for profile lacks the attribute icon and hence no image would be present in the navigation drawer.

In option 4 - The icon attribute is missing for all the menu items.

Hence, the correct option for this question is option 1.

Q7/17 ✓ (4/4 Marks)

Suppose each option of the navigation drawer has a fragment attached to it. How would you open a new fragment?

Given you are at HomeFragment and you need to open ProfileFragment.

A
```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(HomeFragment, profileFragment)
```

B
```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.open(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

C ✓
```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

D
```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.commit()
```

Explanation

In order to open a new fragment, initialise the fragment transaction like this:

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
```

Now replace the current frame with the frame of the ProfileFragment like this:

```
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
```

Finally commit the transaction like this:

```
fragmentTransaction.commit()
```

Hence the correct option becomes:

```
val fragmentTransaction = supportFragmentManager.beginTransaction()
val profileFragment = ProfileFragment()
fragmentTransaction.replace(R.id.frame, profileFragment)
fragmentTransaction.commit()
```

Q8/17 ✓ (4/4 Marks)

How do we add the component 1 to the navigation drawer?

A Create a header layout say drawer_header.xml and add the line given below in the navigation view tag:
app:headerLayout="@layout/drawer_header" ✓

B Add a LinearLayout inside the navigation view and then inside it add the respective elements.

C Create the complete navigation drawer inside a LinearLayout and then add the components.

D Add the attributes app:drawer_image and app_drawerTitle in the navigation view tag.

Explanation

In order to add the drawer header just create a separate layout file for header and then add the code given below as attribute to the NavigationView:

app:headerLayout="@layout/drawer_header"

Q9-10/17 Reading Comprehension ✗ (3/5 Marks)

Passage

Consider the table (HIGHSCORES) of items given below which represents the schema of the table in SQLite DB and answer the questions that follow:

Name	Score
Abc	100
Pqr	150
Lmno	120
Xyz	200
Ghi	320

Solution

Q9/17 ✗ (0/2 Marks)

Write an SQL query to fetch all the data of the given table.

A "SELECT * FROM TABLE HIGHSCORES" ✗

B "SELECT FROM TABLE HIGHSCORES"

C "SELECT ALL FROM TABLE HIGHSCORES"

D "SELECT * FROM HIGHSCORES" ✓

Explanation

The SELECT statement is used to select data from a database. The syntax of the query is:

SELECT * FROM table_name

Q10/17 ✓ (3/3 Marks)

Create a data model to store the player names and high scores.

A
```
class HighScores(val playerName: String, val score: Int)
```

B
```
data class HighScores(val playerName: String, val score: Int)
```
✓

C
```
data HighScores(val playerName: String, val score: Int)
```

D
```
data class HighScores(val playerName: String, val score: String)
```

Explanation

The data model is created by using the data keyword before declaring the class. Also, here the name is a String while the score is an Integer and hence the data model will become:

```
data class HighScores(val playerName: String, val score: Int)
```

Q11/17 **Multiple Choice Question** ✓ (1/1 Marks)

Which View is used to display long lists with custom data?

A <ScrollView/>

B <RecyclerView> ✓

C A collection of <RelativeLayout>

D A collection of <LinearLayout>

Explanation

In order to display a long list with either same number of elements or dynamic number of elements, we use the RecyclerView.

Consider an element of a list with an image and title. Create the ViewHolder class for the same.

Given :

id of Image : imgThumbnail
id of title: txtTitle

A ✓

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

B

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder() {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

C

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val imgThumbNail: ImageView= findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = findViewById(R.id.txtTitle)
}
```

D

```
class MyViewHolder(view: View) : ViewHolder(view) {
        val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
        val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

Explanation

The ViewHolder class for the given View having an image and a text would be:

```
class MyViewHolder(view: View) : RecyclerView.ViewHolder(view) {
    val imgThumbNail: ImageView= view.findViewById(R.id.imgThumbnail)
    val txtTitle: TextView = view.findViewById(R.id.txtTitle)
}
```

Given the JSON named as jsonObject:

```
{
    "id": "1",
    "name": "Internshala Trainings",
    "rating": "4.5",
    "cost_for_one": "1350",
}
```

Pull out the value of cost_for_one using Kotlin.

A
```
val cost = JSON().getString("cost_for_one")
```

B
```
val cost = jsonObject.getValue("cost_for_one")
```

C
```
val cost = jsonObject.getString(cost_for_one)
```

[✓ D]
```
val cost = jsonObject.getString("cost_for_one")
```

Explanation

Since the required entity is String, we fetch it using the getString() method of the JSON class.

Q14/17 **Multiple Choice Question** ✓ (2/2 Marks)

Which of these manifest permissions are required for connecting the app to the internet?

A android.permission.INTERNET_ACCESS

B android.permission.NETWORK_ACCESS

[✓ C android.permission.INTERNET]

D android.permission.internet.FULL_ACCESS

Explanation

android.permission.INTERNET is the required permission to connect an app to the internet. Without this the app cannot make any connection to the internet.

Q15/17 Multiple Choice Question ✓ (4/4 Marks)

How do you keep the spalsh screen for 2 seconds and then automatically change it to the next screen?

Given:

```
private fun openNewActivity() {
          val intent = Intent(this@SplashActivity, LoginActivity::class.java)
          startActivity(intent)
          finish()
 }
```

A

```
Handler.postDelayed({
               openNewActivity()
        }, 2000)
```

[✓ B]

```
Handler().postDelayed({
               openNewActivity()
        }, 2000)
```

C

```
Handler().delay({
               openNewActivity()
        }, 2000)
```

D

```
Handler().postDelayed({
               openNewActivity()
        }, 2)
```

Explanation

We use the handler to keep the thread for 2 seconds and then start a new activity.

Q16/17 Multiple Choice Question ✓ (2/2 Marks)

Suppose a request is made and the response is sent to the main thread. Soon after it the same request is repeated. How long will it take for the second request to fetch the data?

Given: We are using Volley to send the request.

A It will take the same amount of time as it took the first time.

B It will take lesser amount of time for the request to fetch data as this time the items would be picked up from the cache instead of the server. ✓

C It will take more time as the number of requests on the server have increased and it will take extra additional buffer to send the data.

D This will depend on the speed of the internet in the device which is used to make the request.

Explanation

The responses from a volley requests are saved in the cache of the device and stays there for some time so that when a new request is made within a few minutes, the same data can be called.

Q17/17 **Multiple Choice Question** ✓ (2/2 Marks)

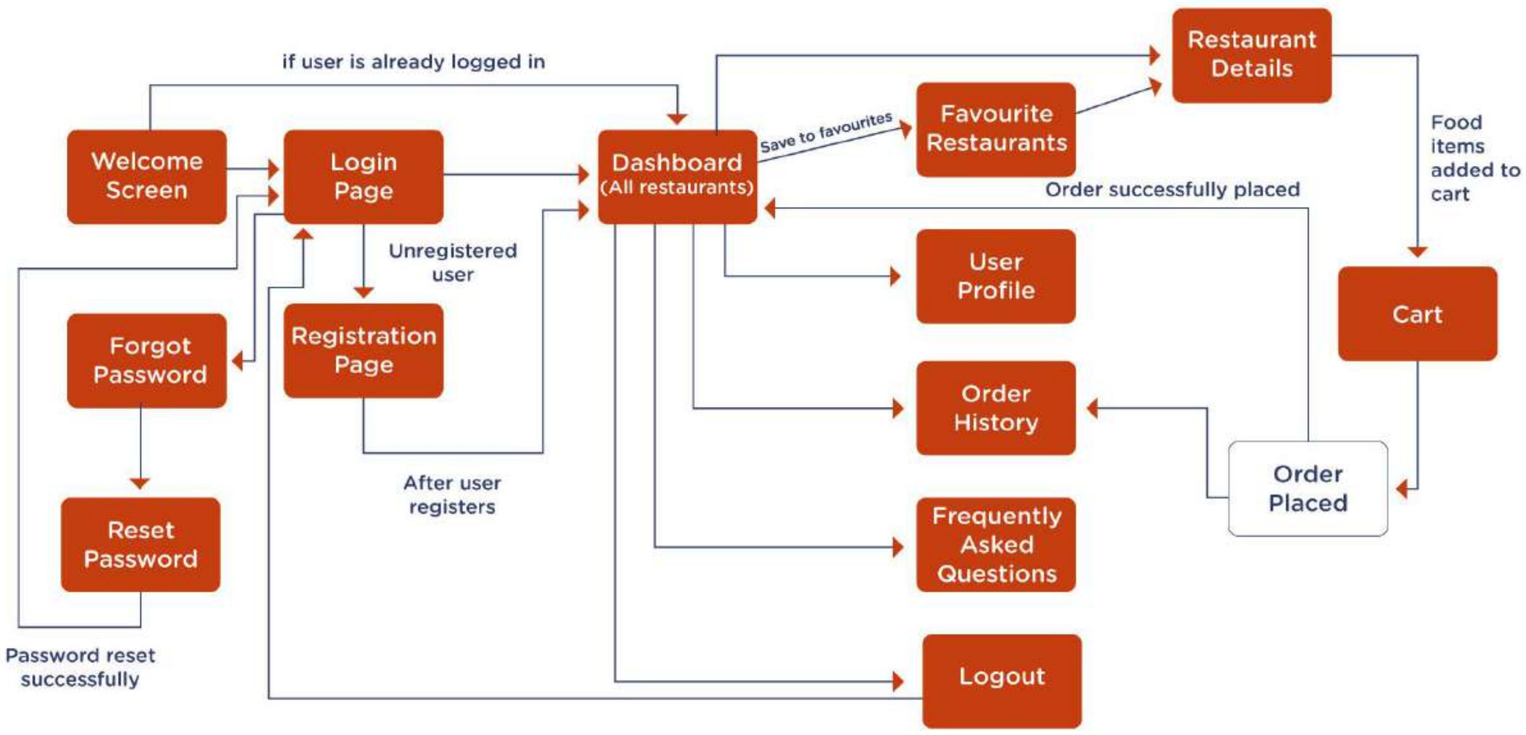
What is the benefit of creating a signed APK?

A The Android system uses the certificate as a means of identifying the author of an application and allowing only the certified users' apps to be installed in the devices.

B The Android system uses the certificate as a means of identifying the author of an application. In case any explicit content is found in the app, the owner is penalised for it.

C The Android system uses the certificate as a means of identifying the author of an application and establishing trust relationships between applications. ✓

D We can upload unsigned apk also as an anonymous person, however in such cases the developers are not identified and the revenue generated by the app is gone to Google.



if user is already logged in
Welcome Screen
Login Page
Dashboard (All restaurants)
Save to favourites
Favourite Restaurants
Restaurant Details
Food items added to cart
Cart
Order successfully placed
Unregistered user
Registration Page
After user registers
Forgot Password
Reset Password
Password reset successfully
User Profile
Order History
Frequently Asked Questions
Logout
Order Placed

World of Kotlin (Assignment)

PROBLEM STATEMENT:

In this module, you have learned about the following in Kotlin:

1. Variables and Operators
2. Functions
3. Arrays, Lists, and Strings
4. Conditionals and Loops
5. Exception Handling
6. Classes, Objects, and Interfaces

If you still do not feel confident with the above, we recommend you to take a quick recap of the module - don't worry, practice makes one perfect! But if you are feeling confident and raring to go, it is time to put your knowledge to work.

In our day-to-day life, we come across various problems that can be solved with the help of programming. We will try to solve a similar problem with the help of Kotlin programming and your problem-solving skills.

For this assignment, we will try to automate the different morning tasks which most of us perform before going to class in college. We will create a Robot which will perform these tasks for us to simplify our lives.

Every day in the morning we need to complete some set of predefined tasks that remain constant. We can list the generic tasks as below:

1. Ring the alarm on time
2. Make coffee
3. Heat the water to a suitable temperature
4. Pack your bag (Keep only appropriate books for the day)
5. Cook breakfast and lunch
6. Iron the clothes

From now on a Kotlin Robot will complete the above set of tasks for us.

SPECIFICATIONS:

About the Robot:

Consider the Robot as your personal assistant. You will give the Robot a name by which you will call it. You also need to define the functions performed by the Robot. These functions will be the same as above i.e. the tasks which it will perform for you in the morning. Feel free to add/modify the above-listed tasks as per your choice.

Ring the alarm:

The Robot will ring the alarm every day at a specified time which will be set by you. You also need to set the days on which the alarm will not ring.

Make coffee:

The Robot will make coffee for you every day. You need to tell the Robot about the following details:

1. How do you like your coffee (Black/With Milk).
2. How much sugar you'll take.

You can also customize the preferences according to the day of the week.

Heat the water:

You need to tell the Robot about the temperature you like for your bathing water. Additional information which can be added is whether you'll be bathing on a particular day or not. Although, we recommend that you take baths every day ;).

Pack your bag:

The Robot will keep the books in your bag as per your timetable. You need to input the timetable into your Robot's memory so that it works accordingly.

Cook breakfast and lunch:

The Robot will cook your food according to your taste. You can let the Robot know which food items do you like for breakfast and lunch and it will choose randomly from the list provided by you. Here is a tip for you. To randomly choose an option from a list you can use the *.random()* method on the list.

Iron the clothes:

Just tell the Robot what you want to wear before taking the bath and the clothes will be ready for you as soon as you come out after bathing.

The above tasks done by the Robot will surely simplify your life. As most of the work will be done by your Robot, you can choose to have an extra hour of sleep. However, make sure to give the commands to the Robot before you sleep for the night, else everything will be ruined and you'll be late for the class.

To make the Robot, you need to create a project in Kotlin that will be responsible for all the above-listed tasks. Just recall the concepts taught to you in this module and try to approach the problem keeping those concepts in mind. So put your programming hats on, it's time to get the work done and sleep a bit more every day. All the best!

*(*If you are not able to figure out the approach to this problem, don't worry we have made a cheat sheet for you that you can see on next page)*

Submission

Create the project in IntelliJ IDEA and after completing it, **Click on File -> Export to Zip file**. Now upload the created zip file to the progress tracker.

After uploading it, you will get the solution for the assignment. Steps to open the solution:

1. Unzip the folder.
2. Open IntelliJ IDEA.
3. Click File -> Open.
4. Now select the folder inside the unzipped solution folder.

Cheat Sheet:

1. For making the Robot, create a class Robot and give its name as a primary parameter.

2. The tasks performed by the Robot will become the functions of this class. Each function will simply print the tasks being performed.

3. The variants for each function will become the parameters of those functions. For storing similar items, try to make use of ArrayList and its functions.

4. After this use the object-oriented techniques using this Robot class in your main function.

5. Also, for some detailed objects like ingredients for coffee, create a data class Coffee and put its parameters in those. You can do this for other functions as well.

All the best! Happy Coding :)

Android Kick-off (Assignment)

PROBLEM STATEMENT:

In this module you have learnt about the following topics:

1. Different types of Layouts (Linear Layout, Relative Layout, and ScrollView)
2. Various widgets (TextView, EditText, ImageView, Buttons, etc)
3. Intents and Shared Preferences

If you would like one more revision and some more practice, we recommend you to go back and go through the module once more. But if you are feeling confident, it is time to put the knowledge to work.

Whenever you install a new app, there are a few common things which we come across every time:

1. A welcome/start page that displays the app logo and disappears after a few seconds. This is known as the **Splash page**. This page contains only a background and an Image placed in the center of the page.
2. The next is the **Login page** which contains the two Edit texts and a Button to log in. Along with these you also find some text which generally says '*If you do not have an account, Register*' and '*Forgot Password*'.
3. When you click on the registration text, you are sent to a new page which requires you to fill some details like name, phone number, email address, password and then you can register with the application. This page is the **Registration page** of the application.
4. Similarly, clicking on the forgot password text sends you to some other page having different functionalities.

For this assignment, you need to make the above four functionalities in an application. You might be thinking that many of these functionalities require internet and data filled in the fields are stored at some place. Yes, this happens and we will be learning that in the upcoming modules. But for now we will simply just make a functional UI for the same. We will provide you with some samples which you can refer to. However, we strongly recommend that you try to use your own imagination and make the UI look even better.

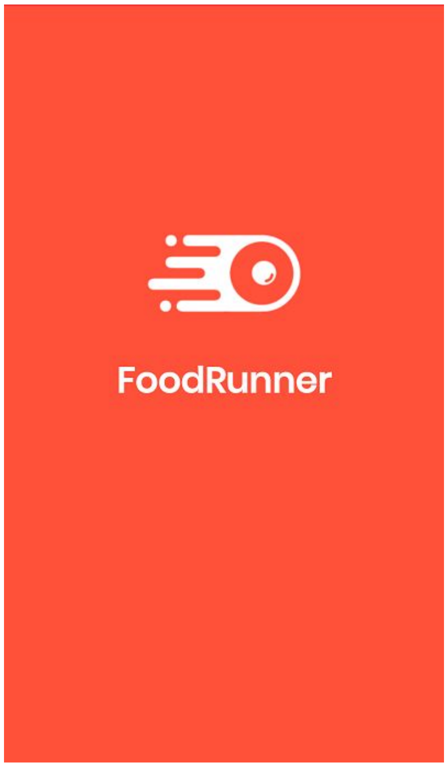
SPECIFICATIONS:

Welcome Page (aka Splash Page)

This will be the first page that gets displayed when the user opens the app. This page will contain the app name and/or app logo. You can also use any custom background for this page.

This page will be displayed for 2 seconds and then it moves on to the login page without any user interaction. The automatic exiting of the page is done by the usage of threads. We do not have to learn more about them here. We will learn about them in-depth when we reach the concept of Multithreading in the next module. This is a very important topic, so for now, we would like you to Google about exiting the splash (this is a very vast subject and you'll need your best googling skills in order to excel in the subject).

Refer to the screenshot below and try to make it.



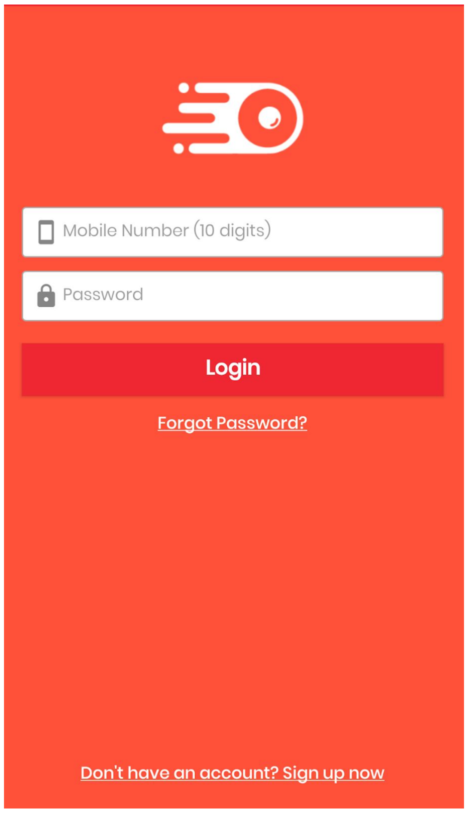
*(*Don't worry if you fail to find the functionality on Google. Refer the cheat code given at the end of this document).*

Login Page

The login page is displayed after the welcome/splash page. The user should be able to enter the mobile number and password and click on the Login button present below it.

For now, clicking on the Login button should take the user to a new blank screen where the user is greeted with a welcome message and can view the credentials entered. This will be done by sending the data using the Intents.

Clicking on the forgot password text will take the user to the forgot password page while clicking on the registration text will redirect the user to the registration page.
Refer to the screenshot below for a better understanding and try to make the page.

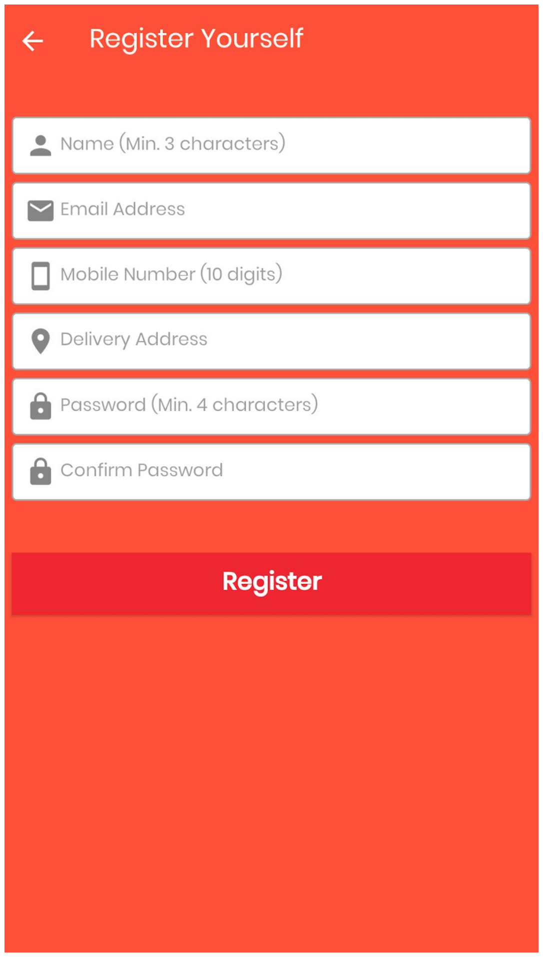


Bonus:
Save the login instance as true in shared preferences and when the user opens the app for the next time, she is not asked to log in again. This can be a bit tricky, so try to implement it without crashing the app.

Registration Page

This page contains the following fields namely: **Name, Email Address, Mobile Number, Delivery Address, and Password** which users will input and register for the app. For now, you only need to create the page and the functionality for this will be made later.

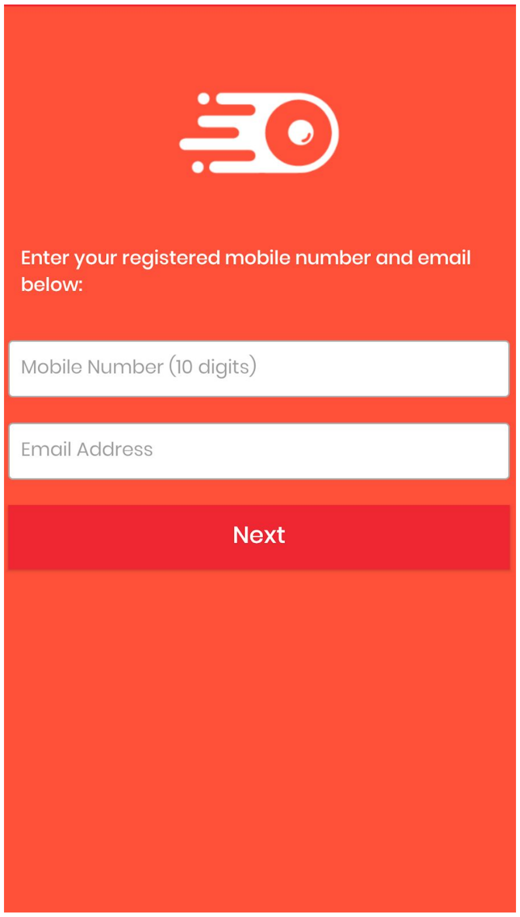
On clicking the register button, the user is taken to a blank page where the entered information will be displayed. Similar to the login page, this will be done with the help of Intents.



Forgot Password Page

This page will contain only two fields where the user is requested to enter the mobile number and email address.

On clicking the next button the user is again redirected to the blank screen where the entered information is displayed.


Enter your registered mobile number and email below:

Submission

Create the project in Android Studio and after completing it, **Click on File -> Export to Zip file**. Now upload the created zip file to the progress tracker.

After uploading it, you will get the solution for the assignment. Steps to open the solution:

1. Unzip the folder.
2. Open Android Studio.
3. Click File -> Open.
4. Now select the folder inside the unzipped solution folder. Make sure the folder you are selecting has this( ) icon.

Important :
Kindly try to complete the assignments as they will help you in completing the final project of the training.

Cheat Code:

Splash page thread implementation: In the onCreate() method, just put the startActivity() method inside a Handler()

```
Handler().postDelayed({
  val startAct = Intent(this@SplashActivty, LoginActivity::class.java)
  startActivity(startAct)
}, 2000)
```

All the best!

Higher-Order Functionalities (Assignment)

PROBLEM STATEMENT:

In this module you have learned about the following topics listed below:

1. Navigation drawer
2. Fragments
3. Recycler View
4. Fetching/Sending data from/to Internet
5. SQLite Database

Hope you are feeling confident and excited about what you have learned till now and are ready to put it to use to build something great. If you are not, no worries, revise the concepts as many times as you need till you feel you have gained a good grasp on the subject.

This assignment would be a continuation of the previous one where we made the splash, login page, registration page and the forgot password page and they all ended up in a blank page where the credentials were displayed. Now it is time to make that app further.

The pages to be made will be the

1. **Home page**
2. **The favorites page**
3. **Profile page**
4. **FAQ page**

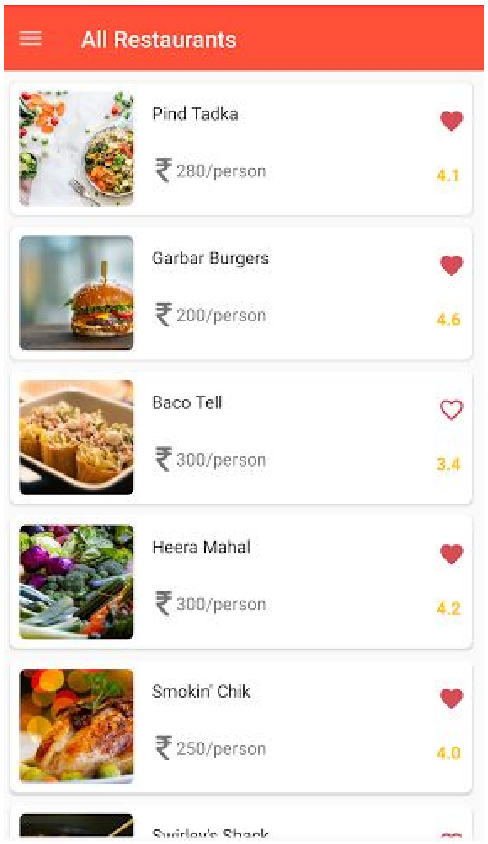
All these pages will be fragments and will open from the navigation drawer and some data will be fetched from an external server just like we did earlier*.

*(*Don't worry we will provide you with the links to fetch the JSON data from the server)*

SPECIFICATIONS:

Home Page

This would be the page which gets displayed when the user logs in to the application. This page would contain the recycler view which contains the list of restaurants. These restaurants will be fetched from an external server by sending a GET request to the below link:
*<**http://13.235.250.119/v2/restaurants/fetch_result/**>
The result obtained would be an array of JSON objects. Kindly use Postman to check the result of the request. You need to set up this data into the recycler view in such a way that the page looks somewhat similar to the screenshot below:

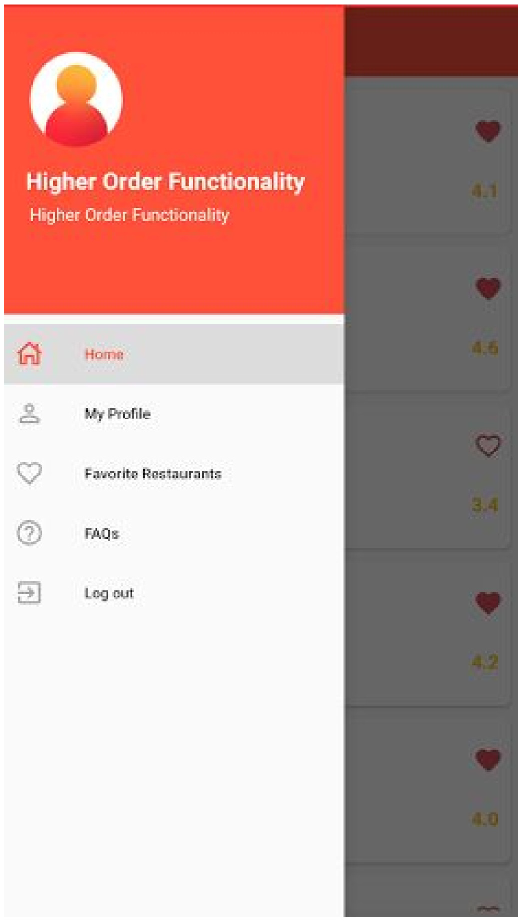


Along with this, the small heart icons are used to save any particular restaurant as favorite. This is explained in more detail in the section for **Favorites Page**.

** You will be required to send the headers along with the request as taught. You can also find the code in the cheat sheet below*

Navigation Drawer

Along with the search bar, the navigation drawer is also attached in the home page. The navigation drawer contains the route to different screens. Refer the screenshot below to get the idea of how this will look.

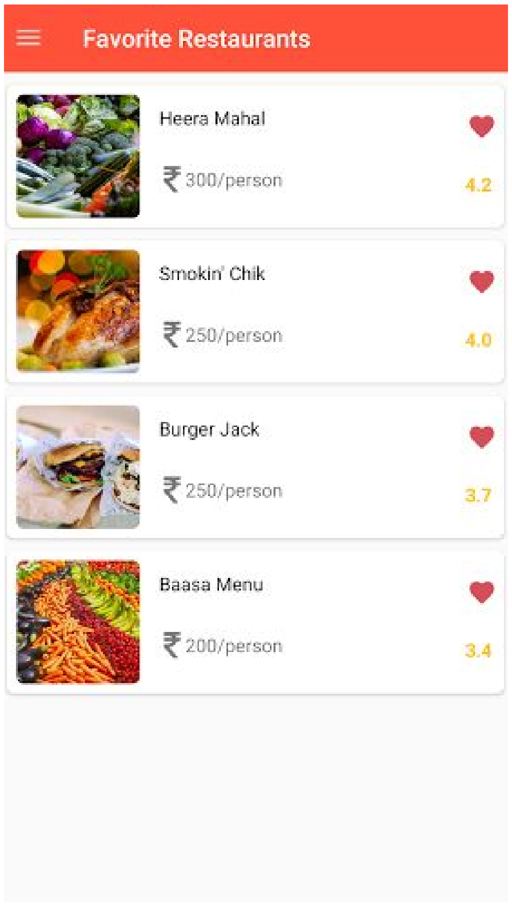


We encourage you to customize this according to your choice.

Favorites Page

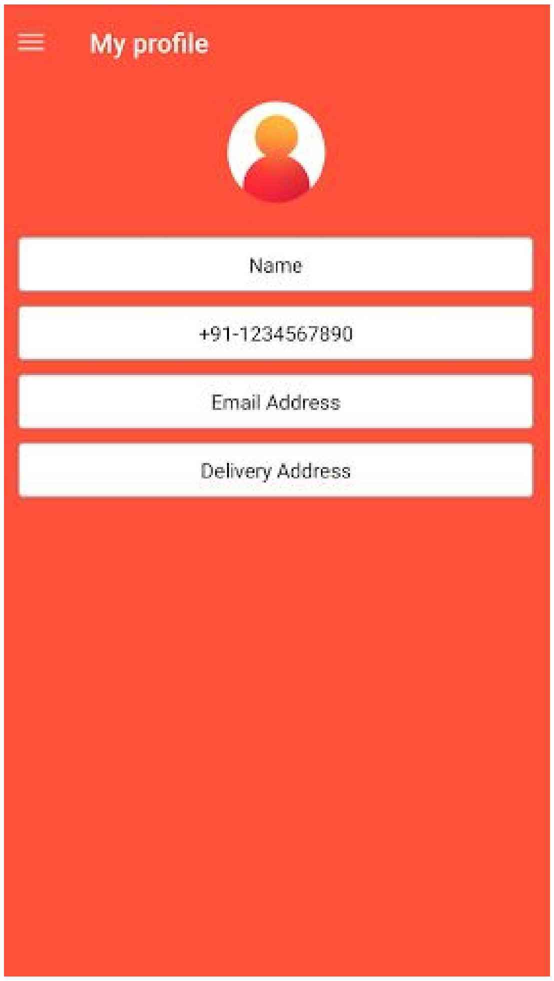
This page will contain the list of the restaurants which were marked as a favorite by you. The favorite restaurants will be saved in the local DB i.e. the SQLite database. The database will contain the list of the restaurants saved as favorites and in the favorite fragment, we can directly populate our recycler view with the data from the DB.

The favorite fragment will look and behave exactly similar to the home page but the restaurants listed there would be the user's favorite restaurants. Refer the screenshot below:



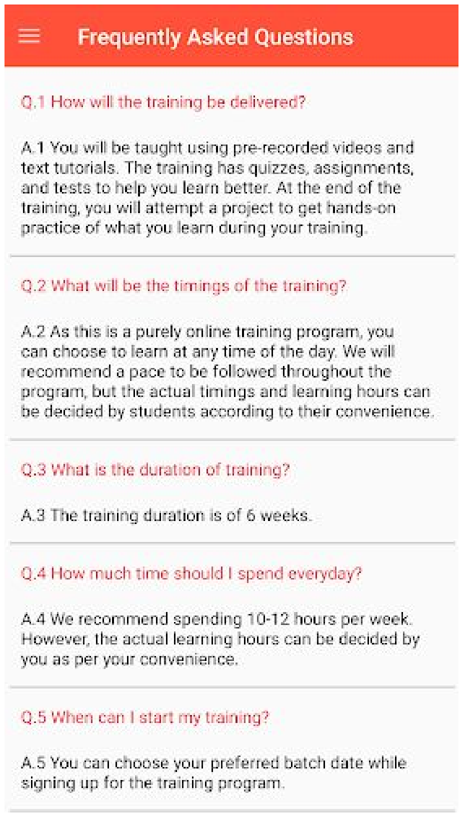
Profile Page

This page also opens up from the navigation drawer. The page contains the details you entered during the registration for the application. A sample profile page would look like the screenshot below:



Frequently Asked Question(FAQ) page

This page will contain a static list of questions. It is up to you as to what questions you want to put there. This page will contain static data i.e. some questions and answers. You can hardcode these questions and answers. Refer the sample screenshot below:



Q.1 How will the training be delivered?

A.1 You will be taught using pre-recorded videos and text tutorials. The training has quizzes, assignments, and tests to help you learn better. At the end of the training, you will attempt a project to get hands-on practice of what you learn during your training.

Q.2 What will be the timings of the training?

A.2 As this is a purely online training program, you can choose to learn at any time of the day. We will recommend a pace to be followed throughout the program, but the actual timings and learning hours can be decided by students according to their convenience.

Q.3 What is the duration of training?

A.3 The training duration is of 6 weeks.

Q.4 How much time should I spend everyday?

A.4 We recommend spending 10-12 hours per week. However, the actual learning hours can be decided by you as per your convenience.

Q.5 When can I start my training?

A.5 You can choose your preferred batch date while signing up for the training program.

<u>Note:</u>

The given screenshots are just for better understanding. Feel free (we recommend) to customize the look and feel of the app. You can choose your own icons, colors, and images in the app. Only the data sent from the API is sent by us and cannot be edited.

Submission

Create the project in Android Studio and after completing it, **Click on File -> Export to Zip file**. Now upload the created zip file to the progress tracker.

After uploading it, you will get the solution for the assignment. Steps to open the assignment:

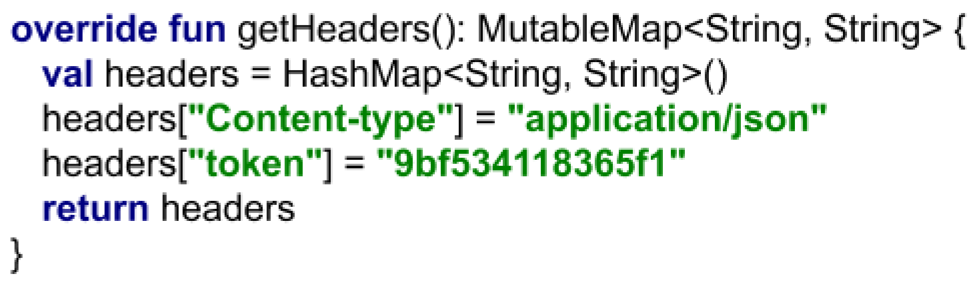
1. Unzip the folder
2. Open Android Studio
3. Click Open
4. Now select the folder inside the unzipped solution folder. Make sure the folder you are selecting has this( ) icon.

Important :

Kindly try to complete the assignment as it will help you in completing the final project of the training

Cheat Code:

Use the below code along with the request for headers:



```
override fun getHeaders(): MutableMap<String, String> {
   val headers = HashMap<String, String>()
   headers["Content-type"] = "application/json"
   headers["token"] = "9bf534118365f1"
   return headers
}
```

Also, the above token will not work. Kindly use the token provided to you in the training.

Bookhub

```
│   │           ├── activity
│   │           │   ├── DescriptionActivity.kt
│   │           │   └── MainActivity.kt
│   │           ├── adapter
│   │           │   ├── DashboardRecyclerAdapter.kt
│   │           │   └── FavouriteRecyclerAdapter.kt
│   │           ├── database
│   │           │   ├── BookDao.kt
│   │           │   ├── BookDatabase.kt
│   │           │   └── BookEntity.kt
│   │           ├── fragment
│   │           │   ├── AboutAppFragment.kt
│   │           │   ├── DashboardFragment.kt
│   │           │   ├── FavouritesFragment.kt
│   │           │   └── ProfileFragment.kt
│   │           ├── model
│   │           │   └── Book.kt
│   │           └── util
│   │               └── ConnectionManager.kt
│   └── res
│       ├── layout
│       │   ├── activity_description.xml
│       │   ├── activity_main.xml
│       │   ├── drawer_header.xml
│       │   ├── fragment_about_app.xml
│       │   ├── fragment_dashboard.xml
│       │   ├── fragment_favourites.xml
│       │   ├── fragment_profile.xml
│       │   ├── recycler_dashboard_single_row.xml
│       │   └── recycler_favourite_single_row.xml
│       ├── menu
│       │   ├── menu_dashboard.xml
│       │   └── menu_drawer.xml
│       ├── values
│       │   ├── book_app_icon_background.xml
│       │   ├── colors.xml
│       │   ├── strings.xml
│       │   └── styles.xml
│       └── xml
│           └── network_security_config.xm
│
```

1. activity/MainActivity.kt

```
package com.internshala.bookhub.activity

class MainActivity : AppCompatActivity() {

    lateinit var drawerLayout: DrawerLayout
    lateinit var coordinatorLayout: CoordinatorLayout
    lateinit var toolbar: Toolbar
    lateinit var frameLayout: FrameLayout
    lateinit var navigationView: NavigationView
```

```kotlin
var previousMenuItem: MenuItem? = null

override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    drawerLayout = findViewById(R.id.drawerLayout)
    coordinatorLayout = findViewById(R.id.coordinatorLayout)
    toolbar = findViewById(R.id.toolbar)
    frameLayout = findViewById(R.id.frame)
    navigationView = findViewById(R.id.navigationView)
    setUpToolbar()

    openDashboard()

    val actionBarDrawerToggle = ActionBarDrawerToggle(
        this@MainActivity,
        drawerLayout,
        R.string.open_drawer,
        R.string.close_drawer
    )

    drawerLayout.addDrawerListener(actionBarDrawerToggle)
    actionBarDrawerToggle.syncState()

    navigationView.setNavigationItemSelectedListener {

        if (previousMenuItem != null){
            previousMenuItem?.isChecked = false
        }

        it.isCheckable = true
        it.isChecked = true
        previousMenuItem = it

        when(it.itemId){
            R.id.dashboard -> {
                openDashboard()
                drawerLayout.closeDrawers()

            }
            R.id.favourites -> {
                supportFragmentManager.beginTransaction()
                    .replace(
                        R.id.frame,
                        FavouritesFragment()
                    )
                    .commit()

                supportActionBar?.title = "Favourites"
                drawerLayout.closeDrawers()
            }
            R.id.profile -> {
                supportFragmentManager.beginTransaction()
                    .replace(
                        R.id.frame,
                        ProfileFragment()
                    )
                    .commit()

                supportActionBar?.title = "Profile"
                drawerLayout.closeDrawers()
            }
            R.id.aboutApp -> {
                supportFragmentManager.beginTransaction()
                    .replace(
                        R.id.frame,
                        AboutAppFragment()
                    )
                    .commit()

                supportActionBar?.title = "About App"
                drawerLayout.closeDrawers()
            }
        }
        return@setNavigationItemSelectedListener true
    }
    }

    fun setUpToolbar() {
        setSupportActionBar(toolbar)
        supportActionBar?.title = "Toolbar Title"
        supportActionBar?.setHomeButtonEnabled(true)
        supportActionBar?.setDisplayHomeAsUpEnabled(true)
    }

    override fun onOptionsItemSelected(item: MenuItem): Boolean {

        val id = item.itemId

        if (id == android.R.id.home){
            drawerLayout.openDrawer(GravityCompat.START)
        }

        return super.onOptionsItemSelected(item)
    }

    fun openDashboard(){
        val fragment = DashboardFragment()
        val transaction = supportFragmentManager.beginTransaction()
        transaction.replace(R.id.frame, fragment)
        transaction.commit()
        supportActionBar?.title = "Dashboard"
        navigationView.setCheckedItem(R.id.dashboard)
    }

    override fun onBackPressed() {
        val frag = supportFragmentManager.findFragmentById(R.id.frame)

        when(frag){
            !is DashboardFragment -> openDashboard()

            else -> super.onBackPressed()
        }
    }
}
```

2. activity/DescriptionActivity.kt

```kotlin
package com.internshala.bookhub.activity

class DescriptionActivity : AppCompatActivity() {

    lateinit var txtBookName: TextView
    lateinit var txtBookAuthor: TextView
    lateinit var txtBookPrice: TextView
    lateinit var txtBookRating: TextView
    lateinit var imgBookImage: ImageView
    lateinit var txtBookDesc: TextView
    lateinit var btnAddToFav: Button
    lateinit var progressBar: ProgressBar
    lateinit var progressLayout: RelativeLayout

    lateinit var toolbar: Toolbar

    var bookId: String? = "100"

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_description)

        txtBookName = findViewById(R.id.txtBookName)
        txtBookAuthor = findViewById(R.id.txtBookAuthor)
        txtBookPrice = findViewById(R.id.txtBookPrice)
        txtBookRating = findViewById(R.id.txtBookRating)
        imgBookImage = findViewById(R.id.imgBookImage)
        txtBookDesc = findViewById(R.id.txtBookDesc)
        btnAddToFav = findViewById(R.id.btnAddToFav)
        progressBar = findViewById(R.id.progressBar)
        progressBar.visibility = View.VISIBLE
        progressLayout = findViewById(R.id.progressLayout)
        progressLayout.visibility = View.VISIBLE

        toolbar = findViewById(R.id.toolbar)
        setSupportActionBar(toolbar)
```

```kotlin
supportActionBar?.title = "Book Details"

if (intent != null) {
    bookId = intent.getStringExtra("book_id")
} else {
    finish()
    Toast.makeText(
        this@DescriptionActivity,
        "Some unexpected error occurred!",
        Toast.LENGTH_SHORT
    ).show()
}

if (bookId == "100") {
    finish()
    Toast.makeText(
        this@DescriptionActivity,
        "Some unexpected error occurred!",
        Toast.LENGTH_SHORT
    ).show()
}

val queue = Volley.newRequestQueue(this@DescriptionActivity)
val url = "http://13.235.250.119/v1/book/get_book/"

val jsonParams = JSONObject()
jsonParams.put("book_id", bookId)


if (ConnectionManager().checkConnectivity(this@DescriptionActivity)) {
    val jsonRequest =
        object : JsonObjectRequest(Request.Method.POST, url, jsonParams, Response.Listener {

            try {

                val success = it.getBoolean("success")
                if (success) {
                    val bookJsonObject = it.getJSONObject("book_data")
                    progressLayout.visibility = View.GONE

                    val bookImageUrl = bookJsonObject.getString("image")
                    Picasso.get().load(bookJsonObject.getString("image"))
                        .error(R.drawable.default_book_cover).into(imgBookImage)
                    txtBookName.text = bookJsonObject.getString("name")
                    txtBookAuthor.text = bookJsonObject.getString("author")
                    txtBookPrice.text = bookJsonObject.getString("price")
                    txtBookRating.text = bookJsonObject.getString("rating")
                    txtBookDesc.text = bookJsonObject.getString("description")

                    val bookEntity = BookEntity(
                        bookId?.toInt() as Int,
                        txtBookName.text.toString(),
                        txtBookAuthor.text.toString(),
                        txtBookPrice.text.toString(),
                        txtBookRating.text.toString(),
                        txtBookDesc.text.toString(),
                        bookImageUrl
                    )

                    val checkFav = DBAsyncTask(applicationContext, bookEntity, 1).execute()
                    val isFav = checkFav.get()

                    if (isFav) {
                        btnAddToFav.text = "Remove from Favourites"
                        val favColor = ContextCompat.getColor(
                            applicationContext,
                            R.color.colorFavourite
                        )
                        btnAddToFav.setBackgroundColor(favColor)
                    } else {
                        btnAddToFav.text = "Add to Favourites"
                        val noFavColor =
                            ContextCompat.getColor(applicationContext, R.color.colorPrimary)
                        btnAddToFav.setBackgroundColor(noFavColor)
                    }

                    btnAddToFav.setOnClickListener {
```

```kotlin
                if (!DBAsyncTask(
                        applicationContext,
                        bookEntity,
                        1
                    ).execute().get()
                ) {

                    val async =
                        DBAsyncTask(applicationContext, bookEntity, 2).execute()
                    val result = async.get()
                    if (result) {
                        Toast.makeText(
                            this@DescriptionActivity,
                            "Book added to favourites",
                            Toast.LENGTH_SHORT
                        ).show()

                        btnAddToFav.text = "Remove from favourites"
                        val favColor = ContextCompat.getColor(applicationContext,
R.color.colorFavourite)
                        btnAddToFav.setBackgroundColor(favColor)
                    } else {
                        Toast.makeText(
                            this@DescriptionActivity,
                            "Some error occurred!",
                            Toast.LENGTH_SHORT
                        ).show()
                    }
                } else {

                    val async = DBAsyncTask(applicationContext, bookEntity,
3).execute()
                    val result = async.get()

                    if (result){
                        Toast.makeText(
                            this@DescriptionActivity,
                            "Book removed from favourites",
                            Toast.LENGTH_SHORT
                        ).show()

                        btnAddToFav.text = "Add to favourites"
                        val noFavColor =
                            ContextCompat.getColor(applicationContext,
R.color.colorPrimary)
                        btnAddToFav.setBackgroundColor(noFavColor)
                    } else {
                        Toast.makeText(
                            this@DescriptionActivity,
                            "Some error occurred!",
                            Toast.LENGTH_SHORT
                        ).show()
                    }

                }
              }

            } else {
              Toast.makeText(
                  this@DescriptionActivity,
                  "Some Error Occurred!",
                  Toast.LENGTH_SHORT
              ).show()
            }

          } catch (e: Exception) {
            Toast.makeText(
                this@DescriptionActivity,
                "Some error occurred!",
                Toast.LENGTH_SHORT
            ).show()
          }

        }, Response.ErrorListener {

          Toast.makeText(this@DescriptionActivity, "Volley Error $it", Toast.LENGTH_SHORT)
            .show()
```

```
                }) {
                    override fun getHeaders(): MutableMap<String, String> {
                        val headers = HashMap<String, String>()
                        headers["Content-type"] = "application/json"
                        headers["token"] = "9bf534118365f1"
                        return headers
                    }
                }

            queue.add(jsonRequest)
        } else {
            val dialog = AlertDialog.Builder(this@DescriptionActivity)
            dialog.setTitle("Error")
            dialog.setMessage("Internet Connection is not Found")
            dialog.setPositiveButton("Open Settings") { text, listener ->
                val settingsIntent = Intent(Settings.ACTION_WIRELESS_SETTINGS)
                startActivity(settingsIntent)
                finish()
            }

            dialog.setNegativeButton("Exit") { text, listener ->
                ActivityCompat.finishAffinity(this@DescriptionActivity)
            }
            dialog.create()
            dialog.show()
        }
    }

    class DBAsyncTask(val context: Context, val bookEntity: BookEntity, val mode: Int) :
        AsyncTask<Void, Void, Boolean>() {
        /*
        Mode 1 -> Check DB if the book is favourite or not
        Mode 2 -> Save the book into DB as favourite
        Mode 3 -> Remove the favourite book
        * */
        val db = Room.databaseBuilder(context, BookDatabase::class.java, "books-db").build()

        override fun doInBackground(vararg p0: Void?): Boolean {

            when (mode) {
                1 -> {// Check DB if the book is favourite or not
                    val book: BookEntity? = db.bookDao().getBookById(bookEntity.book_id.toString())
                    db.close()
                    return book != null
                }
                2 -> {// Save the book into DB as favourite
                    db.bookDao().insertBook(bookEntity)
                    db.close()
                    return true
                }
                3 -> {// Remove the favourite book
                    db.bookDao().deleteBook(bookEntity)
                    db.close()
                    return true
                }
            }
            return false
        }
    }
}
```

3. adapter/DashboardRecyclerAdapter.kt

```
package com.internshala.bookhub.adapter

class DashboardRecyclerAdapter(val context: Context, val itemList: ArrayList<Book>) :
RecyclerView.Adapter<DashboardRecyclerAdapter.DashboardViewHolder>() {
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): DashboardViewHolder {
        val view =
LayoutInflater.from(parent.context).inflate(R.layout.recycler_dashboard_single_row, parent, false)

        return DashboardViewHolder(view)
    }

    override fun getItemCount(): Int {
        return itemList.size
    }

    override fun onBindViewHolder(holder: DashboardViewHolder, position: Int) {
        val book = itemList[position]
        holder.txtBookName.text = book.bookName
        holder.txtBookAuthor.text = book.bookAuthor
        holder.txBookPrice.text = book.bookPrice
        holder.txtBookRating.text = book.bookRating
        //holder.imgBookImage.setImageResource(book.bookImage)

Picasso.get().load(book.bookImage).error(R.drawable.default_book_cover).into(holder.imgBookImage)

        holder.llContent.setOnClickListener {
            val intent = Intent(context, DescriptionActivity::class.java)
            intent.putExtra("book_id", book.bookId)
            context.startActivity(intent)
        }

    }

    class DashboardViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val txtBookName: TextView = view.findViewById(R.id.txtBookName)
        val txtBookAuthor: TextView = view.findViewById(R.id.txtBookAuthor)
        val txBookPrice: TextView = view.findViewById(R.id.txtBookPrice)
        val txtBookRating: TextView = view.findViewById(R.id.txtBookRating)
        val imgBookImage: ImageView = view.findViewById(R.id.imgBookImage)
        val llContent: LinearLayout = view.findViewById(R.id.llContent)
    }
}
```

4. adapter/FavouriteRecyclerAdapter.kt

```
package com.internshala.bookhub.adapter

class FavouriteRecyclerAdapter(val context: Context, val bookList: List<BookEntity>) :
    RecyclerView.Adapter<FavouriteRecyclerAdapter.FavouriteViewHolder>() {

    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): FavouriteViewHolder {
        val view = LayoutInflater.from(parent.context)
            .inflate(R.layout.recycler_favourite_single_row, parent, false)

        return FavouriteViewHolder(view)
    }

    override fun getItemCount(): Int {
        return bookList.size
    }

    override fun onBindViewHolder(holder: FavouriteViewHolder, position: Int) {

        val book = bookList[position]

        holder.txtBookName.text = book.bookName
        holder.txtBookAuthor.text = book.bookAuthor
        holder.txtBookPrice.text = book.bookPrice
        holder.txtBookRating.text = book.bookRating

Picasso.get().load(book.bookImage).error(R.drawable.default_book_cover).into(holder.imgBookImage)
    }

    class FavouriteViewHolder(view: View) : RecyclerView.ViewHolder(view) {
        val txtBookName: TextView = view.findViewById(R.id.txtFavBookTitle)
        val txtBookAuthor: TextView = view.findViewById(R.id.txtFavBookAuthor)
        val txtBookPrice: TextView = view.findViewById(R.id.txtFavBookPrice)
        val txtBookRating: TextView = view.findViewById(R.id.txtFavBookRating)
        val imgBookImage: ImageView = view.findViewById(R.id.imgFavBookImage)
        val llContent: LinearLayout = view.findViewById(R.id.llFavContent)
    }
}
```

5. database/BookDao.kt

```
package com.internshala.bookhub.database

import androidx.room.Dao
import androidx.room.Delete
import androidx.room.Insert
import androidx.room.Query
import com.internshala.bookhub.model.Book
```

```
@Dao
interface BookDao {

    @Insert
    fun insertBook(bookEntity: BookEntity)

    @Delete
    fun deleteBook(bookEntity: BookEntity)

    @Query("SELECT * FROM books")
    fun getAllBooks(): List<BookEntity>

    @Query("SELECT * FROM books WHERE book_id = :bookId")
    fun getBookById(bookId: String): BookEntity
}
```

6. database/BookEntity

```
package com.internshala.bookhub.database

import androidx.room.Dao
import androidx.room.Delete
import androidx.room.Insert
import androidx.room.Query
import com.internshala.bookhub.model.Book

@Dao
interface BookDao {
    @Insert
    fun insertBook(bookEntity: BookEntity)
    @Delete
    fun deleteBook(bookEntity: BookEntity)
    @Query("SELECT * FROM books")
    fun getAllBooks(): List<BookEntity>
    @Query("SELECT * FROM books WHERE book_id = :bookId")
    fun getBookById(bookId: String): BookEntity
}
```

7. fragment/AboutAppFragment.kt

```
class AboutAppFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_about_app, container, false)
    }
}
```

8. fragment/DashboardFragment.kt

```
package com.internshala.bookhub.fragment
class DashboardFragment : Fragment() {

    lateinit var recyclerDashboard: RecyclerView
    lateinit var layoutManager: RecyclerView.LayoutManager
    lateinit var recyclerAdapter: DashboardRecyclerAdapter
    lateinit var progressLayout: RelativeLayout
    lateinit var progressBar: ProgressBar
    val bookInfoList = arrayListOf<Book>()

    var ratingComparator = Comparator<Book>{book1, book2 ->

        if (book1.bookRating.compareTo(book2.bookRating, true) == 0) {
            // sort according to name if rating is same
            book1.bookName.compareTo(book2.bookName, true)
        } else {
            book1.bookRating.compareTo(book2.bookRating, true)
        }
    }

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view = inflater.inflate(R.layout.fragment_dashboard, container, false)

        setHasOptionsMenu(true)

        recyclerDashboard = view.findViewById(R.id.recyclerDashboard)
        progressLayout = view.findViewById(R.id.progressLayout)
        progressBar = view.findViewById(R.id.progressBar)
        progressLayout.visibility = View.VISIBLE
        layoutManager = LinearLayoutManager(activity)


        val queue = Volley.newRequestQueue(activity as Context)

        val url = "http://13.235.250.119/v1/book/fetch_books/"

        if (ConnectionManager().checkConnectivity(activity as Context)){
            val jsonObjectRequest = object : JsonObjectRequest(Request.Method.GET, url, null,
Response.Listener {

                // Here we will handle the response
                try {
                    progressLayout.visibility = View.GONE
                    val success = it.getBoolean("success")

                    if (success){

                        val data = it.getJSONArray("data")
                        for (i in 0 until data.length()){
                            val bookJsonObject = data.getJSONObject(i)
                            val bookObject = Book(
                                bookJsonObject.getString("book_id"),
                                bookJsonObject.getString("name"),
                                bookJsonObject.getString("author"),
                                bookJsonObject.getString("rating"),
                                bookJsonObject.getString("price"),
                                bookJsonObject.getString("image")
                            )
                            bookInfoList.add(bookObject)
                            recyclerAdapter = DashboardRecyclerAdapter(activity as Context,
bookInfoList)

                            recyclerDashboard.adapter = recyclerAdapter

                            recyclerDashboard.layoutManager = layoutManager

                        }

                    } else {
                        Toast.makeText(activity as Context, "Some Error Occurred!",
Toast.LENGTH_SHORT).show()
                    }
                } catch (e: JSONException) {
                    Toast.makeText(activity as Context, "Some unexpected error occurred!",
Toast.LENGTH_SHORT).show()
                }

            }, Response.ErrorListener {

                //Here we will handle the errors
                if (activity != null){
                    Toast.makeText(activity as Context, "Volley error occurred!",
Toast.LENGTH_SHORT).show()
                }

            }){
                override fun getHeaders(): MutableMap<String, String> {
                    val headers = HashMap<String, String>()
                    headers["Content-type"] = "application/json"
                    headers["token"] = "9bf534118365f1"
                    return headers
                }
            }

            queue.add(jsonObjectRequest)

        } else {
            val dialog = AlertDialog.Builder(activity as Context)
            dialog.setTitle("Error")
            dialog.setMessage("Internet Connection is not Found")
            dialog.setPositiveButton("Open Settings"){ _, _ ->
                val settingsIntent = Intent(Settings.ACTION_WIRELESS_SETTINGS)
                startActivity(settingsIntent)
                activity?.finish()
            }
            dialog.setNegativeButton("Exit") { _, _ ->
                ActivityCompat.finishAffinity(activity as Activity)
            }
            dialog.create()
            dialog.show()
        }

        return view
    }
    override fun onCreateOptionsMenu(menu: Menu?, inflater: MenuInflater?) {
        inflater?.inflate(R.menu.menu_dashboard, menu)
    }

    override fun onOptionsItemSelected(item: MenuItem?): Boolean {

        val id = item?.itemId
        if (id == R.id.action_sort){
            Collections.sort(bookInfoList, ratingComparator)
            bookInfoList.reverse()
        }

        recyclerAdapter.notifyDataSetChanged()
        return super.onOptionsItemSelected(item)
    }
}
```

9. fragment/FavouritesFragment.kt

```
package com.internshala.bookhub.fragment
class FavouritesFragment : Fragment() {

    lateinit var recyclerFavourite: RecyclerView
    lateinit var progressLayout: RelativeLayout
    lateinit var progressBar: ProgressBar
    lateinit var layoutManager: RecyclerView.LayoutManager
    lateinit var recyclerAdapter: FavouriteRecyclerAdapter
    var dbBookList = listOf<BookEntity>()

    override fun onCreateView(
        inflater: LayoutInflater, container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // Inflate the layout for this fragment
        val view = inflater.inflate(R.layout.fragment_favourites, container, false)

        recyclerFavourite = view.findViewById(R.id.recyclerFavourite)
        progressLayout = view.findViewById(R.id.progressLayout)
        progressBar = view.findViewById(R.id.progressBar)

        layoutManager = GridLayoutManager(activity as Context, 2)

        dbBookList = RetrieveFavourites(activity as Context).execute().get()

        if (activity != null) {
            progressLayout.visibility = View.GONE
            recyclerAdapter = FavouriteRecyclerAdapter(activity as Context, dbBookList)
            recyclerFavourite.adapter = recyclerAdapter
            recyclerFavourite.layoutManager = layoutManager
        }

        return view
    }


    class RetrieveFavourites(val context: Context) : AsyncTask<Void, Void, List<BookEntity>>() {

        override fun doInBackground(vararg p0: Void?): List<BookEntity> {
            val db = Room.databaseBuilder(context, BookDatabase::class.java, "books-db").build()

            return db.bookDao().getAllBooks()
        }

    }
}
```

10. fragment/ProfileFragment.kt

```
package com.internshala.bookhub.fragment

class ProfileFragment : Fragment() {

 override fun onCreateView(
 inflater: LayoutInflater, container: ViewGroup?,
 savedInstanceState: Bundle?
 ): View? {
 // Inflate the layout for this fragment
 return inflater.inflate(R.layout.fragment_profile, container, false)
 }
}
```

11. model/Book.kt

```
package com.internshala.bookhub.model

data class Book(
    val bookId: String,
    val bookName: String,
    val bookAuthor: String,
    val bookRating: String,
    val bookPrice: String,
    val bookImage: String
)
```

12. util/ConnectionManager.kt

```
package com.internshala.bookhub.util

import android.content.Context
import android.net.ConnectivityManager
import android.net.NetworkInfo

class ConnectionManager {

    fun checkConnectivity(context: Context): Boolean {

        val connectivityManager = context.getSystemService(Context.CONNECTIVITY_SERVICE) as
ConnectivityManager

        val activeNetwork: NetworkInfo? = connectivityManager.activeNetworkInfo

        if (activeNetwork?.isConnected != null){
            return activeNetwork.isConnected
        } else {
            return false
        }
    }
}
```

13. layout/activity_description.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#ffffff"
    tools:context=".activity.DescriptionActivity">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="?attr/actionBarSize"
        android:background="@color/colorPrimary"
        android:minHeight="?attr/actionBarSize"
        android:theme="@style/ThemeOverlay.AppCompat.Dark" />

    <ScrollView
```

```xml
android:layout_width="match_parent"
android:layout_height="match_parent"
android:layout_below="@+id/toolbar"
android:layout_marginTop="20dp"
android:layout_above="@id/btnAddToFav">

<RelativeLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content">
    <LinearLayout
        android:id="@+id/llContent"
        android:layout_width="match_parent"
        android:layout_height="120dp"
        android:orientation="horizontal"
        android:weightSum="6">
        <ImageView
            android:id="@+id/imgBookImage"
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="1.5"
            android:padding="5dp"
            android:scaleType="centerCrop"
            android:src="@mipmap/ic_launcher" />
        <RelativeLayout
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="3.3">
            <TextView
                android:id="@+id/txtBookName"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:padding="4dp"
                android:text="Name of the book"
                android:textColor="#000000"
                android:textSize="16sp"
                android:textStyle="bold" />
            <TextView
                android:id="@+id/txtBookAuthor"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@id/txtBookName"
                android:padding="4dp"
                android:text="Name of the author"
                android:textColor="#000000"
                android:textSize="13sp" />
            <TextView
                android:id="@+id/txtBookPrice"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@id/txtBookAuthor"
                android:padding="4dp"
                android:text="Rs. 299"
                android:textColor="#357a38"
                android:textSize="14sp"
                android:textStyle="bold" />
        </RelativeLayout>
        <TextView
            android:id="@+id/txtBookRating"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1.2"
            android:drawableLeft="@drawable/ic_ratings"
            android:drawablePadding="4dp"
            android:padding="4dp"
            android:text="4.5"
            android:textColor="#ffc828"
            android:textSize="18sp"
            android:textStyle="bold" />
    </LinearLayout>
    <RelativeLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_below="@id/llContent">
        <TextView
            android:id="@+id/txtAboutTheBookStatic"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerHorizontal="true"
```

```
                android:layout_marginTop="4dp"
                android:padding="6dp"
                android:text="About the book:"
                android:textSize="16sp"
                android:textStyle="bold" />
            <TextView
                android:id="@+id/txtBookDesc"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@id/txtAboutTheBookStatic"
                android:layout_margin="10dp"
                android:padding="6dp"
                android:text="@string/hello_blank_fragment"
                android:textColor="#000000"
                android:textSize="18sp" />
        </RelativeLayout>
    </RelativeLayout>
</ScrollView>
<Button
    android:id="@+id/btnAddToFav"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_alignParentBottom="true"
    android:background="@color/colorPrimary"
    android:text="@string/add_to_favourites"
    android:textColor="#ffffff"
    android:textSize="19sp"
    android:textStyle="bold" />
<RelativeLayout
    android:id="@+id/progressLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:visibility="gone"
    android:background="#ffffff">
    <ProgressBar
        android:id="@+id/progressBar"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerInParent="true" />
</RelativeLayout>
</RelativeLayout>
```

14. layout/activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawerLayout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".activity.MainActivity">
    <androidx.coordinatorlayout.widget.CoordinatorLayout
        android:id="@+id/coordinatorLayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent">
        <com.google.android.material.appbar.AppBarLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:elevation="0dp"
            android:theme="@style/ThemeOverlay.AppCompat.Dark">
            <androidx.appcompat.widget.Toolbar
                android:id="@+id/toolbar"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:background="@color/colorPrimary"
                android:minHeight="?attr/actionBarSize"
                android:theme="@style/ThemeOverlay.AppCompat.Dark"
                app:layout_scrollFlags="scroll|enterAlways"/>
        </com.google.android.material.appbar.AppBarLayout>
        <FrameLayout
            android:id="@+id/frame"
            android:layout_width="fill_parent"
            android:layout_height="fill_parent"
            app:layout_behavior="@string/appbar_scrolling_view_behavior"/>
    </androidx.coordinatorlayout.widget.CoordinatorLayout>
    <com.google.android.material.navigation.NavigationView
```

```xml
        android:id="@+id/navigationView"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        app:headerLayout="@layout/drawer_header"
        app:menu="@menu/menu_drawer" />
</androidx.drawerlayout.widget.DrawerLayout>
```

15. layout/drawer_header.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="240dp"
    android:background="@color/colorPrimary"
    android:orientation="vertical">

    <ImageView
        android:layout_width="120dp"
        android:layout_height="120dp"
        android:layout_marginTop="20dp"
        android:padding="10dp"
        android:src="@drawable/bookhub_drawer_icon" />
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:padding="10dp"
        android:text="@string/navigation_drawer"
        android:textColor="#ffffff"
        android:textSize="18sp" />
</LinearLayout>
```

16. layout/fragment_about_app.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".fragment.AboutAppFragment">
    <TextView
        android:id="@+id/txtAboutApp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/about_app_fragment"
        android:textSize="20sp"
        android:padding="10dp"/>
</RelativeLayout>
```

17. layout/ragment_dashboard.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".fragment.DashboardFragment">
    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerDashboard"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="5dp"/>
    <RelativeLayout
        android:id="@+id/progressLayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#fff">
        <ProgressBar
            android:id="@+id/progressBar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"/>
    </RelativeLayout>
</RelativeLayout>
```

18. layout/fragment_favourites.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".fragment.FavouritesFragment">

    <androidx.recyclerview.widget.RecyclerView
        android:id="@+id/recyclerFavourite"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="2dp"/>
    <RelativeLayout
        android:id="@+id/progressLayout"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="#ffffff">
        <ProgressBar
            android:id="@+id/progressBar"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_centerInParent="true"/>

    </RelativeLayout>
</RelativeLayout>
```

19. layout/fragment_profile.xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".fragment.ProfileFragment">
    <ImageView
        android:id="@+id/imgProfilePic"
        android:layout_width="180dp"
        android:layout_height="180dp"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="50dp"
        android:src="@drawable/profile_pic_android" />
    <TextView
        android:id="@+id/txtName"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/imgProfilePic"
        android:layout_marginTop="10dp"
        android:padding="10dp"
        android:text="John Doe"
        android:textAlignment="center"
        android:textColor="#111111"
        android:textSize="18sp"
        android:textStyle="bold" />
    <TextView
        android:id="@+id/txtEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txtName"
        android:layout_marginTop="10dp"
        android:padding="10dp"
        android:text="jonathan@anyone.com"
        android:textAlignment="center"
        android:textColor="#111111"
        android:textSize="18sp" />
    <TextView
        android:id="@+id/txtPhone"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txtEmail"
        android:layout_marginTop="10dp"
        android:padding="10dp"
        android:text="+11 - 11111111"
        android:textAlignment="center"
        android:textColor="#111111"
        android:textSize="18sp" />
    <TextView
        android:id="@+id/txtAddress"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txtPhone"
        android:layout_marginTop="10dp"
        android:padding="10dp"
        android:text="Manhattan, NY"
        android:textAlignment="center"
        android:textColor="#111111"
        android:textSize="18sp" />
</RelativeLayout>
```

20. layout/recycler_dashboard_single_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="140dp"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:background="#ffffff"
    android:layout_margin="6dp"
    app:cardCornerRadius="4dp">

    <LinearLayout
        android:id="@+id/llContent"
        android:layout_width="match_parent"
        android:layout_height="140dp"
        android:background="#ffffff"
        android:orientation="horizontal"
        android:weightSum="6">

        <!--The weight sum property is used to divide the layout into
        different parts and then giving each layout a particular weight
        gives it that amount of space on the screen-->
        <!--Since each parent layout will have a specific weight, we need
        not give any width to those layout-->
        <ImageView
            android:id="@+id/imgBookImage"
            android:layout_width="0dp"
            android:layout_height="120dp"
            android:layout_weight="1.5"
            android:padding="5dp"
            android:scaleType="centerCrop"
            android:src="@mipmap/ic_launcher" />
        <RelativeLayout
            android:layout_width="0dp"
            android:layout_height="match_parent"
            android:layout_weight="3.3">
            <TextView
                android:id="@+id/txtBookName"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:padding="8dp"
                android:text="Name of the book"
                android:textColor="#000000"
                android:textSize="18sp" />
            <TextView
                android:id="@+id/txtBookAuthor"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@id/txtBookName"
                android:padding="8dp"
                android:text="Name of the Author"
                android:textSize="15sp" />
            <TextView
                android:id="@+id/txtBookPrice"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                android:layout_below="@id/txtBookAuthor"
                android:layout_alignParentBottom="true"
                android:padding="8dp"
                android:text="Rs. 299"
                android:textColor="#357a38"
                android:textSize="15sp"
                android:textStyle="bold" />
        </RelativeLayout>
        <!--Many times we see that texts have an image along with them to
        their right or left. This is added with the help of the
        drawableLeft (or drawableRight and so on) attribute-->
        <TextView
            android:id="@+id/txtBookRating"
            android:layout_width="0dp"
            android:layout_height="wrap_content"
            android:layout_weight="1.2"
            android:drawableLeft="@drawable/ic_ratings"
            android:drawablePadding="4dp"
            android:padding="4dp"
            android:text="4.5"
            android:textColor="#ffca28"
            android:textSize="15sp"
            android:textStyle="bold">
        </TextView>
    </LinearLayout>
</androidx.cardview.widget.CardView>
```

21. layout/recycler_favourite_single_row.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="220dp"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_margin="4dp"
    android:background="#ffffff"
    app:cardCornerRadius="10dp"
    android:orientation="vertical">
    <LinearLayout
        android:id="@+id/llFavContent"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:weightSum="11">
        <ImageView
            android:id="@+id/imgFavBookImage"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_weight="5"
            android:background="#f5f5f5"
            android:scaleType="fitCenter"
            android:src="@drawable/default_book_cover" />
        <TextView
            android:id="@+id/txtFavBookTitle"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_margin="2dp"
            android:layout_weight="1.5"
            android:padding="4dp"
            android:text="Hello World"
            android:textAlignment="center"
            android:textAllCaps="false"
            android:textColor="#000000"
            android:textSize="16sp"
            android:textStyle="bold" />
        <View
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:layout_marginLeft="4dp"
            android:layout_marginRight="4dp"
            android:background="#f5f5f5" />
        <TextView
            android:id="@+id/txtFavBookAuthor"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_margin="2dp"
            android:layout_weight="1.5"
            android:padding="4dp"
            android:text="Android Inc."
            android:textAlignment="center"
            android:textSize="13sp" />
        <View
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:layout_marginLeft="4dp"
            android:layout_marginRight="4dp"
            android:background="#f5f5f5" />
        <TextView
            android:id="@+id/txtFavBookPrice"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            android:layout_margin="2dp"
            android:layout_weight="1.5"
            android:padding="4dp"
            android:text="Rs. 499"
            android:textAlignment="center"
            android:textColor="#357a38"
            android:textSize="13sp"
            android:textStyle="bold" />

        <View
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:layout_marginLeft="4dp"
            android:layout_marginRight="4dp"
            android:background="#f5f5f5" />
        <TextView
            android:id="@+id/txtFavBookRating"
            android:layout_width="wrap_content"
            android:layout_height="0dp"
            android:layout_gravity="center"
            android:layout_margin="2dp"
            android:layout_weight="1.5"
            android:drawableLeft="@drawable/ic_ratings"
            android:drawablePadding="4dp"
            android:padding="4dp"
            android:text="4.5"
            android:textAlignment="center"
            android:textColor="#ffca28"
            android:textSize="13sp"
            android:textStyle="bold" />
        <View
            android:layout_width="match_parent"
            android:layout_height="1dp"
            android:layout_marginLeft="4dp"
            android:layout_marginRight="4dp"
            android:background="#f5f5f5" />

    </LinearLayout>
</androidx.cardview.widget.CardView>
```

22. layout/menu/menu_dashboard.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

    <item
        android:id="@+id/action_sort"
        android:icon="@drawable/ic_action_sort"
        android:title="@string/action_sort"
        android:orderInCategory="100"
        app:showAsAction="always"/>
</menu>
```

23. layout/menu/menu_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <group android:checkableBehavior="single">
        <item
            android:id="@+id/dashboard"
            android:icon="@drawable/ic_dashboard"
            android:title="@string/dashboard" />
        <item
            android:id="@+id/favourites"
            android:icon="@drawable/ic_favourites"
            android:title="@string/favourites" />
        <item
            android:id="@+id/profile"
            android:icon="@drawable/ic_profile"
            android:title="@string/profile" />
        <item
            android:id="@+id/aboutApp"
            android:icon="@drawable/ic_about_app"
            android:title="@string/about_app" />
    </group>
</menu>
```

24. layout/menu/values/book_app_icon_background.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="book_app_icon_background">#ABBCE0</color>
</resources>
```

25. layout/menu/values/colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#EE6C00</color>
    <color name="colorPrimaryDark">#C23E10</color>
    <color name="colorAccent">#D81B60</color>
    <color name="colorFavourite">#311b92</color>
</resources>
```

26. layout/menu/values/strings.xml

```
<resources>
    <string name="app_name">BookHub</string>
    <string name="dashboard">Dashboard</string>
    <string name="favourites">Favourites</string>
    <string name="profile">Profile</string>
    <string name="about_app">About App</string>
    <string name="navigation_drawer">Navigation Drawer</string>
    <string name="open_drawer">Open Drawer</string>
    <string name="close_drawer">Close Drawer</string>

    <!-- TODO: Remove or change this placeholder text -->
    <string name="hello_blank_fragment">Hello blank fragment</string>
    <string name="dashboard_fragment">Dashboard Fragment</string>
    <string name="favourites_fragment">Favourites Fragment</string>
    <string name="profile_fragment">Profile Fragment</string>
    <string name="about_app_fragment">About App Fragment</string>
    <string name="add_to_favourites">Add to Favourites</string>
    <string name="action_sort">Sort</string>
</resources>
```

27. layout/menu/values/styles.xml

```
<resources>
    <!-- Base application theme. -->
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
        <!-- Customize your theme here. -->
        <item name="colorPrimary">@color/colorPrimary</item>
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
        <item name="colorAccent">@color/colorAccent</item>
    </style>
</resources>
```

28. layout/menu/values/xml/etwork_security_config.xm

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config cleartextTrafficPermitted="true">
        <domain includeSubdomains="true">13.235.250.119</domain>
    </domain-config>
</network-security-config>
```