**Name:** Muhammad Mehdi Ali **Roll No:** BSEF18M015

**GitHub Tutorial:-**

**Steps:-**

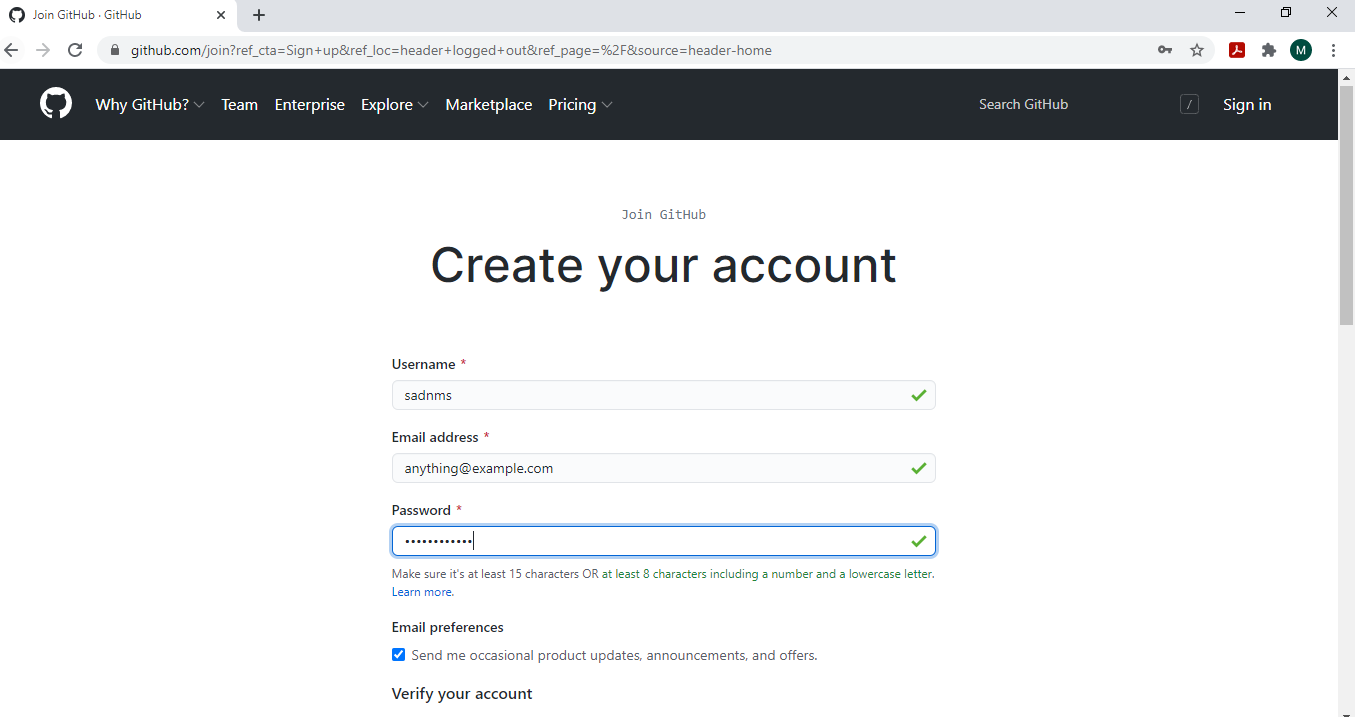
**Getting Started:-**

1. Go to github.com

A screenshot of a computer screen

Description generated with high confidence

1. Click on sign up and make an account



1. Then Sign in to your account.

Graphical user interface, application

Description generated with very high confidence

1. Your account will be shown

Graphical user interface, application

Description generated with very high confidence

1. Click on + button to add a repo

Graphical user interface, application

Description generated with very high confidence

**Making a repository:-**

1. Click on New Repository

Graphical user interface, application

Description generated with very high confidence

1. Give details of the repository

Graphical user interface, text, application, email

Description generated with very high confidence

1. Click on Create Repository
2. Copy The link of your repository we will use it later

Graphical user interface, text, application, email

Description generated with very high confidence

1. Create a directory and open git bash in it

A picture containing text

Description generated with very high confidence

**Manage repository locally and git add, commit and push commands:-**

1. Clone repository locally using git clone command and url copied at step 9

Text

Description generated with very high confidence

12)Create a file and add data to it

Graphical user interface, text, application

Description generated with high confidence

1. Now push this txt file to main repo using git add, commit and push commands

Text

Description generated with very high confidence

1. Changes will be applied at main repository at github.com too.

A screenshot of a computer

Description generated with high confidence

**Git Pull:-**

1. Edit the file locally

Graphical user interface, text, application

Description generated with high confidence

1. Now update the main repository by using git add, commit and push commands

Text

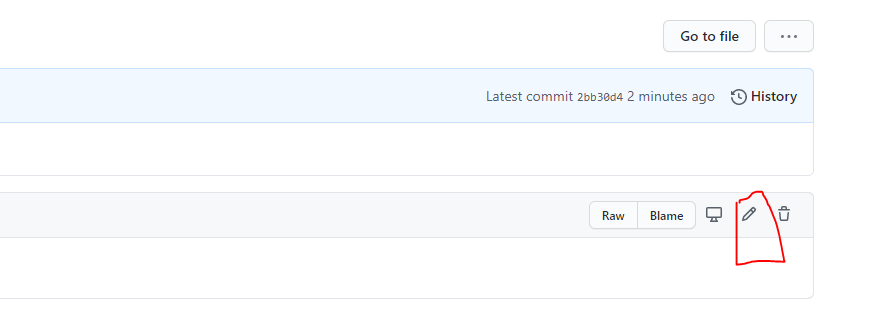
Description generated with very high confidence

1. Main Repository will be updated

A screenshot of a computer

Description generated with high confidence

1. Now to edit from online github editor click on edit button at top right

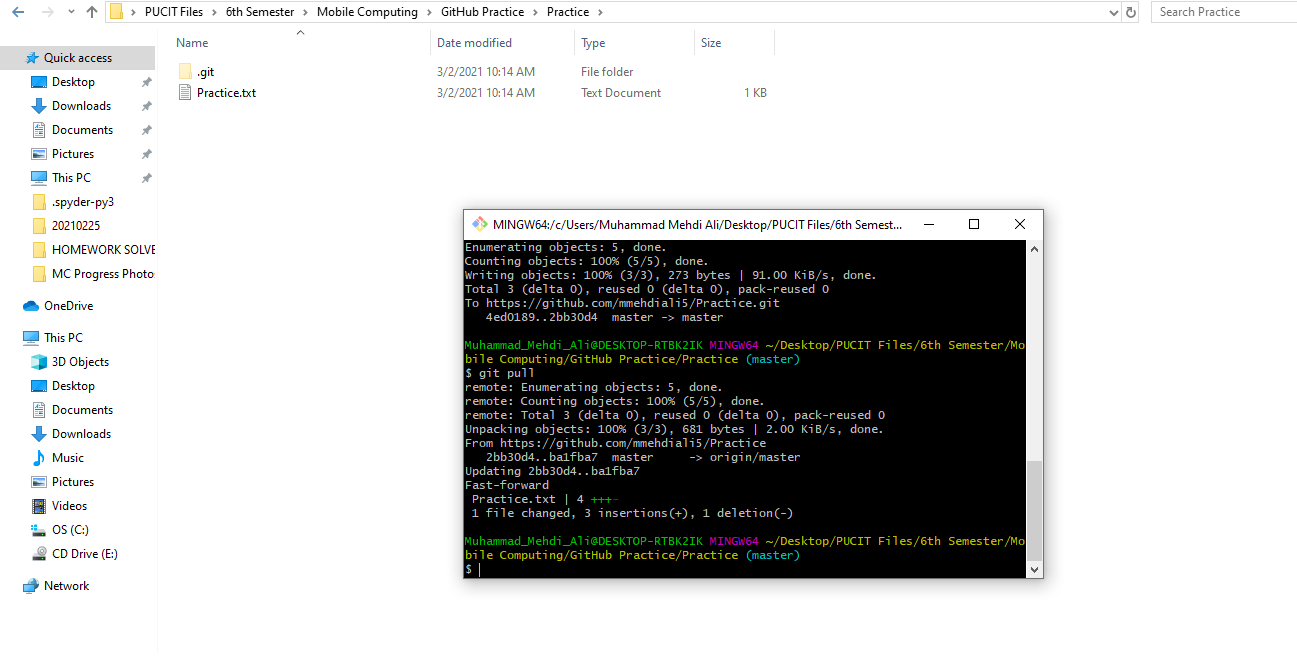


18)Edit the document there

Graphical user interface, text, application, email

Description generated with very high confidence

19)Now to update local repo we will use git pull command



20) Local repo will be updated

Graphical user interface, text, application

Description generated with high confidence

**MERGE CONFLICTS:-**

Suppose that two people change same line of the file one person change it locally and other on main branch. In this case if we want to push changes to main repository we will get error of merge conflicts which must be resolved before pushing

1. Open the file in text Editor and change a line in it

Text

Description generated with very high confidence

1. Now change the line at main repo and commit

Graphical user interface, text, application, email

Description generated with high confidence

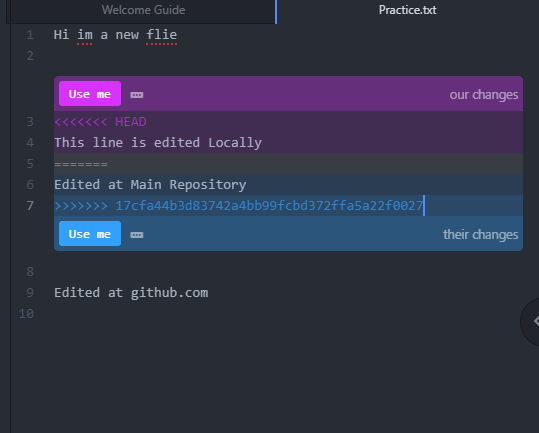
1. Now when we try to push from local repository we get an conflict error message



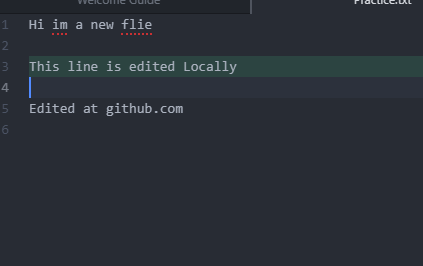
1. To resolve this use git pull to pull the file from main repository



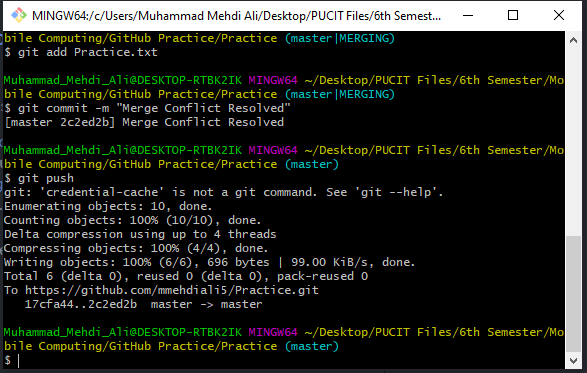
1. When you open the file you will see buttons use me click on the line which you want to retain



1. Choose one of the lines



1. Now we can push successfully and merge conflict is resolved



1. Main repo will also be updated

Graphical user interface, application, Word

Description generated with very high confidence

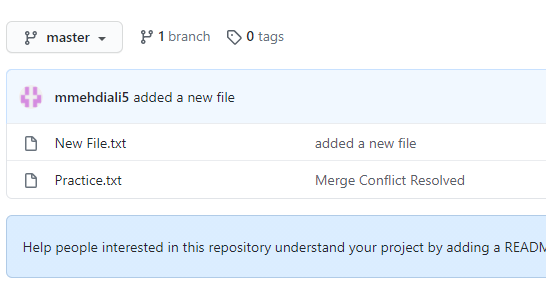
**Git Remove**

1. Let us add a file in our repository

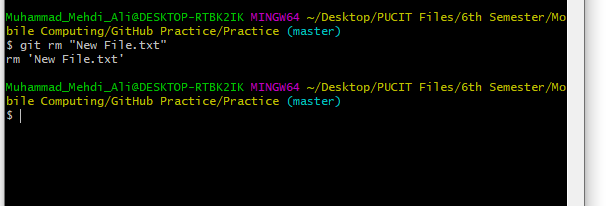
Graphical user interface, text, application

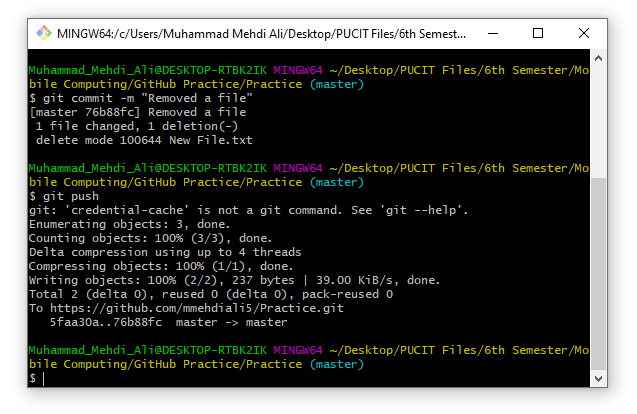
Description generated with very high confidence

1. Now we check it on our main repository

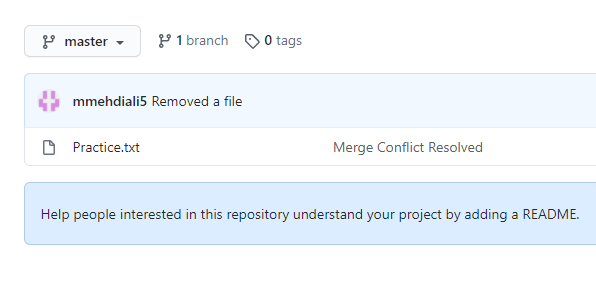


1. Now we remove it using git remove and commit changes and push to main repo





1. Now, we verify it on our main repo



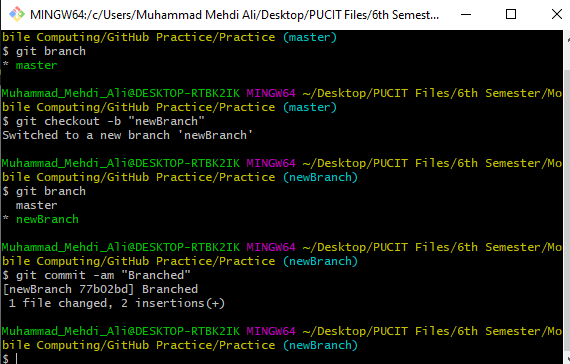
**Git Branch**

1. Let us create a branch in our project so that we can make changes in it separately and marge when required. We use git checkout to navigate among branches. Now we add some text to it.

Graphical user interface, application, Word

Description generated with very high confidence

1. Now we commit it.



1. When we checkout to master we can see that master branch is at was before branching

Graphical user interface, application

Description generated with very high confidence

1. Now we merge it with master and delete the branch. We can see that now master have branch text also

Graphical user interface, application

Description generated with very high confidence

**ANDROID STUDIO**

**First Program:-**

1. First we open our android studio and start a new project.
2. Then we select empty project option
3. After loading we go to our activity\_main.xml
4. We add a text “Application Submission” and adjust the margins

Graphical user interface, application

Description generated with very high confidence

1. Now we add some input text of Name, Address and Email

Graphical user interface, application

Description generated with very high confidence

1. Now we add some checkboxes and submit button

Graphical user interface, application

Description generated with very high confidence

**Layouts:-**

1. In this layout elements are sequentially inserted

Graphical user interface, application

Description generated with very high confidence

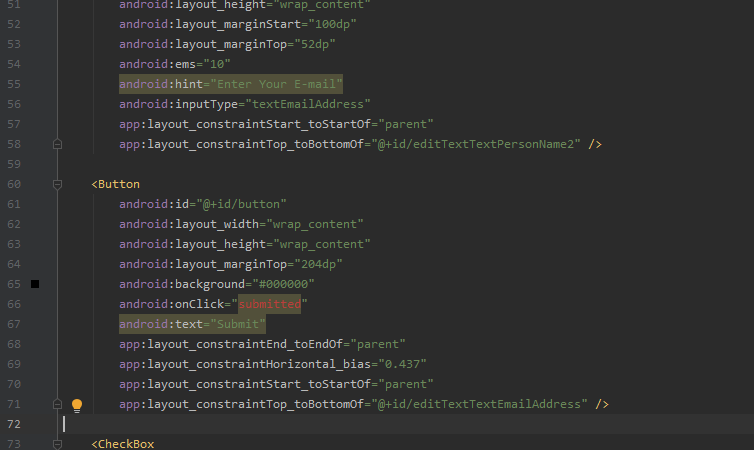
A picture containing text

Description generated with very high confidence

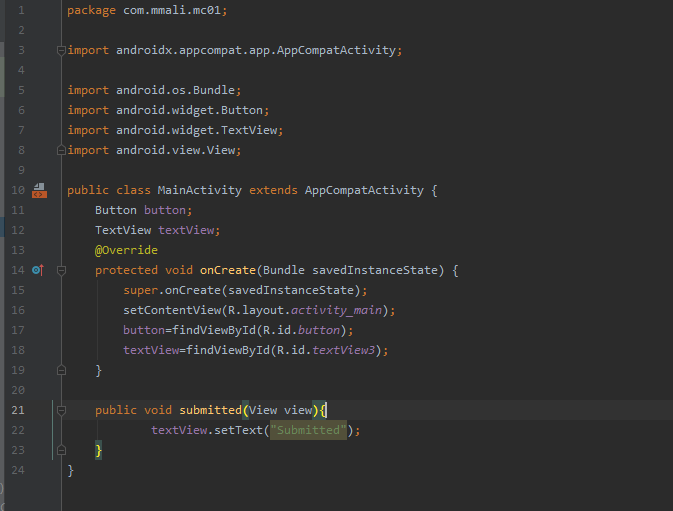


**OnClick:-**

1. Now we will assign an onclick function to our submit function
2. Go to code and add onClick in button tag and write name of the function



1. Now to remove the error we will define the function in main



1. Output on android device

Graphical user interface, application

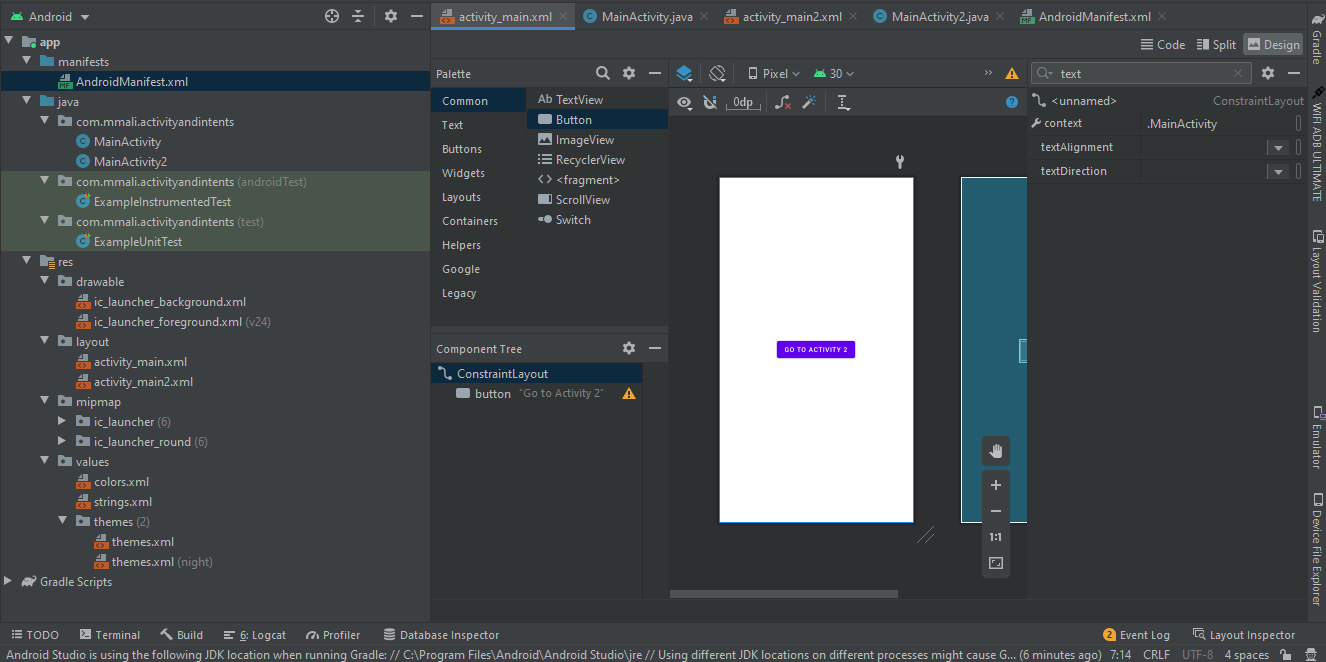
Description generated with high confidence

Graphical user interface, application

Description generated with high confidence

**Activiity and Intend**

1. We first create a main activity with a button in it having text Go to avtivity 2.



1. Now we create an empty activity

Graphical user interface, application

Description generated with very high confidence

1. Now we add 2 buttons here and give them onClick functions

Text

Description generated with very high confidenceGraphical user interface, application

Description generated with very high confidence

1. Now we apply onClick to go to activity 2 from main activity

Text

Description generated with very high confidence

1. Now we will see the output on android phone

Graphical user interface, application, Teams

Description generated with very high confidenceGraphical user interface, application, Teams

Description generated with very high confidence

Graphical user interface, application

Description generated with very high confidence

**ACTIVITY LIFECYCLE**

1. There are three activities performed in application:-

* OnCreate()
* OnStart()
* OnResume()
* OnPause()
* OnStop()
* OnDestroy()

1. When an activity destroys it’s data is also destroyed.
2. For example let us create a counter which keeps count of clicks on button

Text

Description generated with very high confidence

Graphical user interface, application

Description generated with high confidence

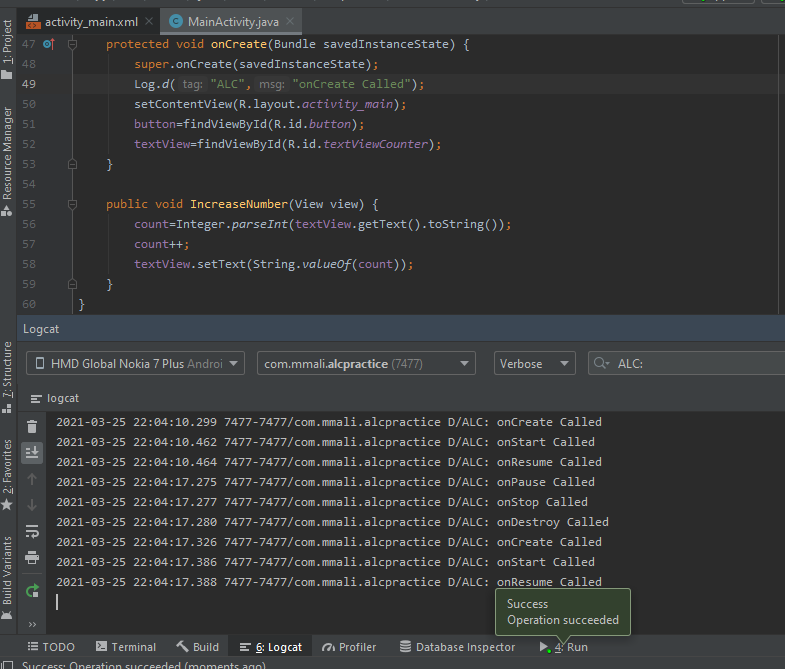
4)Now when we try to increase the count and rotate it we see that it resets to zero its because new activity is started and previous activity is destroyed.

Graphical user interface, application

Description generated with high confidenceGraphical user interface, text, application

Description generated with high confidence

5)Now we log the activities to see what is happening



6)Now to solve this issue we will save the previous state of activity

Text

Description generated with very high confidence

1. Now if we increase count and rotate screen the previous activity state is preserved and count is same as before

Graphical user interface, text, application

Description generated with high confidence

Graphical user interface, text, application

Description generated with high confidence

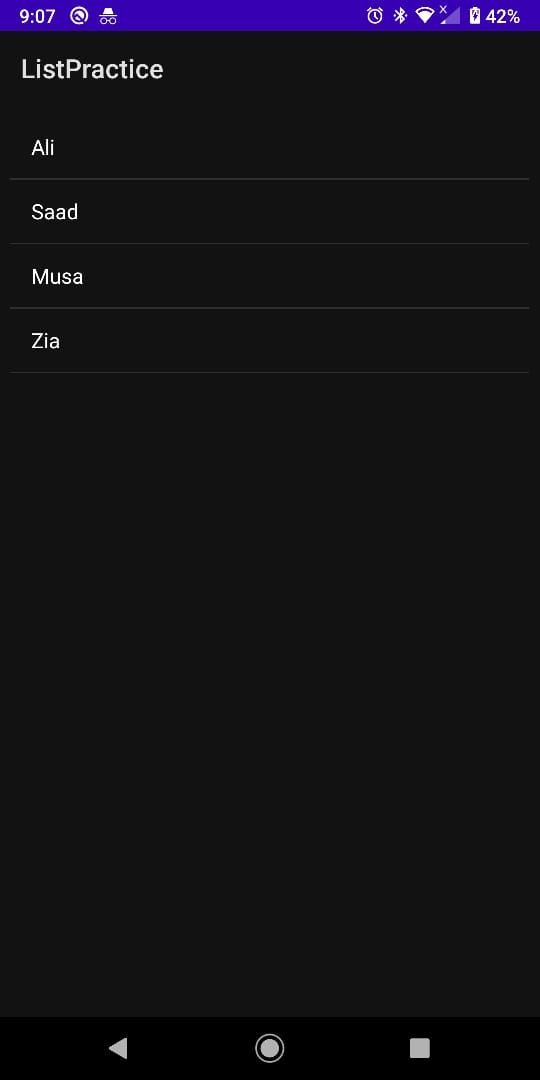
**ADAPTERS**

1)Adapters convert data from a data source in UI Components eg. ListView, GridView etc

2)Let Us make a code for a listView.

Text

Description generated with very high confidence



3)Now we do this procedure for ArrayList

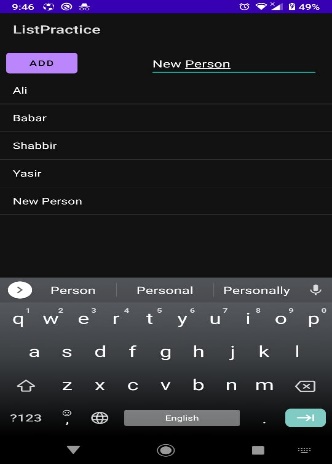
Text

Description generated with very high confidenceText

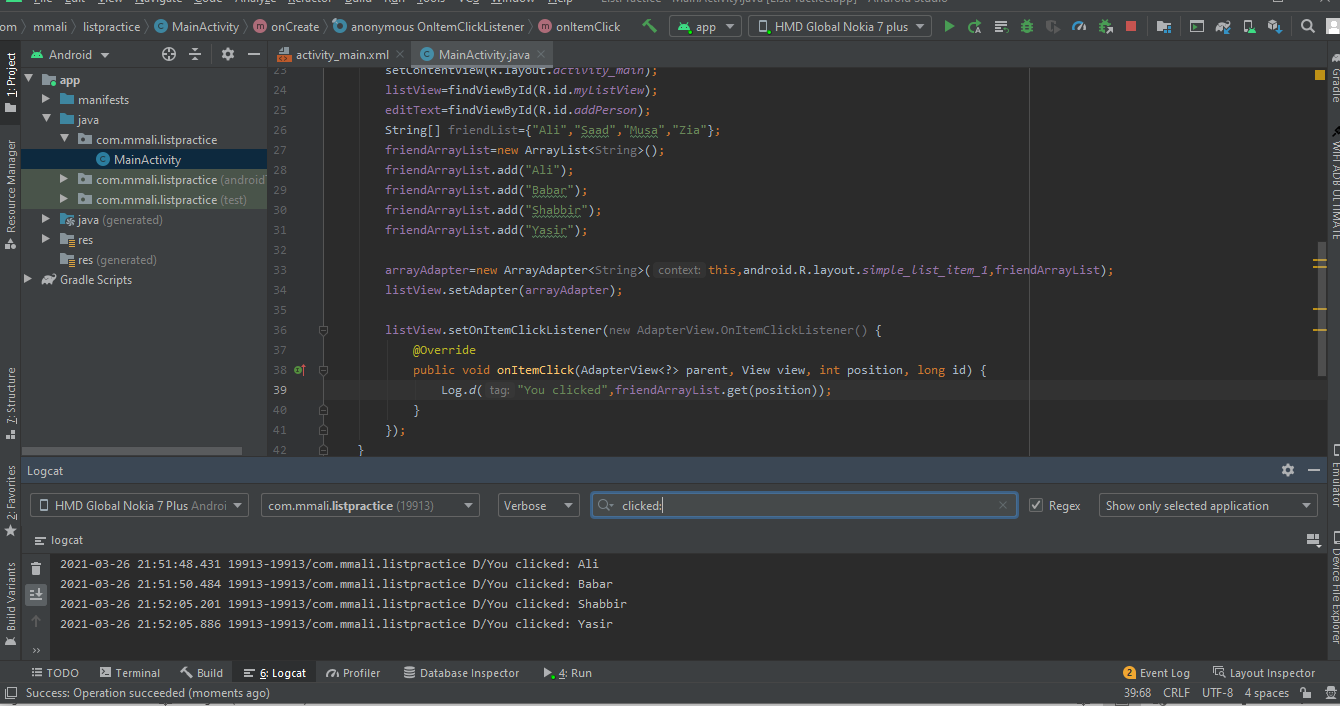
Description generated with high confidence

4)Now we add record taking input from user

Text

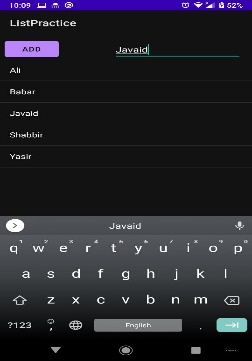
Description generated with very high confidence

5)Now we try to add some clickEventListner to list Items

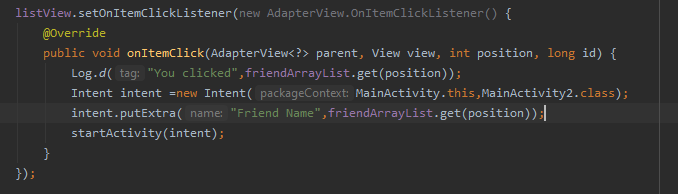


6)Now we update list using notifyDataSetChanged so that whenever our list is changed the listview refresh itself

Text

Description generated with very high confidence

7)Now let’s transfer list Item on click to other activity

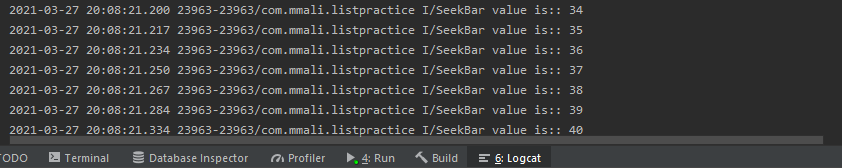
A picture containing shape

Description generated with very high confidence

8)Now let us add a seekBar in it and log to see when we change it’s value

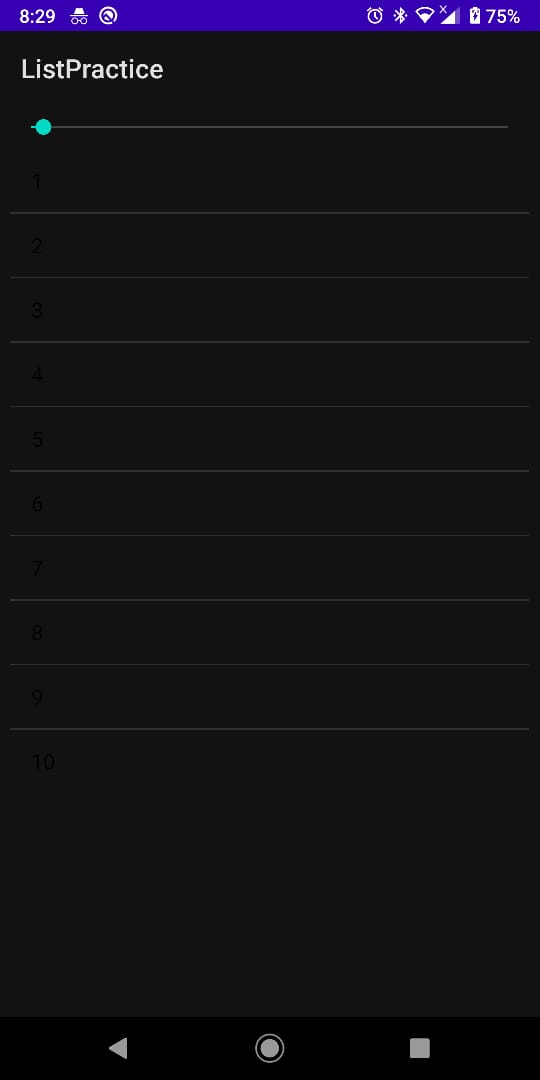
Text

Description generated with very high confidence



9)Now Let us add a list and display table of the current index of seekBar

Text

Description generated with very high confidenceGraphical user interface, application

Description generated with high confidence

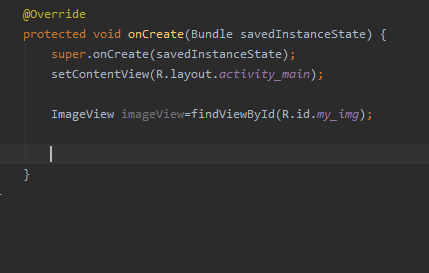
**ANIMATIONS**

1. We add image to activity using ImageView.

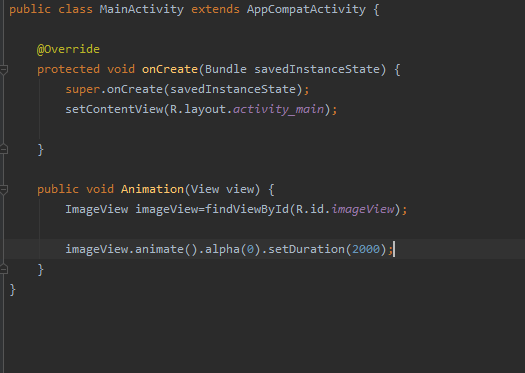
Graphical user interface, application

Description generated with very high confidence

1. Now we get Image in our code and apply animations to it



1. Now we add a fading animation to it

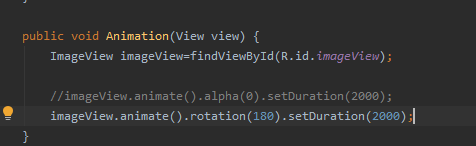


A screenshot of a red car

Description generated with high confidenceA screenshot of a car

Description generated with high confidence

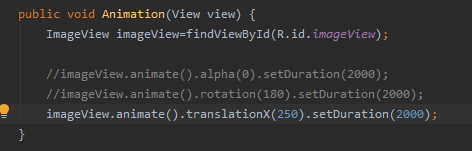
1. Now let’s add rotation animation

A screenshot of a red car

Description generated with high confidenceA picture containing text, screenshot

Description generated with very high confidence

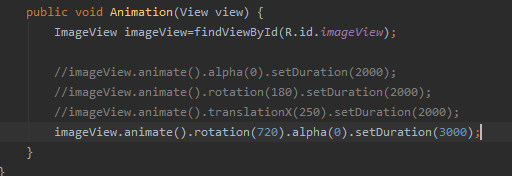
1. Now let’s translate the image on axis



A screenshot of a red car

Description generated with high confidence

1. Now let’s rotate and fade at same time

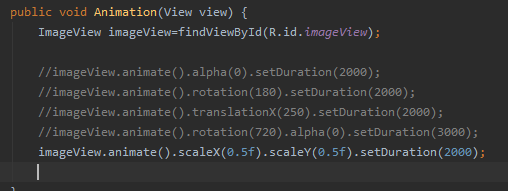


A screenshot of a red car

Description generated with high confidenceA picture containing text

Description generated with very high confidence

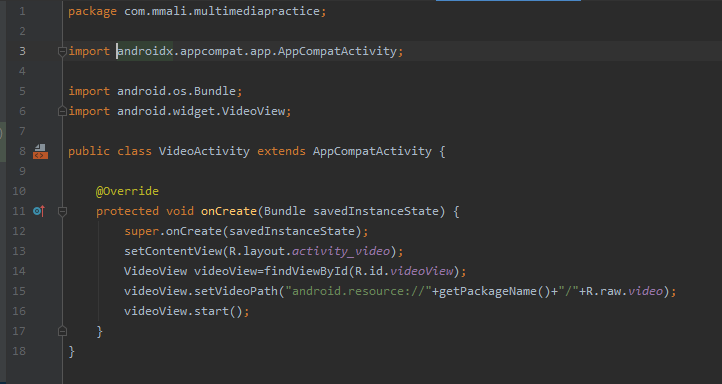
1. Now we add scaling animation



A screenshot of a red car

Description generated with high confidenceA picture containing graphical user interface

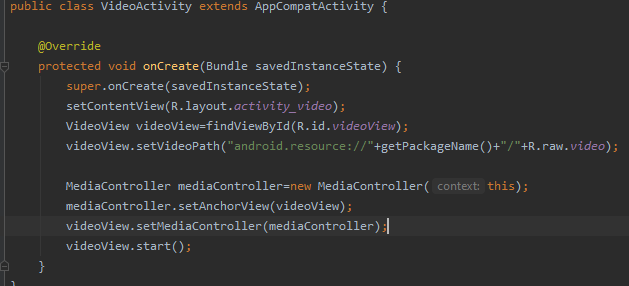
Description generated with very high confidence

1. Now we add video activity as launcher activity and add videoView in it
2. 

Graphical user interface, application

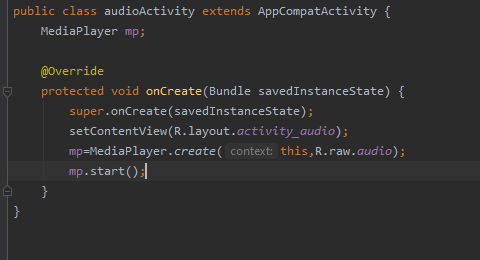
Description generated with very high confidence

1. Now we add media controller to it.

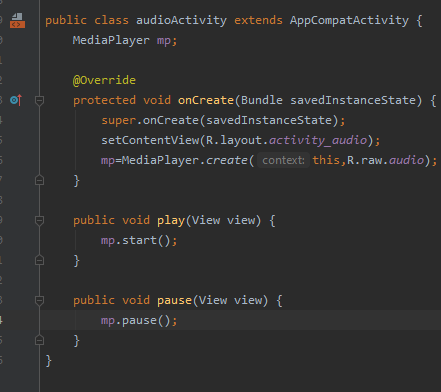
A picture containing text

Description generated with very high confidence

1. Now we add an audio activity and make it as launcher activity



1. Now we add some controller buttons to play and pause the sound

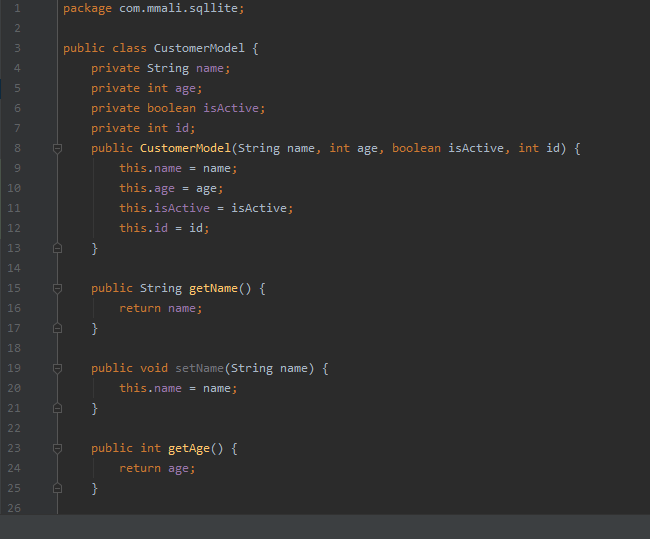
Graphical user interface, application, Teams

Description generated with very high confidence

1. Sound will be played on clicking play button and stopped when click on stop button

**SQLite**

1. SQLite is a file based system which is used as a database system for android development. It provides all RDBMS functionality and uses Structured Query Language.
2. We add a Customer Model class

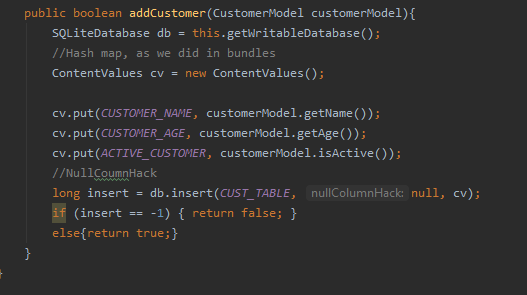


1. Now we add a DBHelper class which inherits SQLDatabaseHelper class to get write database

Text

Description generated with very high confidence

1. Now we add code for adding the customer record



1. Now we add buttons to view and insert record

Text

Description generated with very high confidence

Graphical user interface, application, Teams

Description generated with very high confidence

1. Now we add some data

Graphical user interface, application, Teams

Description generated with very high confidence

1. Now we view name of added data through view button

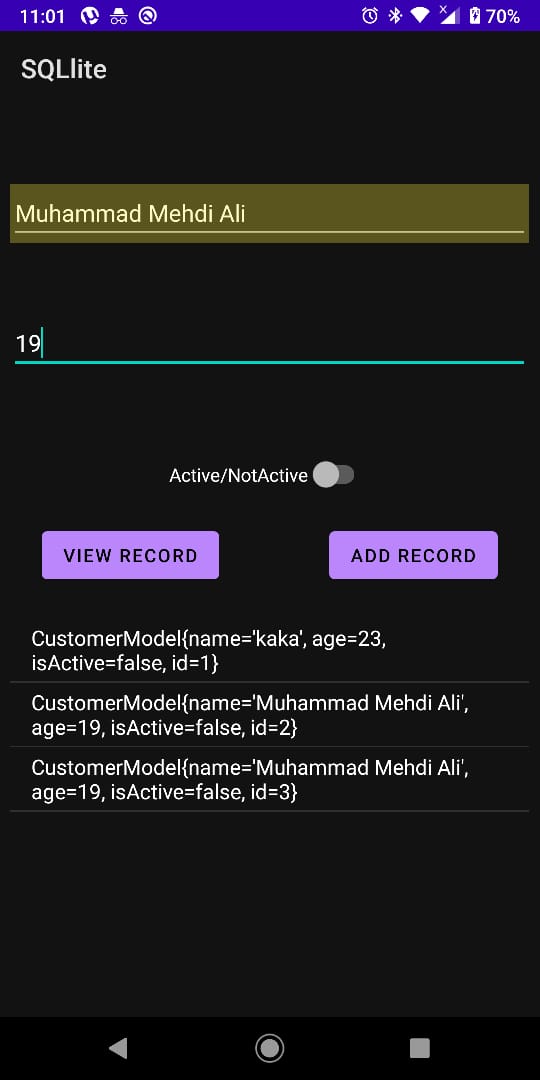
Graphical user interface, application, Teams

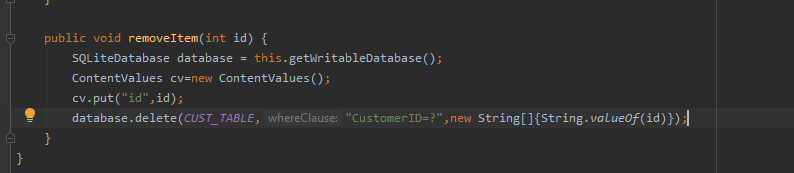
Description generated with very high confidence

1. Now we list the data in list view

Text

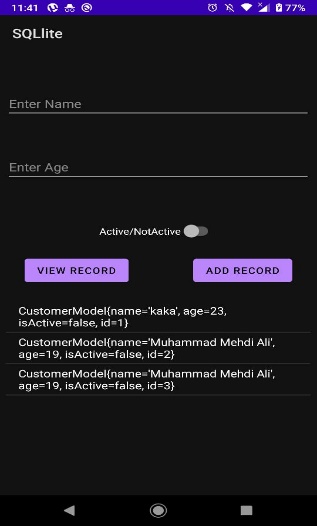
Description generated with very high confidenceText

Description generated with very high confidence

1. Now let’s add a onClick Function on ListItems to delete the clicked Item from database and the list. Text

   Description generated with very high confidence

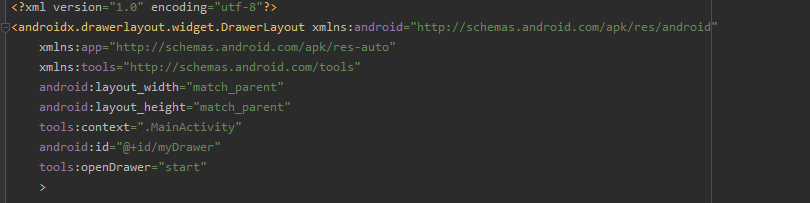
Graphical user interface, application

Description generated with very high confidence

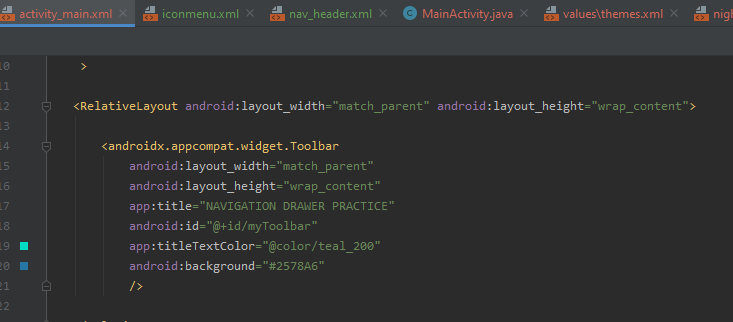
1. Here we removed last 3 entries of list by clicking on them

**NAVIGATION DRAWER**

1. Navigation drawer is used to navigate between different activities. It provides us a menu from which we can select given activities
2. Change Main Activity’s layout to drawer layout

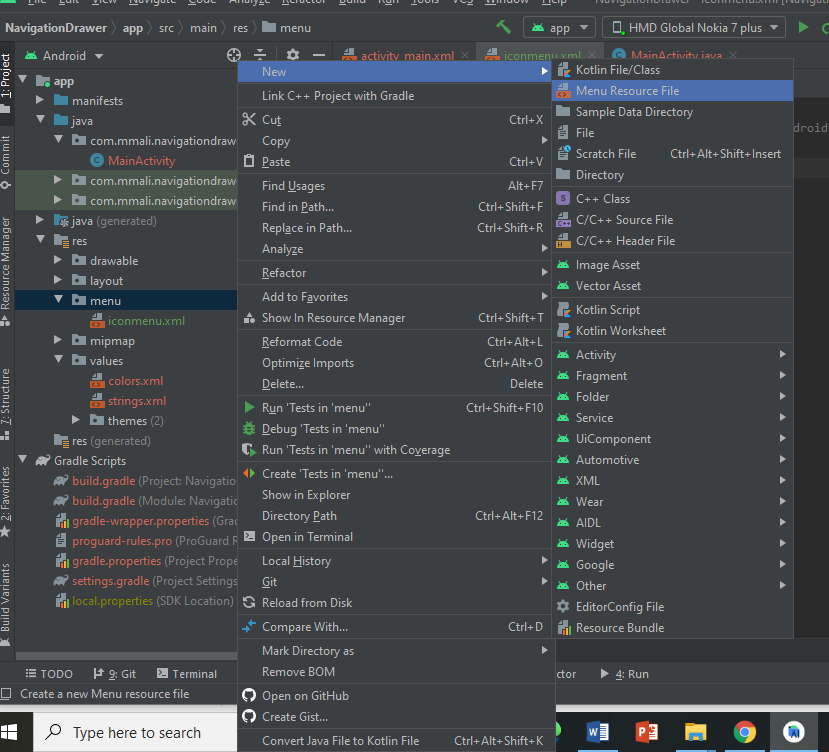


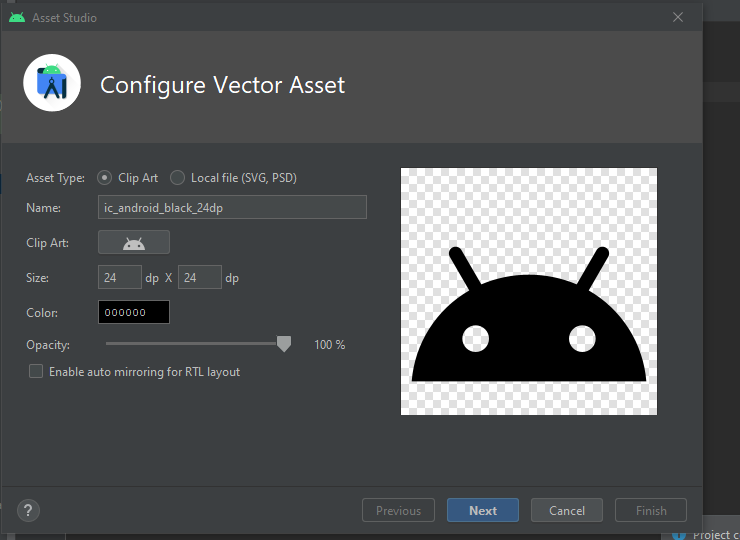
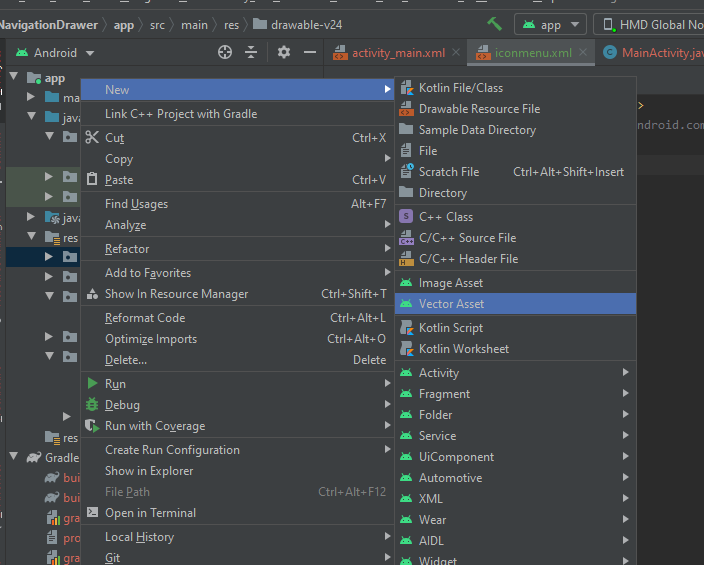
1. Now add a toolbar to main activity



1. Add a menu resource directory and add menu resource file in itGraphical user interface, application

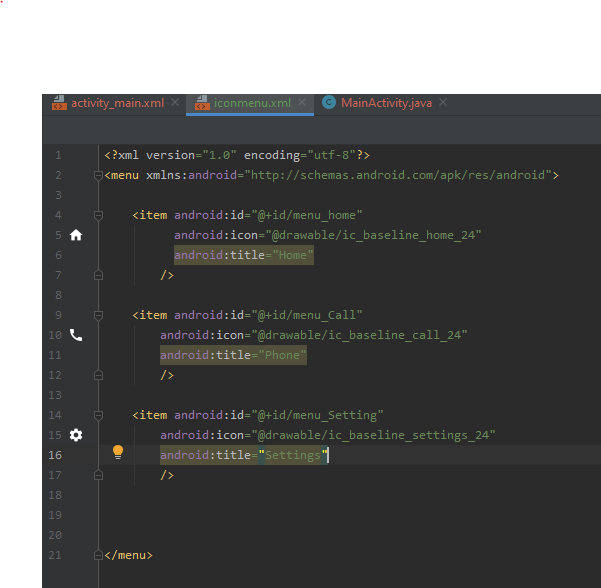
   Description generated with very high confidenceGraphical user interface, application

   Description generated with very high confidence
2. Add icons for menu items using vector asset and select clip arts as you like home, call and settings etc





1. Now add item in menu xml

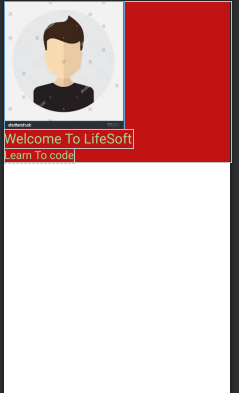
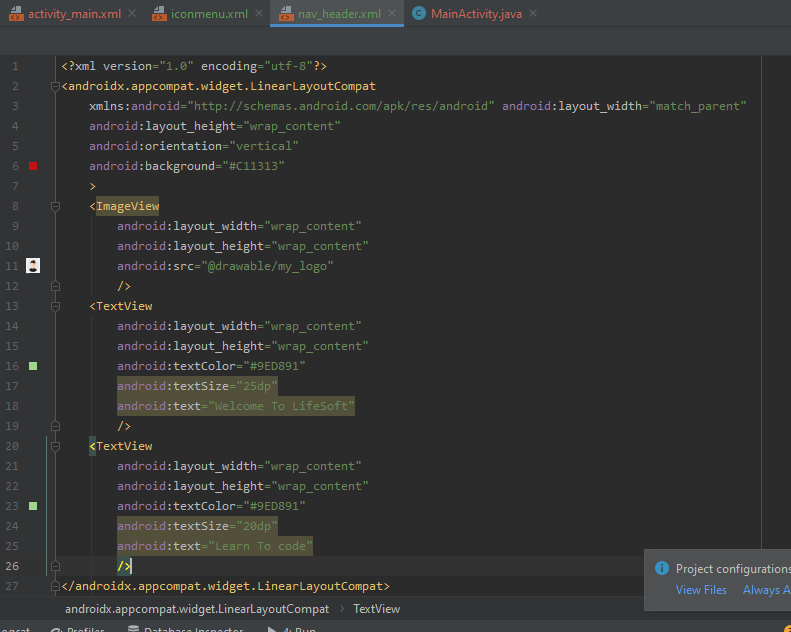


1. Now add a header xml file in layout

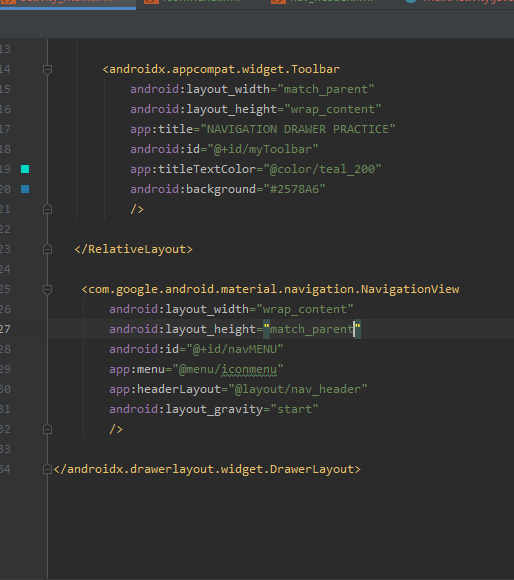
Graphical user interface, application

Description generated with very high confidence

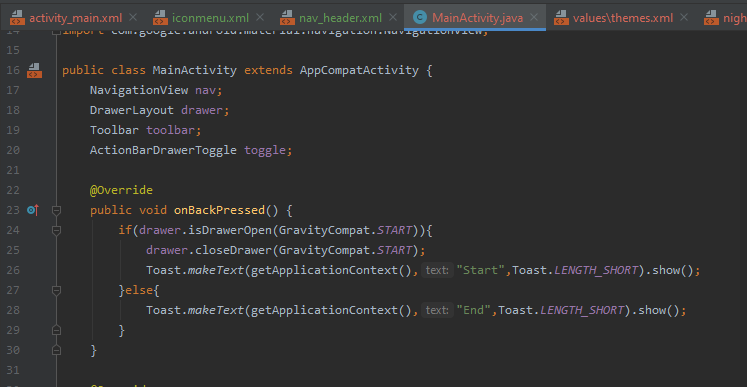
1. Now we design the layout using linear layout



1. Now we add navigation View in the main activity



1. Now we add code in main to handle the navigation drawer functionality

Text

Description generated with very high confidence

1. Now we check the output of navigation drawer on android device



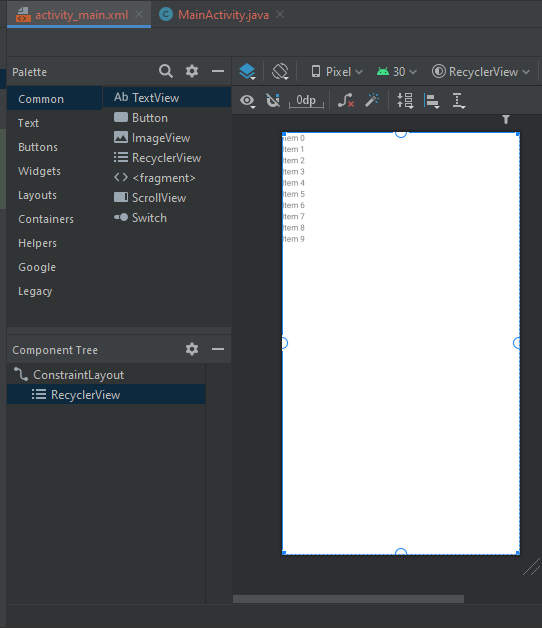
Text

Description generated with high confidenceText

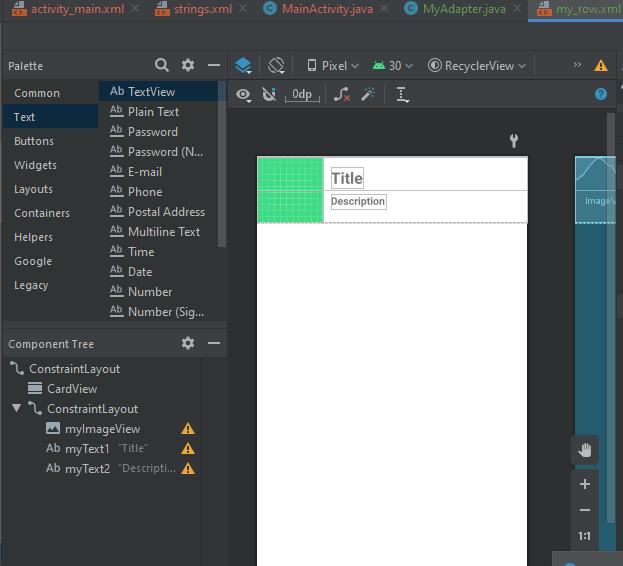
Description generated with high confidence

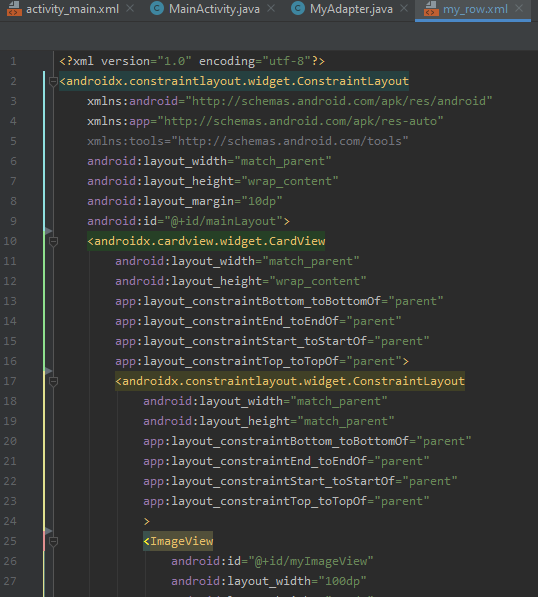
**RECYCLER VIEW**

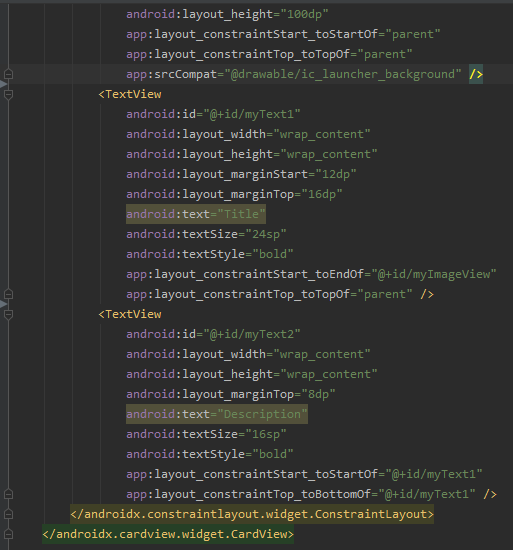
1. RecyclerView makes it easy to efficiently display large sets of data. You supply the data and define how each item looks, and the RecyclerView library dynamically creates the elements when they're needed
2. To implement recycler view we first add a recycler view in our activity



1. Now we define a layout for our list item of recycler view by making a layout in layout folder named my\_row.xml. Here we defined a card and inside it we placed a constraint layout having an image and two textViews.

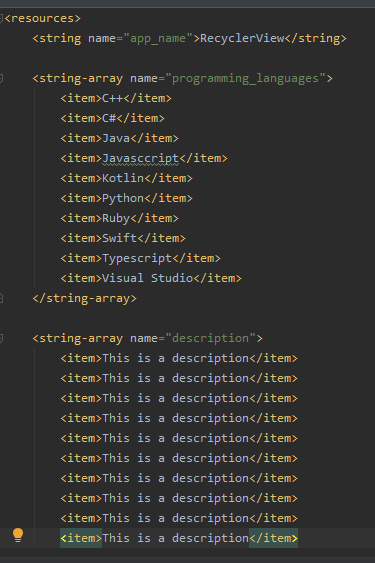




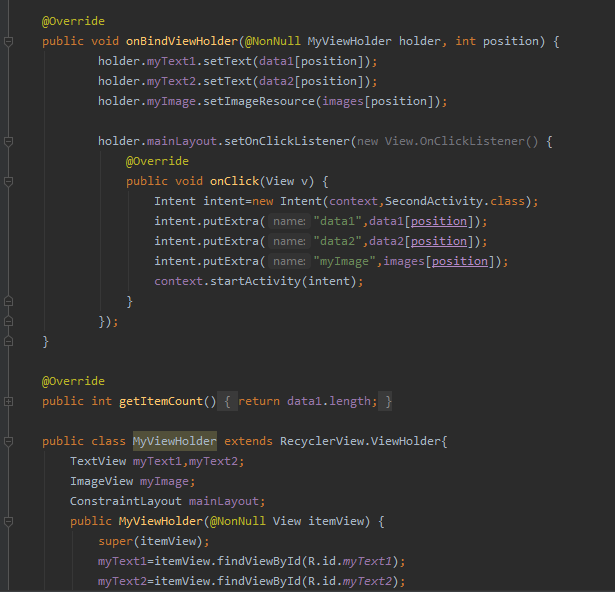
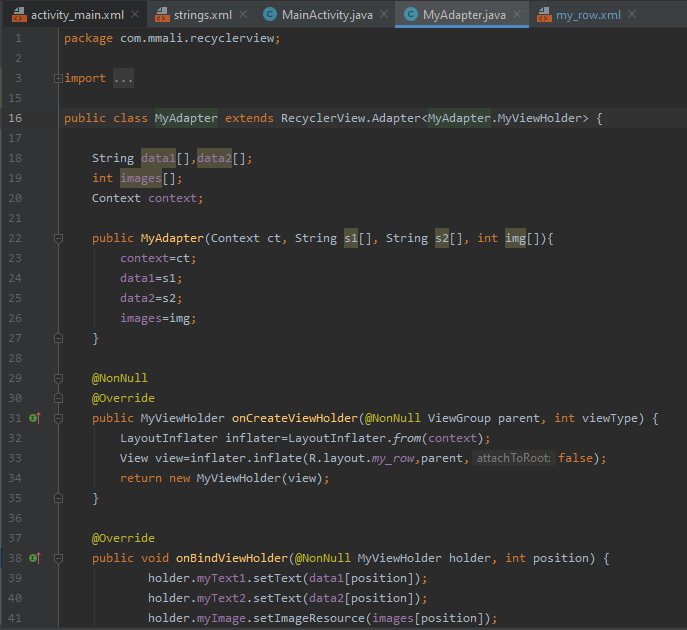
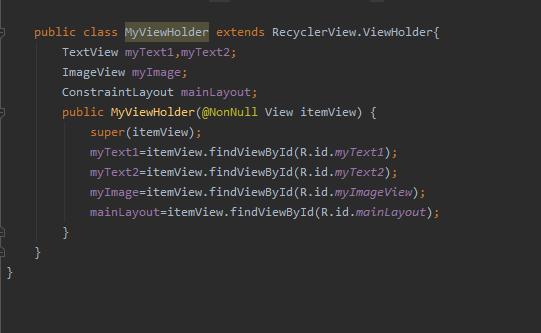


1. Now we add some image and text resources

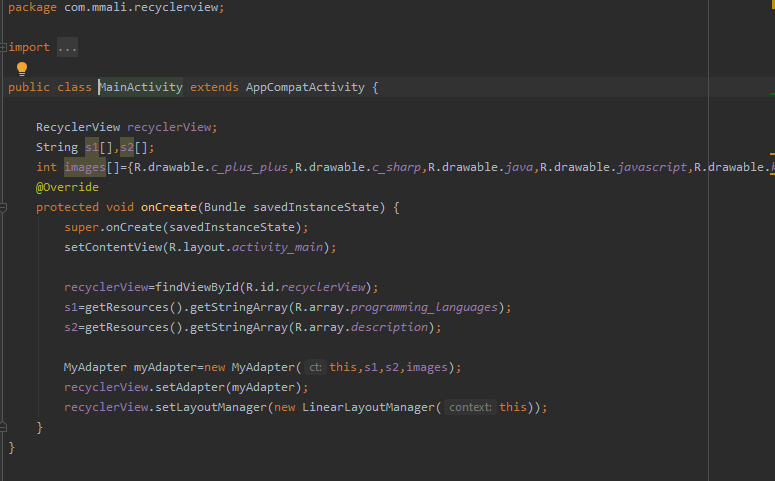
Text

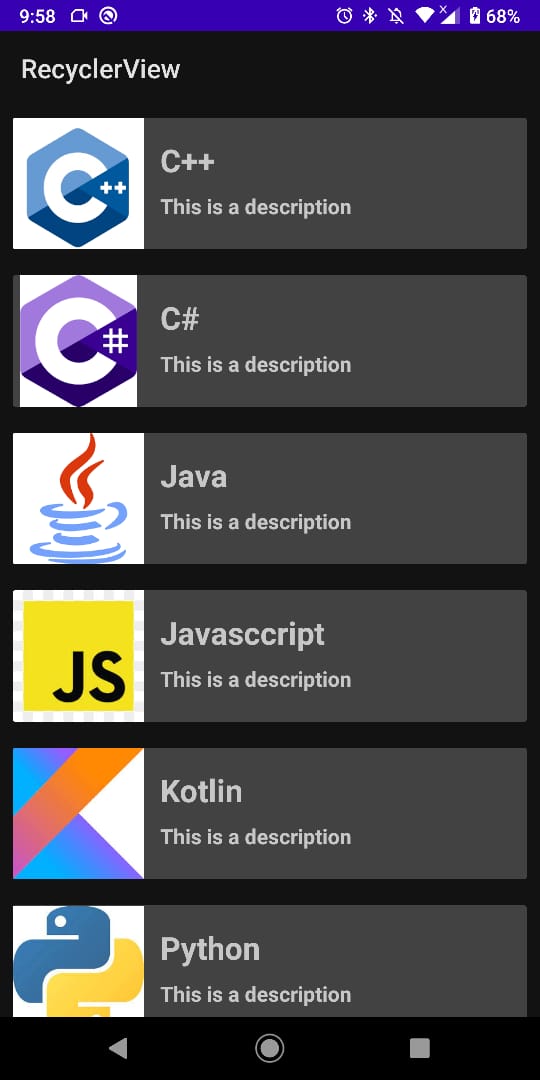
Description generated with very high confidence

1. Now we make a class of Adapter for recycler view functions. We define onCreate, onBind functions in it and also make a view holder class

1. Now we set the adapter and layout manager for recycler view in our main activity





**Dialogs**

1. A dialog is a small window that prompts the user to make decision or enter additional information.
2. On button click we add a dialog having negative and positive buttons

Text

Description generated with very high confidenceText

Description generated with very high confidence

Graphical user interface, application

Description generated with very high confidenceGraphical user interface, application, Teams

Description generated with very high confidence

1. We can also define custom dialogs
2. First we have to create a layout for our dialog box.



1. Now we add the view in dialog using inflater



1. We get following custom layout on our dialog

