**Mobile Computing**

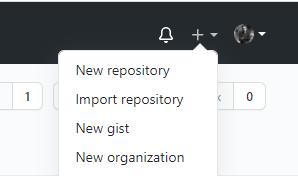


**Submitted to: Sir Haq Nawaz**

**Submitted by: Hafiz Muhammad Ali**

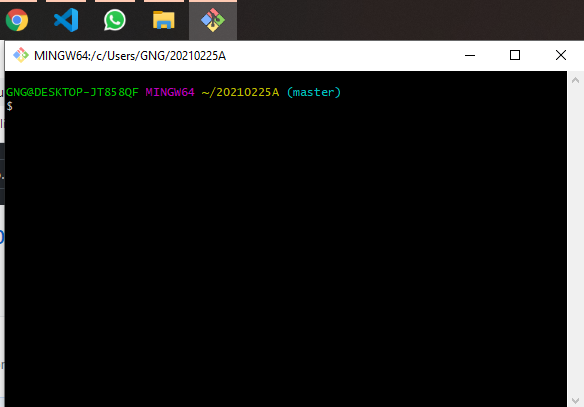
**Roll no: BSEF18A002**

After creating my account on github I have created a new repository on github with this name > “20210225A”





Downloaded git bash

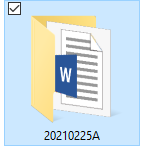


And then cloned my repository from central repository to local repository by using this command

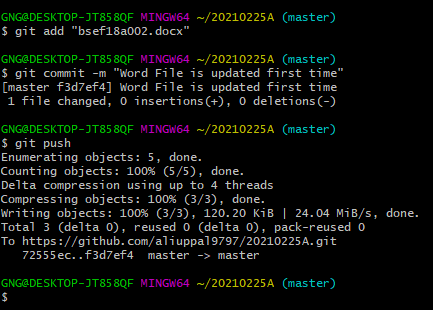
git clone [url]



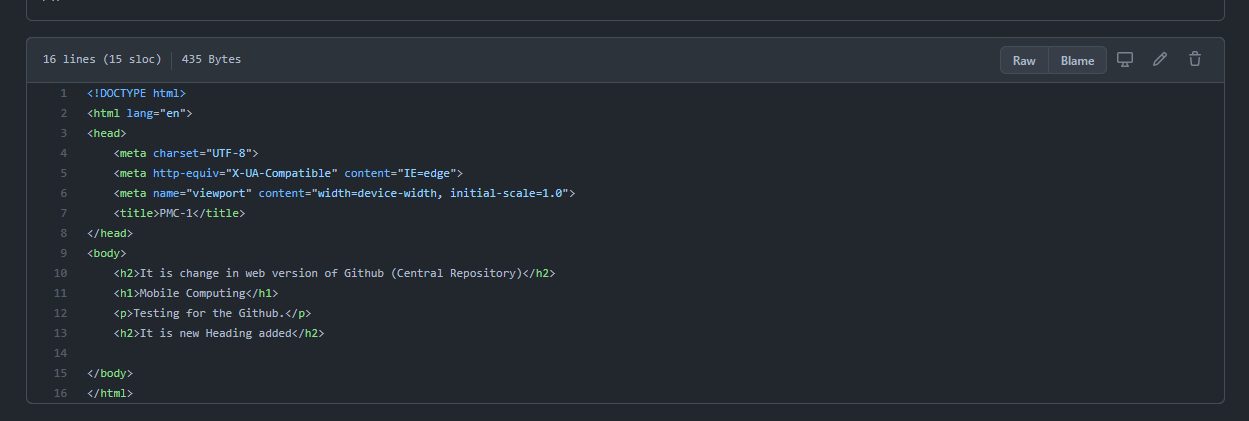
Here is our local repository which is linked with the central repository and in that repository I have created a docx file in it.



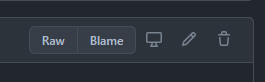
Then I moved to local repository and added some content in docx file and then updated this file on central repository.



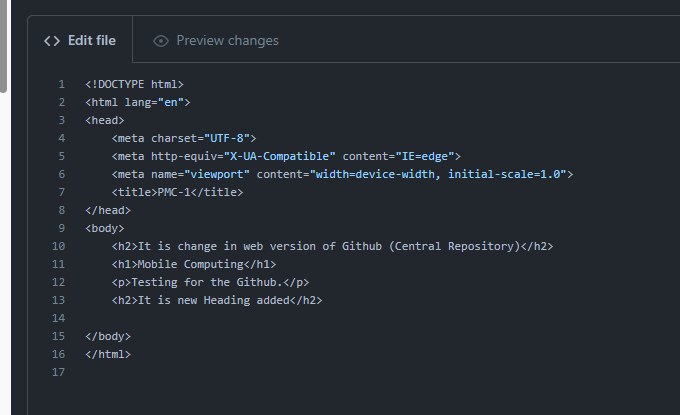
On central repository we open our file and make some changes in it.

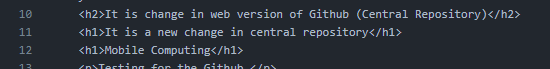


Here we get the edit option.

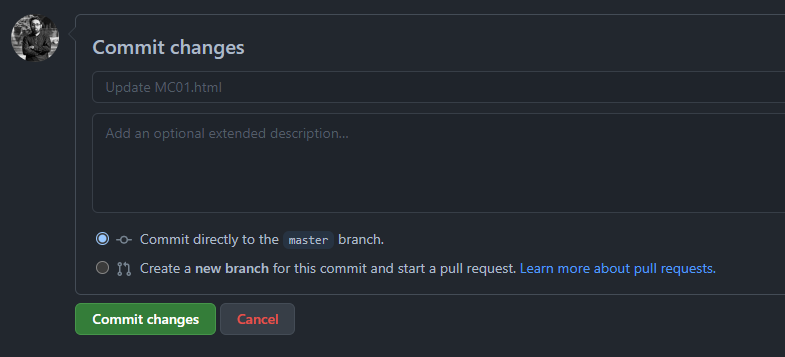


Then on central repository editor will be opened for editing.

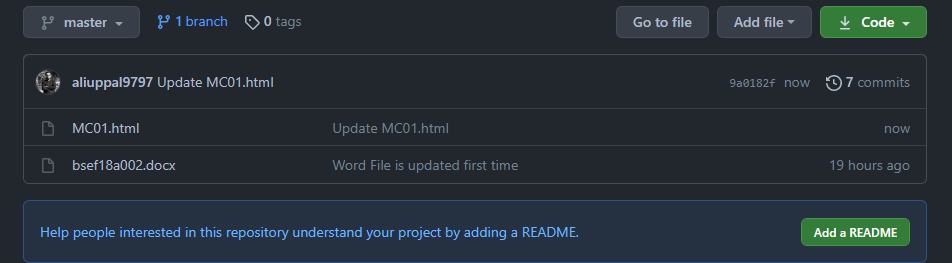


I have added line number 11 here. 

Then we can place a commit message here and then click on Commit Changes option. This will update MC01.html.

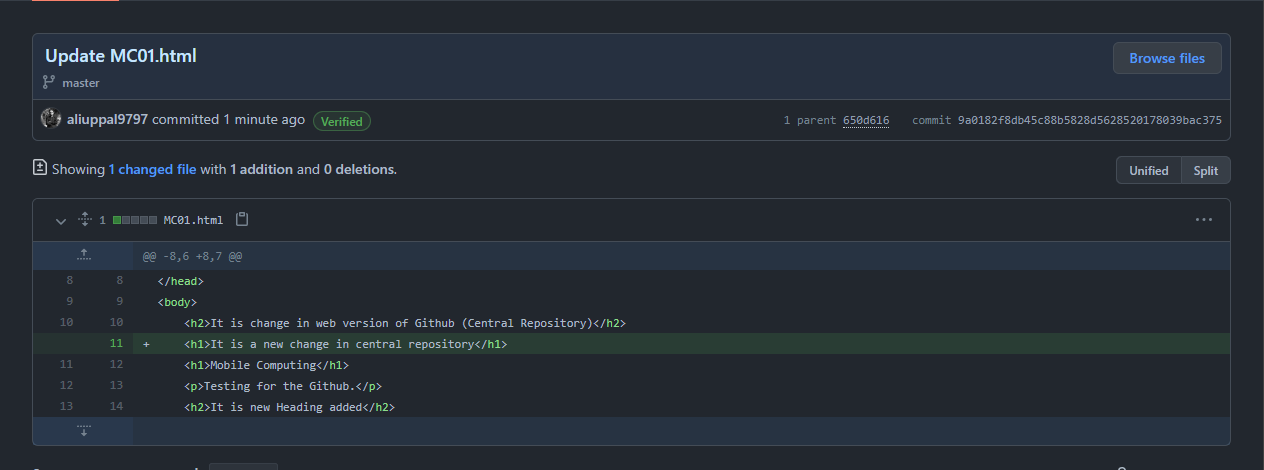


Here we can check that MC01.html is updated and a new commit is added there.

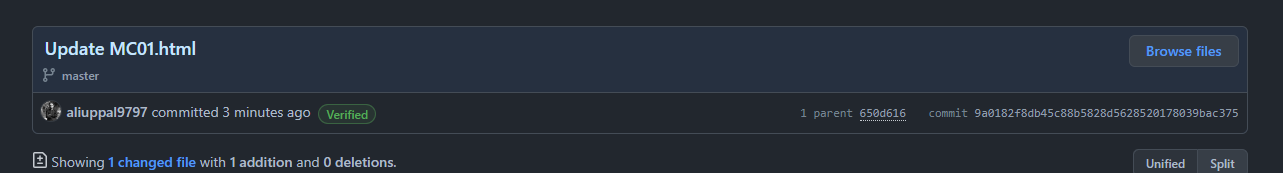


By clicking on the new commit name, we can see the changes in our central repository.

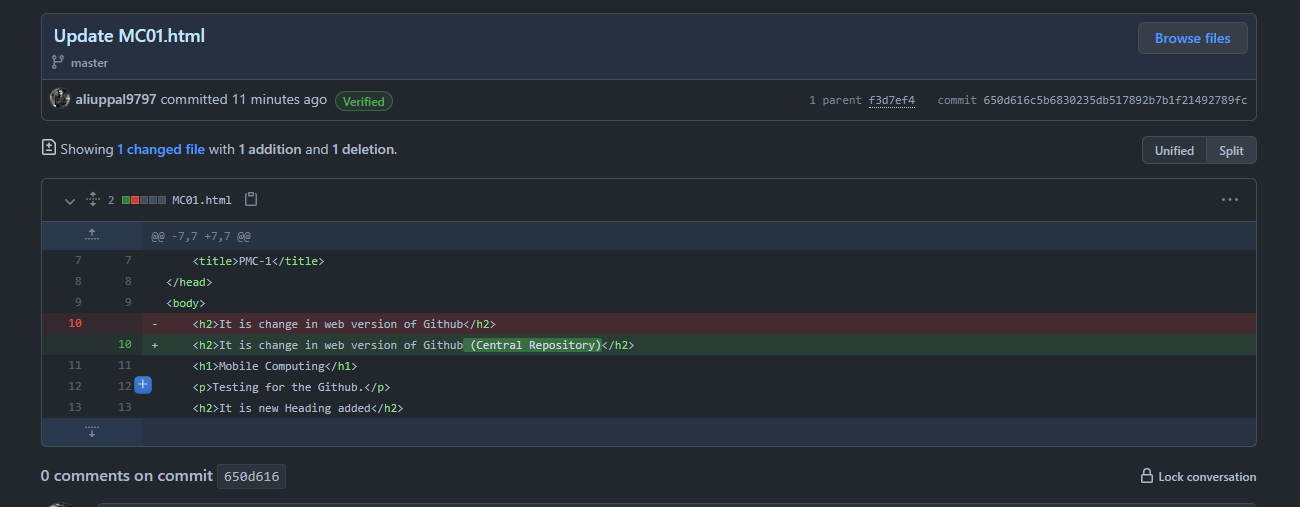
Here we can see that line number 11 is greenish which means this code is added in this repository.



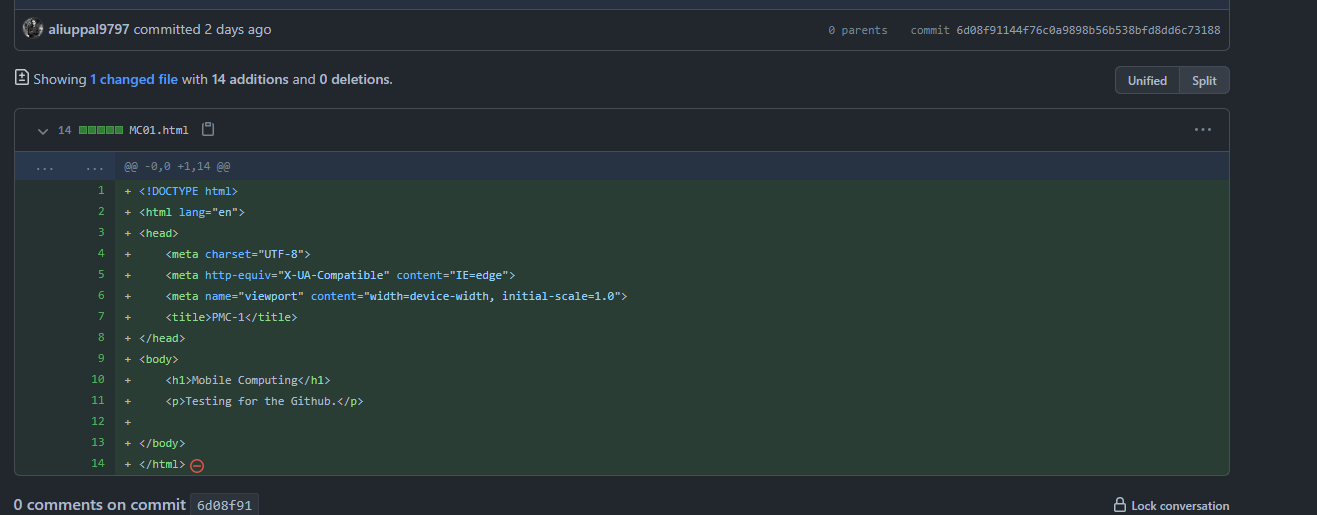
Here we can see it has one parent. We can see its parent by clicking on parent’s file number.



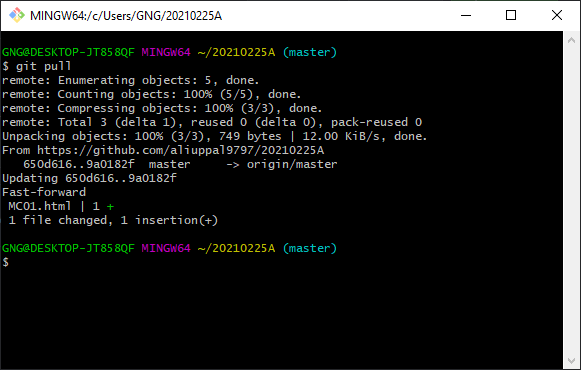
Here is the parent of latest commit. We can see it. And it also has one more parent.



By clicking on its parent’s file number we can see its parent.

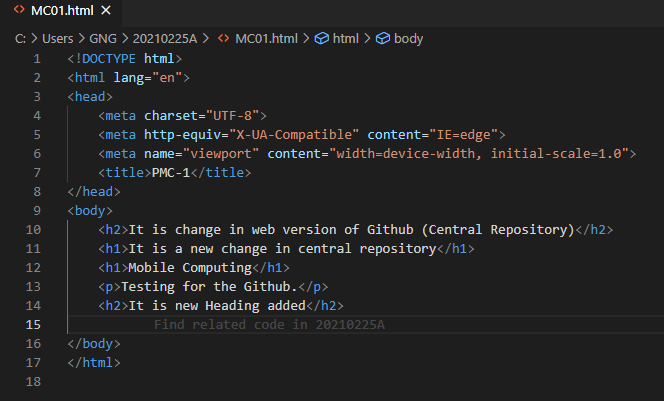


Now we come on git bash and use   
git pull

Command 

Now our local repository is synchronized with the central repository.

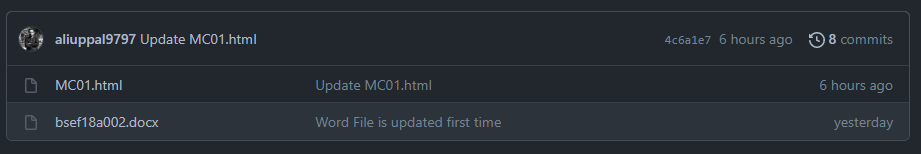
Here we can see that MC01.html on our local repository is updated.

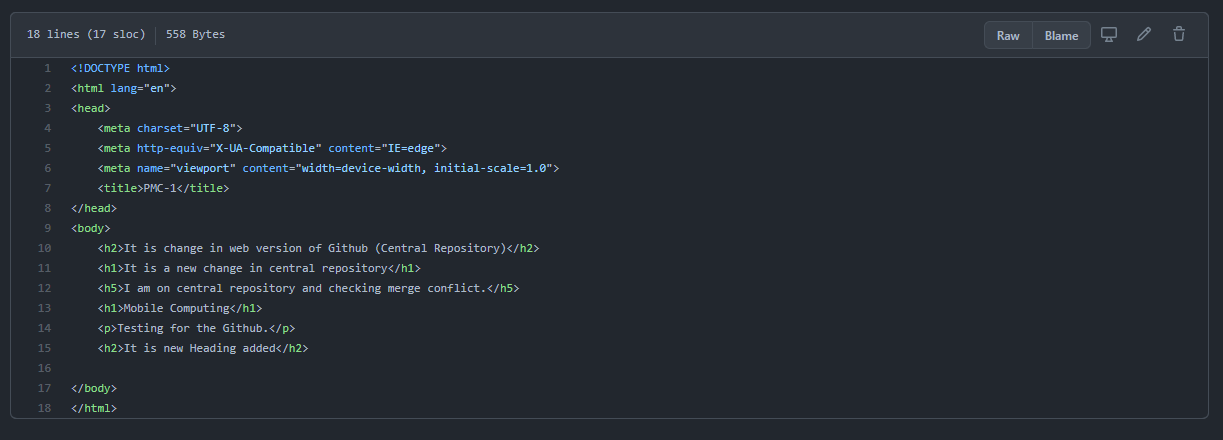


**Merge conflict**

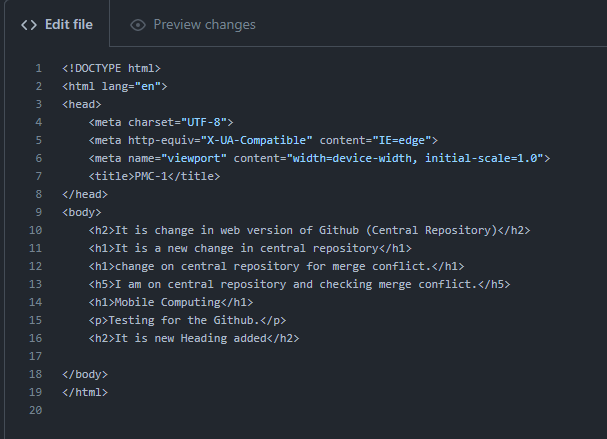
Merge conflict occurs when same line is changed on both repositories (on local and on central repository).

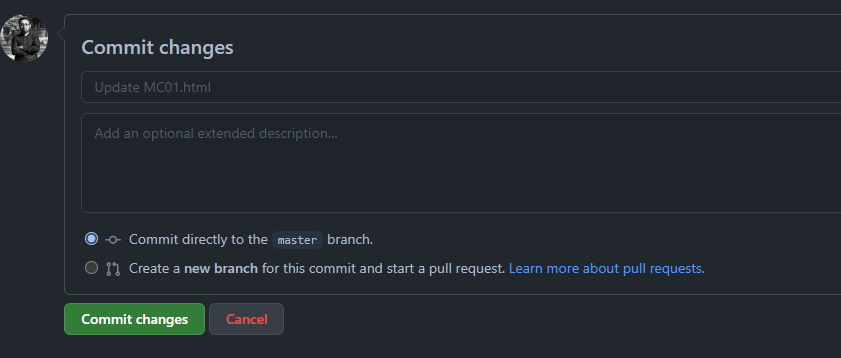
To elaborate merge conflict first of all on central repository we open MC01.html by clicking on its name.

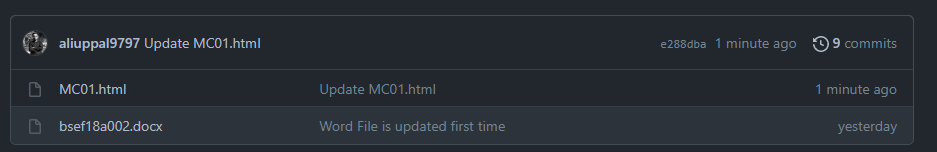


Then this window will appear. 

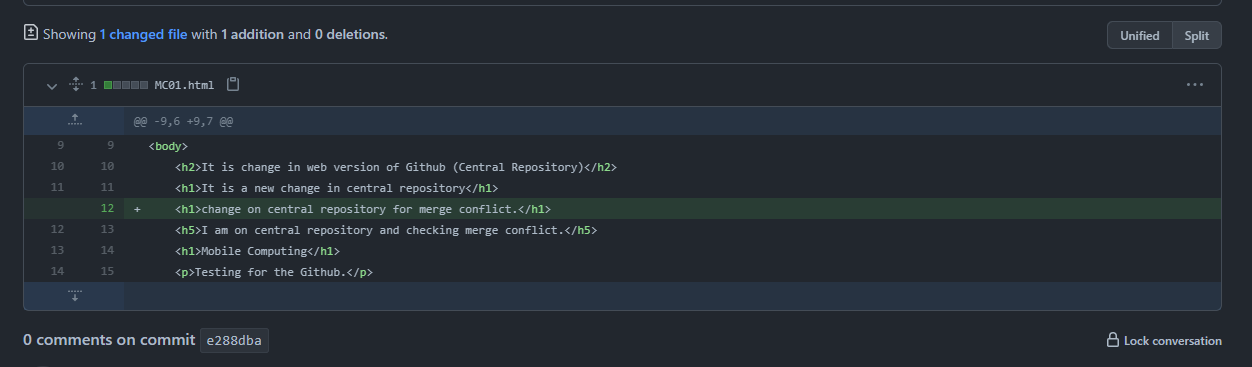
Click on edit option. (which is on upper right corner .

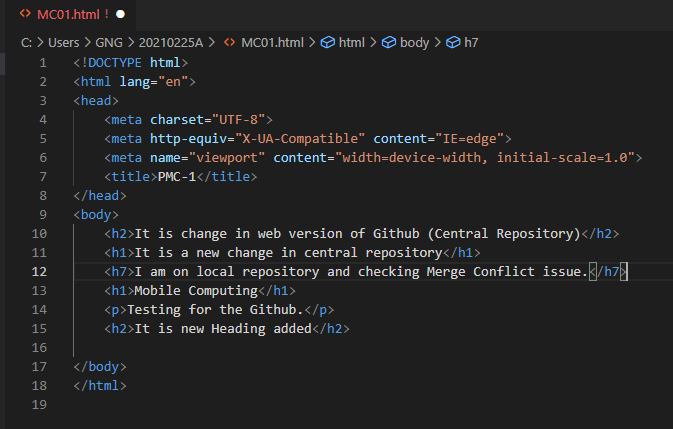
Now we added line number 12 on central repository.

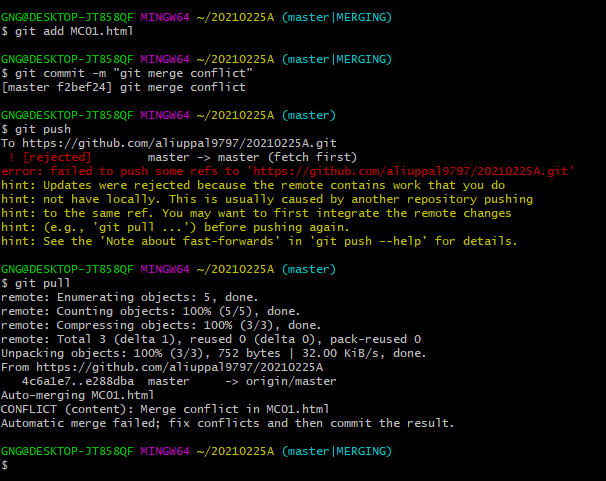
Now after writing commit message and name, click on commit changes. 

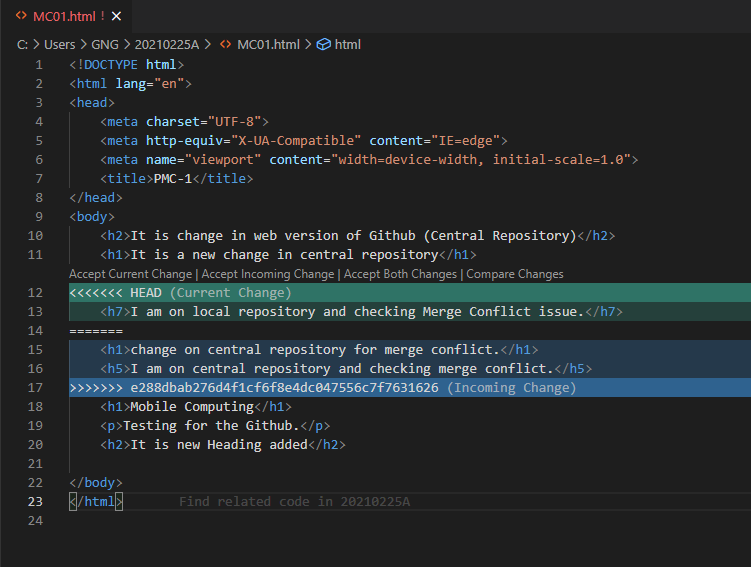
Your commit changed successfully. For verifying that changes. Click on new commit’s option (Update MC01.html) 

You can check line number 12 is changed

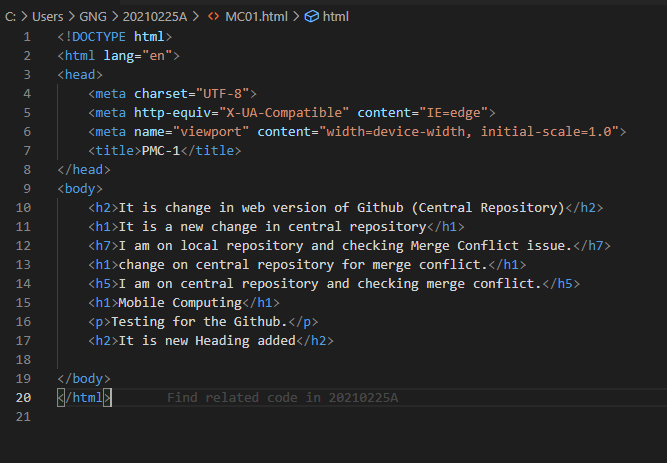


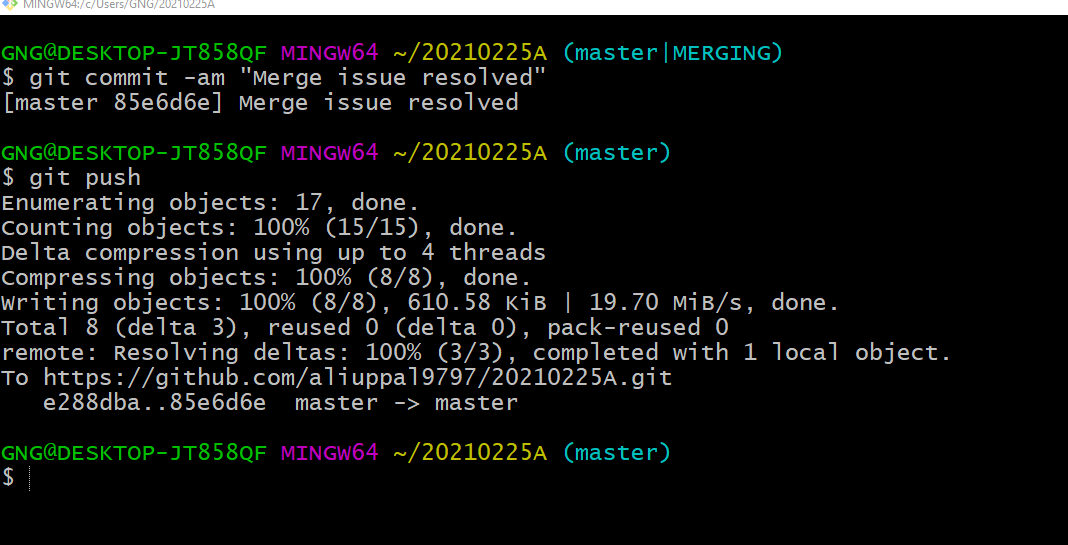
Now on local repository we update MC01.html by adding line number 12. 

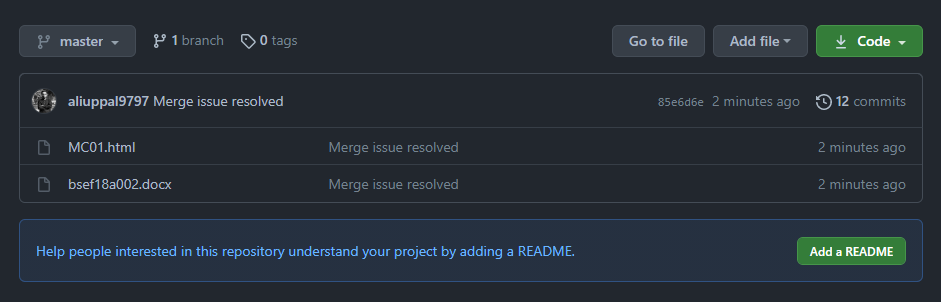
After saving this file on local repository, we open git bash and write these commands on it. And the conflict error arises.

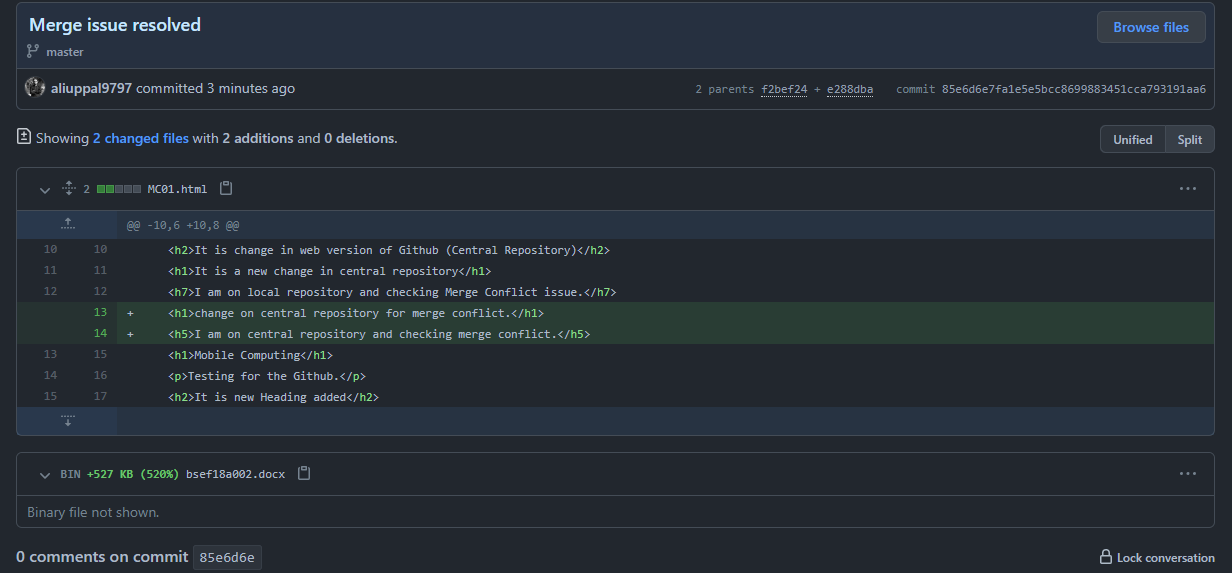
When you open MC01.html on your IDE on local repository. Then you will get these options. 

You can chose one of these options for solving this conflict. 

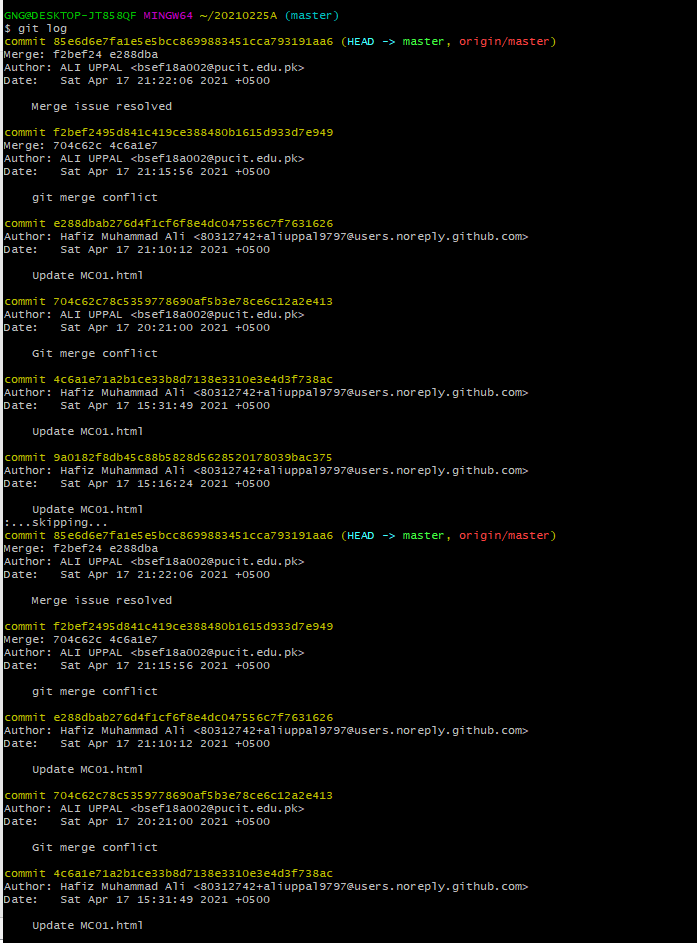
I chosed Accept Both Changes. Now this conflict is resolved and both changes are updated on my local repository. 

After solving conflict issue we open git bash and run these commands. And you can see these is no error anymore. 

For varrifying we go to our central repository. 

Click on new Commit which is Merge Issue Resolved. And you can see your file is updated. 

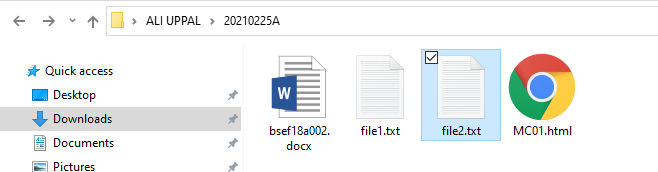
**Git log**

This command is used for checking log of our gits. 

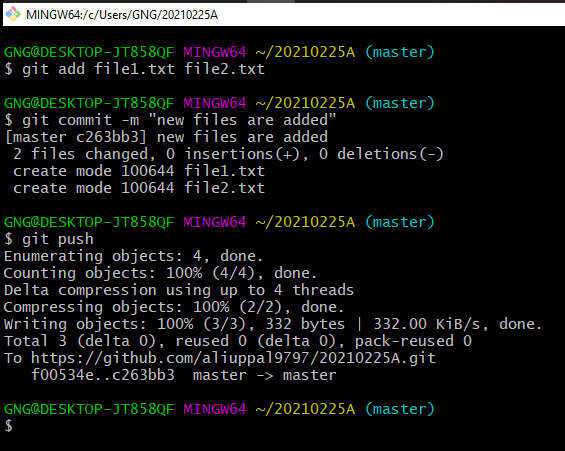
**Git rm**

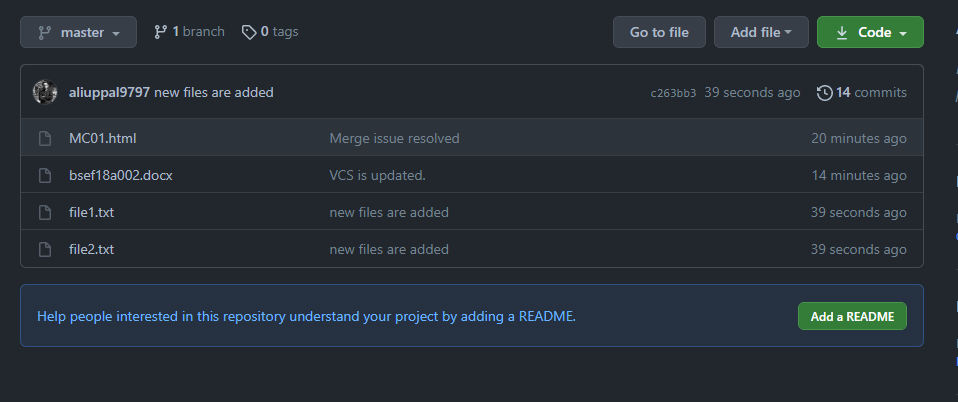
First of all we create 2 new files on our local repository.

* file1.txt
* file2.txt.



now add these two files on our central repository by using these commands on git bash.

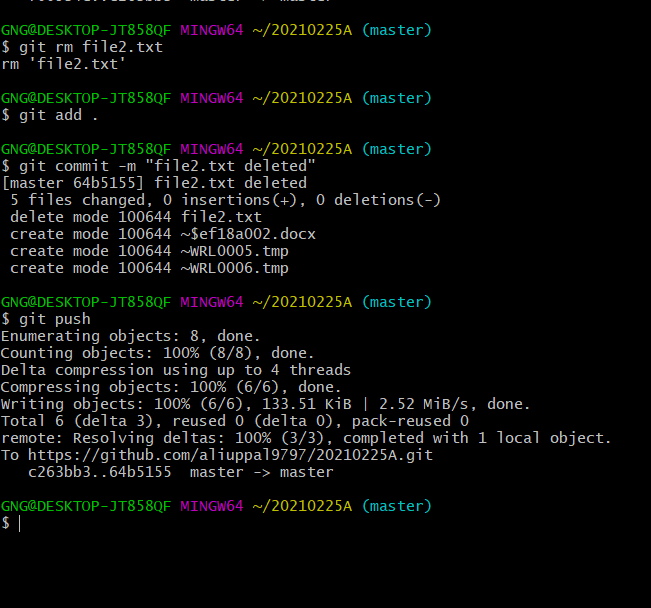


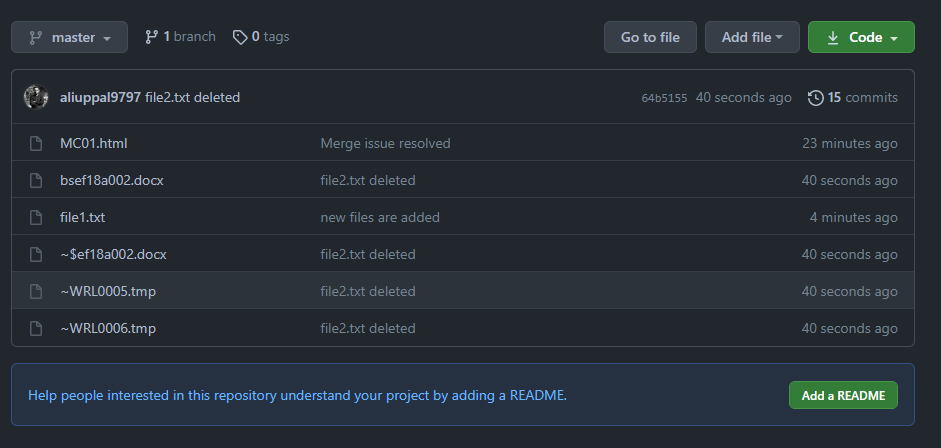
You can see 2 file1.txt and file2.txt is uploaded on our central repository. 

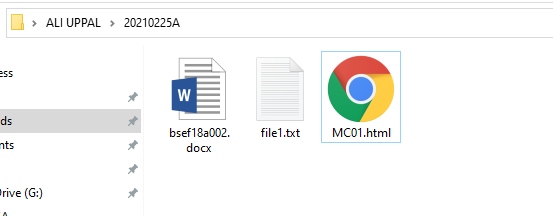
Now we remove file2.txt by using

**Git rm file name**

on git bash



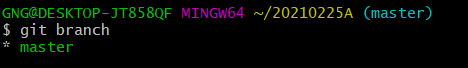
You can see file2.txt is deleted from our central repository. 

File2.txt is also deleted from central repository. 

**BRANCHING**

* git branch
* git checkout
* git merge

git branch

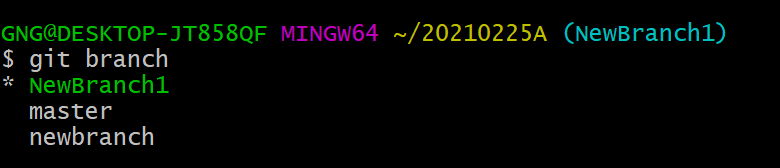
by using git branch we can check our current branch just like this. It shows master as our current branch.

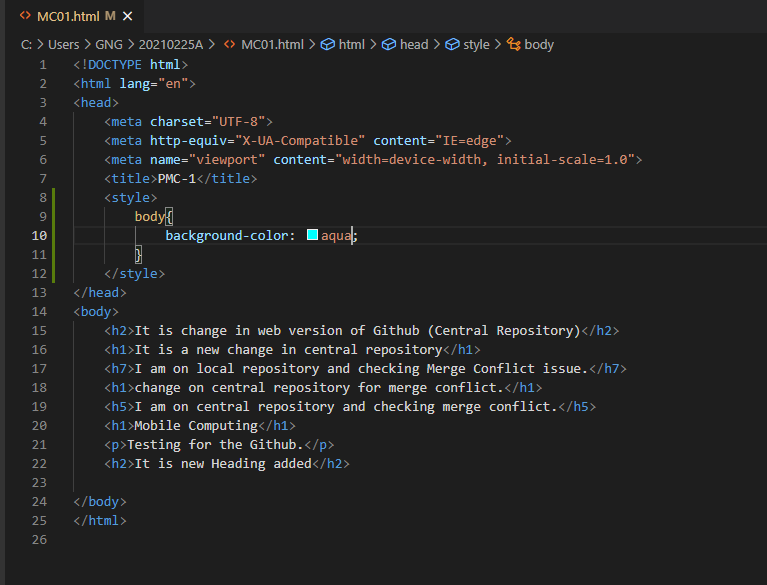
git checkout –b branch\_name

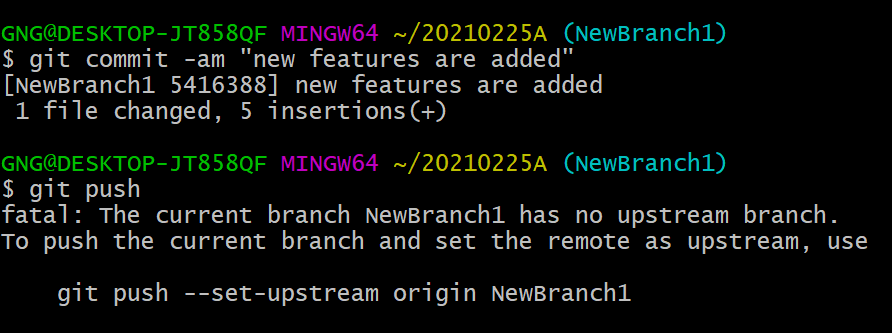
by using this git checkout –b branch\_name, we make a branch just like this.

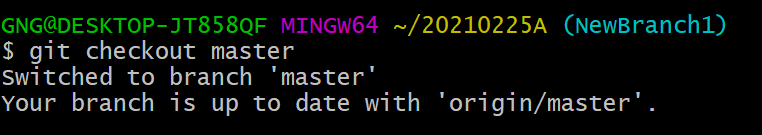


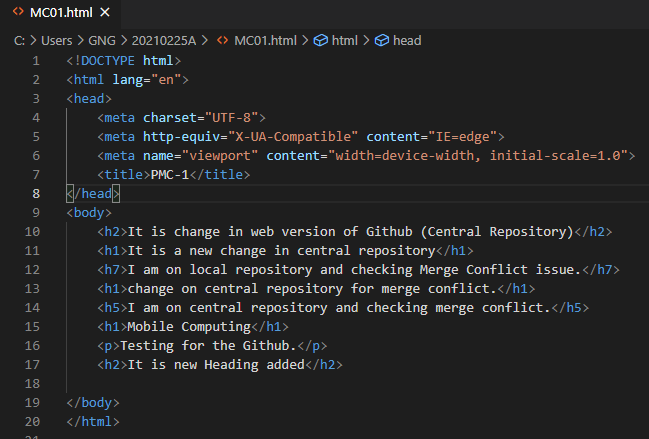
now you can check that your branch is changed from master to newbranch1



now we updated MC01.html and added some CSS in it. 

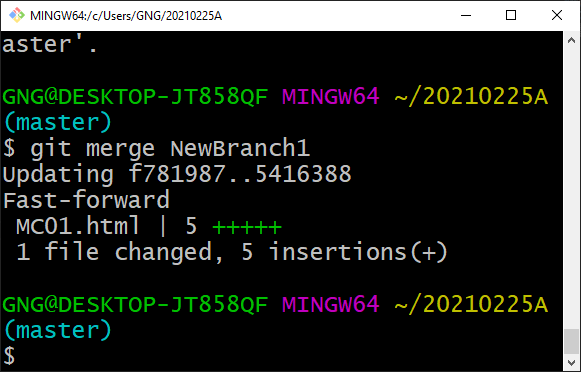
by using git commit –am we added and committed with a single command 

now we move back to master branch 

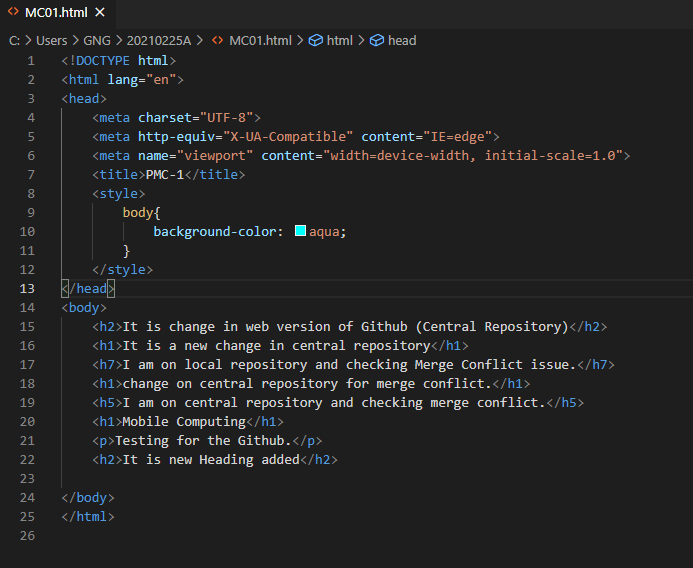
and we can see that our MC01.html file is reloaded by the previous code. And there is no CSS applied here. 

by using

**git merge branch\_name**

this command we merged both branches. 

we can see that both branches are merged now..and the content of both files are updated.

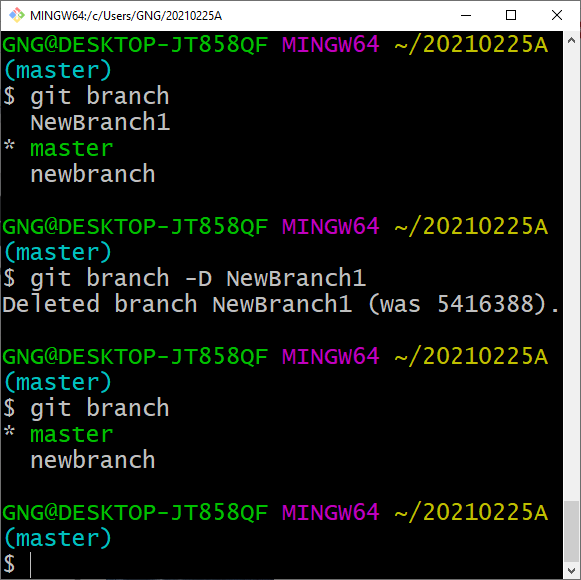


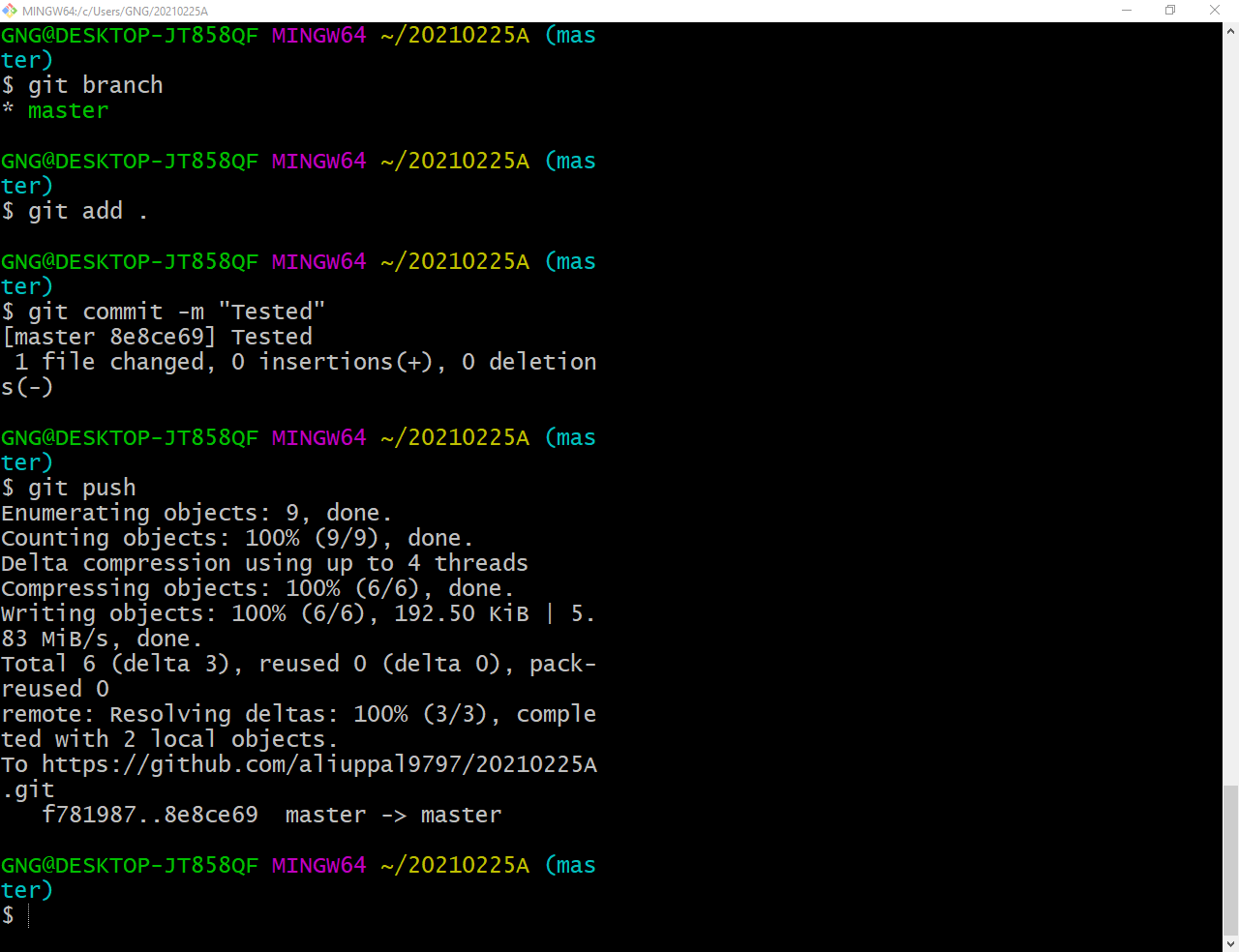
**Delete Branch**

For deleting a branch we use this command

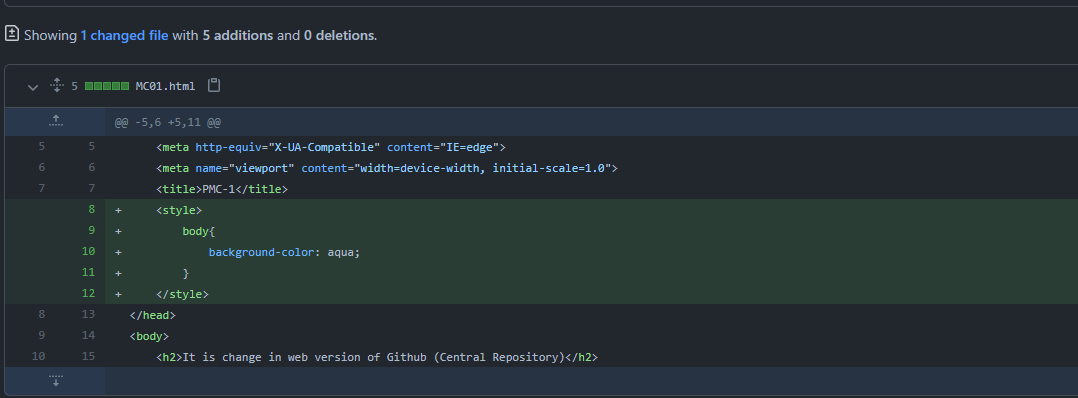
**Git branch –D branch\_name**

Here we can see that NewBranch1 is deleted.

****

At last, we update our last commit.

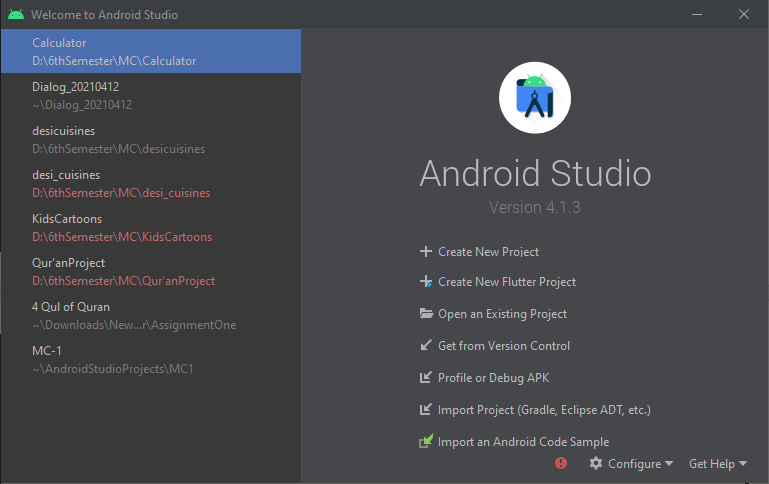
On central repository you can see MC01.html is updated.

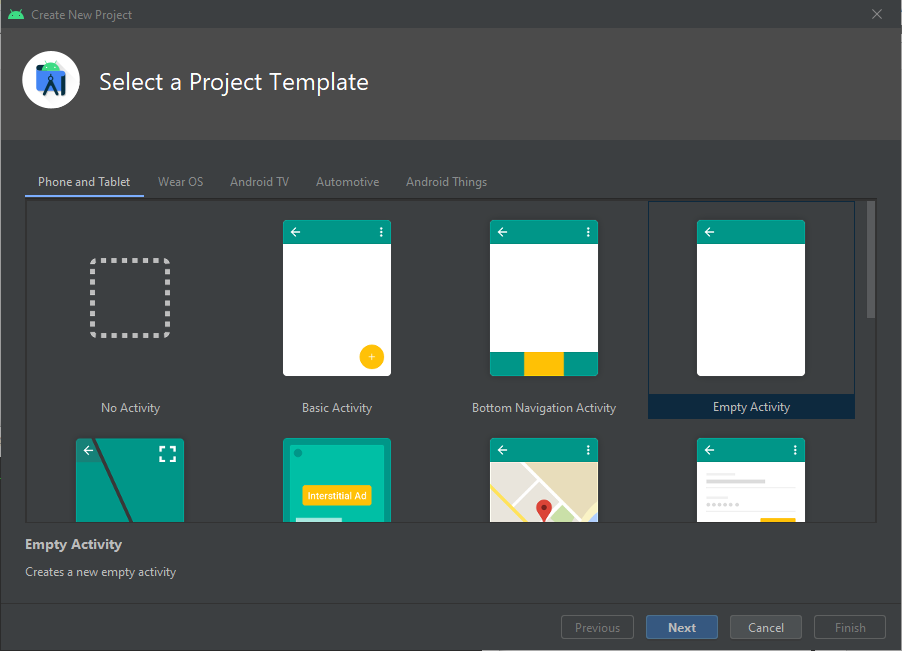


**Android Studio**

## First project

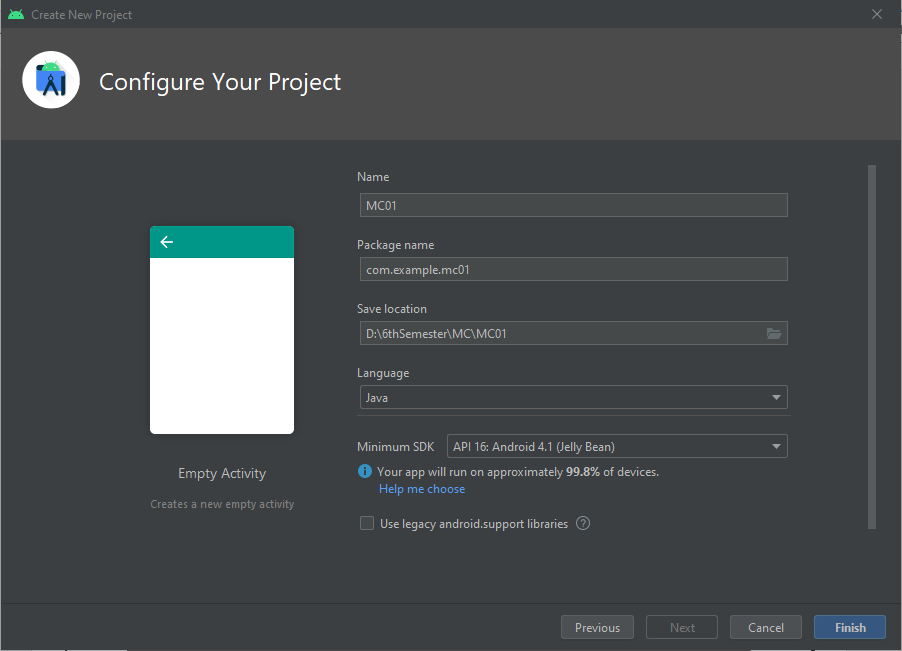
Open android Studio and select Create New Project option. 



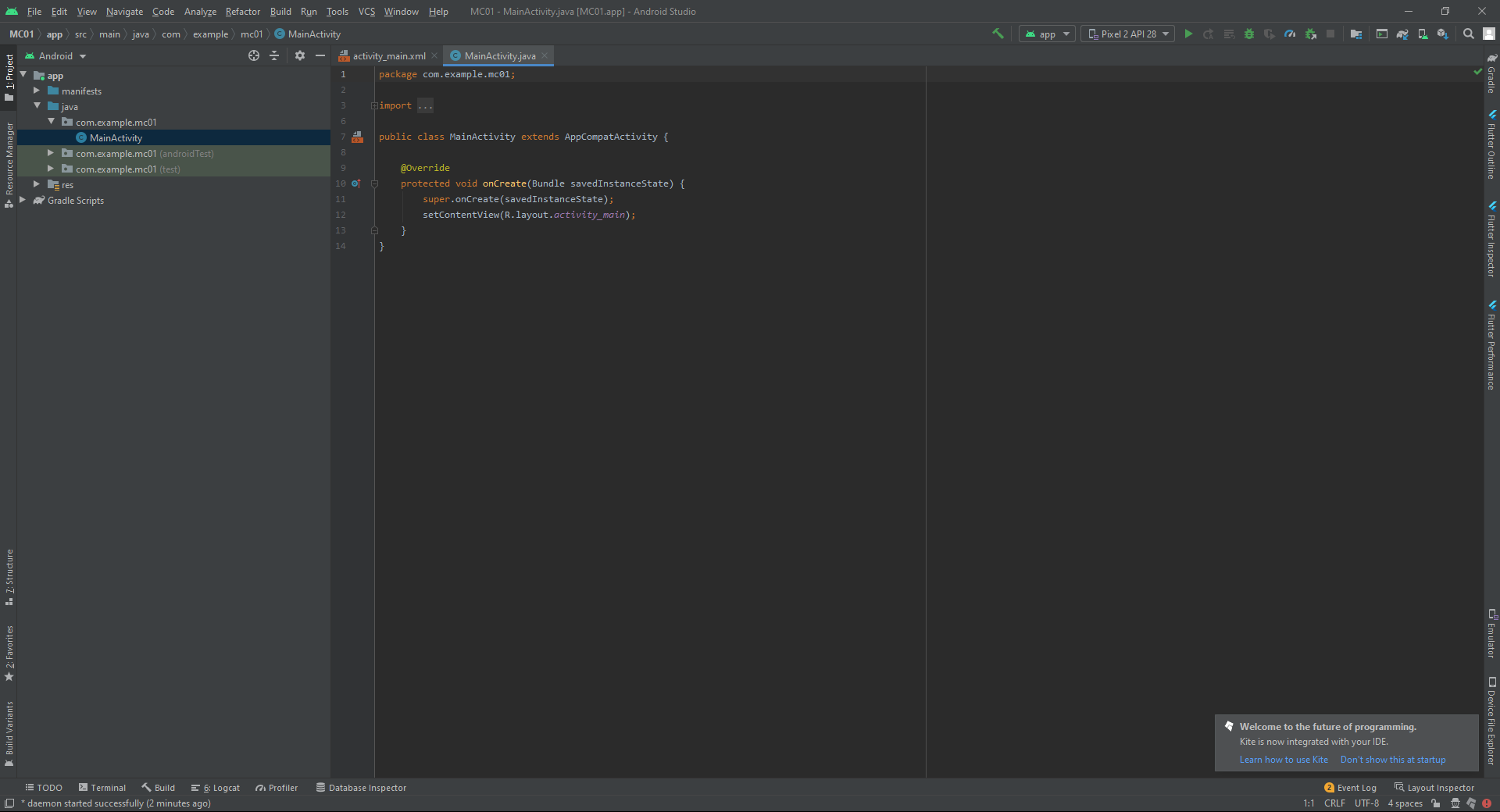
Chose Empty activity and then click on  button.

Write the name for your project and Package Name. after writing these names chose the location where you want to save your project on local disc. From language drop down menu: select Java. And then select Minimum SDK version.

NOTE: try to choose that SDK which will run on Maximum number of devices.

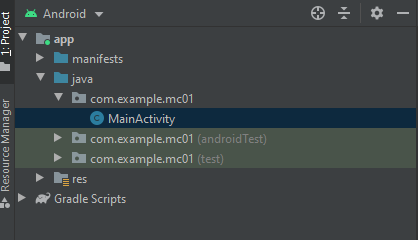
And then click on Finish button.

It will take some time. (it depends on your machine and its efficiency.

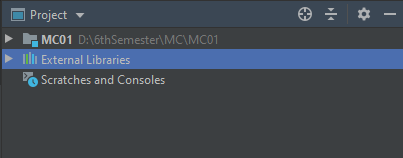
After loading the required files and allocating memory. You will see this window. That means your Project is ready to edit. 

Here we have two different file structures.

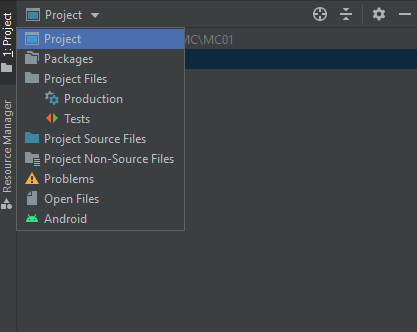
1. **Android**



1. Project



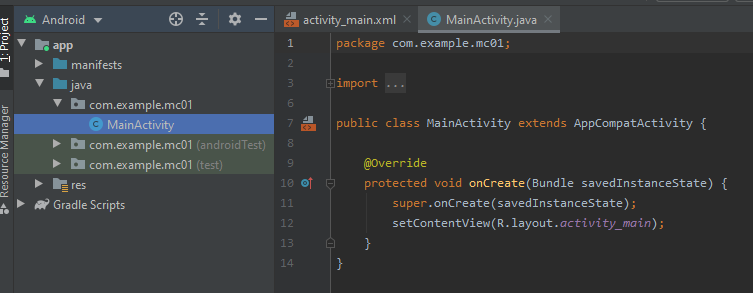
we can also choose these file structures.



But we will work on Android File Structure.

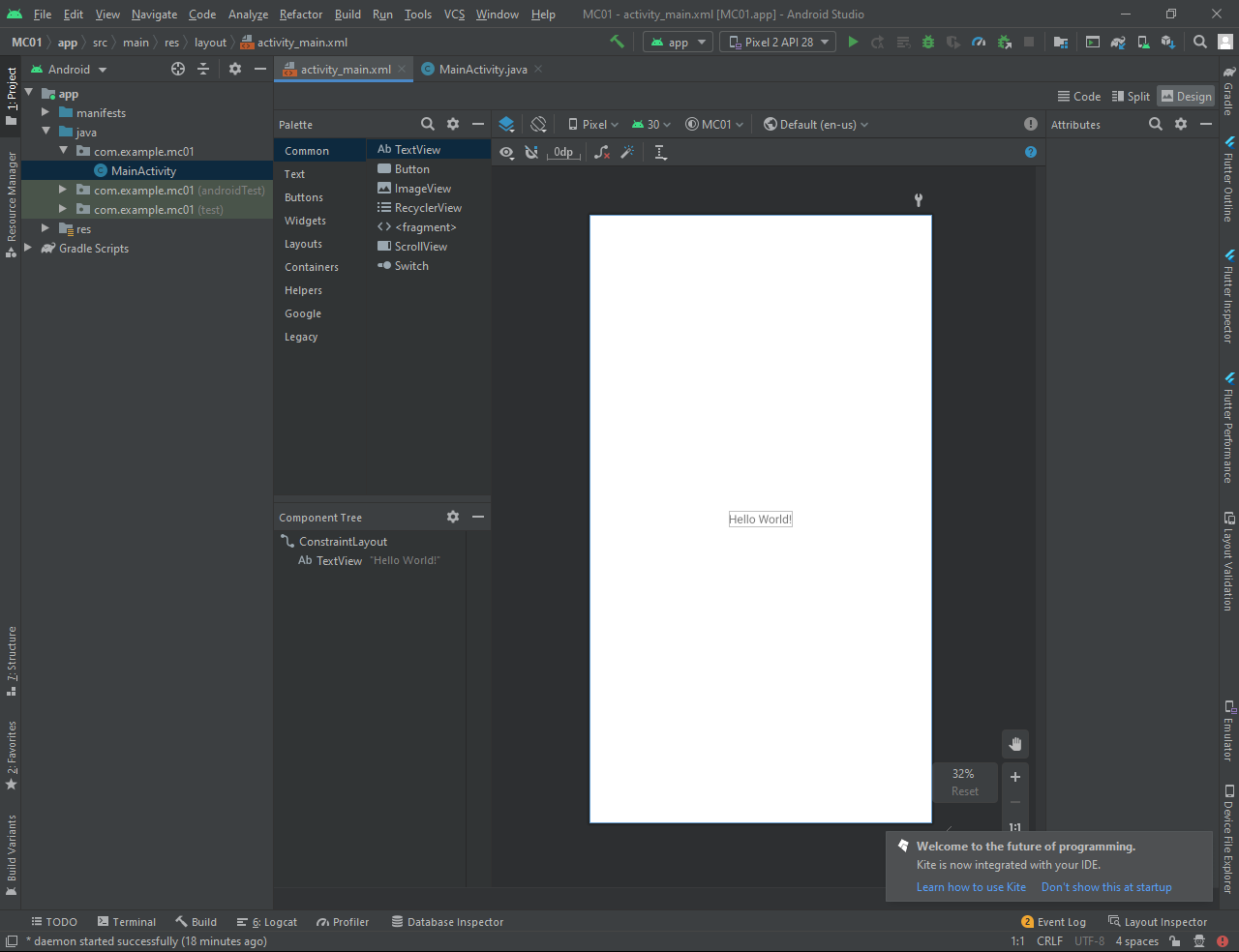
In our project we have two main files.

1. MainAcitvity.java
2. Activity\_main.xml



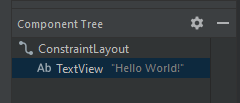
Our coding part will be in MainActivity.java

And our GUI (Graphical User Interface) related things will be in activity\_main.xml



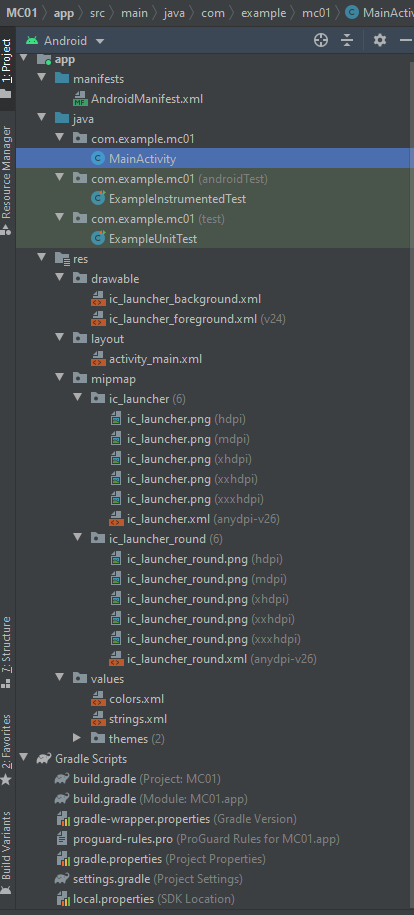
## Component tree

Out components will be placed here.



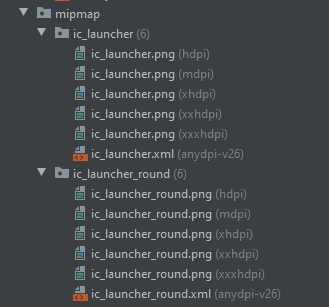
## Project Resources

These project resources files are built in available in our new project by Android Studio



## app icons

Here you can find the icons of different resolutions which will be used for your app. These icons are measured on dpi scale.



You can find these icons on that path.



## buil.gradle (project)

If you are using some third party tools then you can find their information in build.gradle file



And you can find this file here.

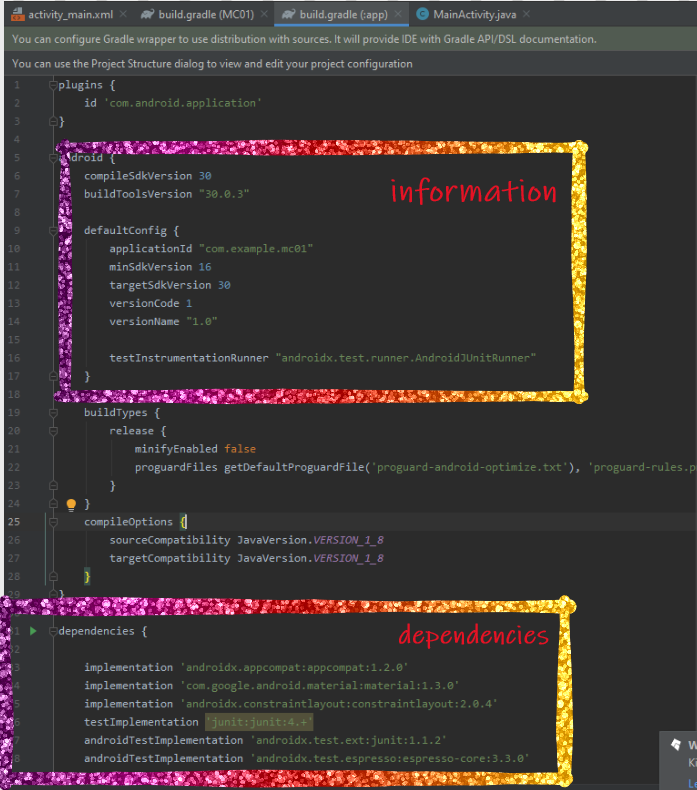


## build.gradle (module app)

You can find all these information there:

* Version name
* Version code
* SDK’s info

And you can find dependencies there which are essential for running your project.

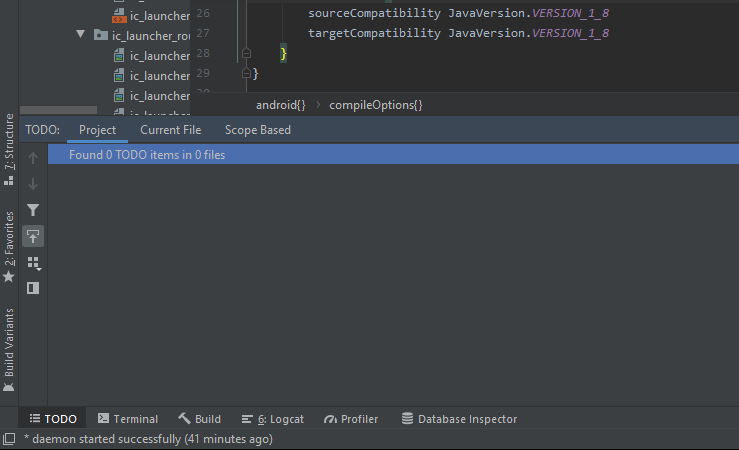


## todo

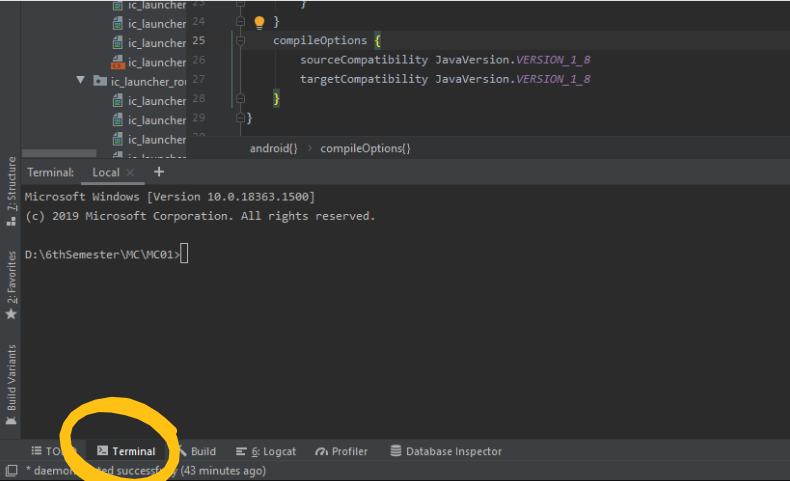
We place our pending things here. This will help us to continue our work from last activities.

And you can find it on lower left corner of the android studio.

**//TODO: message..**



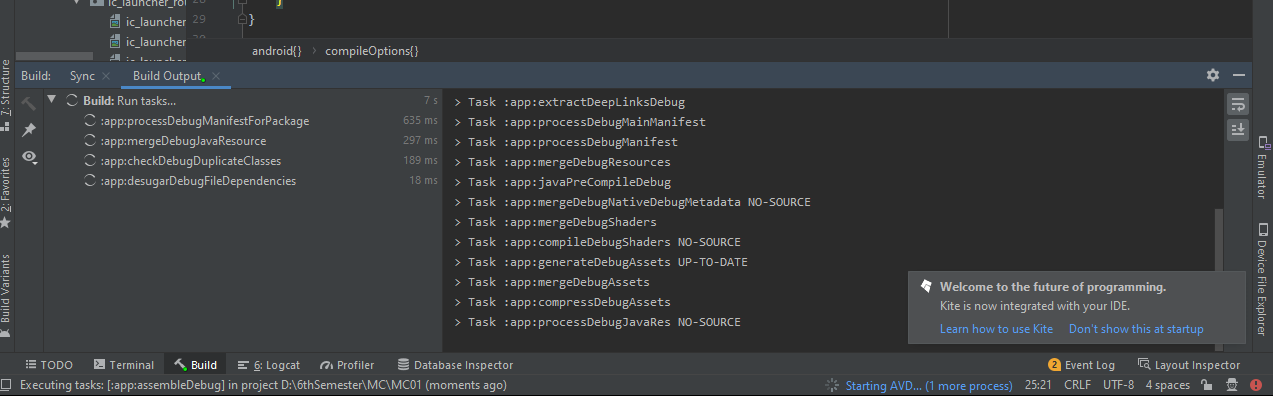
## terminal

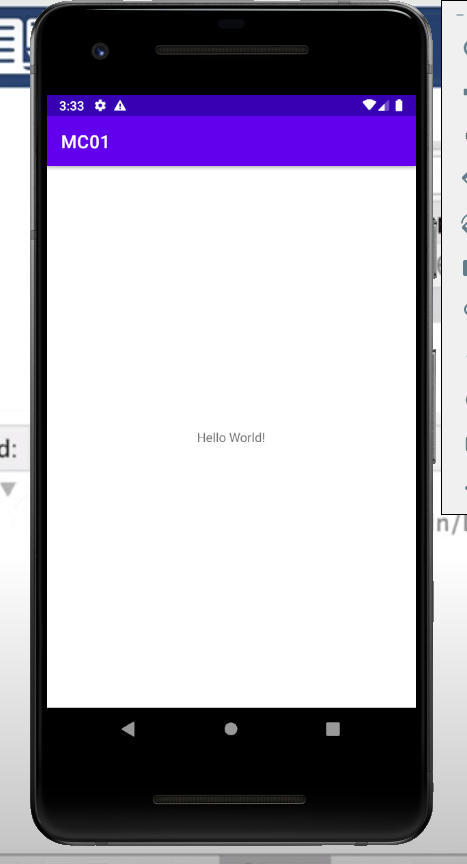
You can run all commands of command prompt on its given terminal

## build

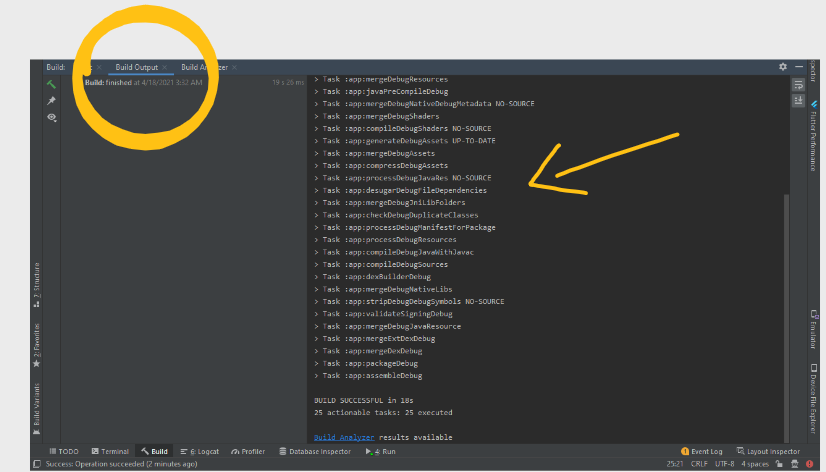
For building your project you can click on this option. Which is available on upper right corner of the android studio.



It will take some time for execution of your project. If there are some errors, then you can see there.

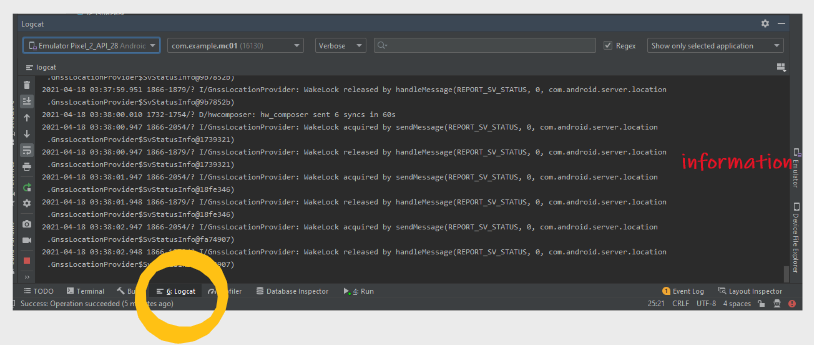
After completing the execution process your app will run on an emulator/your own android device.

You can find the relevent info about your running task here.



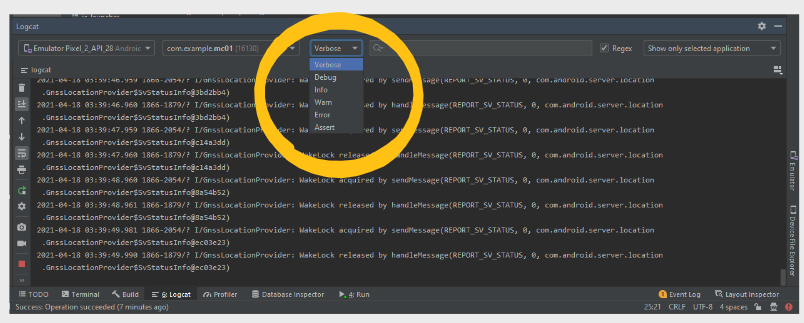
## logcat

Information about your running process are located in LogCat menu.



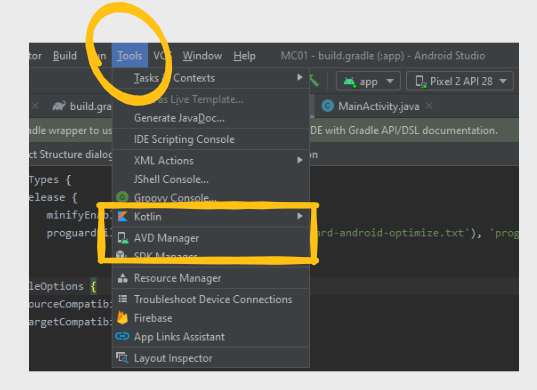
You can see errors/problems/warning there by switching in these options.

* Verbose
* Debug
* Info
* Warn
* Error
* Assert



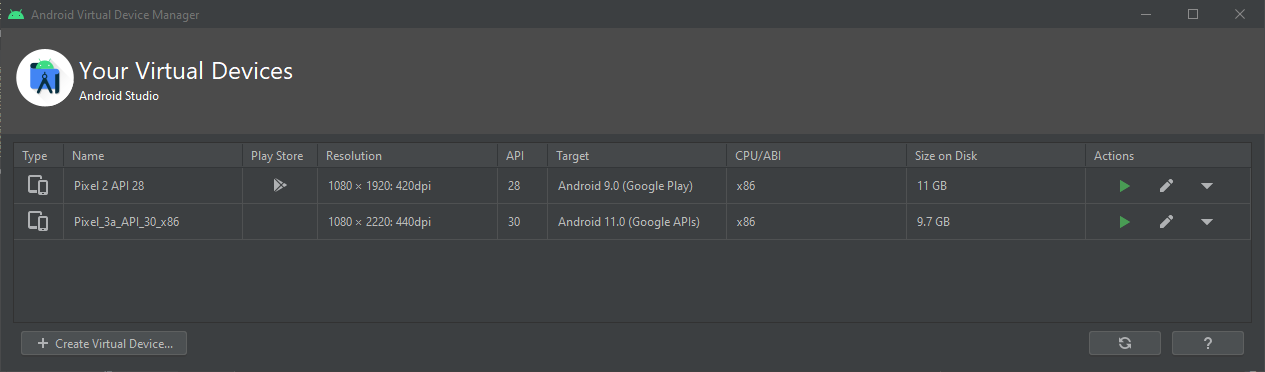
## avd manager

By clicking on AVD manager you can download a virtual device on which you can see your running project’s output.

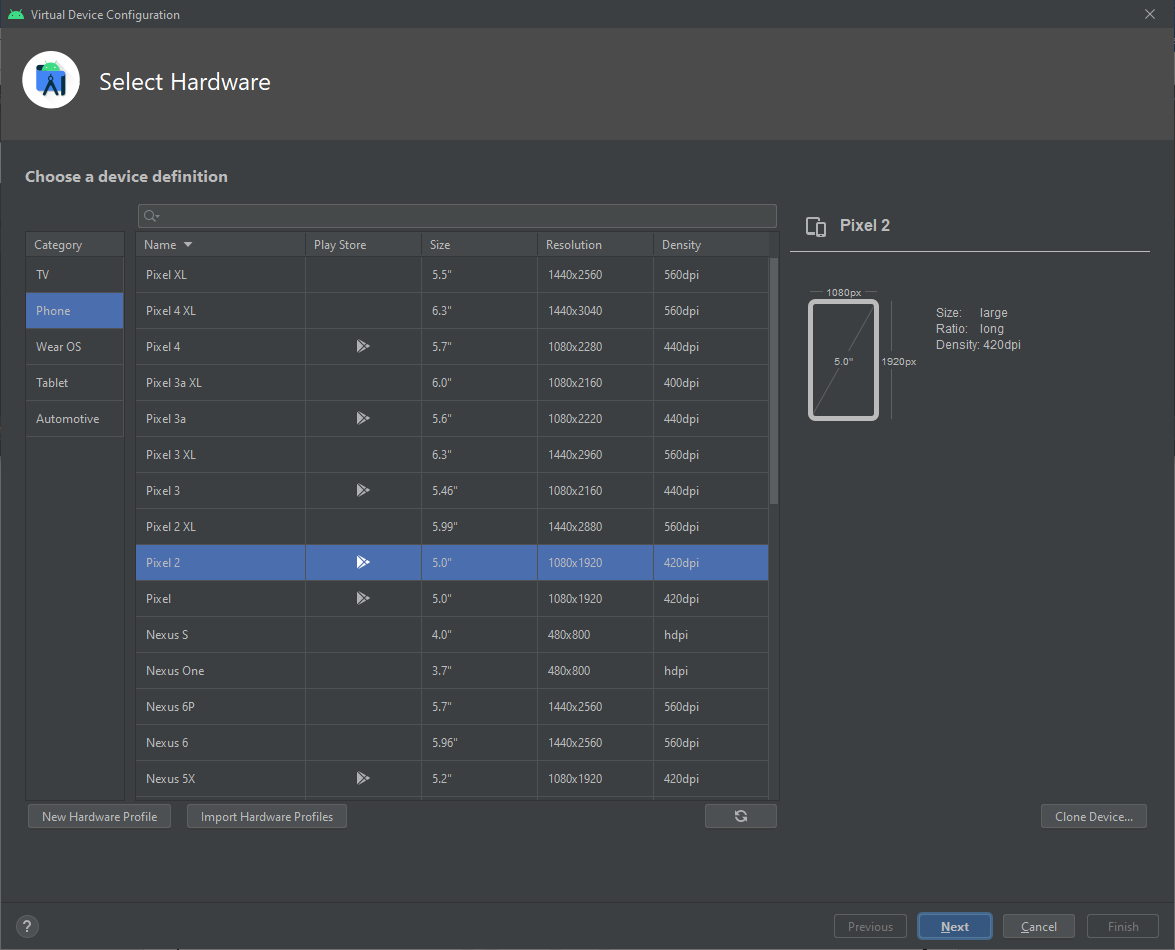


You can download or Create Virtual Device and choose that virtual device for running your app.

Available virtual devices are given below.



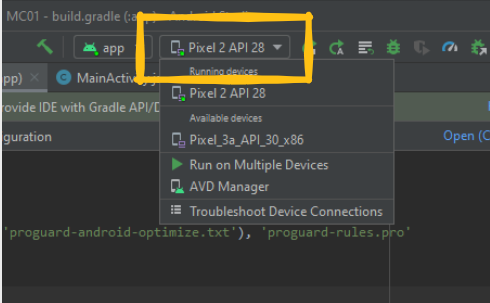
By clicking on **Create Virtual Device** option this window will appear.

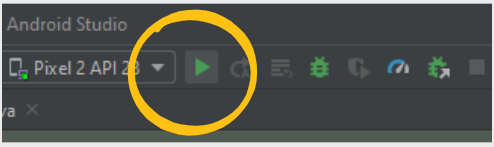


By selecting your desired device click on Next Button.

Now you can run your app on selected device.

From this drop down menu you can choose from your available devices.



Now by clicking on this button your app will build and run on selected virtual device. 

Your virtual device will look like this.

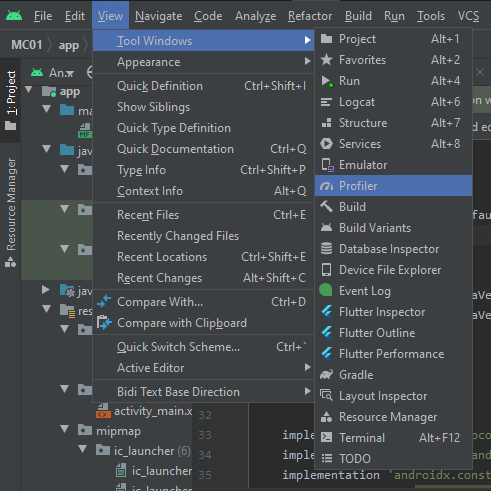


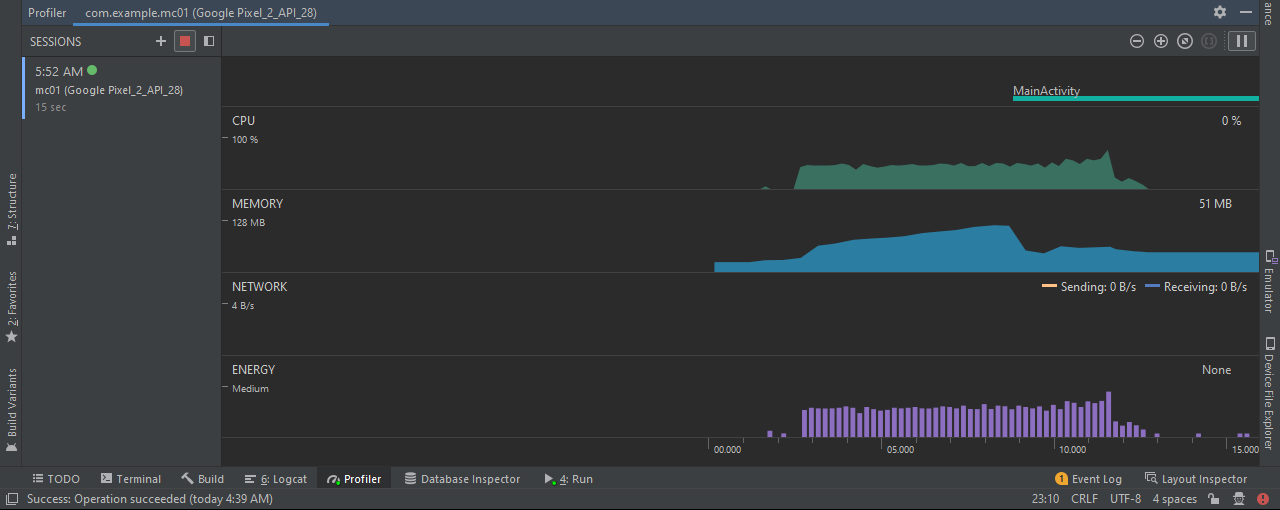
## setting for real devices

For using your own device for developing apps, you have to connect your mobile device to your PC/Laptop with a data cable and then enable Developer Options on your mobile device. Then Enable USB Debugging option on your mobile. By doing these steps you can run your app on your own mobile device.

## android profiler

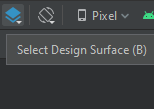
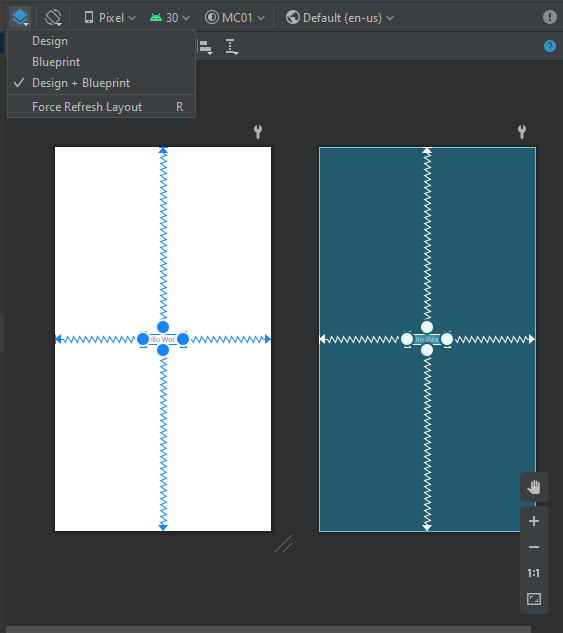
It provides the information regarding the usage of system resources.



Here you can see the usage of resources by your app.

## design and blueprint

On design section you can choose among different designs.

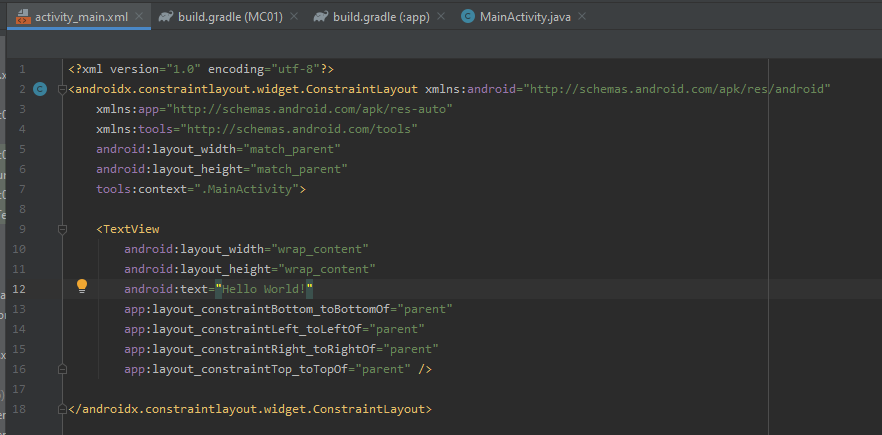
 

## design and text view

By using these options you can changes among the Code view/split view/design view. These options are available on upper right side of the screen.

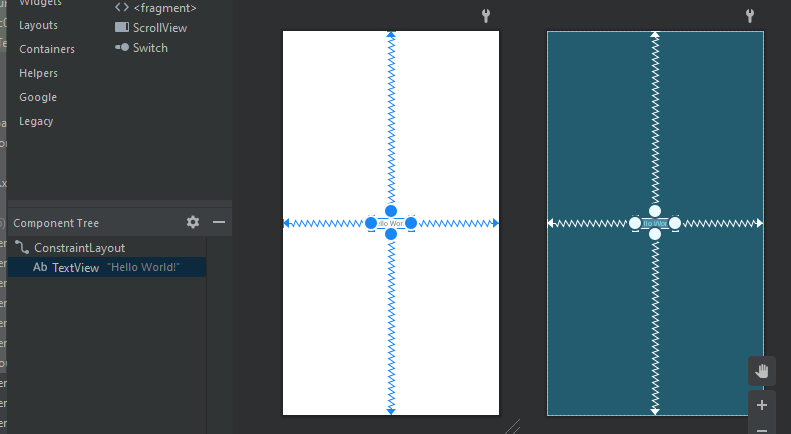


You can also use text view for changing design/content of your app. Here you can update the text of your Text View Item from “Hello World!” to something else or you can change other properties of text view.



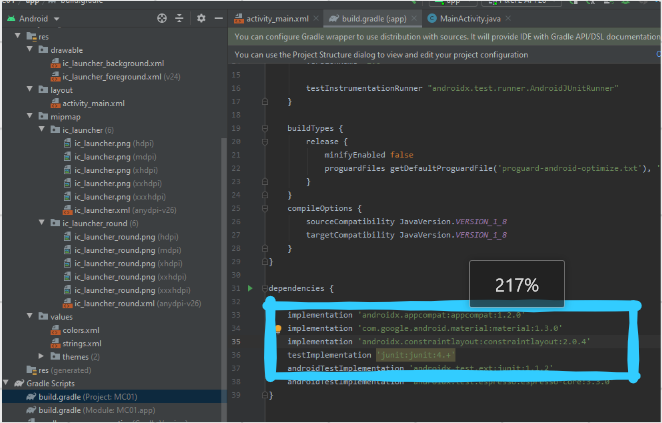
## constraint layout

Here we are using constraint layout which sets the position of our item by setting its upper, lower, left and right constraints.



## dependensies

In **build.gridle** file you can see the dependencies which are used in our project are available. Just like dependency for our layout is given here.



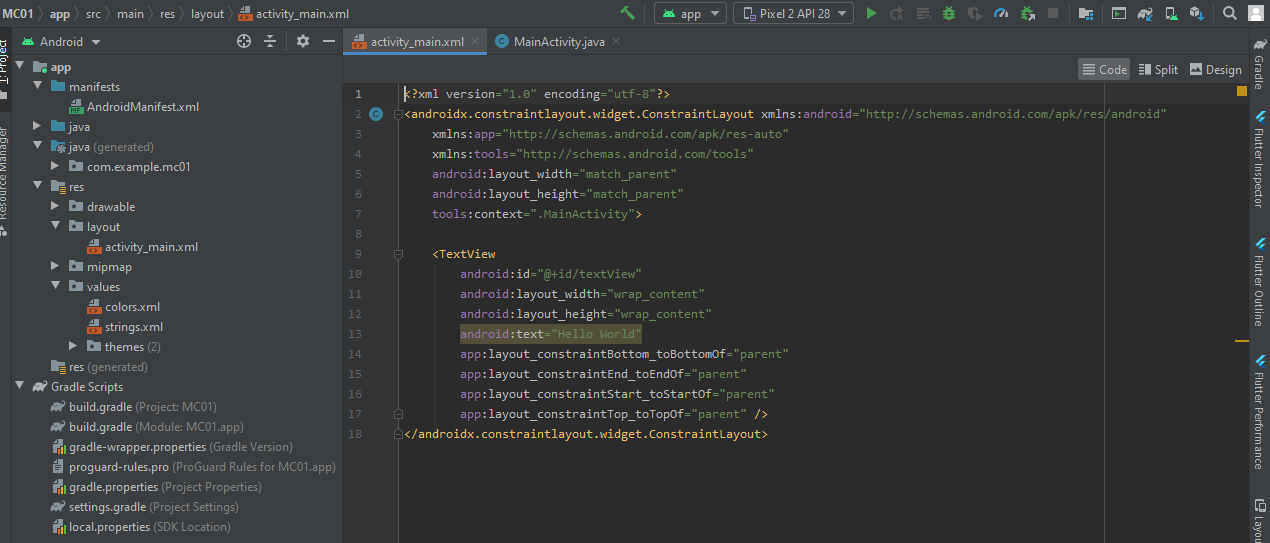
**SCREEN TASKS**

**LEC # 4**

# SCREEN1

First of all create an empty project.

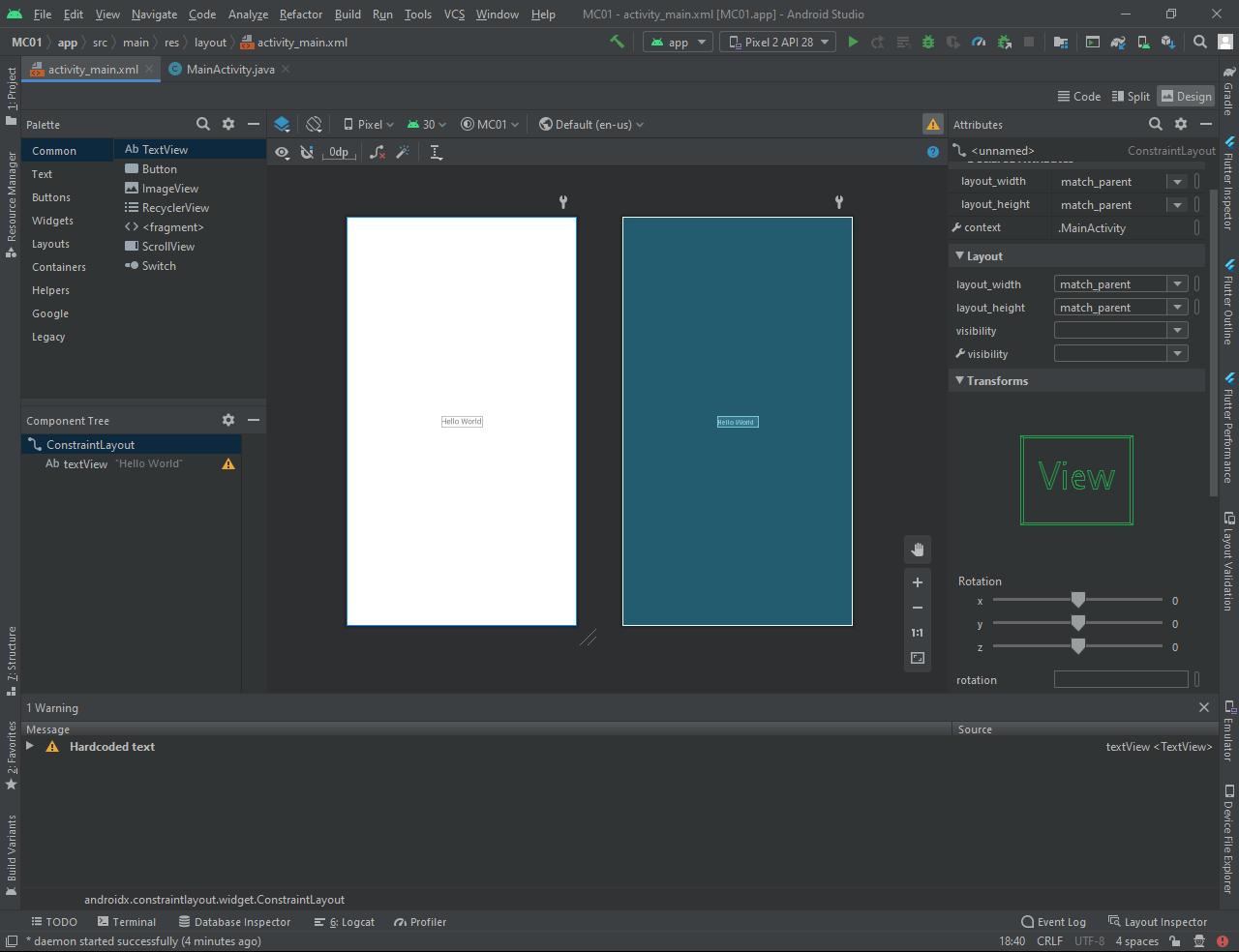
Open activity\_main.xml

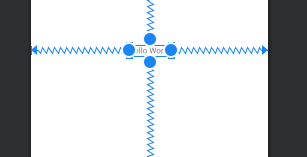


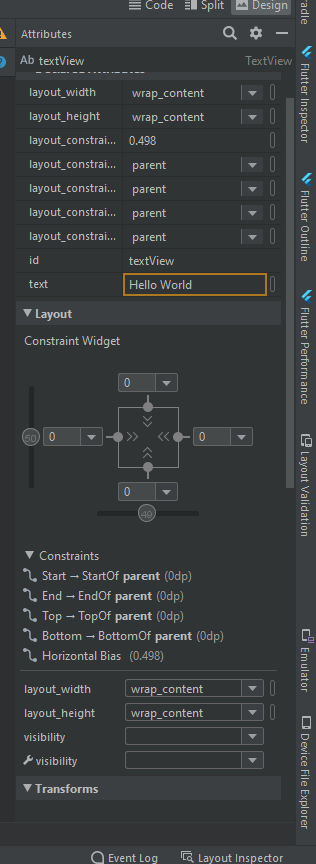
Then from upper right corner select **Design**



This window will appear

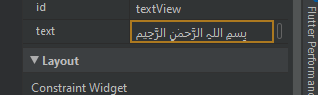


Select text view in which Hello World is written by Clicking on the text.

On right side of the window you will find Attribute option. Select on Attributes then this menu will appear

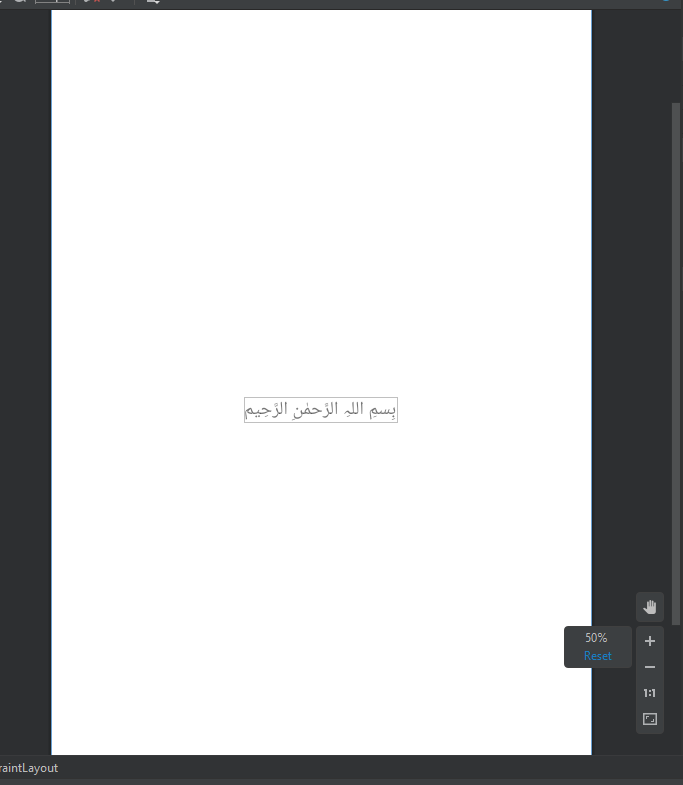
From text option you can change the text of textView item. We write

بِسمِ اللہِ الرَّحمٰنِ الرَّحِيم



Your text will be changed to بِسمِ اللہِ الرَّحمٰنِ الرَّحِيم

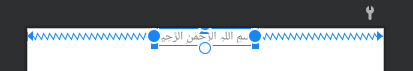
Now you can see your text is updated



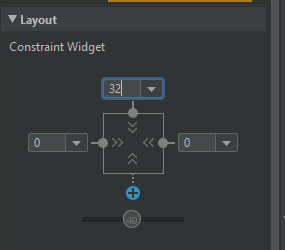
Now click on lower spring/constraint and press Delete key from keyboard



Your text will look like this one.



Now in attributes option, under layout >> constraint Widget, give upper constraint value = 32 and press enter. It will place some space from top.



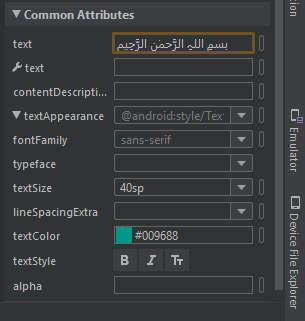
You can see the difference in design



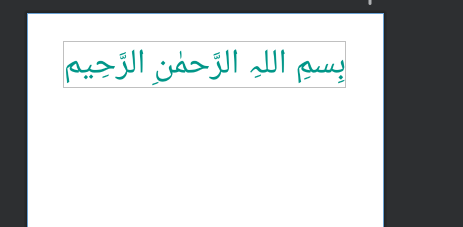
**Now in attributes >> Common Attributes >> textSize**

**Now in attributes >> Common Attributes >> textColor**

Change text size and text color and press enter

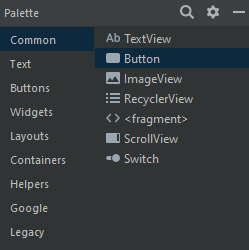
****

You can see your design is updated now



From this menu drag and drop these items to your design

1. textView
2. 3 buttons

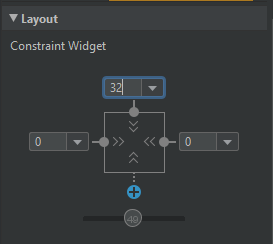




Set the constraints of the text view by selecting the constraint head and fix to the sides (left and right) and upper side with upper text view. Just like



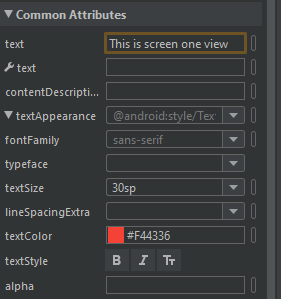
Set Its top margin up to 32



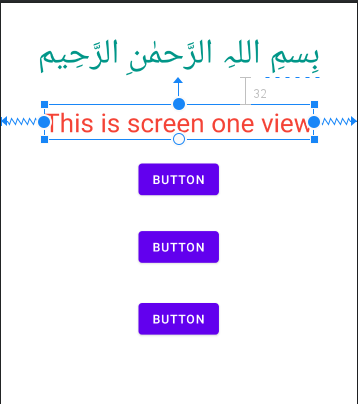
Updated design



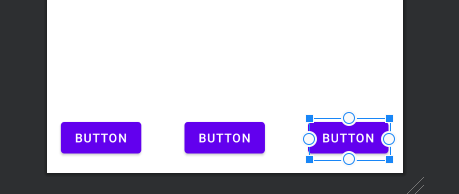
Change its color and text as we did before.



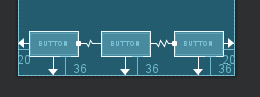
Updated design



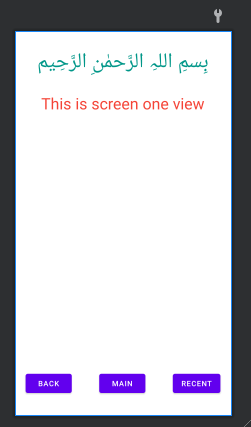
Now drag these buttons to the lower side of the design



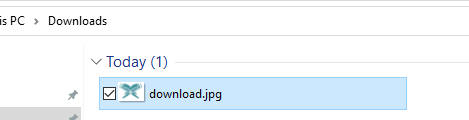
And set their constraints just like this one.



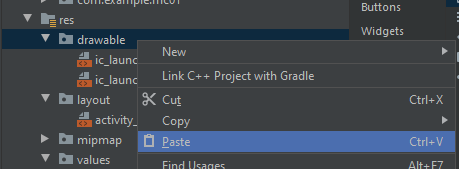
Now update the text of these buttons



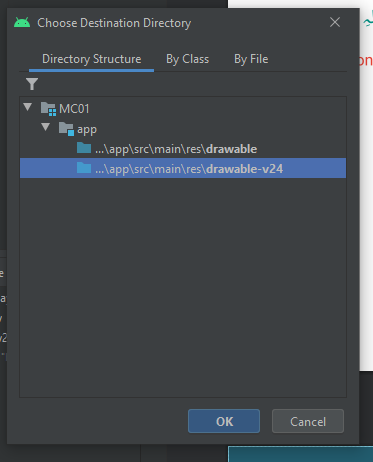
Now copy this image from your local directory



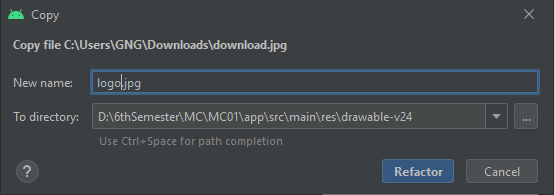
And paste in the res >> drawable folder



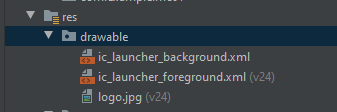
Click on OK



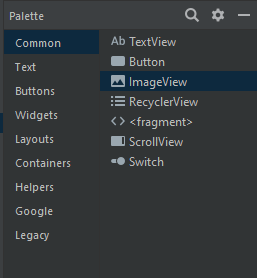
Type name and click on Refactor



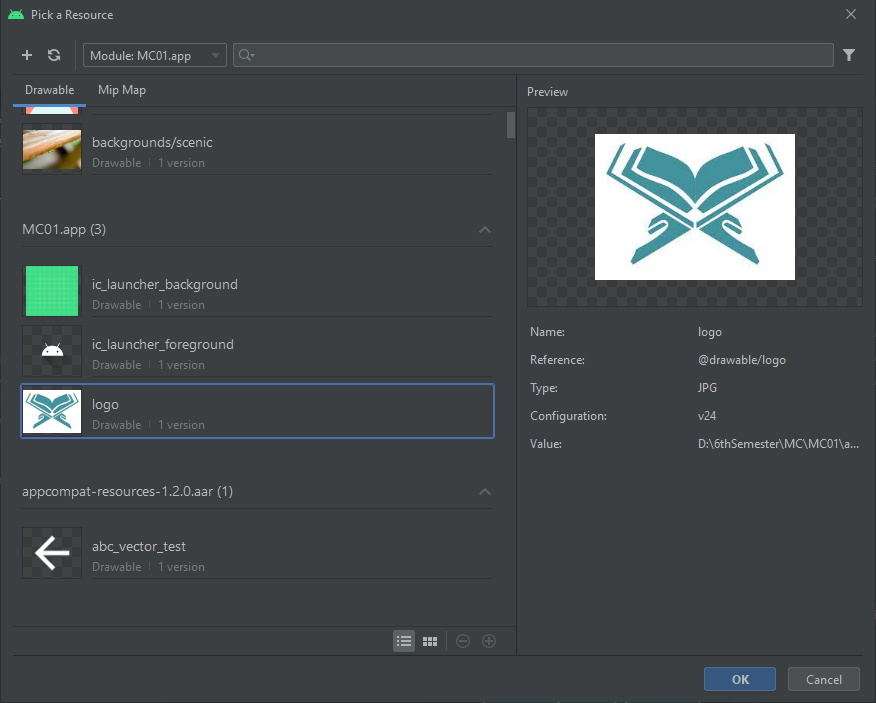
Here you can see logo.png is added in drawable folder

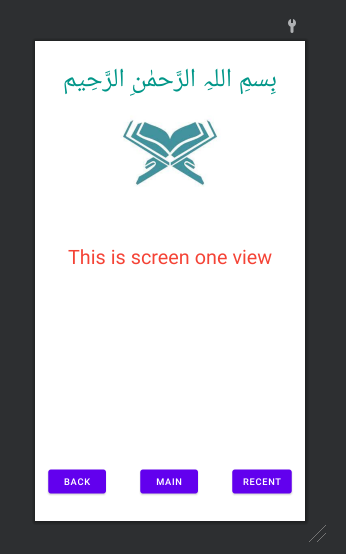


Now add an Image View in design

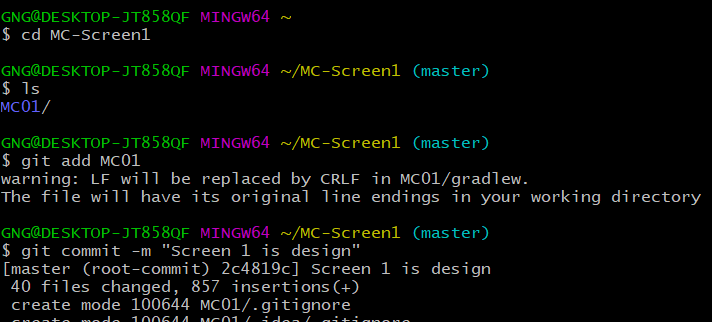


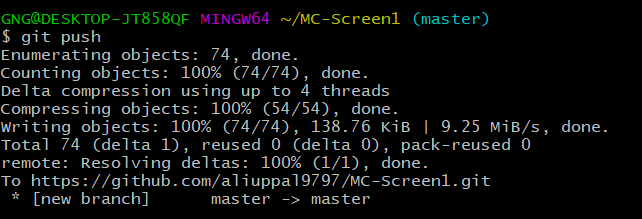
Select your desired image. In my case it is logo. And click on OK



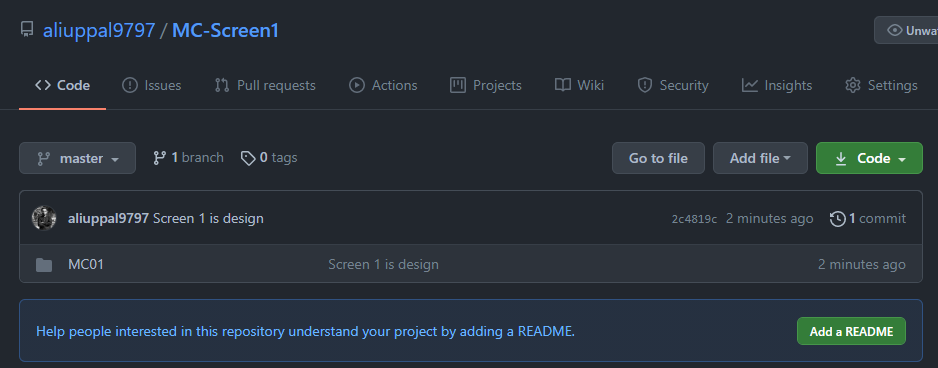
And set the constraints for this image view and text view. And then the final screen is ready now.

Now open git bash and move to MC-Screen1 repository and then add and push Screen 1 project (MC01)



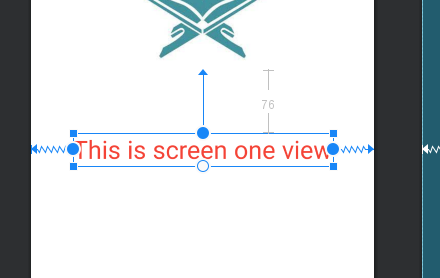


On central repository you can check that screen one is added

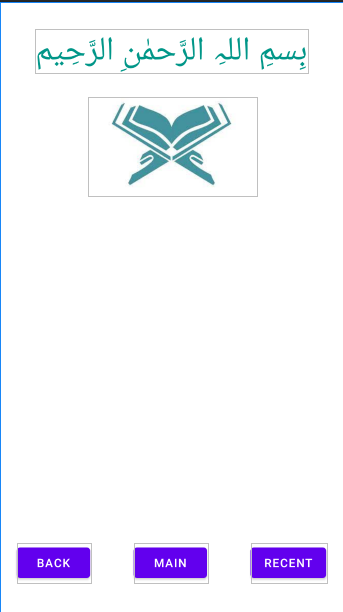


# Screen 2

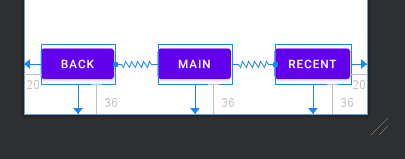
Select this text view and press delete key from key board

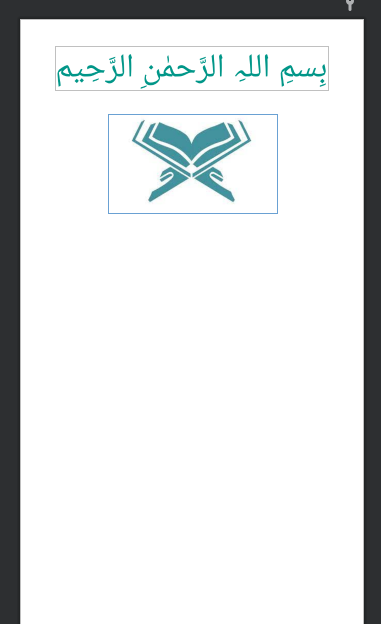


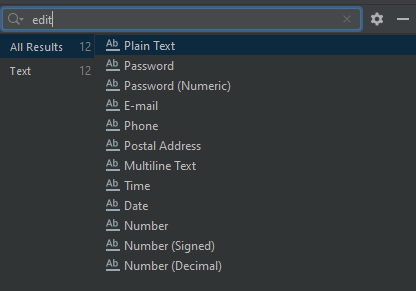
Your screen is updated



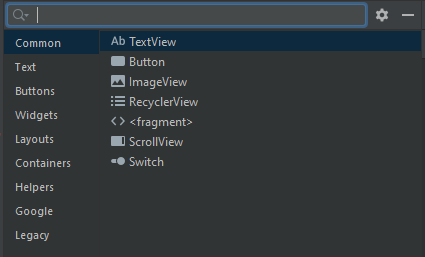
Now remove these buttons too



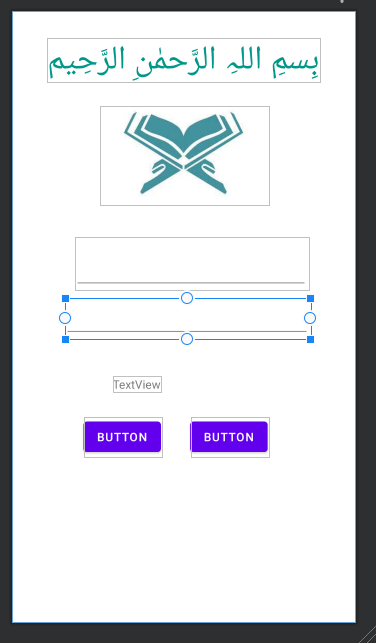
Current layout.  


Now add email and password items in your design from edit.

Now add a text view and two buttons to your design



Our current design will be like this one.

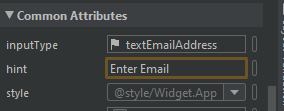


Now we have to set the constraints and size of our items by drag and drop procedure.

Our second screen will look like this. By joining the constraints.



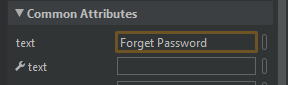
Now select email editor and go in attribute >> Common Attributes >> hint = Enter Email



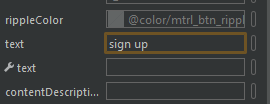
And for password editor follow the same procedure as given above.

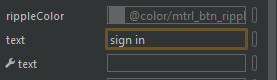


For text editor go in attribute >> Common Attributes >> text

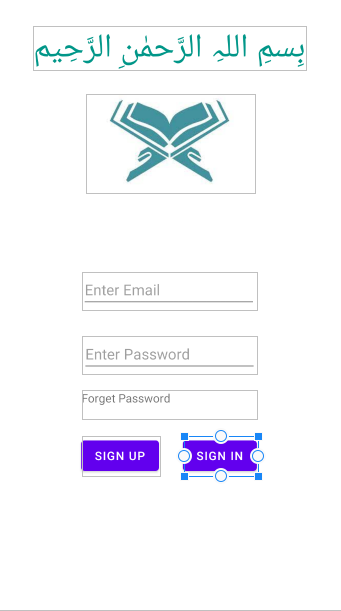


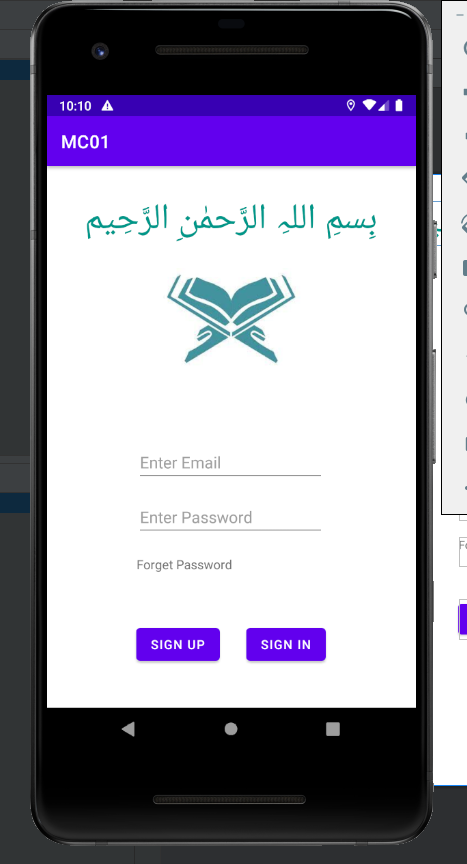
And for buttons, change their text property to Sign up and sign in respectively





Now your second screen is ready now.





Now push your second screen by using these commands

