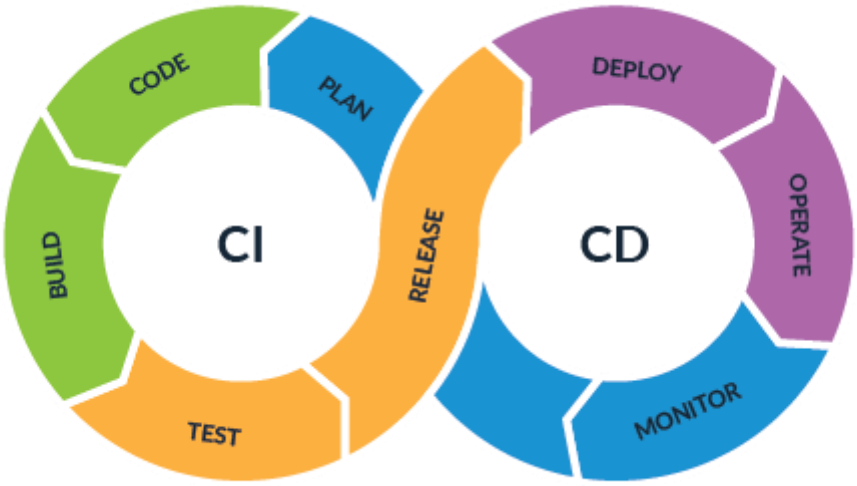
Automate React Native App with CI/CD



By - Rananjaya Chiranthaka Bandara.

Software Engineer.

Contents

Introduction

Mobile app development is the most paid IT industry now adays. Because developers make different kind of mobile applications for the human needs, such as banking apps, social networking apps, medical apps, educational apps, Online conferencing apps etc.

With the covid pandemic most of the people in the world move to the smart devices. That the other reason mobile app development become trading topic. In the development side of those mobile apps software development companies faced different challengers when developing apps with fast deliverable. Because developers need to provide their app to market as soon as possible. In this book ill discussion how to develop react native mobile app and how to integrate the CICD workflow to the mobile app. In the development of software continuous integration and either continuous delivery or continuous deployment are referred to as CI/CD or CICD in software engineering. CI/CD bridges the gap between development and operations activities and teams by mandating automation in application development, testing, and deployment.

Instead of companies releasing new app releases on a monthly or quarterly basis, this can now be done on a weekly, daily, or even multiple times a day basis. This has increased the need for automation in terms of integration, delivery, and deployment, so that apps can be updated without users even noticing or having to intervene in most cases. At most companies, continuous integration combines the work of multiple developers and is usually done at least daily to help detect any bugs early on. Continuous delivery assists you in building your app in such a way that it is ready to be released at any time. Continuous deployment is an automation process that allows changes you make to be deployed instantly through the pipeline.so I this book we'll discuss the way how to use GitHub workflow.

1 Chapter

1.1Environment setup

1. How to Create GitHub account.
2. How to install git for your pc.
3. How to create new git repository.
4. How to write some basic git command.
5. Install react-native CLI and JAVA JDK, Python, Node JS.
6. Setup new React native project.

Chapter 1

1.1 Environment setup

In this book ill discussion about How to instigate CICD to the mobile app development. so, I'll show your practical way to implement that. According to that I'll show you How to create GitHub workflow to your project and how to be automated it. And, ill teach you how to develop some basic mobile application using react native with step-by-step process.

So first of all, I'll show you how to create GitHub account for your project. Simply you need to navigate the https://github.com and click the sign-up button.

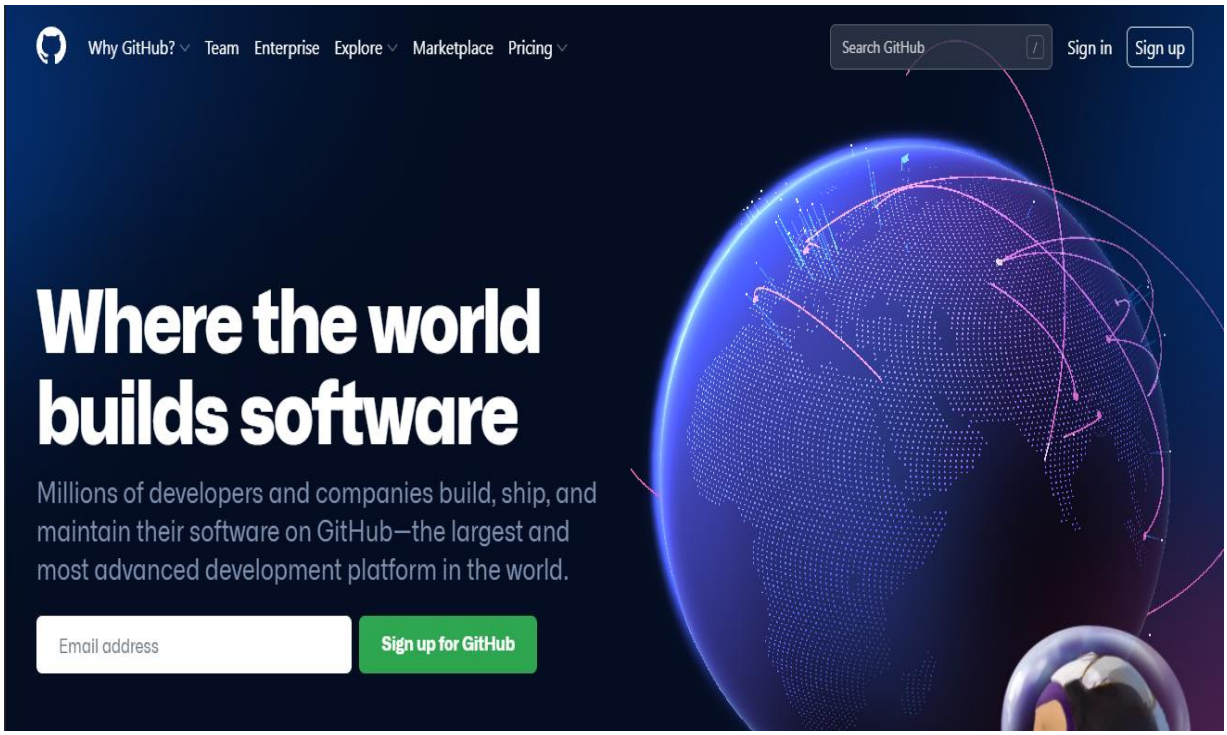


Image 1.1 – Home Page of GitHub

Next you have to give your email and password and click continues and it’ll create you a new GitHub account.

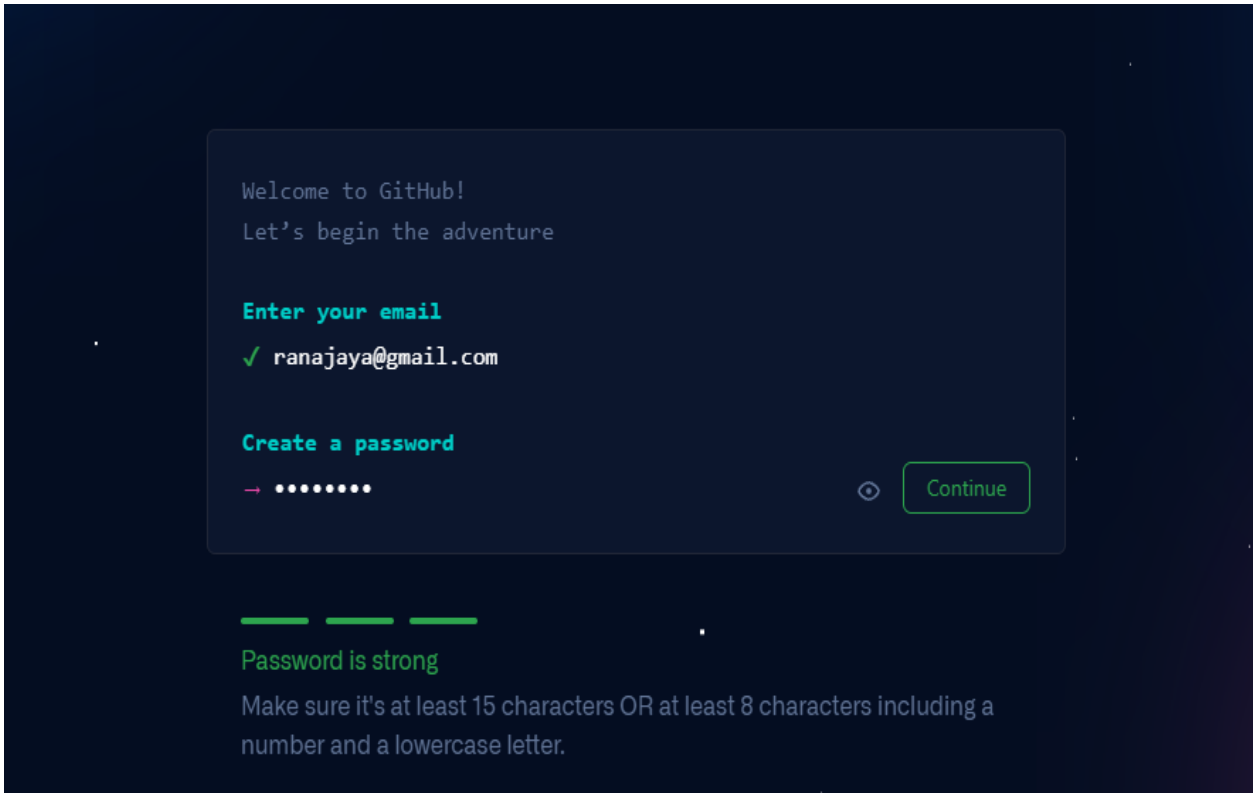


Image 1.2 – Sing-up Page of GitHub

After the create new account with GitHub you ill redirect to this kind of screen.

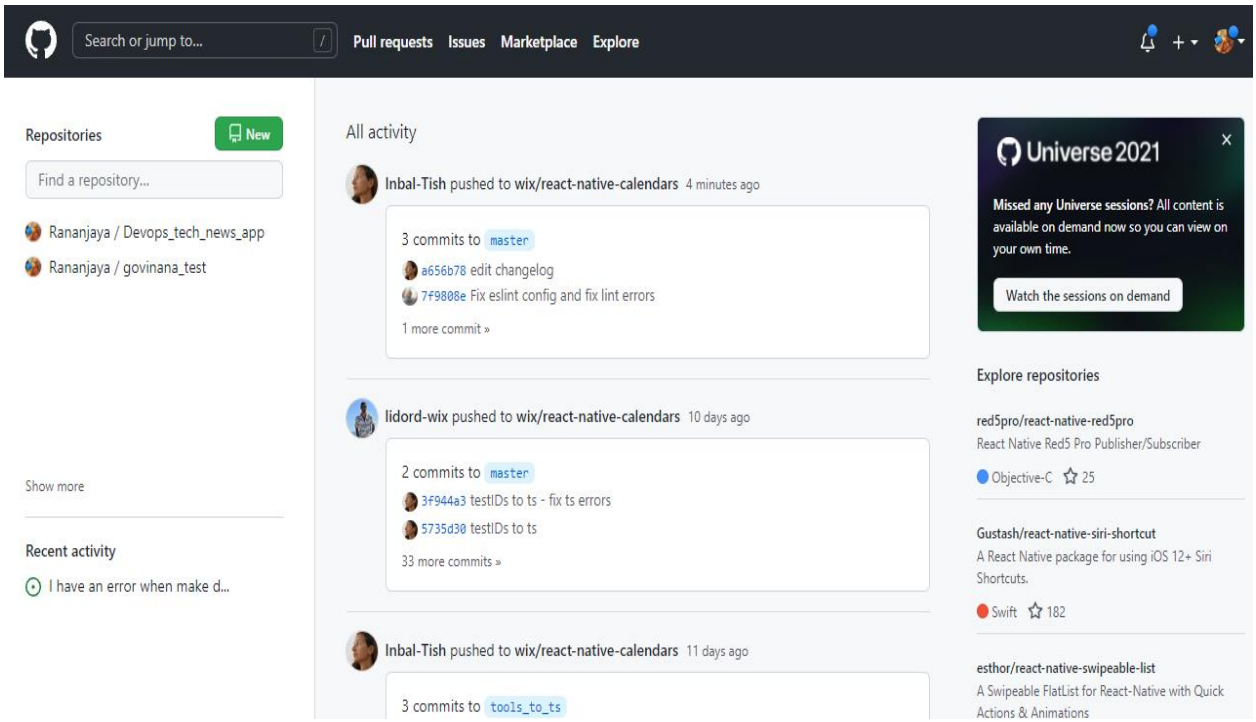


Image 1.3 – GitHub Homepage

Sometime your GitHub home screen different than me. because I have pro version of the GitHub. But it is not issue. So, after the create GitHub account you need to create git repository for our project. For that you need to click new button in top left corner with green color. When you click that button, you ill redirected to this kind of screen.

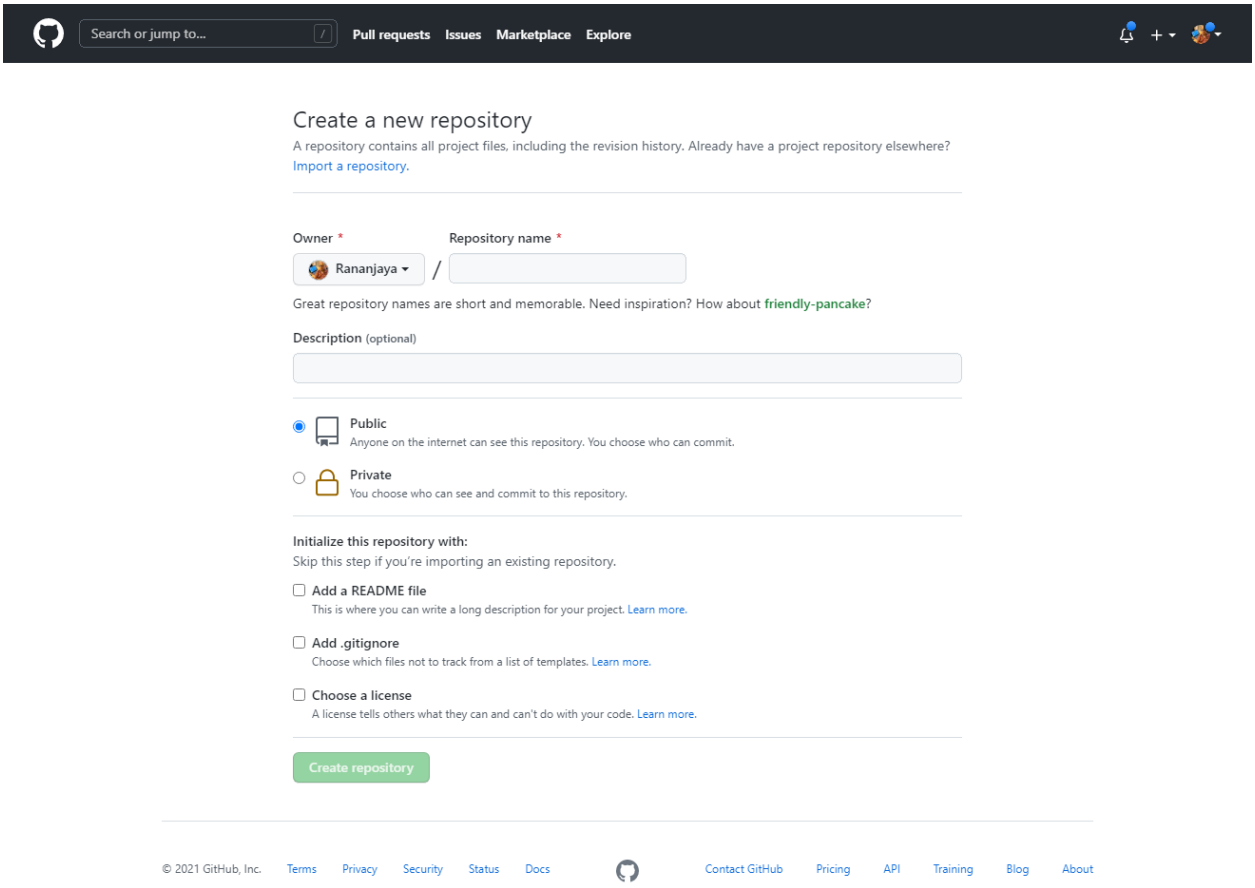


Image 1.4 – GitHub New repository creation

So, in here you need to provide the proper name and description and also you need to decide to make your repository public or private. And also, you can provide some read me file to your repository. After you need to click create repository button in the bottom of the screen and GitHub will create you knew repository.

1.2Local Git installation

So, this project you need to install Git for our windows pc. So that first you need to download git installation package form their official web site. So first you need to visit https://git-scm.com and download the windows installation package. If you have ubuntu or other Linux distribution on your PC you need to install it with this command

sudo apt-get install git

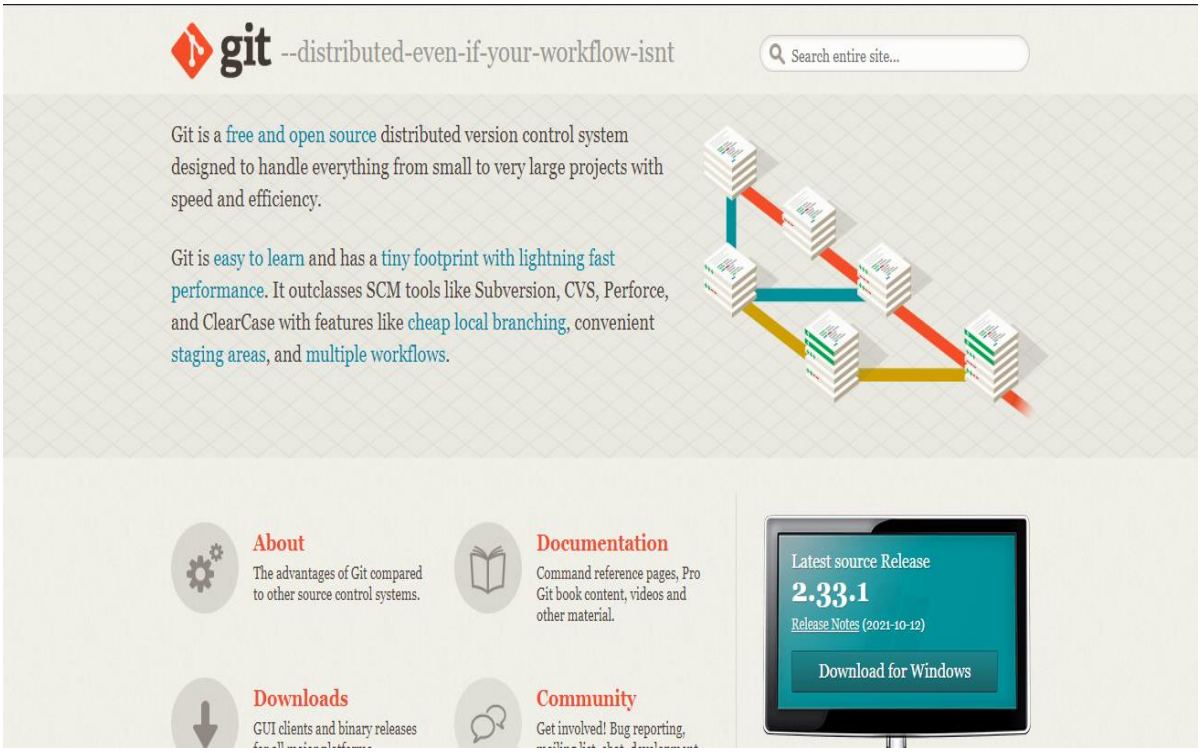


Image 1.5 – https://git-scm.com

After you download the git.exe package for your windows pc you can install it as a normal windows installation package. I will provide you some screenshot of the installation process so you can follow that.

Image 1.6 – Git installation steps 1

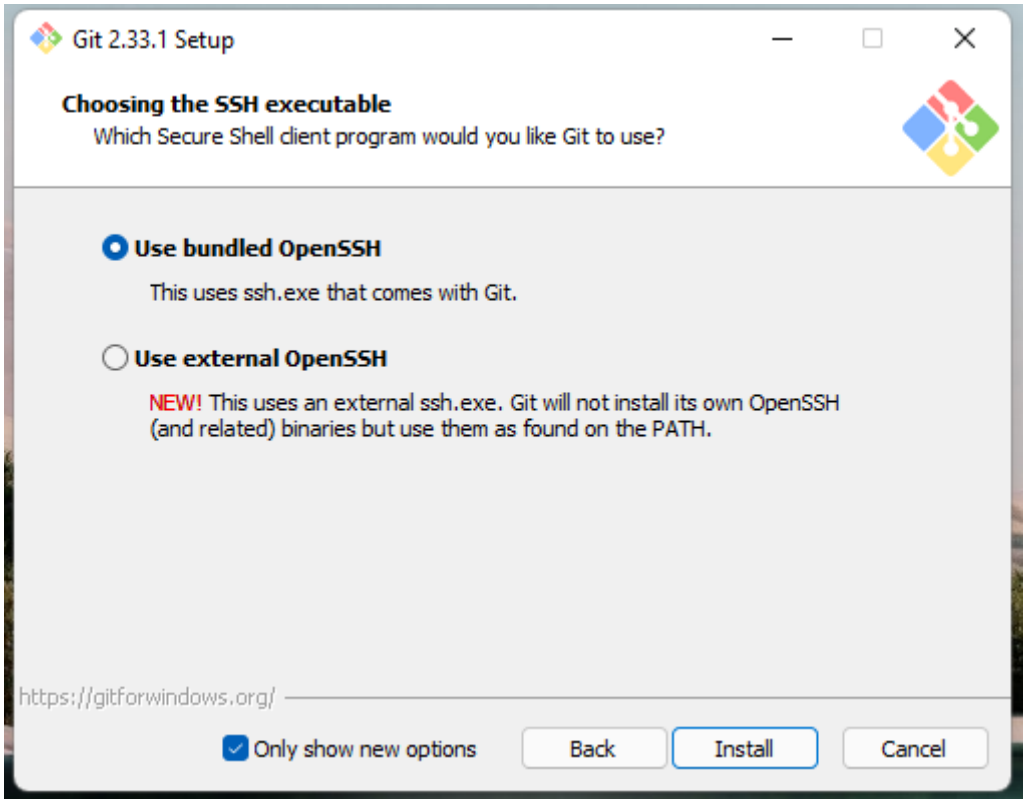


Image 1.7 – Git installation steps 2

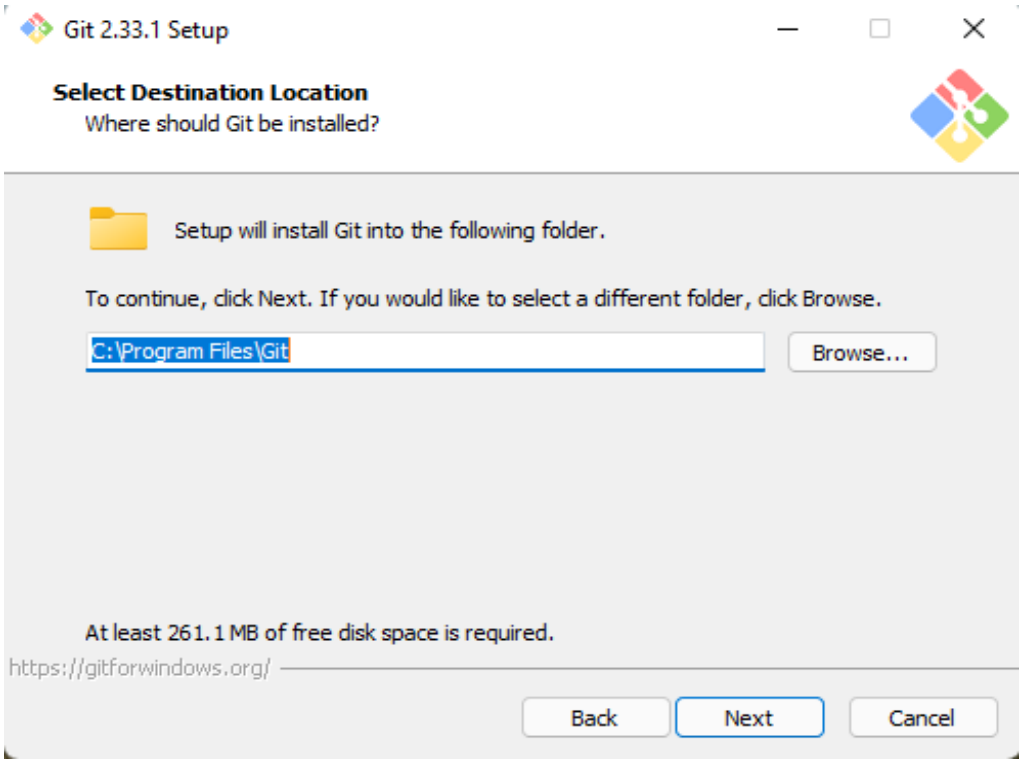


Image 1.7 – Git installation steps 3

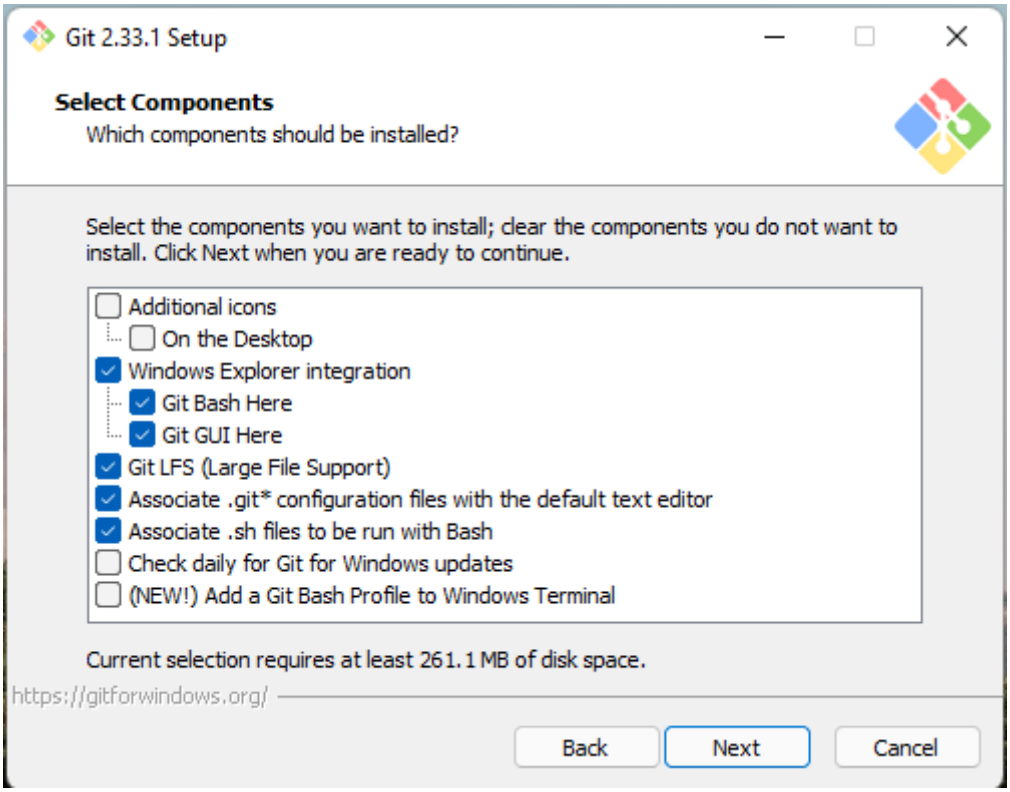


Image 1.8 – Git installation steps 4

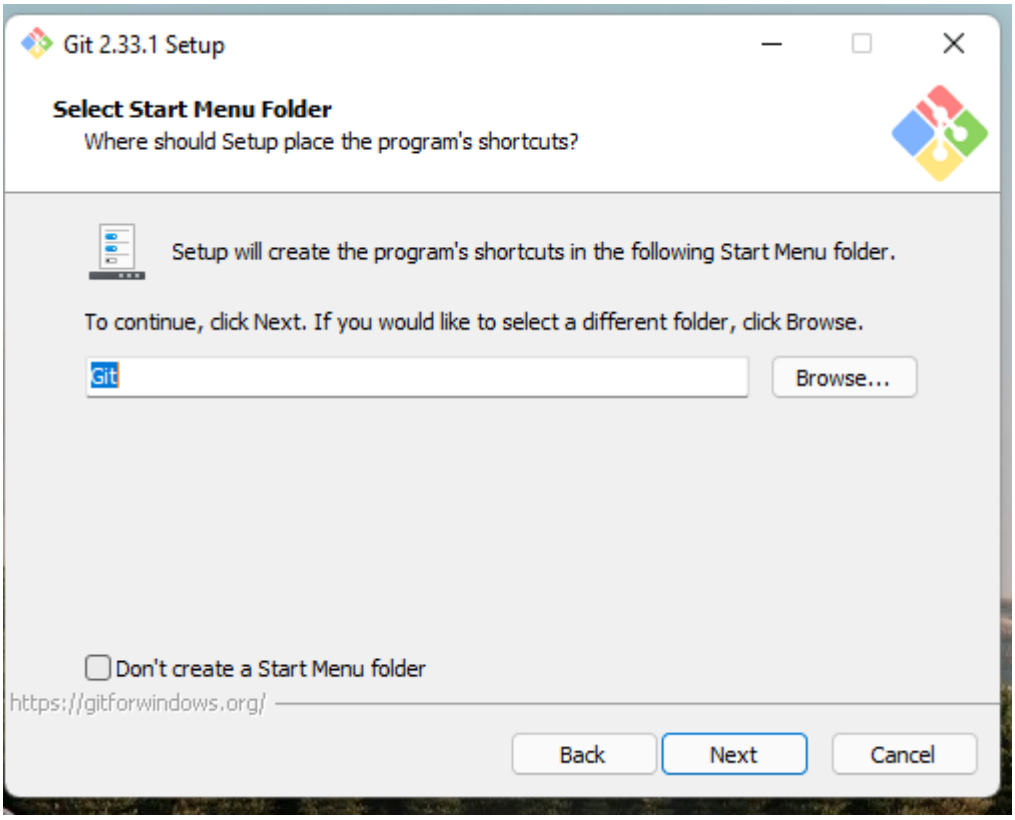


Image 1.9 – Git installation steps 5

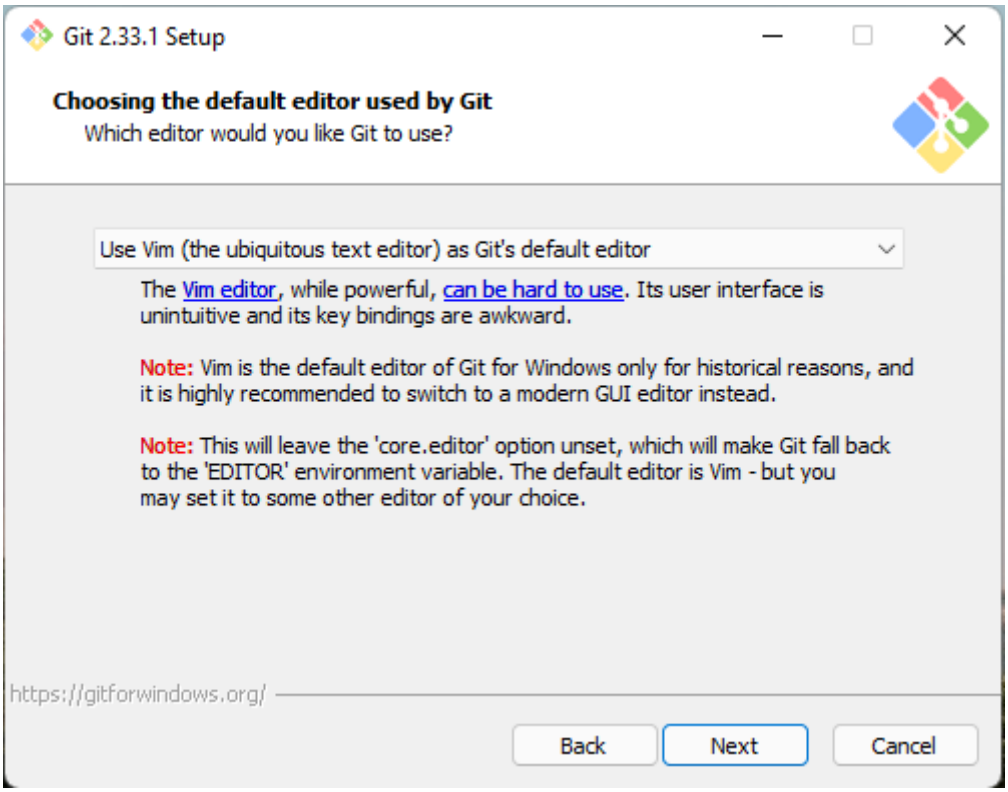


Image 2.0 – Git installation steps 6

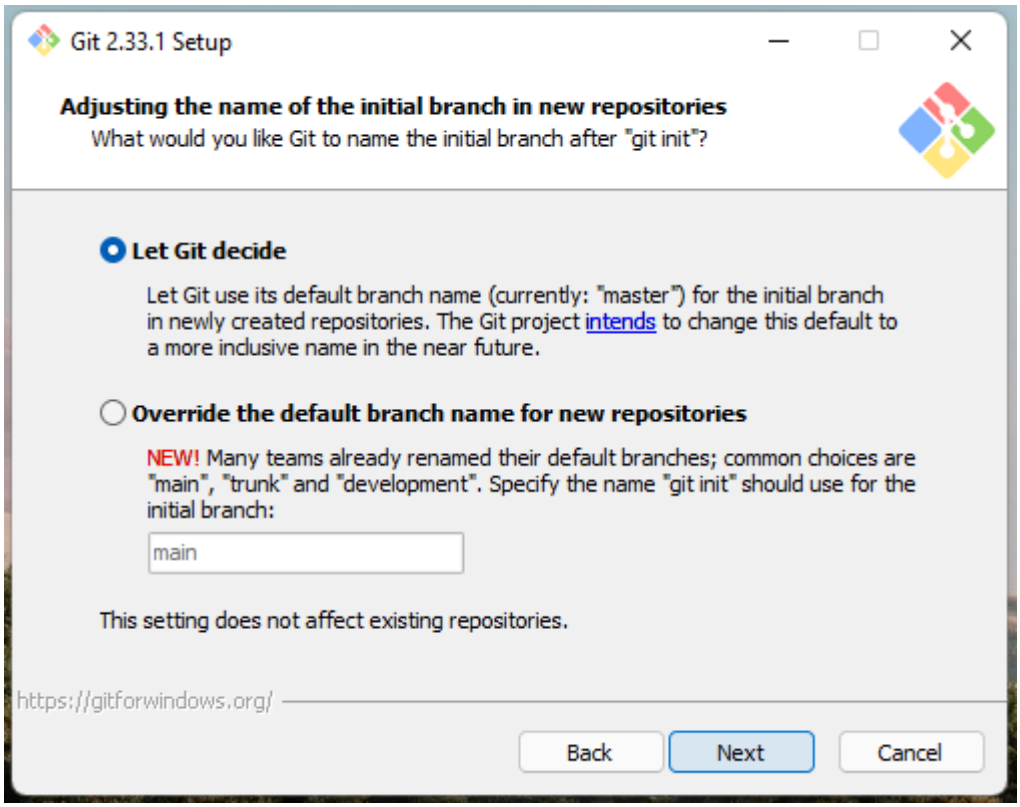


Image 2.1 – Git installation steps 7

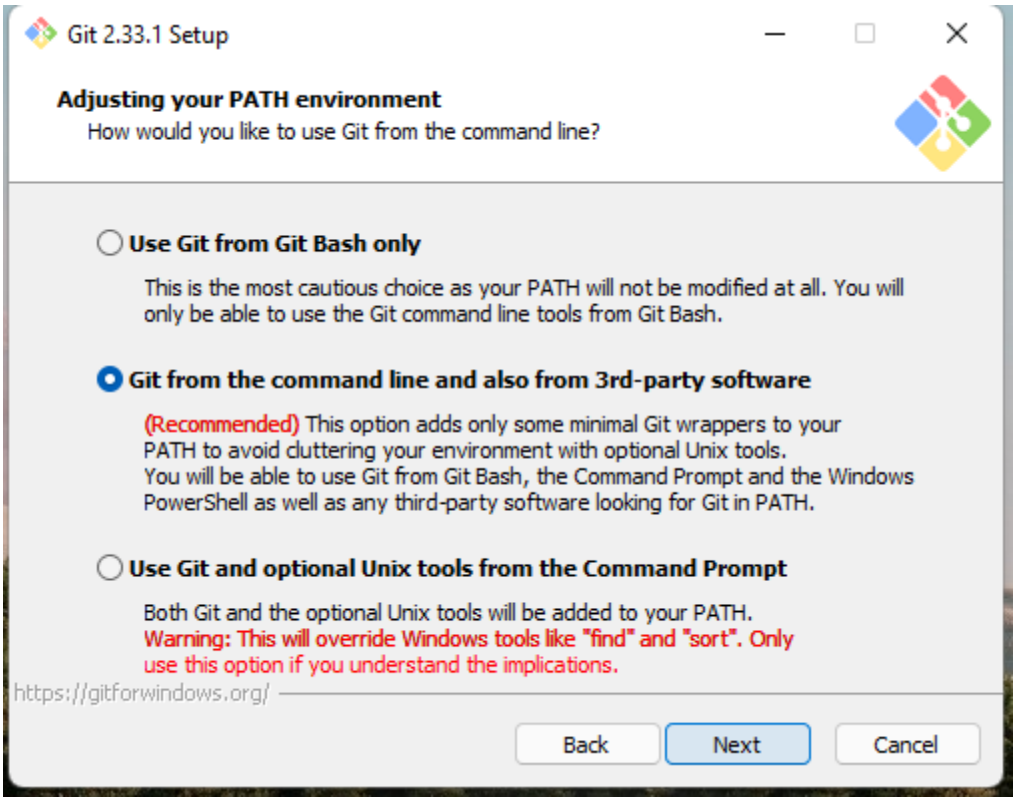


Image 2.2 – Git installation steps 8

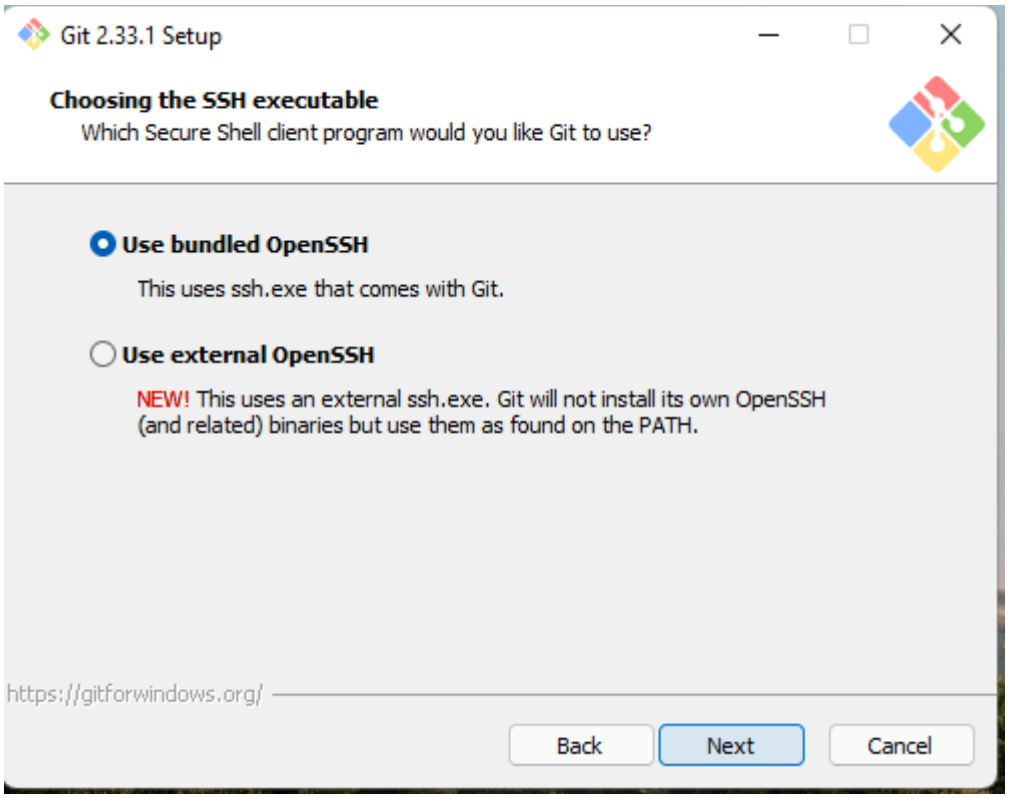


Image 2.3 – Git installation steps 9

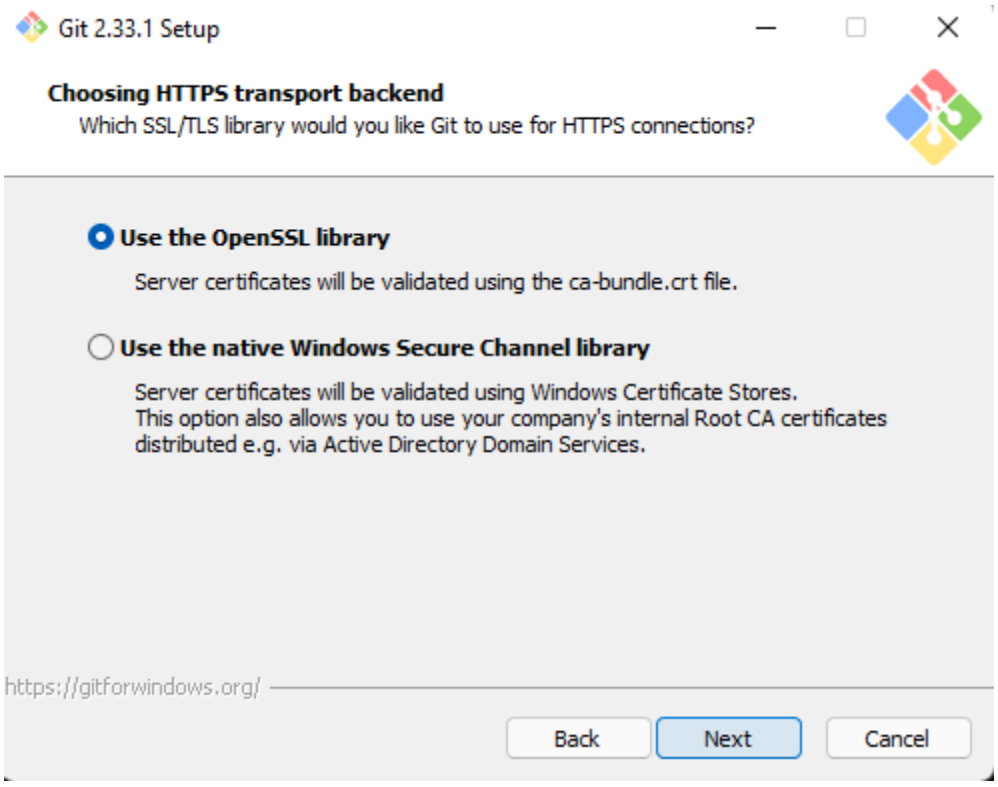


Image 2.4 – Git installation steps 10

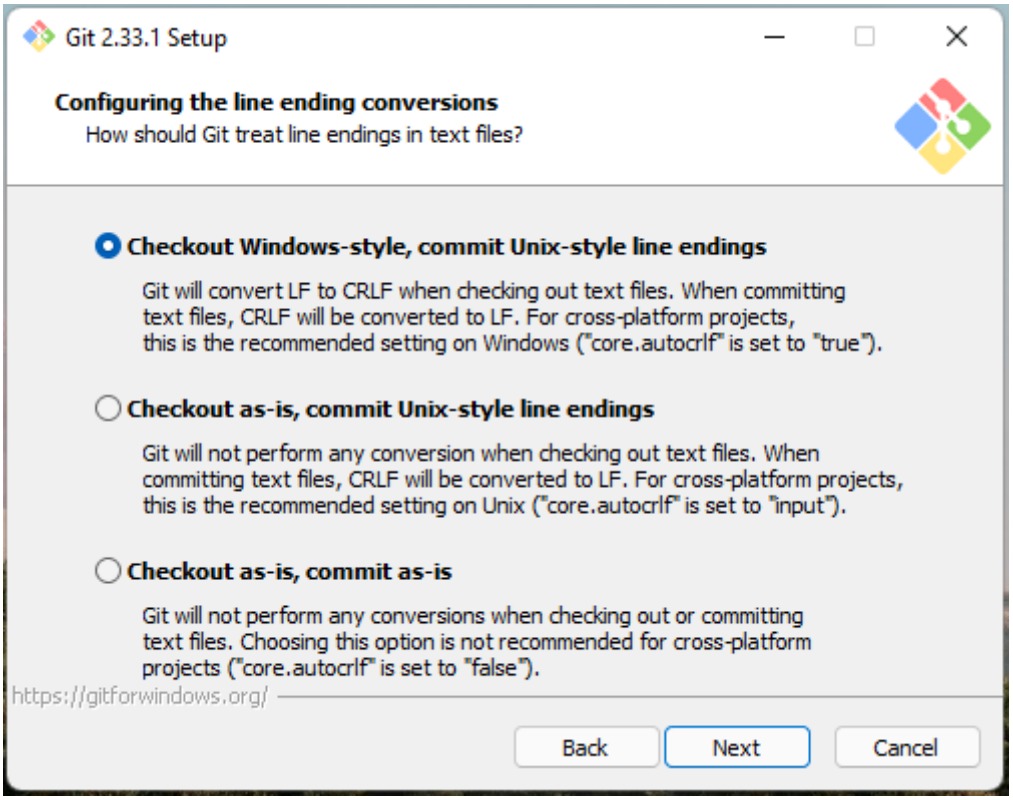


Image 2.5 – Git installation steps 11

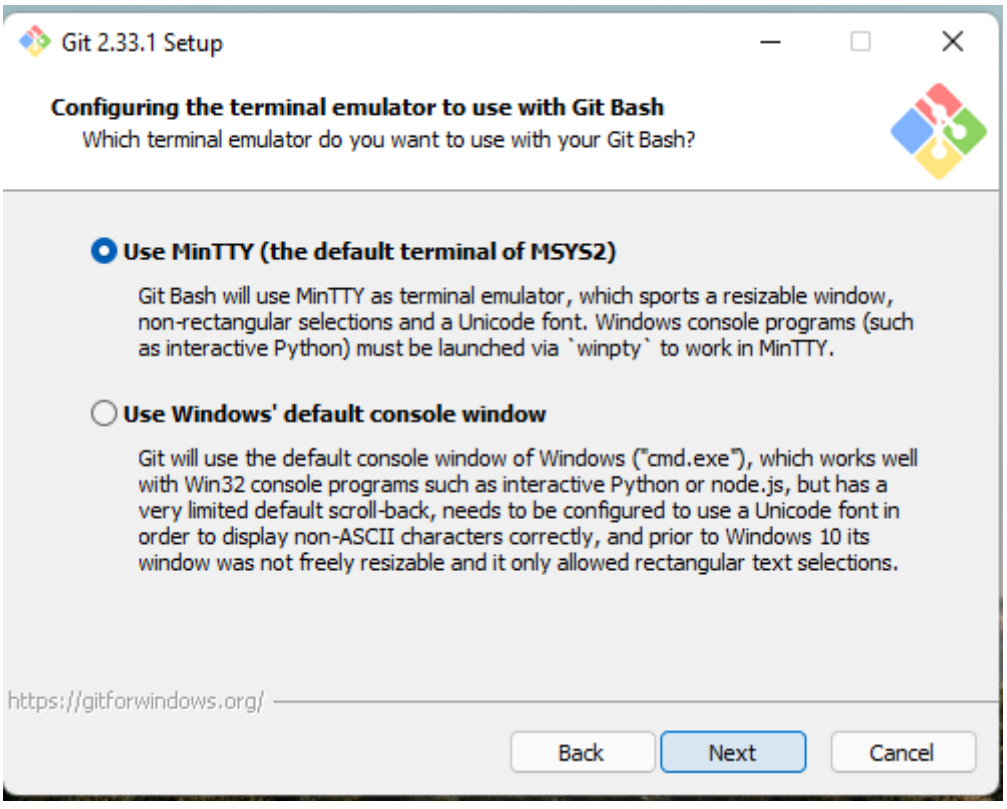


Image 2.6 – Git installation steps 12

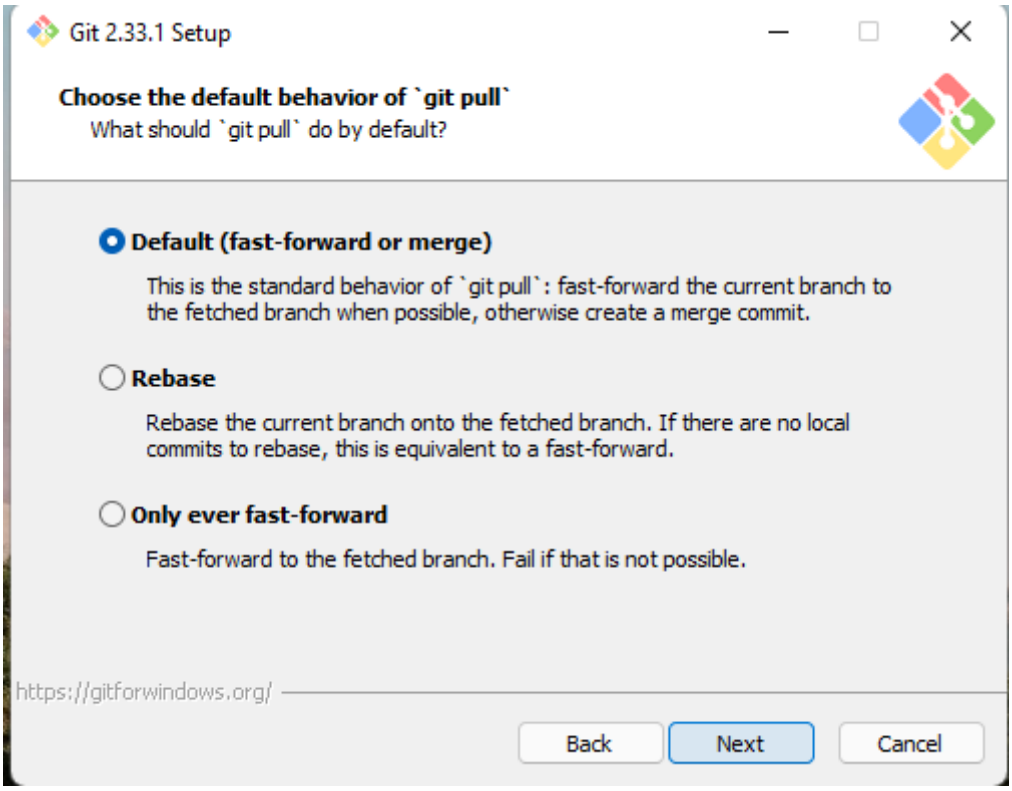


Image 2.7 – Git installation steps 13

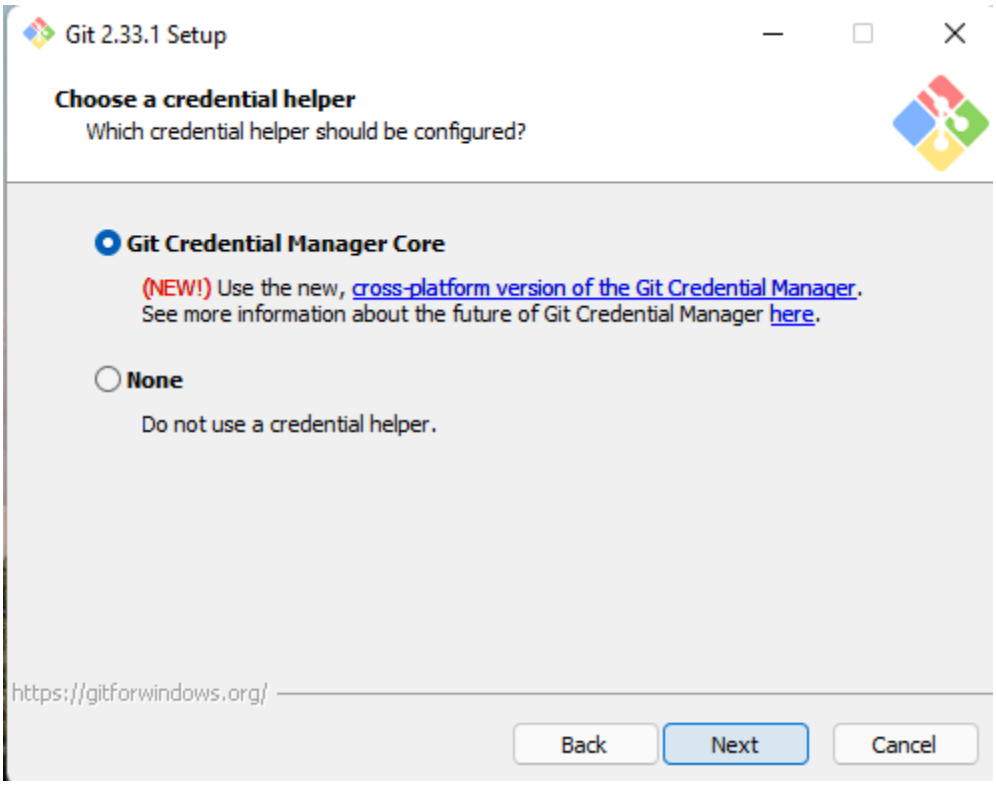


Image 2.8 – Git installation steps 14

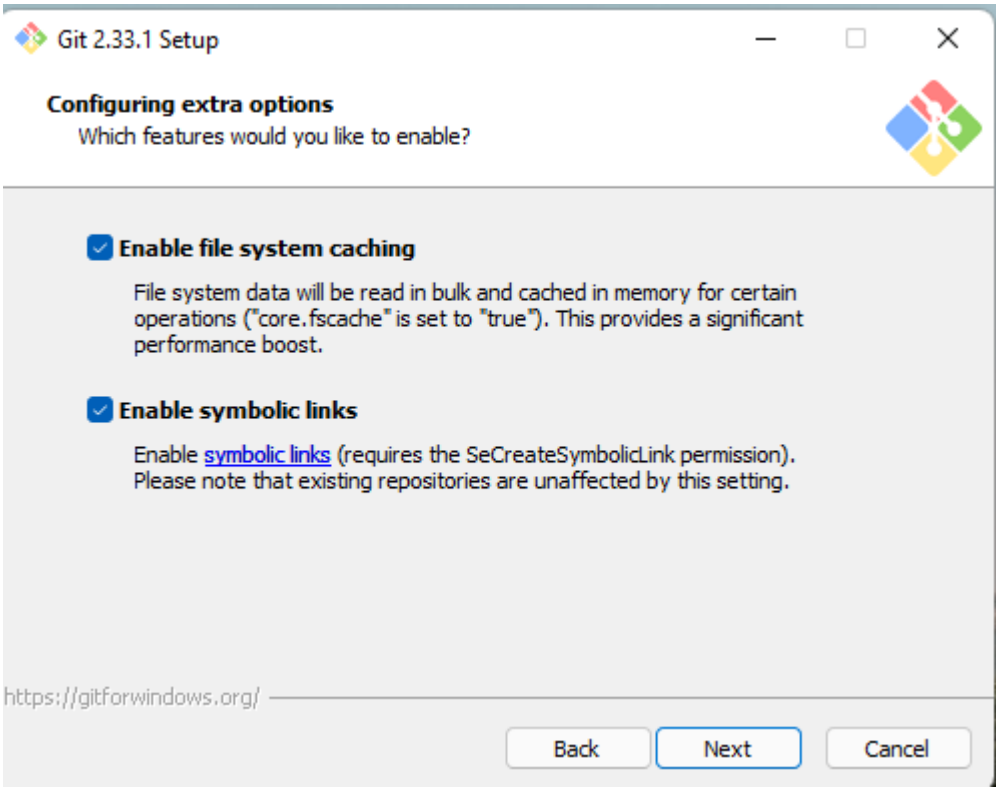


Image 2.9 – Git installation steps 15

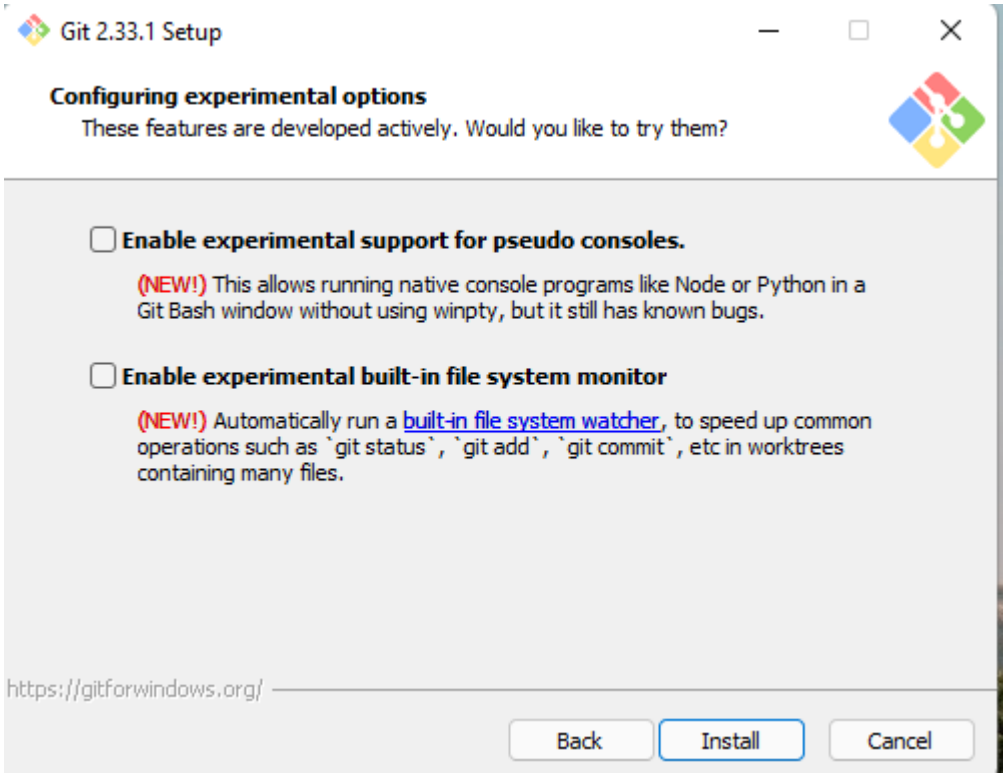


Image 3.0 – Git installation steps 16

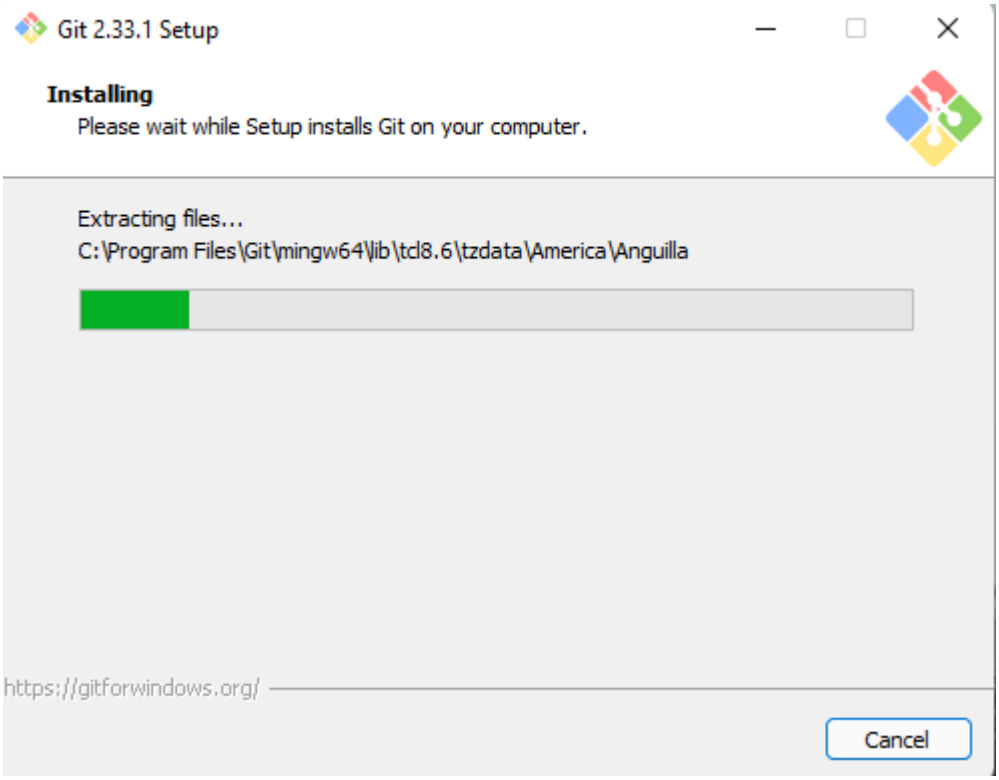


Image 3.1 – Git installation steps 17

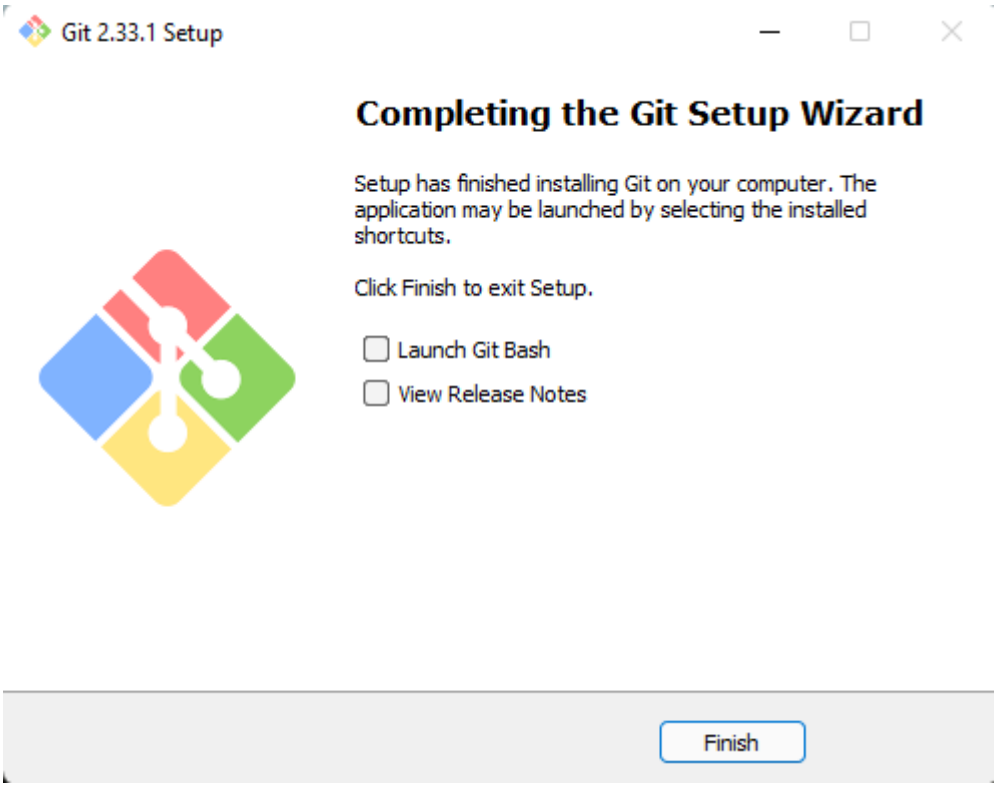


Image 3.2 – Git installation steps 18

- After the follow those step you can install the git for your windows pc successfully.

1.3Git Command

When using git, you need some knowledge in git also and its commands. So, in here ill teach you some git command you need to use this project.

i) Git Clone

Git clone is used to download existing source code from remote git repository like GitHub, bitbucket, GitLab etc. in other word Git clone command makes identically copy of that least version of the project in the remote repository and download or save it into local pc.

e.g. - **git clone <remote repository clone URL you need to clone>**

for example, if you need to download repository form GitHub you fallow it like that.

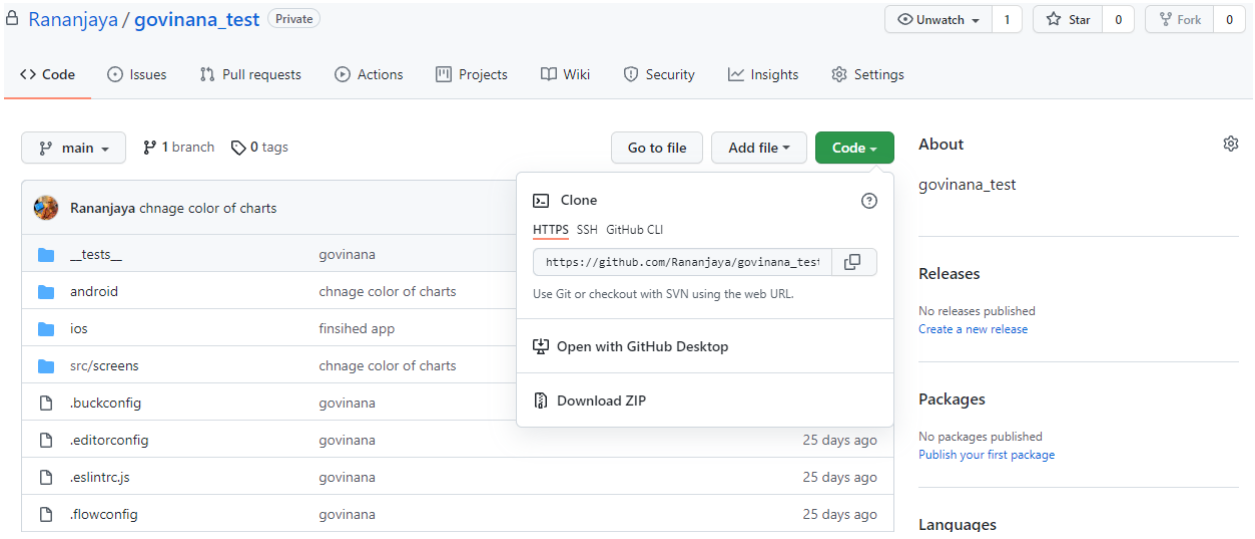


Image 3.3 – GitHub Govinena project Clone example.

So, you can write you git command like this-

git clone https://github.com/Rananjaya/govinana_test.git so, this command makes a copy of the repository and download it into your PC

ii) Git branch

Git branch command is most import command in git world. When we develop big software project several developers working in different branchers in same repository. We can use git branch command for creating, listing, and deleting branches.

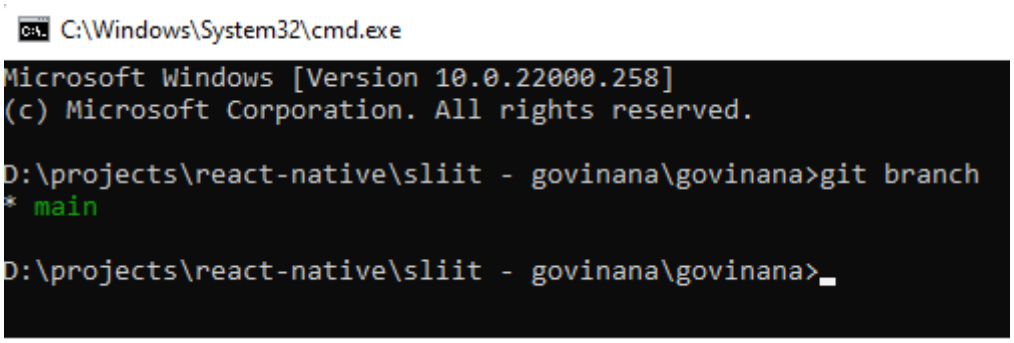


Image 3.4 – Git branch command example.

iii) Git Push

Git push command is also another most important command. This command is used to push the local repository to remote repository branch. You can use git push command like this.

Git push origin main

In here main is our local branch name and we will push to the remote repository. When this time git checks this repository exist or not in the git remote repository and if it not available in remote repository git ill create it for us.

iv) Git checkout

Git checkout is the most use command. To work with branch first you need to switch it. We use git checkout for mostly switch one branch to another.

git checkout <Your branch name>

v) Git status

Git status command is very useful command, and it is mostly used to see the information about current branch.

vi) Git branch

Git branch command also use for –

- Check current branch is updated or not.
- Check if the anything to remain to push or commit.
- Check there are file staged, upstaged, or untracked.

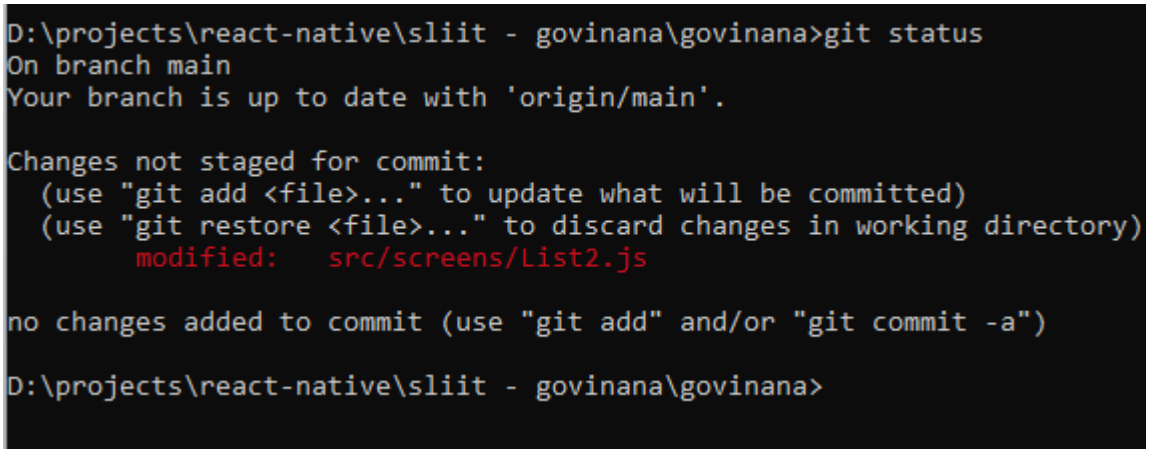


Image 3.5 – Git status command example.

vii)Git add

git add is another useful command. When we create, delete or modify the file those changers happen in our local and won't include our next commit. We use git add command to add those changers. We can use it like that.

git add.

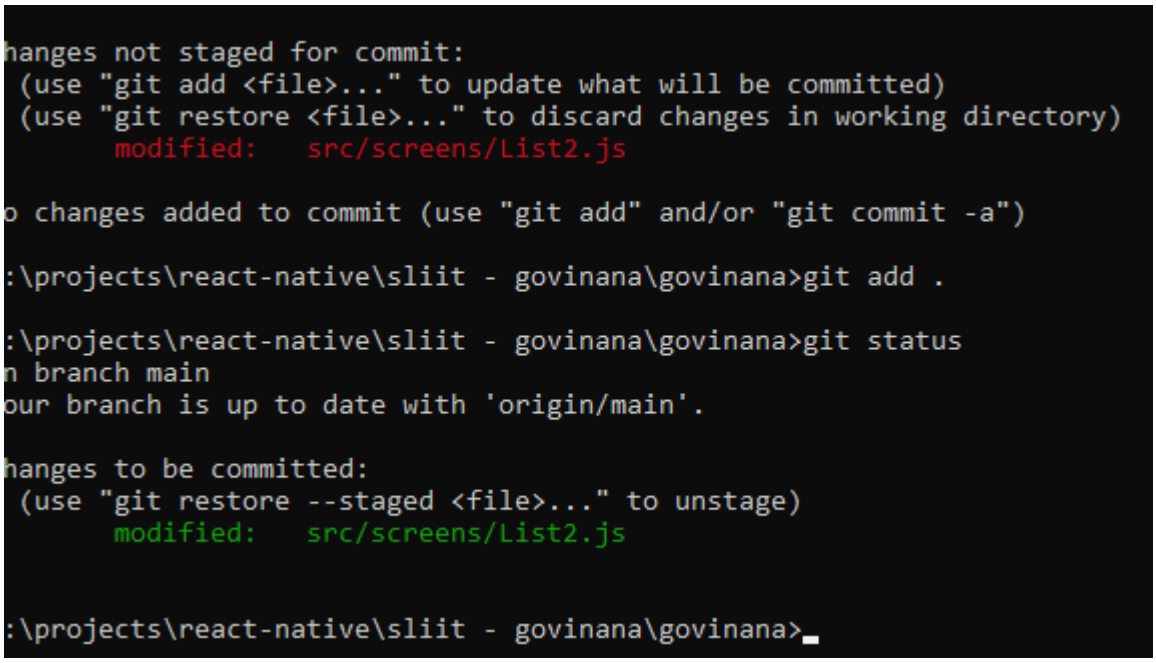


Image 3.6 – git add example

viii) Git commit

This is the most used command of Git. Because every git project we need to commit it finally. Once we reach the certain point of the development, we want to save our changers. Git commit is just like checkpoint. If you have any issue happened in your code, you can go back with certain commit. We also need to write some short message to identify the commit – it is just like brief description. And git commit only save your commits locally unless you push the code to the repository.

git commit -m "Your commit message"

```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/screens/List2.js


D:\projects\react-native\sliit - govinana\govinana>git commit -m "Do some styling"
[main f61bc72] Do some styling
 1 file changed, 1 insertion(+), 1 deletion(-)

D:\projects\react-native\sliit - govinana\govinana>
```

Image 3.7 – git commit example

ix) Git push

After the commit your changers, you need to send it into remote repository. So, git push command does short of that thing. You can writer git push command like that –

git push origin <Your Branch Name to push>

```
:\projects\react-native\sliit - govinana\govinana>git branch
 main

:\projects\react-native\sliit - govinana\govinana>git push origin main
numerating objects: 9, done.
ounting objects: 100% (9/9), done.
elta compression using up to 8 threads
ompressing objects: 100% (4/4), done.
riting objects: 100% (5/5), 410 bytes | 410.00 KiB/s, done.
otal 5 (delta 3), reused 0 (delta 0), pack-reused 0
emote: Resolving deltas: 100% (3/3), completed with 3 local objects.
o https://github.com/Rananjaya/govinana_test.git
   ec5fa66..f61bc72  main -> main

:\projects\react-native\sliit - govinana\govinana>
```

Image 3.8 – git push example

x) Git pull

This command not going use in this project, but I'll show you how to use this. Git pull is another important command to learn. Git pull is used to get the update from the remote server. This command is made by the combination of the git fetch and git marge commands. When we use git pull it is get updated from the remote repository like git fetch and it will immediately add changers to your local. You can use git pull command like this.

git pull <remote repository name>

when we pull data to local repository sometimes it will make some conflicts with our local repository. Sometimes it is automatically merges the branchers without any issues.

xi) Git merge

Git marge is another command we use for marge the branchers together. When we development some project there are multiple developer working parallelly with different branchers in same repository some in the final step of the project we need to merge that one branch to our main master branch. So, this situation we use git marge commend to marge our branchers. You can use git branch command like that.

Git marge <Another remote branch name (of same repository)>

Ok done. Now we discussed most impotent git commands here. Now you can fallow those tutorials with no issue.

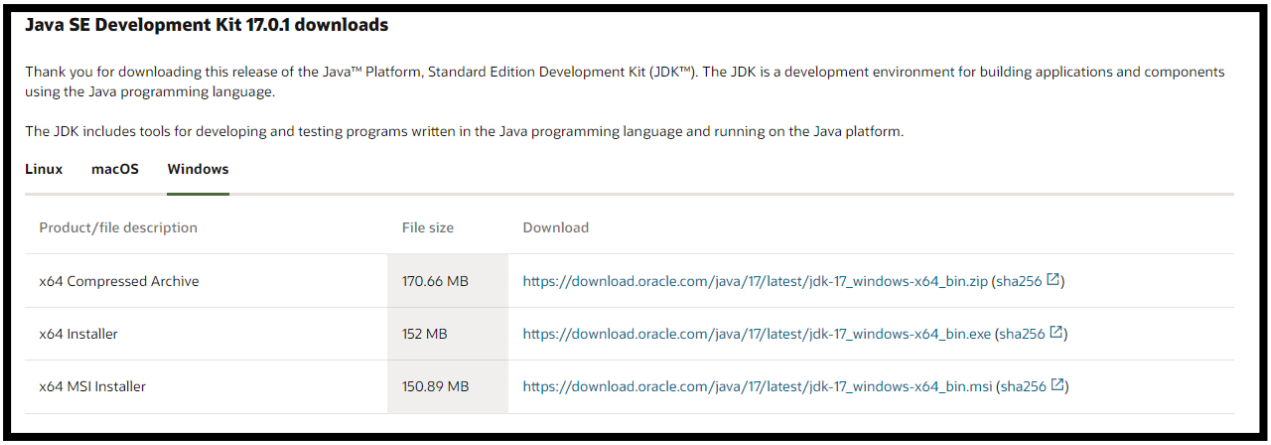
1.4 Install react-native CLI and JAVA JDK, Python, Node JS.

In this book main objective is teach you how to automate react native app with CICD. This section ill explains how to developers some basic react native app. So that I'll show you how to install react native CLI. And also react native CLI required the JAVA. So, I'll show you how to install java JDK step by step also here.

When you install react native CLI first you need to go the react native DOC and read it carefully. You can find the react native official documentation here - https://reactnative.dev/docs/environment-setup. You can install react native CLI to your PC using this command –

npm install –g react-native-cli

We can't run react native on pc without JAVA. so, you need to install java to your pc. First you need to download java JDK. so, this project is using java 8. And you also need to download Payton. Because sometimes react native uses some Payton based libraires. And also, you need to install git to your local computer. I'll explain step by step next. As a first step you need to download java JDK from this link - https://www.oracle.com/java/technologies/downloads/ and download the JDK.exe file and run the JAVA setup. After the install you need to download Payton from this link and install.



Java SE Development Kit 17.0.1 downloads

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications and components using the Java programming language.

The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.

Linux macOS Windows

Product/file description	File size	Download
x64 Compressed Archive	170.66 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.zip (sha256)
x64 Installer	152 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.exe (sha256)
x64 MSI Installer	150.89 MB	https://download.oracle.com/java/17/latest/jdk-17_windows-x64_bin.msi (sha256)

Image 3.9 – JAVA JDK Download

https://www.python.org/downloads

Image 4.0 – Payton Download

and you also need node JS installed on the pc. So simply go to this link and download and install it – https://nodejs.org/en/

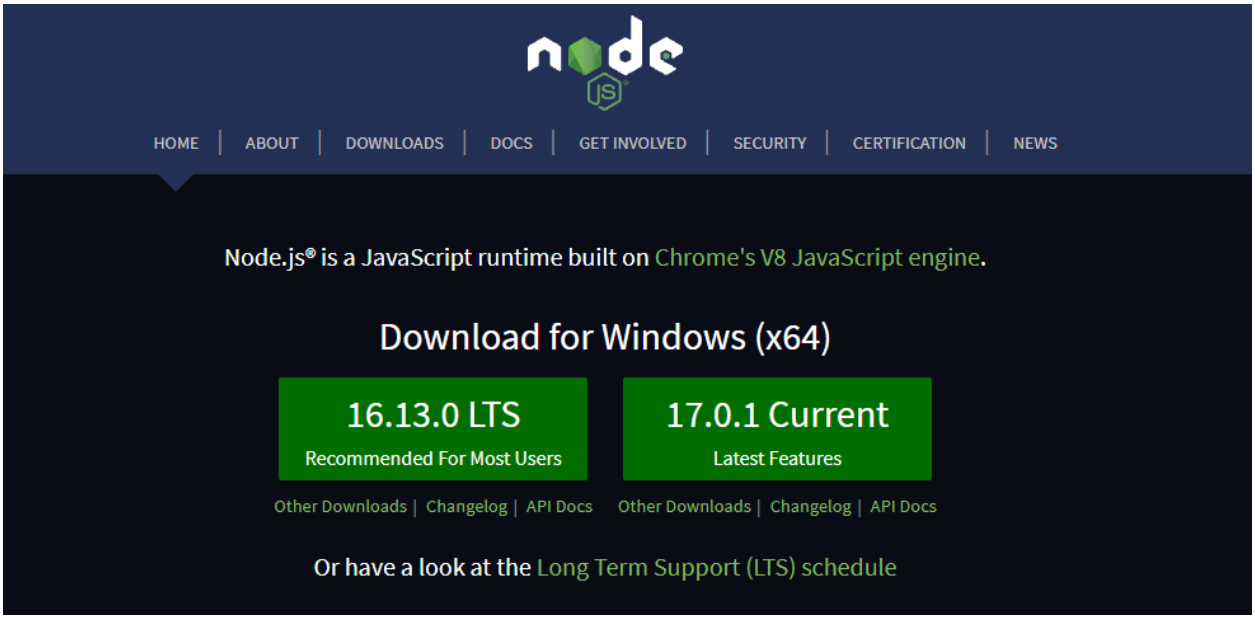


Image 4.1 – Node JS Download.

We use node JS because react native use it and, we need to develop our API with node express that's why we install node JS.

1.5Setup new React native project.

After the installation of the react native CLI and other dependencies, you need to create a new react native project. So simply go to your drive and create a new folder for the project and open the terminal in that location and run this command below.

react-native init Devops_tech_news_app

In my case I'm create react native project and named it as **Devops_tech_news_app**. After you create the new react native project you will see this kind of file structure. Ill describe about this file structure in next step.

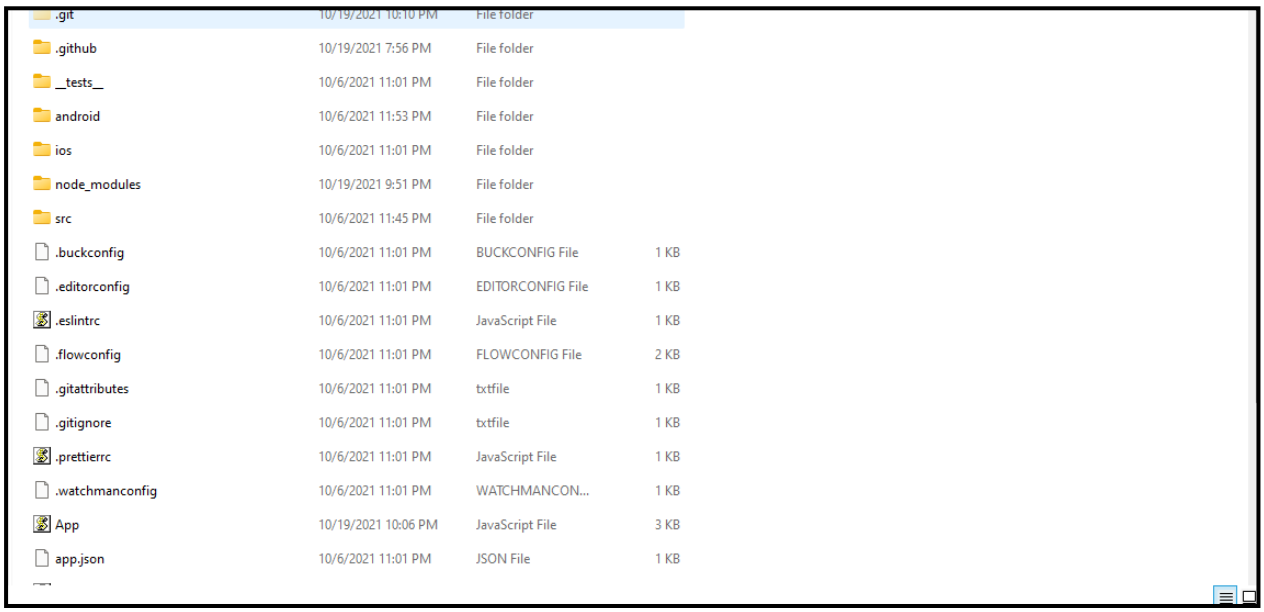


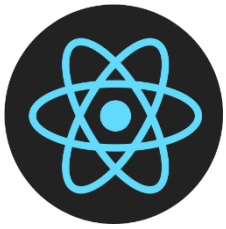
Image 4.2 react native default file structure

Discussion about file structure of react native you I'll see they create many files for us. Actually, this is dummy project react native created for us. According to the above screen you will see them. git and GitHub folder. You can ignore this folder for this time. Because ill describe it when we talk about CICD workflow. In here most impertinent folder is **android** folder. Its conations all configuration of our mobile app. It had **gradle file**, **main java files** and our **AndroidManifest** file. And **src** folder we can locate our all screens and assets.

And you will see another file called **IOS**. React native is Hybrid mobile app development formwork so this IOS folder is used to make IOS configuration with XCode. And node modules folder is containing our **npm** libraries that are helping to run this mobile app.

Chapter 2

Develop react-native Mobile app.



2 Chapter

2.1Development of React native App.

Now you successfully configure your PC to run react native project so let's begin to develop react native project. In this book I'll develop some news app using react native, so I'll use some advances react native components to this project. Don't worry I'll teach you everything step by step in here.

So now we are going to start development. As a first step you need to open your project on any code editor. I 'm using my favorite code editor vscode. You can use vs code or any other code editor for this. You can download VS code form this link – https://code.visualstudio.com

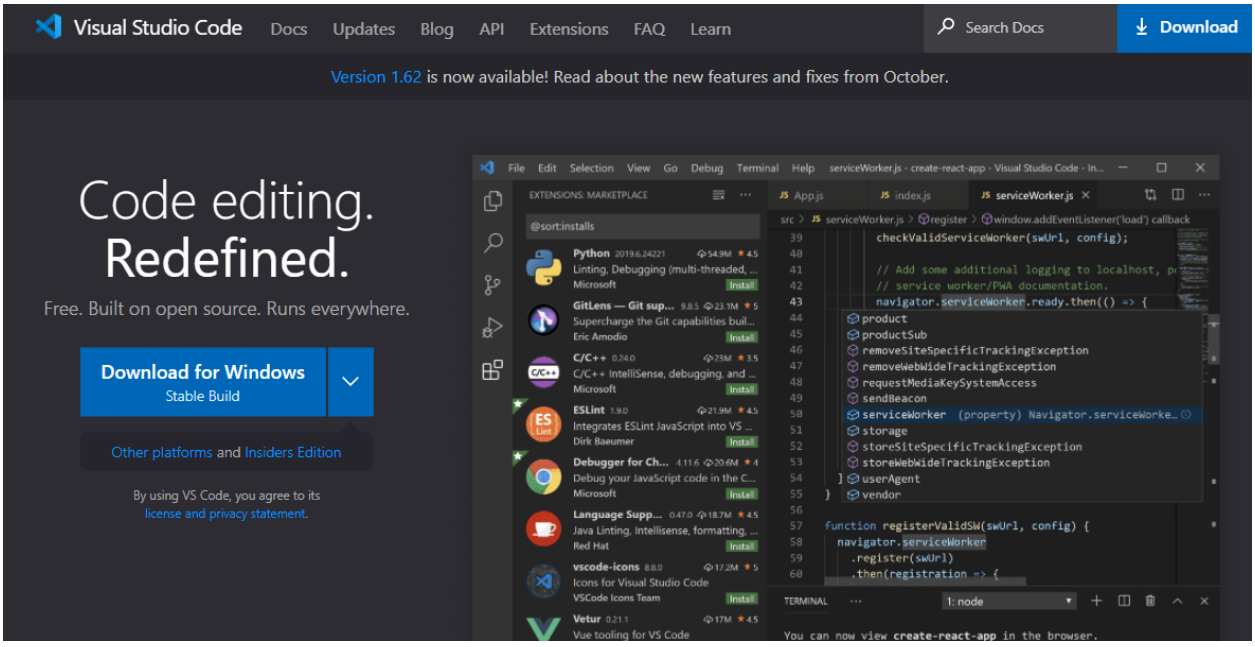


Image 4.3 visual studio code editor download

As a first step I'm going to install react navigation stack and other dependencies. We use react native navigation stack for navigating between screens in android and ios app. You can install react native navigation using this npm command. If you have yarn installed in your pc you can also install this using yarn commands also.

npm install @react-navigation/native @react-navigation/native-stack

You can find this command in react native official documentation here - https://reactnative.dev/docs/navigation

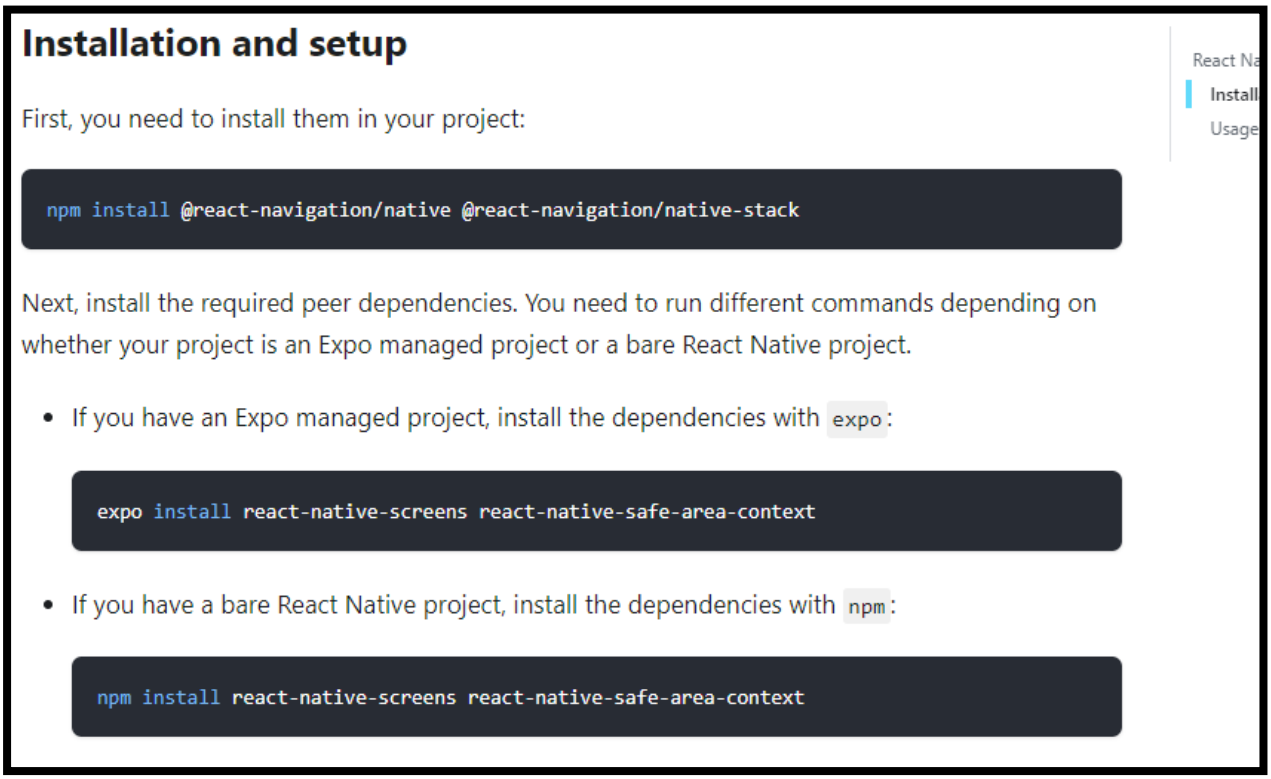


Image 4.4 - Setup react native Navigation

And you need to install react native screens and react-native-safe-area-context for the project.

npm install react-native-screens react-native-safe-area-context

You can confirm your installed libraires by checking pakage.json file located in our project root directory. 4.4 image show how it look like.

Image 4.5 – Installed npm libraires in package.json

After the installing those dependencies, you need to implement those react native navigation to your project. So simply go into your project and select app.js file. App.js is the root component of the react native project. Whatever happens it will be called first. After the navigate to the app JS in react native project remove some dummy code react naive already generated and please the is code below. I'll explain each one after that.

```
import * as React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createNativeStackNavigator } from '@react-navigation/native-stack';
import Home from './src/screens/Home';

const Stack = createNativeStackNavigator();

const MyStack = () => {
return (
```

```
        <NavigationContainer>
        <Stack.Navigator screenOptions={{ headerShown: false }}>
<Stack.Screen name="Home" component={Home} />
</Stack.Navigator>
</NavigationContainer>
);

};

export default MyStack
```

In this code you will see a different structure. If you knew to the react and JavaScript this, I'll heard to first time. But it is not a big deal. In this code the first part is the import part. We need to import our dependencies. In this case we need react, react navigation stack and react navigation container. So, we imported it first. After that we need to create a stack object from react navigation.

Const my stack is the functional component. In react native and react we have both options class components and functional components. Here I'm using functional components because I need to use react hooks in this project.

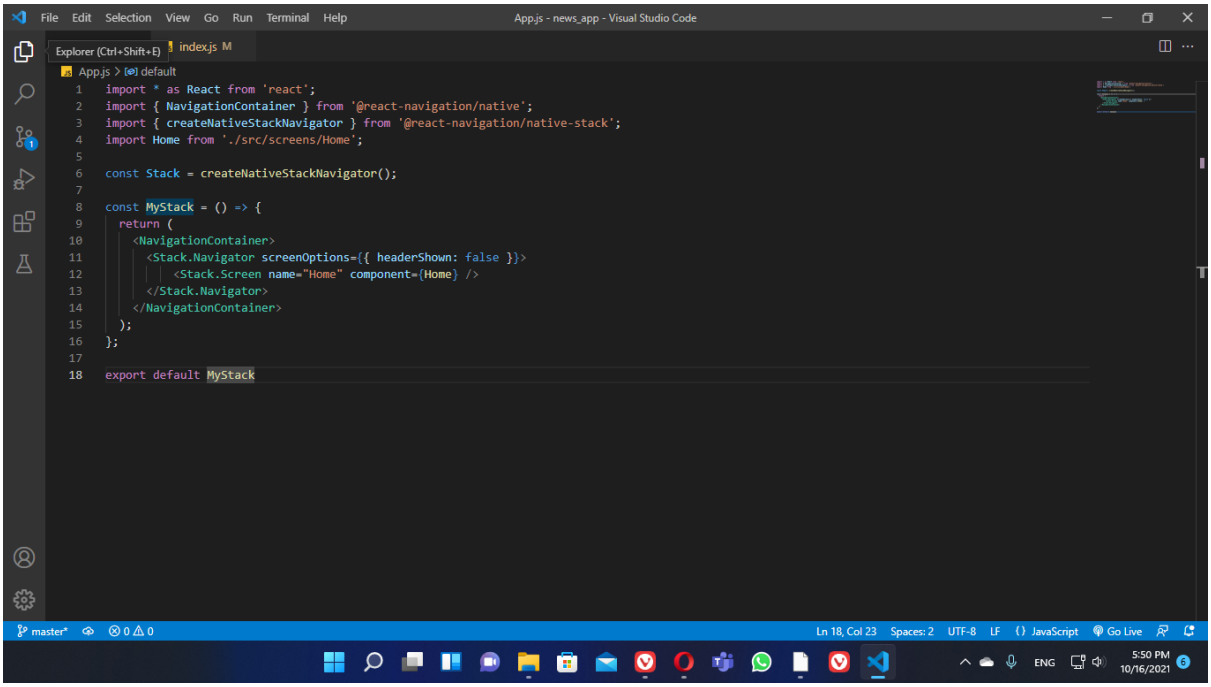


Image 4.6 – Actual look of Stack Navigator

After the above code you need to create a src folder in the root of the project. And inside it you need to create images and a screen folder Just like that. If you have an experience in web development or any other development background you will know what is the purpose of this folder. This folder we used for the store our project's assets and screen.

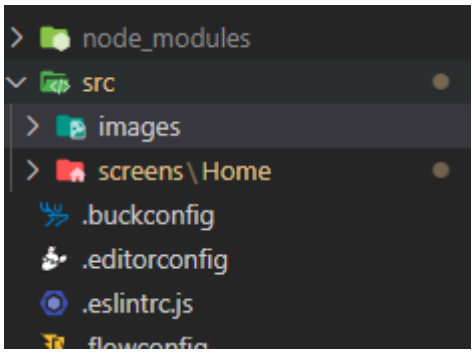


Image 4.7 – src folder

After the create SRC folder you need to create a screens folder and inside there you need to create a JS file called index.js. And placed it in a new folder called Home. After finishing this process, you need to import this screen to app.js. So simply go to the app.js and import it like this –

import Home from './src/screens/Home';

Now we import our screen into the app.Js and now we need to start development. So simply go to the home folder and click index.js and start development.

So, you need to define a functional component inside the index.js. According to

import React from 'react'

```
import { StyleSheet, Text, View } from 'react-native'

const index = () => {
   return (
      <View>
         <Text></Text>
      </View>
   )
}

export default index

const styles = StyleSheet.create({})
```

In the above code I defined the functional component. First, I imported the react and stylesheet, text and view from react native.

Inside the return of the function, you need to define view components. React native uses the JSX tags. So, these JSX tags are just like HTML tags. But the work difference. After the return we need to export the function and also you need to define your styles also.

2.2 UI design

Before development I design my mock design in figma. Figma is the best tool for software mock design. Many developers and designers worldwide use figma. So, if you don't have a figma account please create it and it is free. You can develop windows app also. So, I created a new project on figma and created the news app UI. Here I used Google's material design patterns to design UIs.

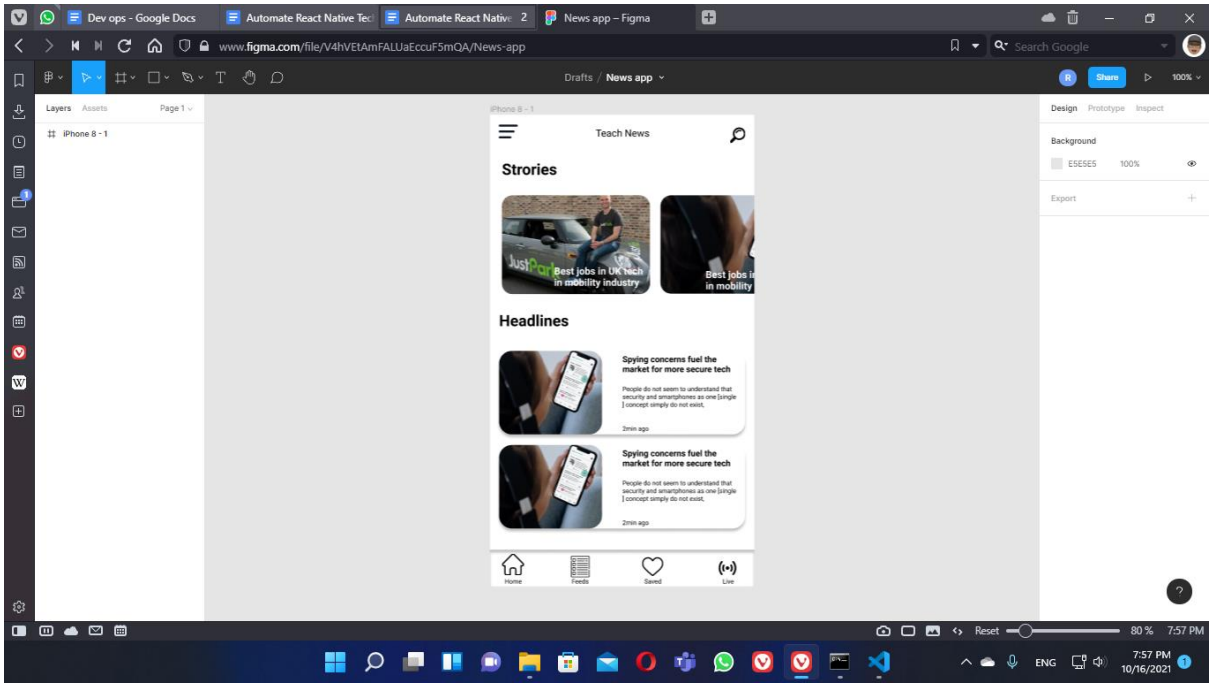


Image 4.8 – Figma Workspace

In the next part I'll teach you how to develop those UI interfaces.

2.3Develop App Interface with React Native.

In the react native project before the initial layout development we need to specify the project root file app.js with the react router. So, in this project we use react stack navigator for the full app routing. I will explain How to install react navigation stack to react native project and how to implement it to App.js

There are many types of navigators available in react navigation. Those are

1. Stack navigator
2. Tab navigator
3. Bottom tab navigator
4. Drawer navigator

Like many navigators available in react navigation. So, in our proposed app only use stack navigator and Bottom tab navigator only.

Image 4.9 – react navigation stack

After the setup the stack navigator in to app.js we can move to app implications.

2.4Develop First Screen

Before the development we need to create screens folder inside our src folder so simply go to the scr folder and create screen folder inside it like this.

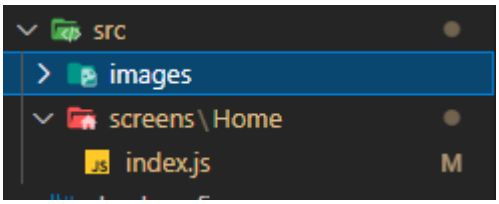


Image 5.0 – Screen Folder inside src folder.

So, after that you need to create home folder inside src folder like above 4.9 image. Now to inside the home folder and create new JS file called index.js. you can use any name for this JS file but when you call it as index.js int will automatically called when we give path to the stack navigator. You can see the index.js folder in 4.9 image. Now we can start development.

So simply go to the index.js using vscode and follow me. First of the development you need to add <Safeareview> as our main wrapper. Because every smartphone has a notch. So, this safe review gives some protected area just like margin to mobile apps from this notch.

As a design we need to add a header first to our app. You can use pre-built header using native base or any other UI framework. But in this project, I'm developing it on my own.

```
<SafeAreaView style={{ flex: 1, backgroundColor: "white" }}>
        {/* header */}
        <View style={styles.header}>
          <View>
            <Image source={left_image} style={styles.left_image} />
          </View>
          <View>
            <Text style={styles.header_text}>Tech News</Text>
          </View>
          <View>
            <Image source={right_icon} style={styles.left_image} />
          </View>
```

……

Styles

……

```
  header: {
    flexDirection: "row",
    justifyContent: "space-between",
    alignItems: "center",
    paddingVertical: (window.width) * 0.03,
    paddingHorizontal: (window.width) * 0.02
  },
```

……

After that we need to develop our header slider. You also can use react native swiper or any other react native slider library for that. But here I develop it on my own react native components. According to that, I'm using flatlist to dynamically integrate the card. Because when the API connects, I need to dynamic those cards. So first I'm going to create an array with news card information just like that.

```
const Top_slider_list = [
  {
    id: 1,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
  {
    id: 2,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
  {
    id: 3,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
];
```

After that you need to map this array using react native flatlist. First you need to import react native flatlist form react native like this. (You need to replace this array when API is available.)

```
import { StyleSheet, Text, View, Image, Dimensions, FlatList, ScrollView } from 'react-native'
```

After that you need to set up a flatlist for the above data array.

```
<ScrollView>
        <Text style={styles.scondery_header_text}>Stories</Text>
        <FlatList
          contentContainerStyle={{ paddingLeft: 10 }}
          horizontal
          showsHorizontalScrollIndicator={false}
          data={Top_slider_list}
          renderItem={({ item }) => <Top_slider tech_data={item} />}

        />
    <Text style={styles.scondery_header_text2}>Headlines</Text>
```

……

As a next part you need to develop the bottom headline tab. I create an array the same as before for data and map it using map () callback.

```
const news = [
  {
    id: 1,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
  {
    id: 2,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
  {
    id: 3,
    text: 'Best jobs in UK tech in mobility industry',
    slid_image: require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
  {
    id: 4,
    text: 'Best jobs in UK tech in mobility industry',
```

```
    slid_image:      require('../../images/_120570688_xayn-mock_woman-hold-iphone-deep-search.jpg'),
  },
];

……….

 const Bottom_cards = () => {
    return (
      <View style={{alignItems:"center",marginTop:(window.width)*0.06}}>
        {news.map((item, index) => (

            <View style={styles.wide_card}>

            </View>

        ))}

      </View>

    );
  }
……..
```

After that we need to call this custom component beside the view like this. And my full app CSS style sheet look like this. You can copy and paste it into your code.

```
const styles = StyleSheet.create({
   header: {
      flexDirection: "row",
      justifyContent: "space-between",
      alignItems: "center",
      paddingTop: (window.width) * 0.03,
      paddingHorizontal: (window.width) * 0.02,

   },
   left_image: {
      height: (window.width) * 0.08,
      width: (window.width) * 0.08
   },
   header_text: {
      fontSize: (window.width) * 0.045,
      color: "#000",
      fontWeight: "bold"
   },
   scondery_header_text: {
```

```
    fontSize: (window.width) * 0.055,
    fontWeight: "bold",
    color: "#000",
    marginHorizontal: 15,
    marginTop: (window.width) * 0.06
},
scondery_header_text2: {
    fontSize: (window.width) * 0.055,
    fontWeight: "bold",
    color: "#000",
    marginHorizontal: 15,
    marginTop: (window.width) * 0.02
},
slider_card: {
    width: (window.width) * 0.7,
    height: (window.width) * 0.4,
    backgroundColor: "white",
    elevation: 15,
    borderRadius: 10,
    marginVertical: 10,
    marginLeft: 6,
    marginRight: 10,
    marginVertical: (window.width) * 0.05,
```

```
        // marginHorizontal: 10
    },
    wide_card: {
        width: (window.width) * 0.95,
        height: (window.width) * 0.35,
        backgroundColor: "#fff",
        elevation: 10,
        marginVertical: 6,
        borderRadius:10,
        flexDirection:"row"
    },
})
```

After developing the app, it will look like this.

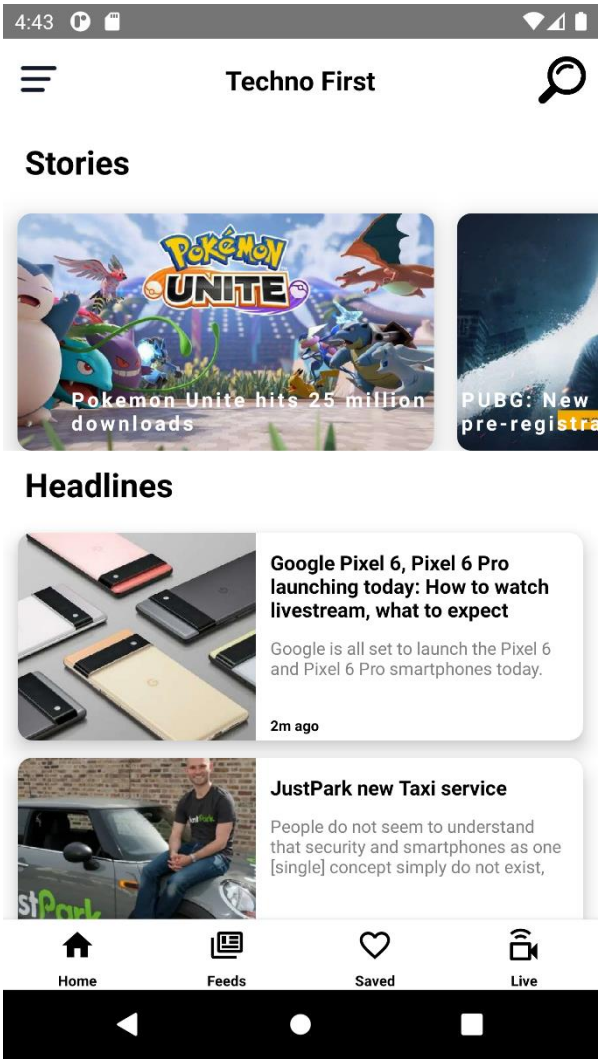


Image 5.1 – Final App screen

2.5Develop API

In our previous code we make our app with hardcoded data. So, we need to develop API for this. In this section I'll teach you how to create MySQL database and how to create table and how to develop Node.js API on starch.

First you need to install XAMPP client for your pc. It will install SQL and it will help us to run the sever on local pc. So first, you need to visit https://www.apachefriends.org/index.html and download the XAMPP.exe file and install in into your pc. So, I'll show you how to install it step by step here.

Image 5.2 – XAMPP Web site.

After the download the exe file form official site you need to install it to your local system. First install this you need to deactivate any running virus guard for some time because sometime virus guard block the xampp installation. After the installation you can continue with virus guard.

Image 5.2 need to disable the virus guard for temporary.

Next you need to Deactivate UAC because UAC will be limited to the writing access to the C drive.

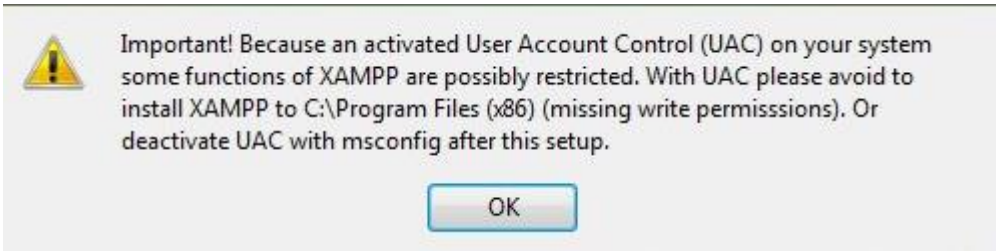


Image 5.3 Detective UAC

So, after the above process xampp setup wizard will append like this.

Image 5.4 Start the setup.

So next step you need to select components so you can select any components form there but ill recommend to this test environment you need to put it as default.

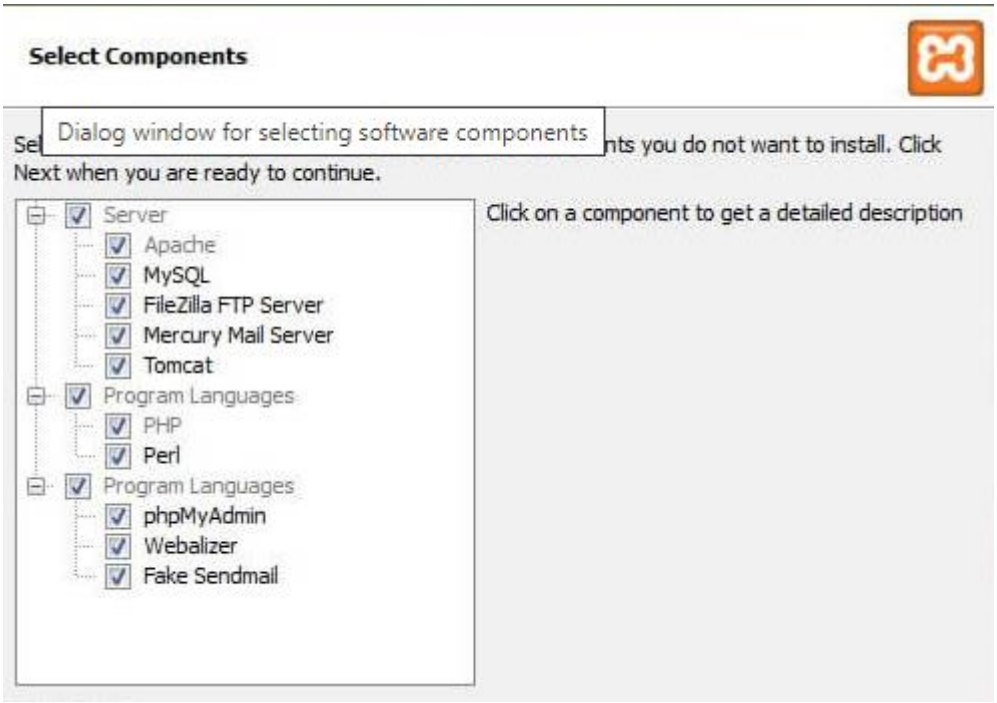


Image 5.5 Select needed components.

In the next part you need to select installation directory so that you can select any directory for that according to your desire. But I recommend using C drive default path for this.

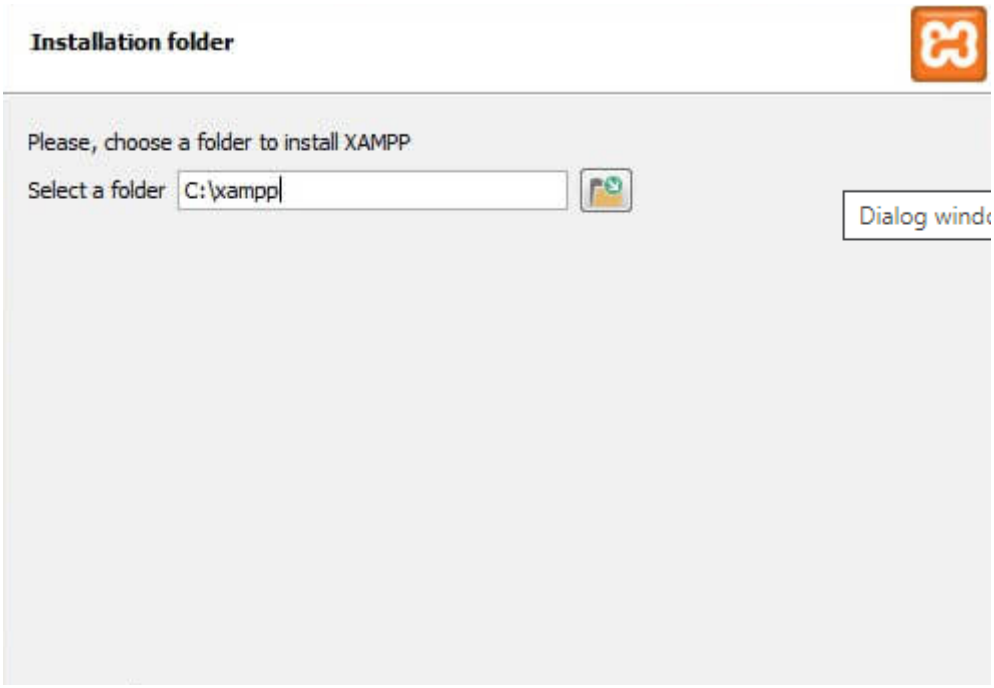


Image 5.6 Select installation location.

After the above process you will be move to the start installation screen in the xampp setup. So, this setup weird will be unpacked all necessary file to your installation location.

Image 5.7 Installation process start.

Sometimes in this installation process will be interpreted by your system firewall. So, this time you need to allow that when firewall pop up detected. So, after the installation you will see the tis type of screen.

Image 5.8 Finished the Installation.

So, after the installation is finished you need to open xampp control panel. It will look like this.

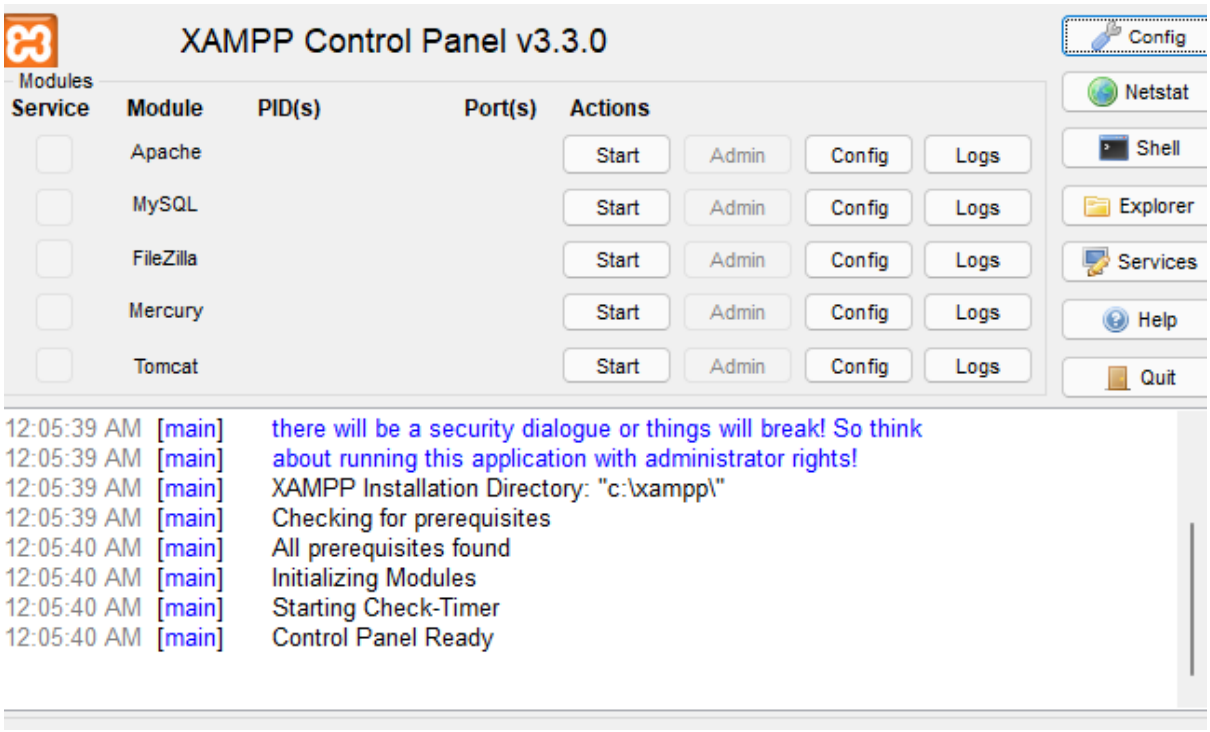


Image 5.9 XAMPP Control Panel.

In this control panel you will see various tool available to configure. Eg that if you need to Apache abd mysql in is there and fizilla also there and tomcat and Mercury also there. And you can open configuration file using config button and you can open shell using shell button like that.

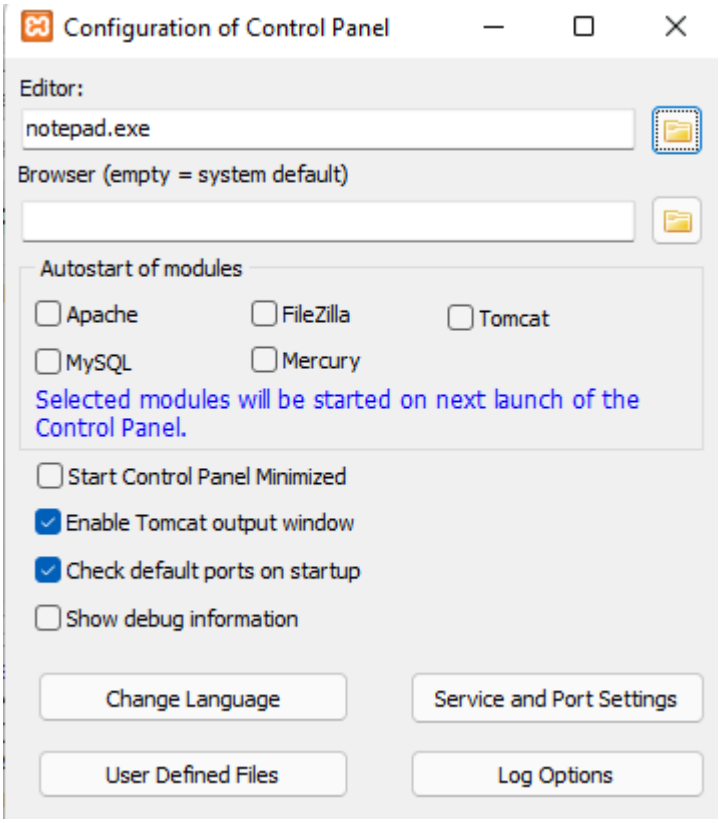


Image 6.0 Open Config file.

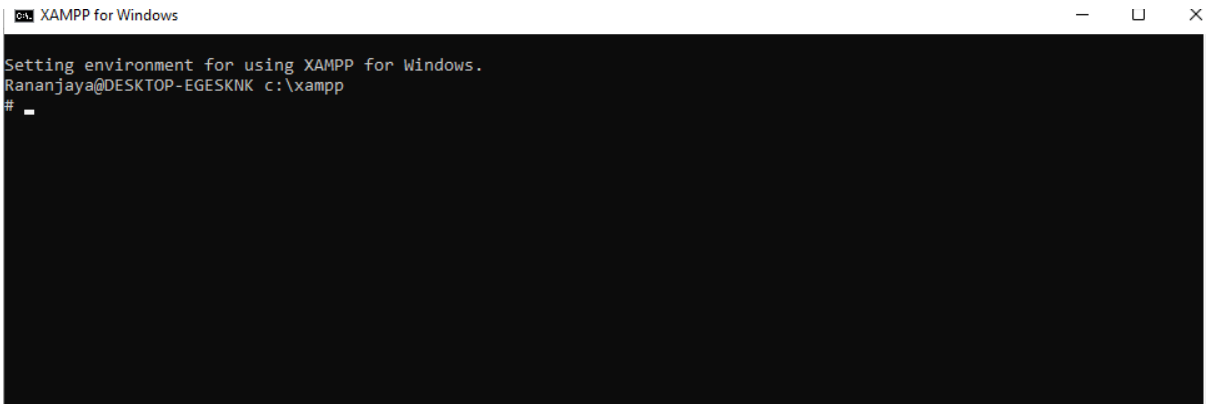


Image 6.1 Open shell

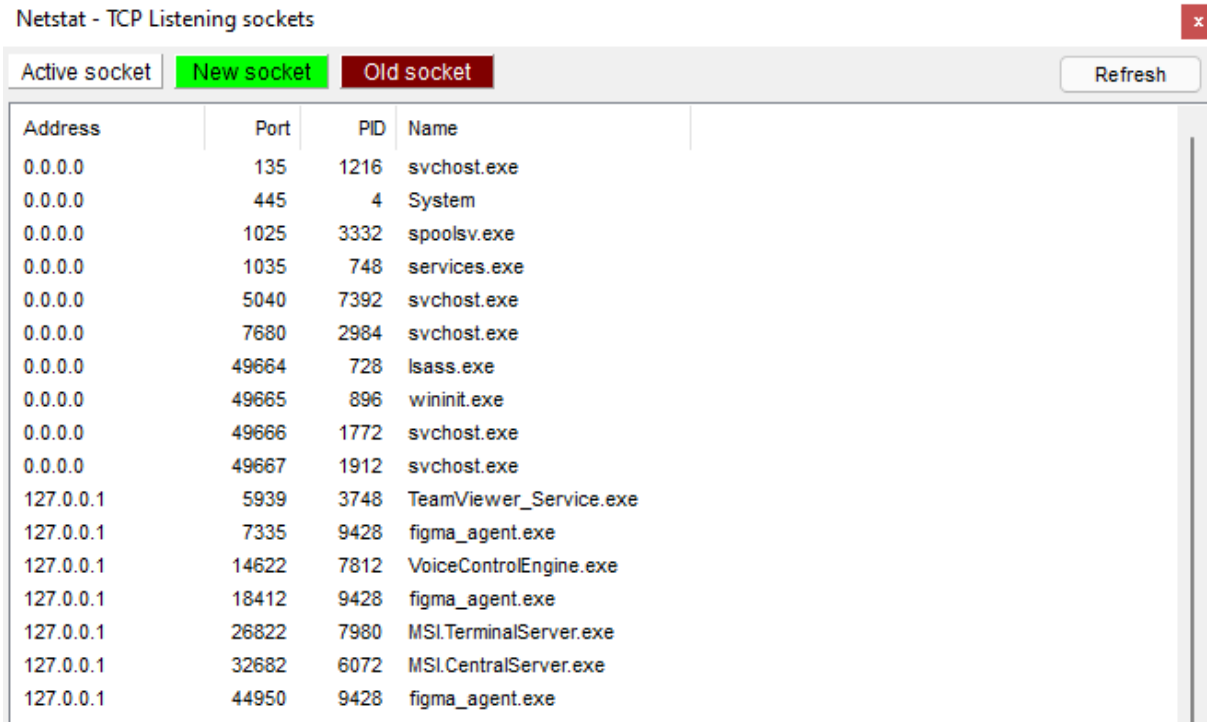


Image 6.2 Open Netstat.

So, in this project we need to Apache and MySQL. So, we need to enable those modules form the xampp server. So that you need to enable those models like this.

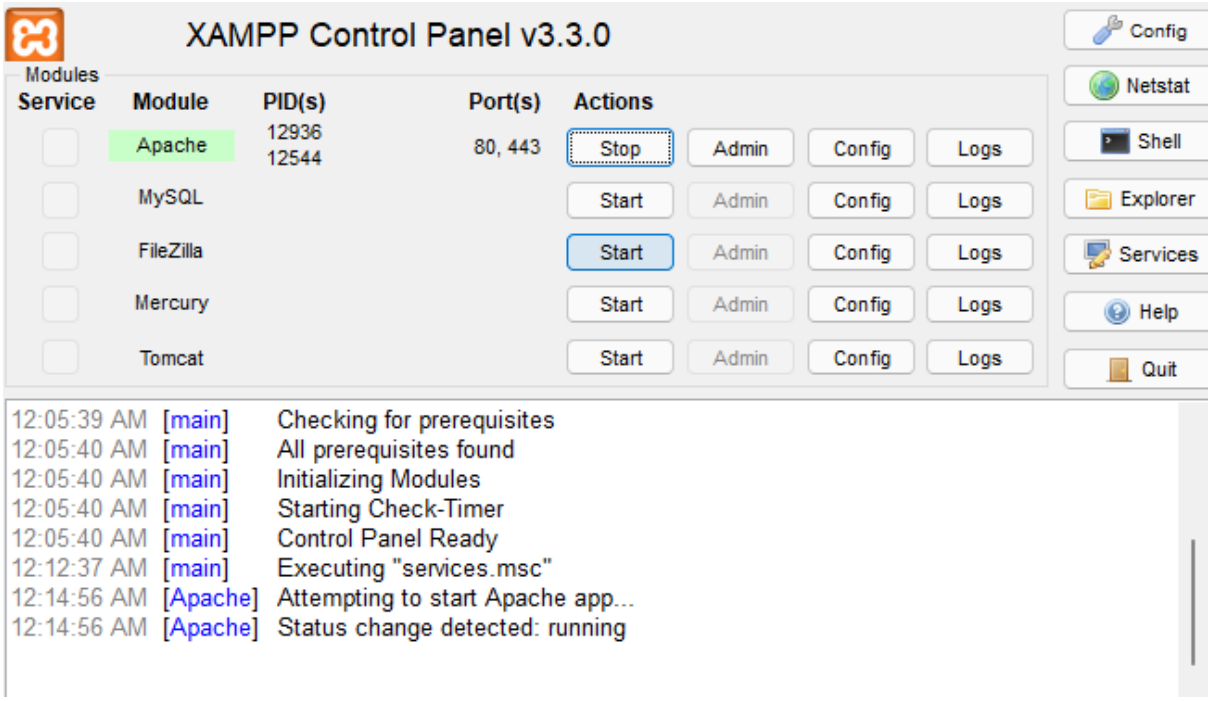


Image 6.3 Start Apache.

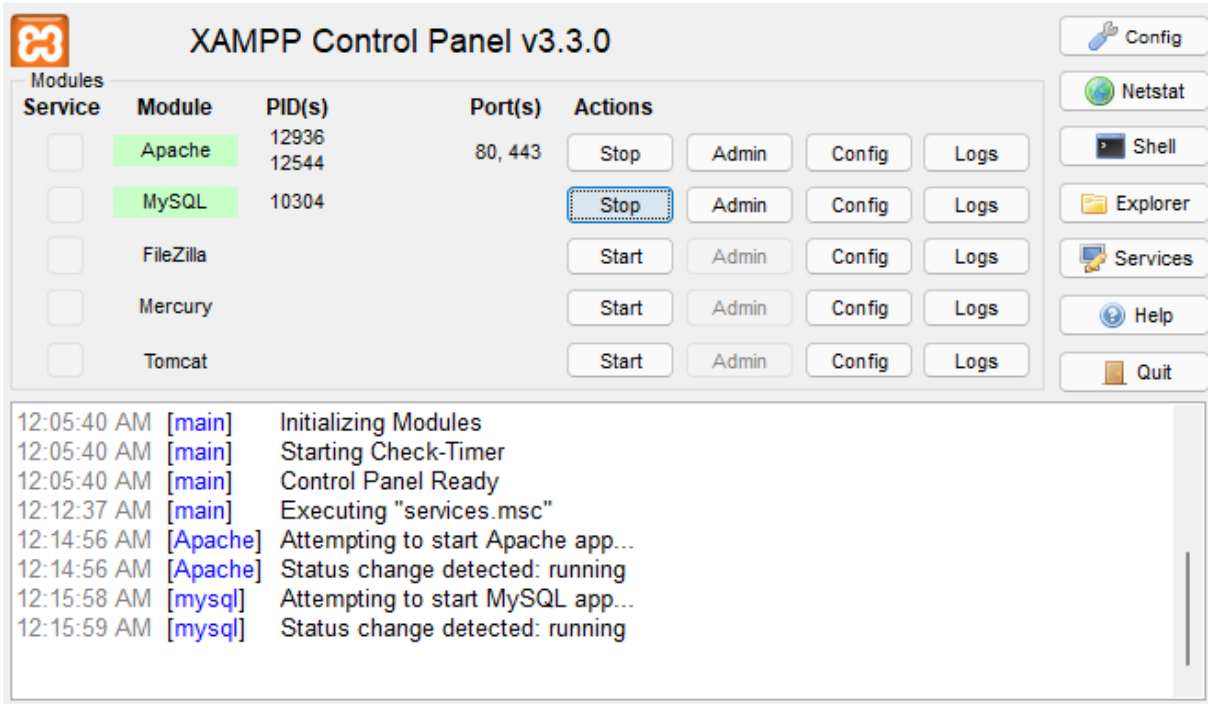


Image 6.4 Start MySQL

Now we are successfully installed and setup locally.

2.6How to make Node JS project

We need to create npm project for out Node express API. So, create Node JS project you need to open your terminal in the project folder and run this command.

npm init

we use this npm init command initialize npm project. When you run this command, it will ask to provide the package name first. So, you can give desire name for that. I gave it as news_app_api like this –

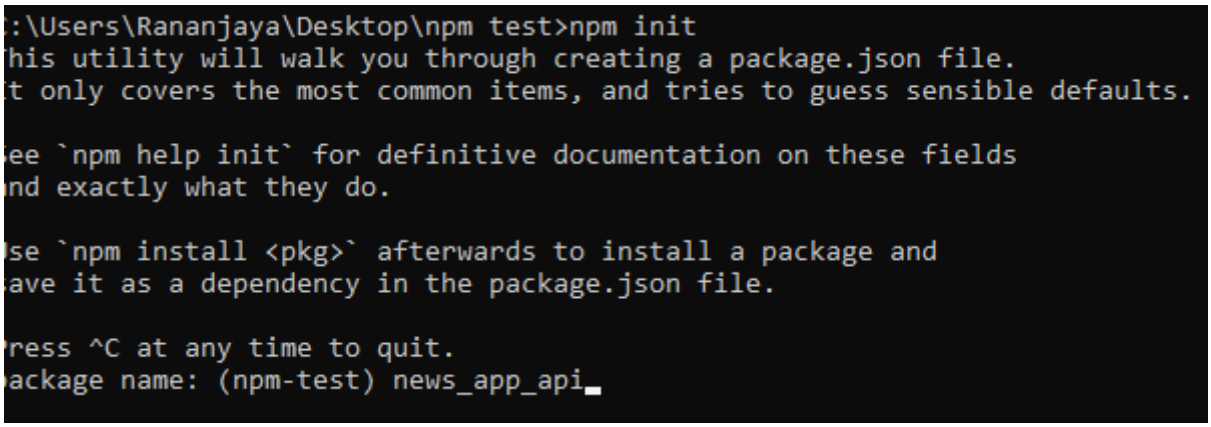


Image 6.5 – npm project creation ask Package name.

You need to give name and hit enter. After the hit enter it will ask version you can give the version or skip it using press enter.

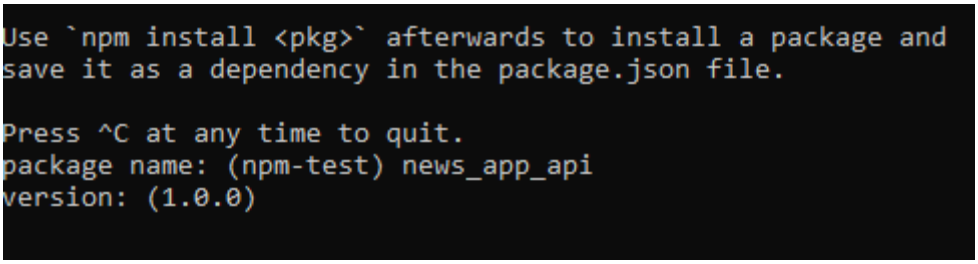


Image 6.6 – npm project creation ask version.

After the hit enter, it will ask to provide some description for the npm app. Ill skip it using press enter

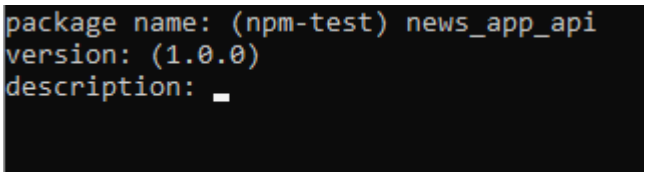


Image 6.7 – npm project creation ask description.

After the hit enter it will ask the entry point of the node JS app. so default it will create index.js file when press the enter.

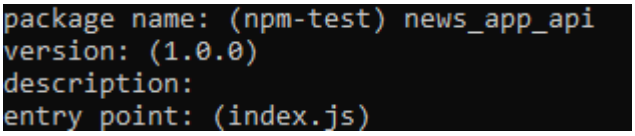


Image 6.8 – npm project creation ask entry point.

Next it will as test command. If you need to provide some unit test command you can use this or you can skip it with press enter.

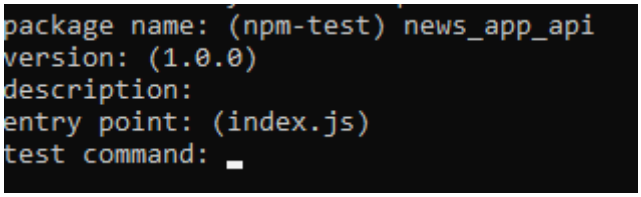


Image 6.9 – npm project creation ask test command.

Next part it will ask to provide git repository and keyword. Ill skip it till now using enter.

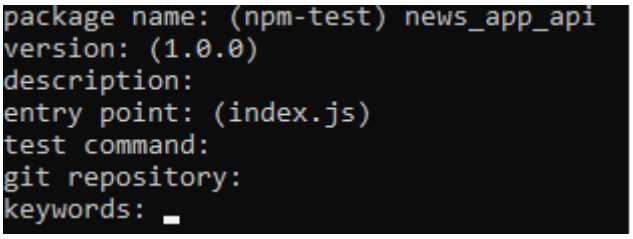


Image 7.0 – npm project creation ask git repository and keywords.

Next part it will ask you author and license of the project. You can skip them by hit enter.

Image 7.1 – npm project creation ask author and license.

After you hit yes it will create our node JS project with package.json file.

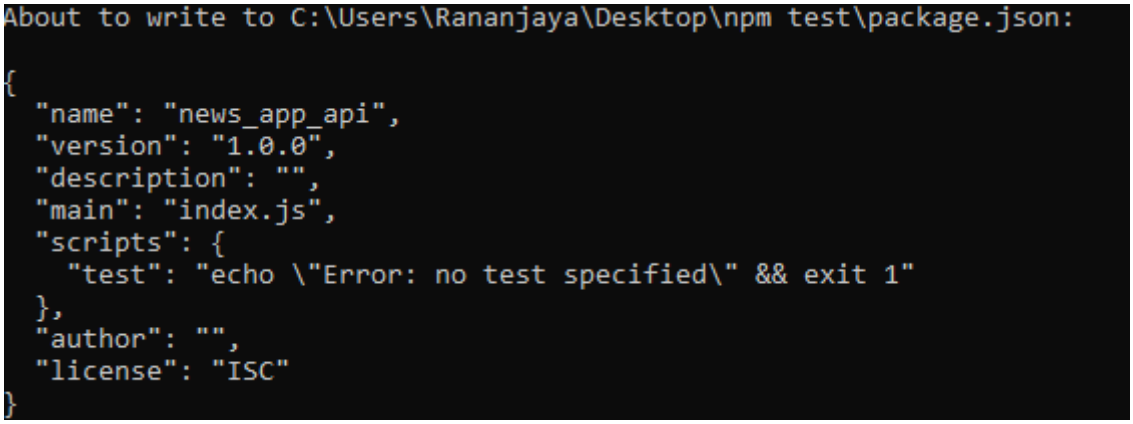


Image 7.2 – npm project creation finished.

Now let's develop API. Before that you need to install some packagers, we need to our API. Those packagers are –

- body-parser
- cookie-parser
- cors
- express
- express-session
- multer
- mysql

you can install them using flowing commands –

- npm i body-parser
- npm i cookie-parser
- npm i cors
- npm i express
- npm i express-session
- npm i multer
- npm i mysql

after the install those Packagers, you need to add this code to your index.js file located in your root directory.

```
const express = require('express')
const mysql = require("mysql")
const app = express()
const cors = require("cors");
const bodyParser = require('body-parser')
const fs = require('fs');


app.use(express.json());
app.use(cors());

//for uplode
app.use(bodyParser.urlencoded({ extended: true }))
app.use(bodyParser.json({ limit: '15MB' }))

app.use(function (req, res, next) {
   var oneof = false;
   if (req.headers.origin) {
      res.header('Access-Control-Allow-Origin',
req.headers.origin);
      oneof = true;
   }
   if (req.headers['access-control-request-method']) {
      res.header('Access-Control-Allow-Methods',
req.headers['access-control-request-method']);
      oneof = true;
   }
   if (req.headers['access-control-request-headers']) {
      res.header('Access-Control-Allow-Headers',
req.headers['access-control-request-headers']);
```

```
        oneof = true;
    }
    if (oneof) {
        res.header('Access-Control-Max-Age', 60 * 60 * 24 * 365);
    }

    // intercept OPTIONS method
    if (oneof && req.method == 'OPTIONS') {
        res.sendStatus(200);
    }
    else {
        next();
    }
});


const routes = require("./routes/routes");
routes(app);

app.listen(3001, () =>{
    console.log ("running on port 3001")
})
```

You can copy and paste this code into your index.js file.

After the copy above code, you need to create model.js file and past this code to your model.js file.

```
const mysql =require('mysql');
const con =mysql.createConnection(
    {
        host:'localhost',
        user:'root',
        password:'',
        database:'tech_news_app'
    },
   {multipleStatements: true}
);
module.exports=con;
```

this model.js file is used to connect our API with MySQL database. You need to specify your database name, username, password, and your hostname here. We will create our database later.

After the create model JS file you need to create another two folders in your project. You need to create **controller** folder and **routes** folder in our root project location. This controller folder is used to store our controllers and routes folder is used to store our routes.js file.

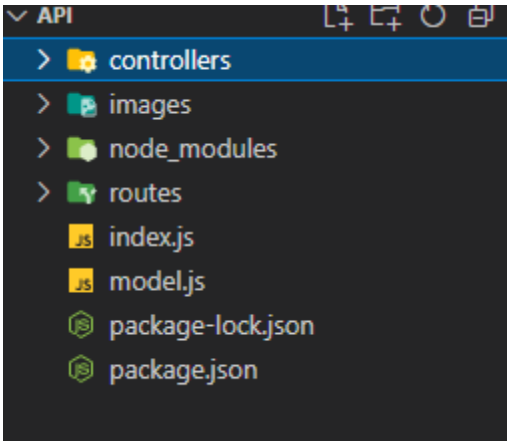


Image 7.3 – node project structure.

So, after that you need to create controllers our app. For that you need to go to controller folder and create TecnewsController.js and copy this code to that file.

```
const mysql = require('../model');


exports.TecnewsController = function(req,res){
   var insertSQL  = 'SELECT * FROM news';
   //var InsertBody = [req.body.mobile];




   mysql.query(insertSQL,function(err,result){
      if(err){
         console.log("rana error",err)
         res.send({err: err})
         res.json({"Error": true, "Message": "Unable to connect to the database. Please contact Support."});
        }else if(result.length > 0){
         console.log("This number avilable")
           res.send({ "Error": false, "Message": "avilable" , "Rows":result});


         // res.send(result)
```

```
        }else{

            res.json({"Error": true, "Message": "Not_avilable"});
        }


    }}
```

After the create controller, you need to go to the route folder and create routes.js file. So simply go to the route folder and create route.js file and past this code.

```
const TecnewsController = require('../controllers/TecnewsController');
const Testcontroller = require('../controllers/testController');

const multer = require('multer');
const fs = require('fs');


const storage = multer.diskStorage({
   destination :function (req,file,cb) {
      cb(null,'images/');
   },
   filename: function (req,file,cb) {
```

```
        cb(null,Date.now() + file.originalname);
        console.log('check file :',file);
    }
});

const upload =multer({storage:storage});


const appRouter = function (app) {
    app.get('/', function (req, res) {
        res.status = 200;
        res.send("Tech_news_API_Running…….");
    });


    app.get('/TecNews',TecnewsController.TecnewsController); //get all tec news
    };
    module.exports = appRouter;
```

so now we configure everything for our API. So next thing config the database for our API. So, let's do it.

2.7 Database configuration.

First the database configuration we need to start our local server first. So simply go to XAMPP and open it and run the Apache and MySQL.

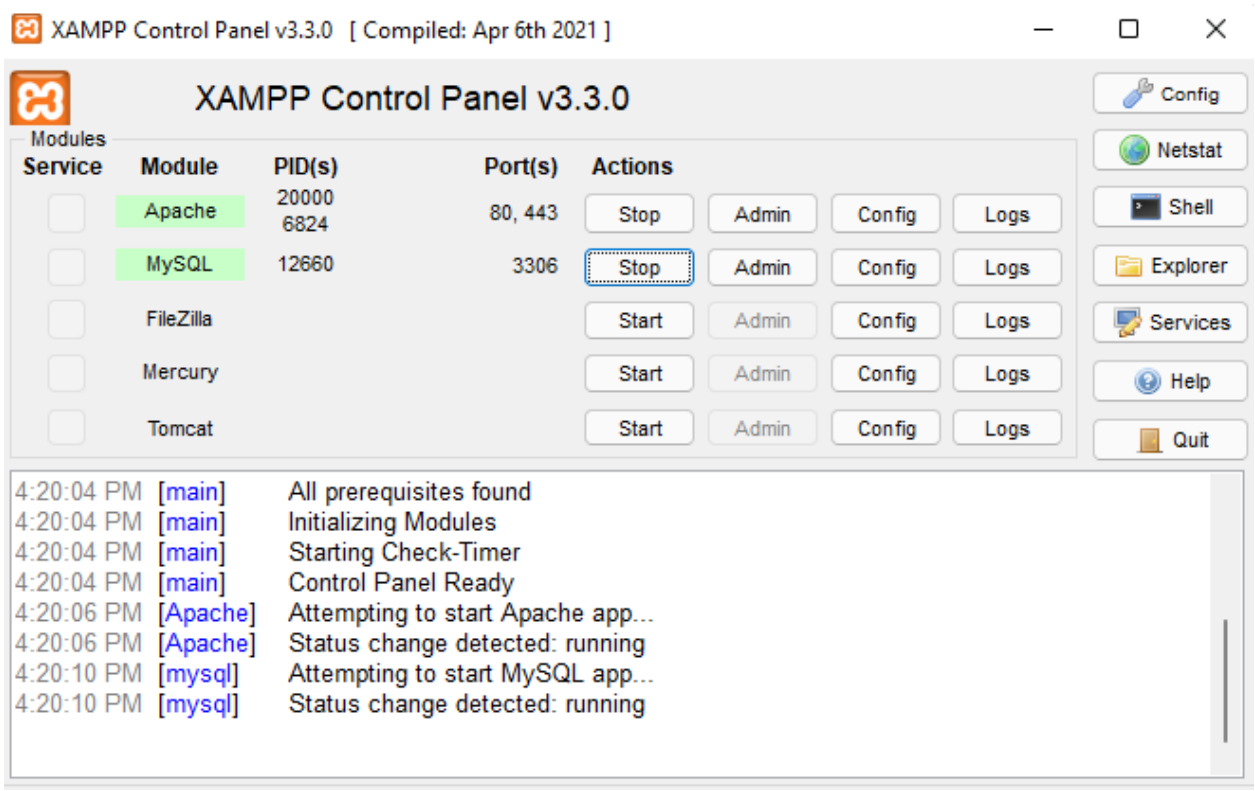


Image 7.4 – XAMPP client start

According to Image 6.1 you need to start Apache and MySQL. After that you need to open your browser and type **http://localhost/phpmyadmin/** in URL Bar.

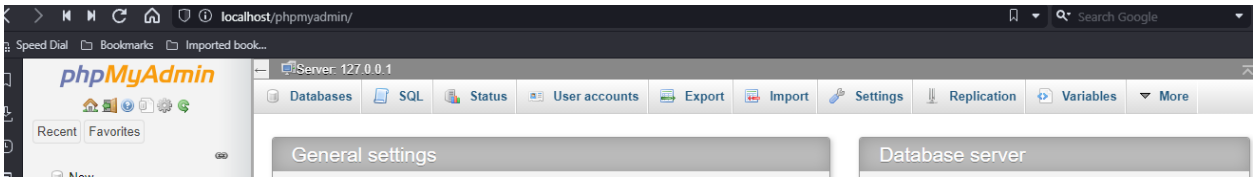


Image 7.5 – phpMyAdmin.

According to Image 6.2 you will see web browser redirect to the phpMyAdmin page.

2.8Create Database

Next Part is the database creation. If you need to create database on phpMyAdmin you need to click new button on left side bar in the phpMyAdmin page.

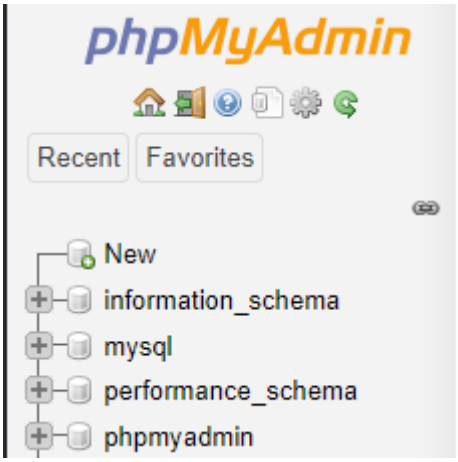


Image 7.6 – phpMyAdmin create new Database.

After the click the new button, you will be redirected to the new screen. In there you need to give your database name.

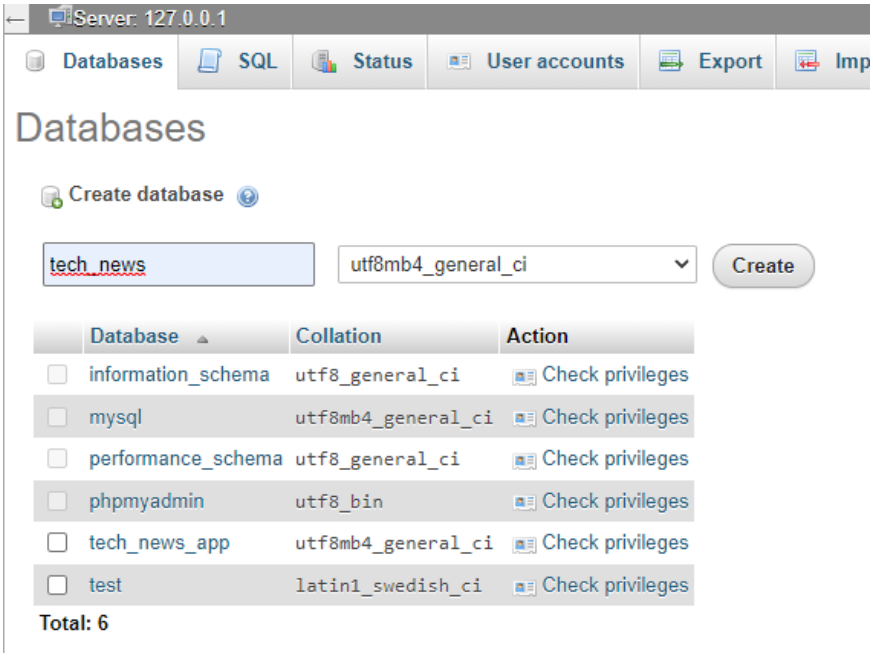


Image 7.7 – phpMyAdmin create new Database.

In here I gave name as tech_news and select utf8mb4 second drop down. And click create button. Now we create new database now phpMyAdmin shows your new database of left side bra of the phpMyAdmin.

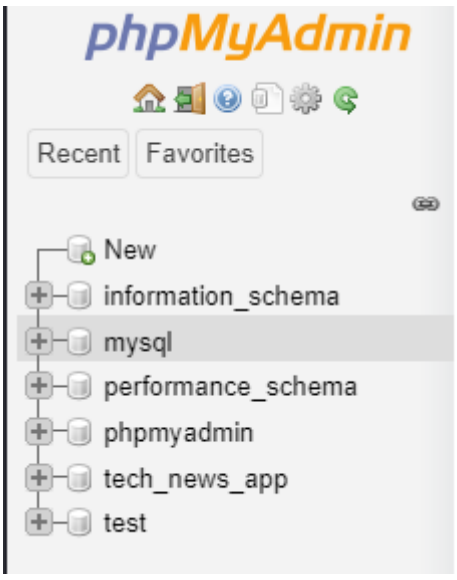


Image 7.8 – newly created database shown here.

Now you need to select the database to create a table inside it. For that you need to click on your database. And go to create database section.

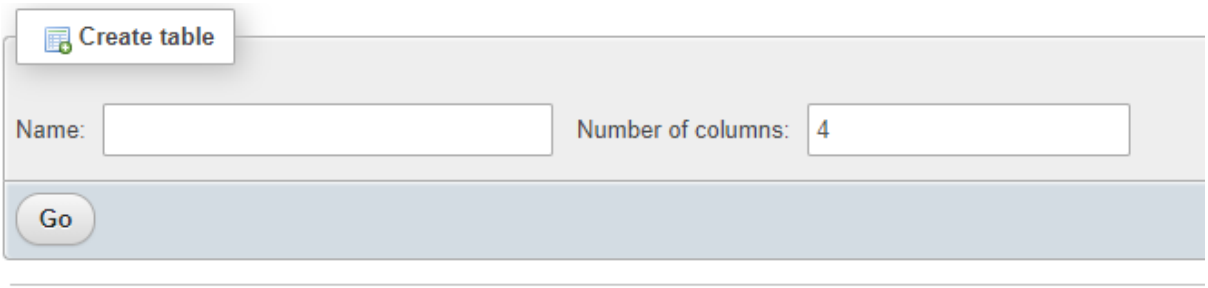


Image 7.9 – Create new Table.

According to Image 6.6 you need to type your Database Table name and number of columns here. In this project I use two database tables for news and slider. So that I create two tables here.

So, I create first table as “news” and add 5 columns. And I create another table as “Slider new” to store data of the slider with 3 columns. So ill show you what is the structure of each table here. So, in here news table id must be primary key. And Image URL must be the long text as it’s type and headline, news and time must be varchar.

And slider table also id need to be primary key and Image URL same as long text and text need to be varchar. So, you need to configure your table like that.

- **News Table.**

id	Image_URL	headline	news	time

- **slider_new**

id	Image_URL	text

According to this structure you need to create tables. After the create the table, we need to insert data to those tables, for that you need to click insert tab in top bar of the phpMyAdmin Screen.

Image 8.0 – Insert Tab in Top Bar.

When you click the insert tab int will redirect you to new screen just like this.

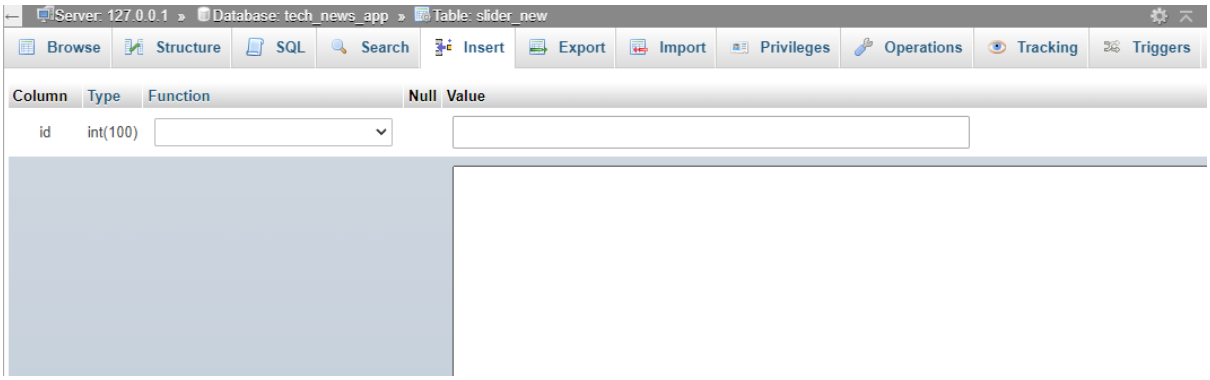


Image 8.1 – Insert Data to Table.

So, you need to add some data and image URL and Insert to data to the table. You can follow this process for your both tables.

After the create both tables you need to add the database details to your API's Model.js file. So that you need to open your file directory of your project and fine the model.js file and open it using VScode. In there you need to give your MySQL database name, hostname and password and username of your local MySQL.

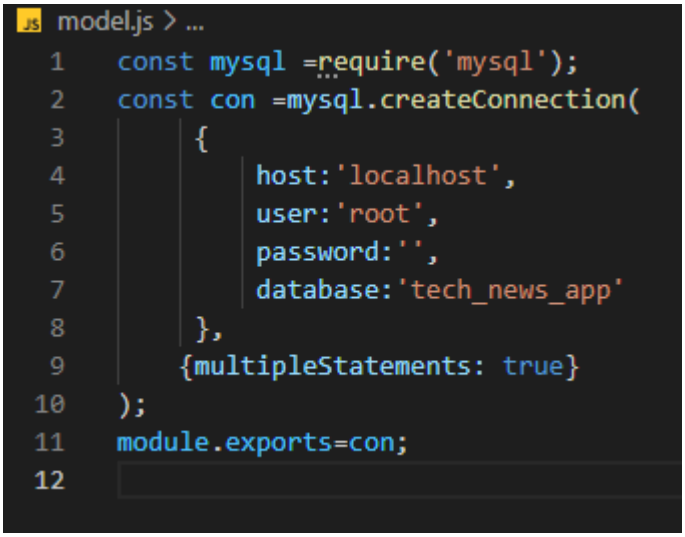


Image 8.2 – Edit model.js file.

Now we have done our API and database configuration successfully. So next part is you need to run this API and test it. Before the test you need to run **npm install**. After npm installation you need to run your API with **npm start** command. You will see this type of messages "**Tech_news_API_Running…….”** if your API have successfully run.

2.9Connect API with Mobile App

For next part we need to connect our API with mobile app. for that you need to make a request to our API routes. I'll show you how to do that part. So as the first step you need to import hooks that we need.

```
import React, {useState, useEffect} from 'react'
```

after that you need to define **useEffect** function.

```
useEffect(() => {

}, []);
```

useEffect function calls when our app component is banding so each time our components rendering it will call. Next part is we need to fetch the data from the API. For that you need to define two functions. Called **get_slides** and **get_bottom_news** like that.

```
async function get_slides() {
        console.log("insdie get slides");
  try{
      fetch("http://192.168.56.1:3001/test", {
          method: "GET",
          headers: {
          //   'Authorization': 'Bearer ' + DEMO_TOKEN
           Accept: 'application/json',
          'Content-Type': 'application/json'
          }
        })
          .then((response) => response.json())
          .then((json) => {
            console.log("headline rana", json.Rows)
            if(json.Rows != "" && json.Rows != ""){
              SetSlide(json.Rows);
              setLoading(false);
            }

          })
```

Image 8.3 – get slides function

……

```
async function get_bottom_news() {

  try{
      fetch("http://192.168.56.1:3001/TecNews", {
          method: "GET",
          headers: {
          //    'Authorization': 'Bearer ' + DEMO_TOKEN
           Accept: 'application/json',
          'Content-Type': 'application/json'
          }
        })
          .then((response) => response.json())
          .then((json) => {
            console.log("ranaaa", json.Rows)
            if(json.Rows != "" && json.Rows != ""){
                var string_Data = JSON.stringify(json.Rows);
                 console.log("data",string_Data)

              SetNews(json.Rows);
              setLoading2(false);
```

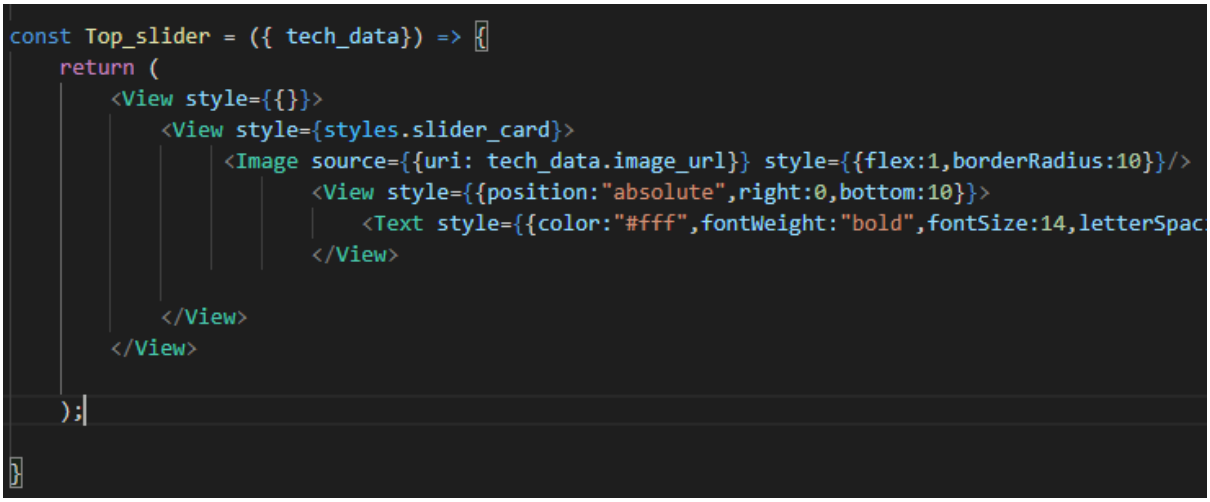
Image 8.4 – get news function

And you need to define those states in your scouse code like that.

```
const [slide, SetSlide] = useState("");
 const [Newslist, SetNews] = useState([]);
 const [Loading, setLoading] = useState(true);
```

Image 8.5 – Define states.

so that we can define our states. Next part is the dynamic the UI with API response. So according to that we pass the states to the UI like that. First, I'll show you how I dynamic news list.



```
const Top_slider = ({ tech_data}) => {
    return (
        <View style={{}}>
            <View style={styles.slider_card}>
                <Image source={{uri: tech_data.image_url}} style={{flex:1,borderRadius:10}}/>
                    <View style={{position:"absolute",right:0,bottom:10}}>
                        <Text style={{color:"#fff",fontWeight:"bold",fontSize:14,letterSpac
                    </View>

            </View>
        </View>

    );

}
```

Image 8.6 – Top slider dynamic with API.

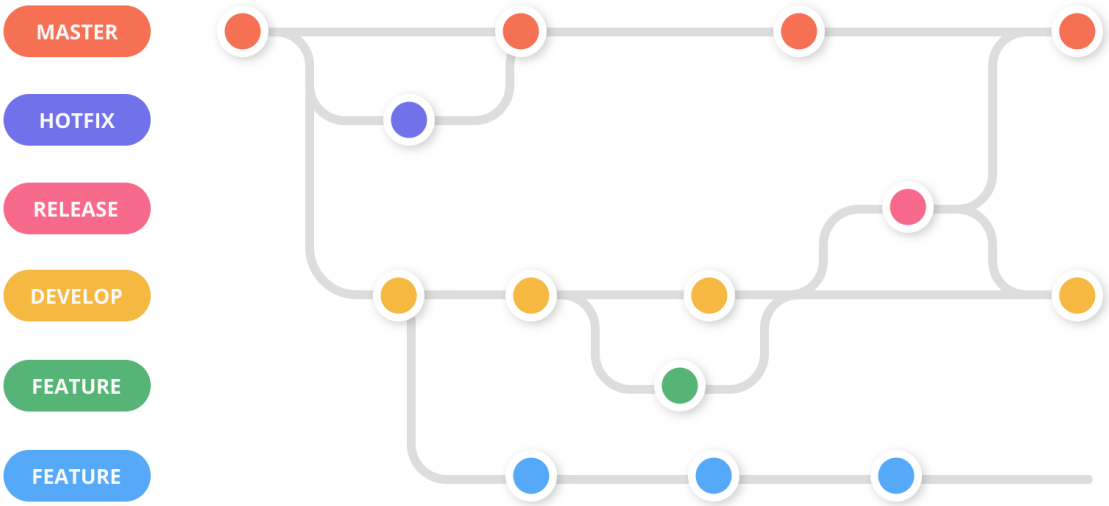
```
const Bottom_cards = () => {
    return (
        <View style={{ alignItems: "center", marginTop: (window.width) * 0.03 }}>
            {Newslist.map((item, index) => {
                return (
                    <View key={index} style={styles.wide_card}>
                        <View>
                            <Image source={{ uri: item.image_url }}
                                style={{
                                    width: (window.width) * 0.4,
                                    height: "100%",
                                    borderTopLeftRadius: 10, borderBottomLeftRadius: 10
                                }} />
                        </View>
                        <View style={{ marginLeft: 10, marginTop: 5, flexShrink: 1 }}>
                            <Text style={{ fontWeight: "bold", color: "#000", marginTop:
                            <Text style={{ fontSize: 12, marginTop: 9, color: "gray" }}>{
                            <Text style={{ fontSize: 10, marginTop: 15, fontWeight: "bold
                        </View>
                    </View>
                )
            })}
```

Image 8.7 – Dynamic Bottom card with API.

Now we dynamic our full app with the API.

Chapter 3

GitHub Workflow.



3 Chapter

3.1Define GitHub workflow.

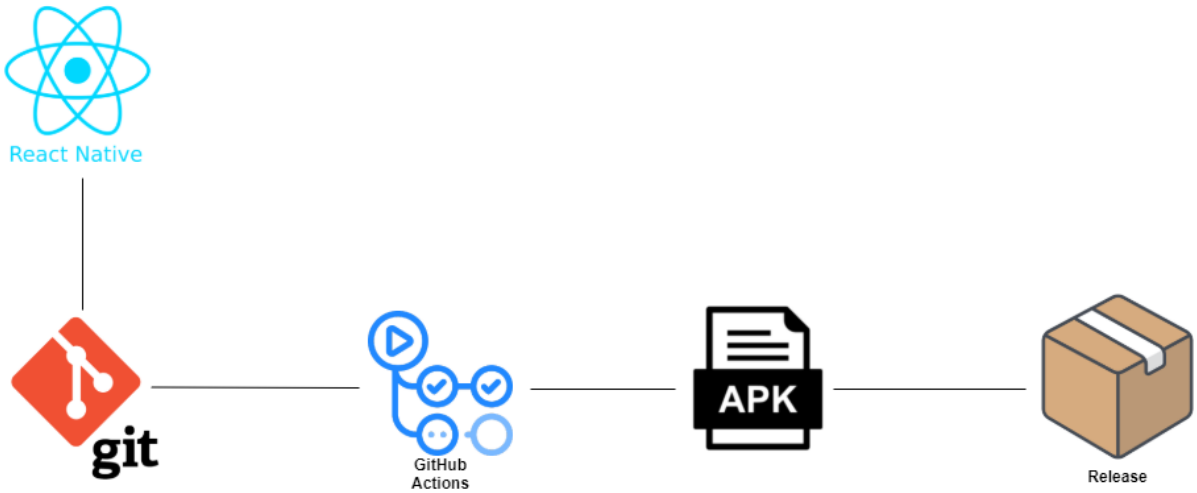


Image 8.8 – GitHub Workflow

GitHub is one of the most popular systems for hosting cloud-based repositories among developers. Continuous Integration (IC) is a method that entails frequently updating new shared code to a repository in order to find mistakes as quickly as feasible. As a result, GitHub recommended not just hosting our code in their repositories, but also automating our projects' compilation and testing stages.

After the GitHub acquired by Microsoft, and they have been introducing new features and functionality, such as GitHub Actions, which are designed to make coding easier. To construct, execute, and deploy their applications nowadays, developers must go through a slew of stages. Installing packages or development environments is a common part of this process. Git Hub Actions develops from here.

So now we are going to define The GitHub workflow process to our React native project. We use GitHub workflow actions scheduled or triggered by an event. The procedure may be used to create, test, package, release, or deploy a GitHub project. Now I'm going to teach you how to use GitHub workflow for you react native project.

3.2Push the Project to the GitHub repository.

As a first step you need push your project to your Git repository you created before. For that you need to go to your project and Open CMD on your project location. As a first step we need to check git status of our project. For that you need to type **git status** command and press enter.

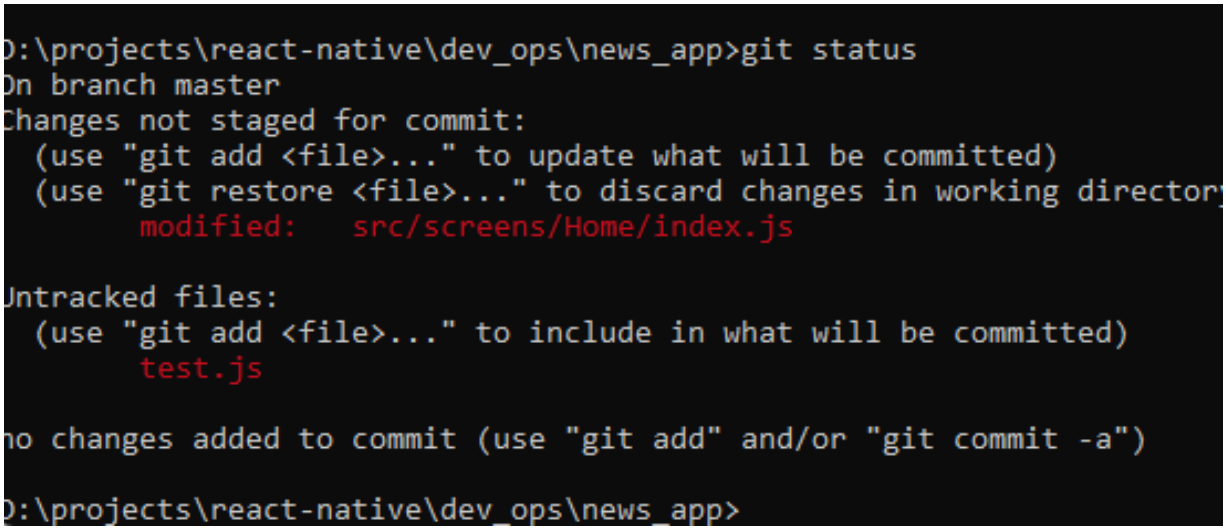


Image 8.9 – Git status command.

After check git status if you see red text like above you need to add them in to git. So, you need to type **git add.** command to add it.

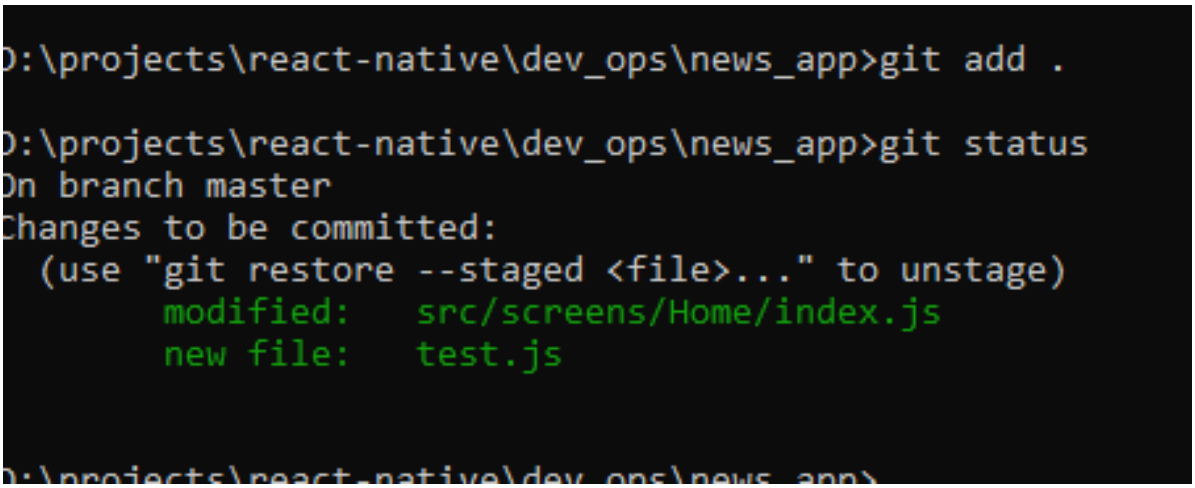


Image 9.0 – Git add. command and git status after the add.

Next part is you need to commit your changers before the push. For that you need to run git commit command.

Git commit -m “first commit”

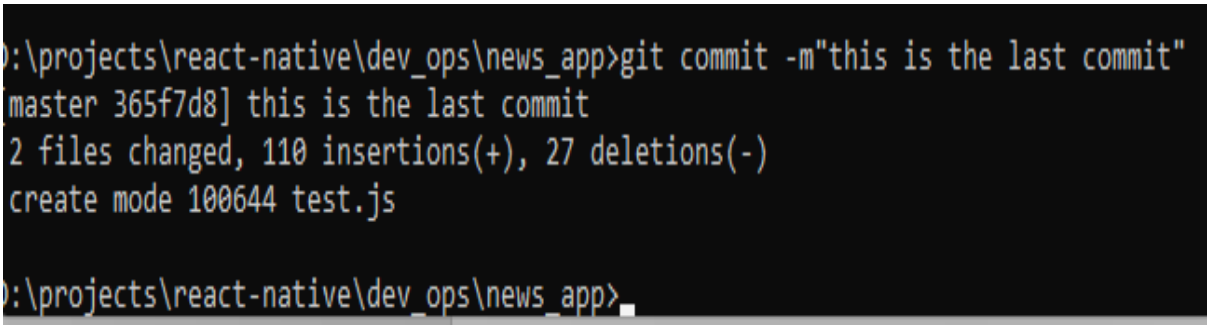


Image 9.1 – Git commit Command.

After the git add command and git status if you see this type of green text, you project is ready to push.

As a next part you need to push your project into GitHub repo. For that you need to run git push command. I'll show you how to do that.

Git push origin master

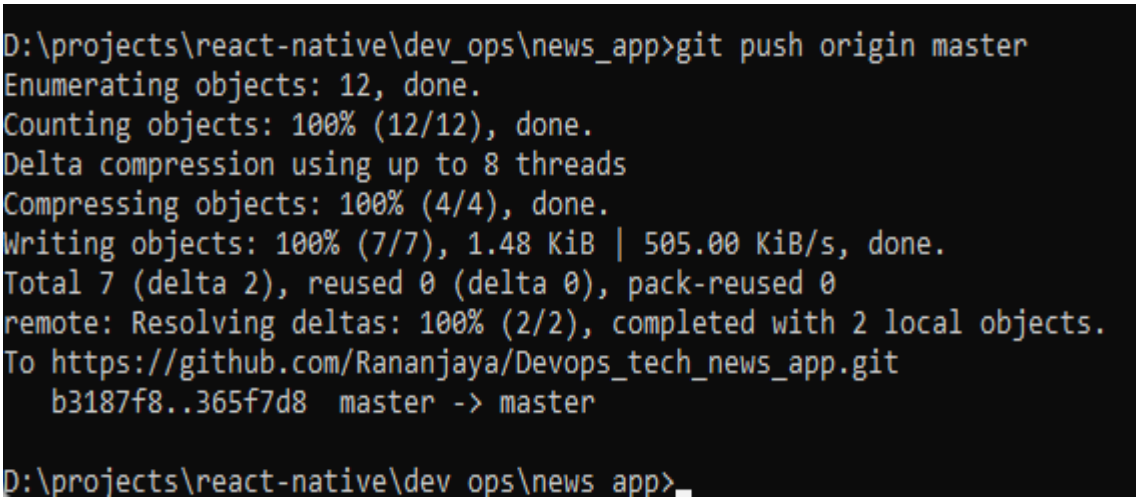


Image 9.2 – Git push command.

3.3Setup GitHub workflow.

For setup the GitHub workflow you need to go to your GitHub dashboard and find your project. After the navigate the project, you need to and find the action tab on the top tab bar in GitHub.

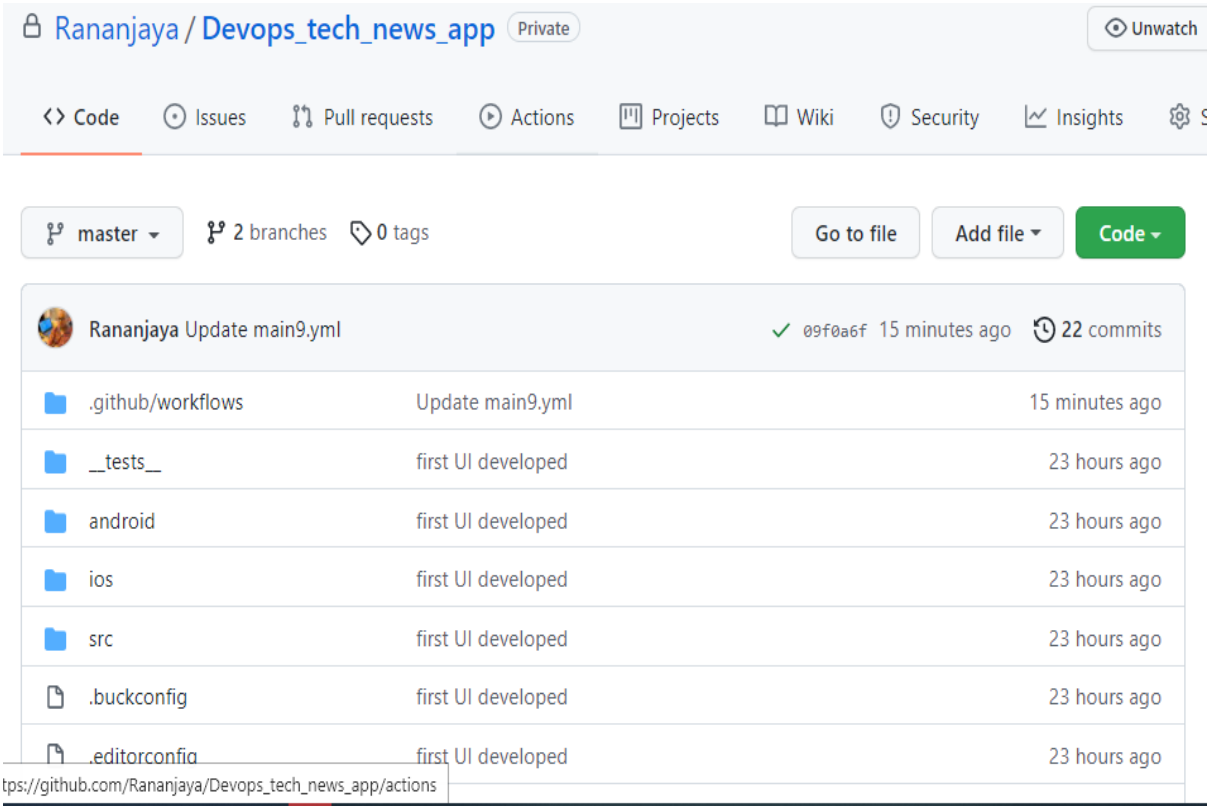


Image 9.3 – GitHub actions tab

When you click the actions, you will be redirected to another new screen called workflow. In there you need to click new workflow button to create new workflow.

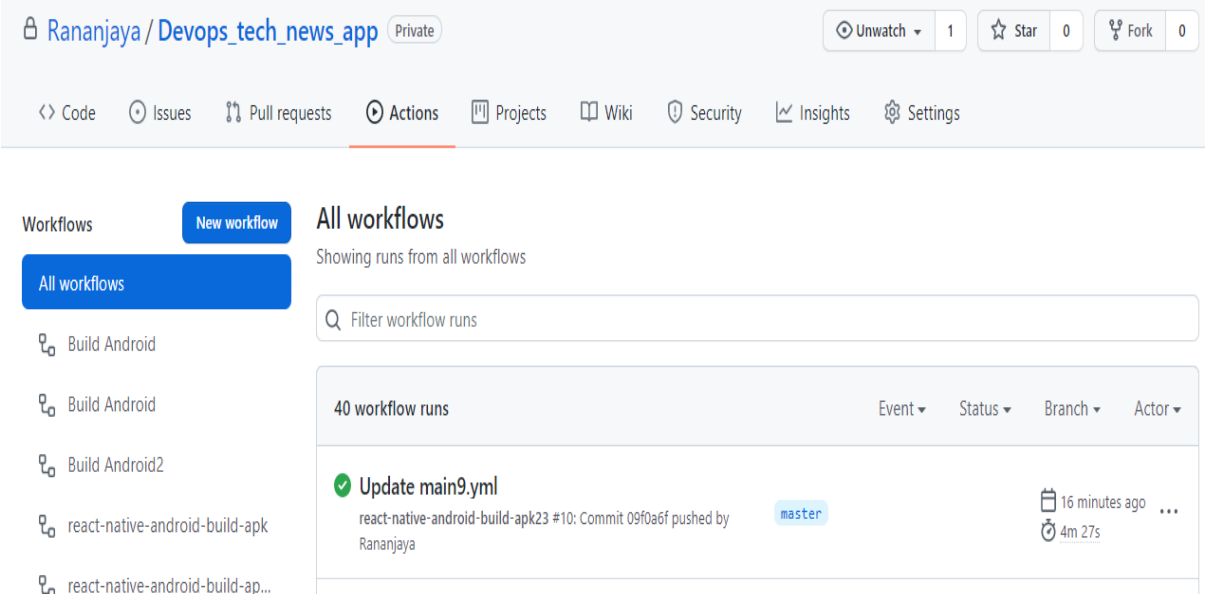


Image 9.4 – create new workflow button.

After that you will redirected to new screen called choose a workflow template.

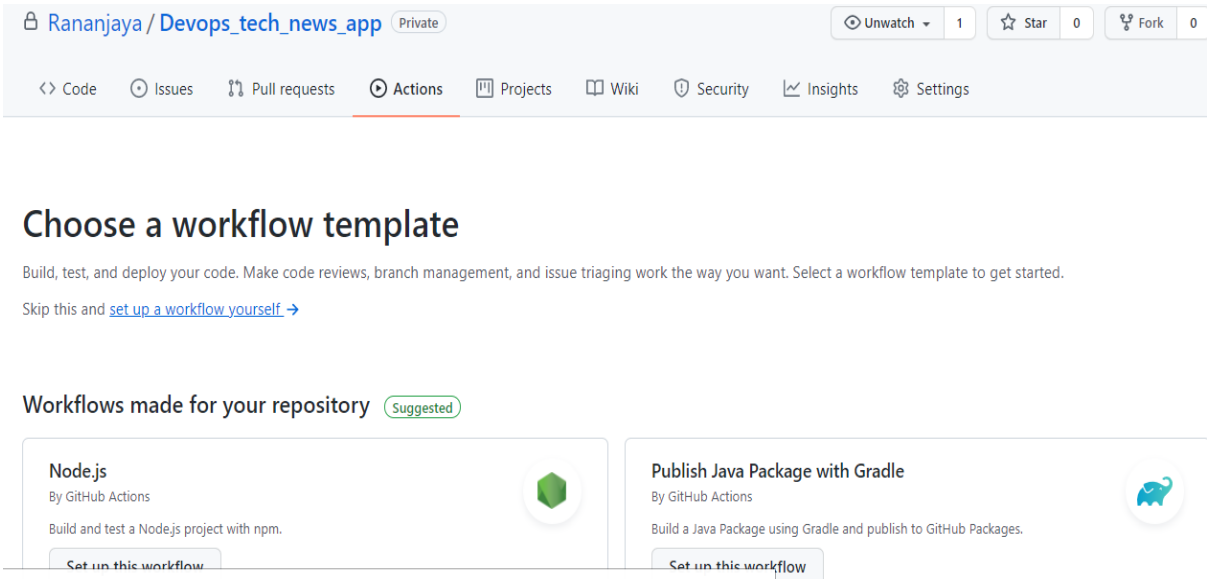


Image 9.5 – choose a workflow template screen

In here you need to click the link called setup a workflow yourself link. When you click that link you will redirect new scree that will be workflow file editor. In tire your need to define your workflow. So ill attached workflow code to here and you can also follow it.

```yaml
name: react-native-android-build-apk23
on:
  push:
    branches:
      - master
jobs:
  install-and-test:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Install npm dependencies
        run: |
          npm install
  build-android:
    needs: install-and-test
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - name: Install npm dependencies
        run: |
          npm install
      - name: Get Permission gradlew
        run: cd android && chmod +x ./gradlew
      - name: Build Android Release
        run: |
          cd android && ./gradlew assembleRelease
```

```
    - name: Upload Artifact
      uses: actions/upload-artifact@v1
      with:
        name: app-release.apk
        path: android/app/build/outputs/apk/release/
```

Image 9.6 – Workflow

in this code first we define the workflow name. and next we need to give branch name when this workflow triggers push that branch. So, I put “Master branch” for this. And next part we define the jobs. Jobs means that what is the process this workflow do to automate this app. so, we define jobs like this.

```
install-and-test:
  runs-on: ubuntu-latest
  steps:
    - uses: actions/checkout@v2
    - name: Install npm dependencies
      run: |
        npm install
build-android:
  needs: install-and-test
  runs-on: ubuntu-latest
  steps:
    - uses: actions/checkout@v2
    - name: Install npm dependencies
      run: |
        npm install
    - name: Get Permission gradlew
      run: cd android && chmod +x ./gradlew
```

```yaml
- name: Build Android Release
  run: |
    cd android && ./gradlew assembleRelease
- name: Upload Artifact
  uses: actions/upload-artifact@v1
  with:
    name: app-release.apk
    path: android/app/build/outputs/apk/release/
```

Image 9.7 – Workflow

in above code first we will define necessary step to checkout some branchers and give some step to build release apk.

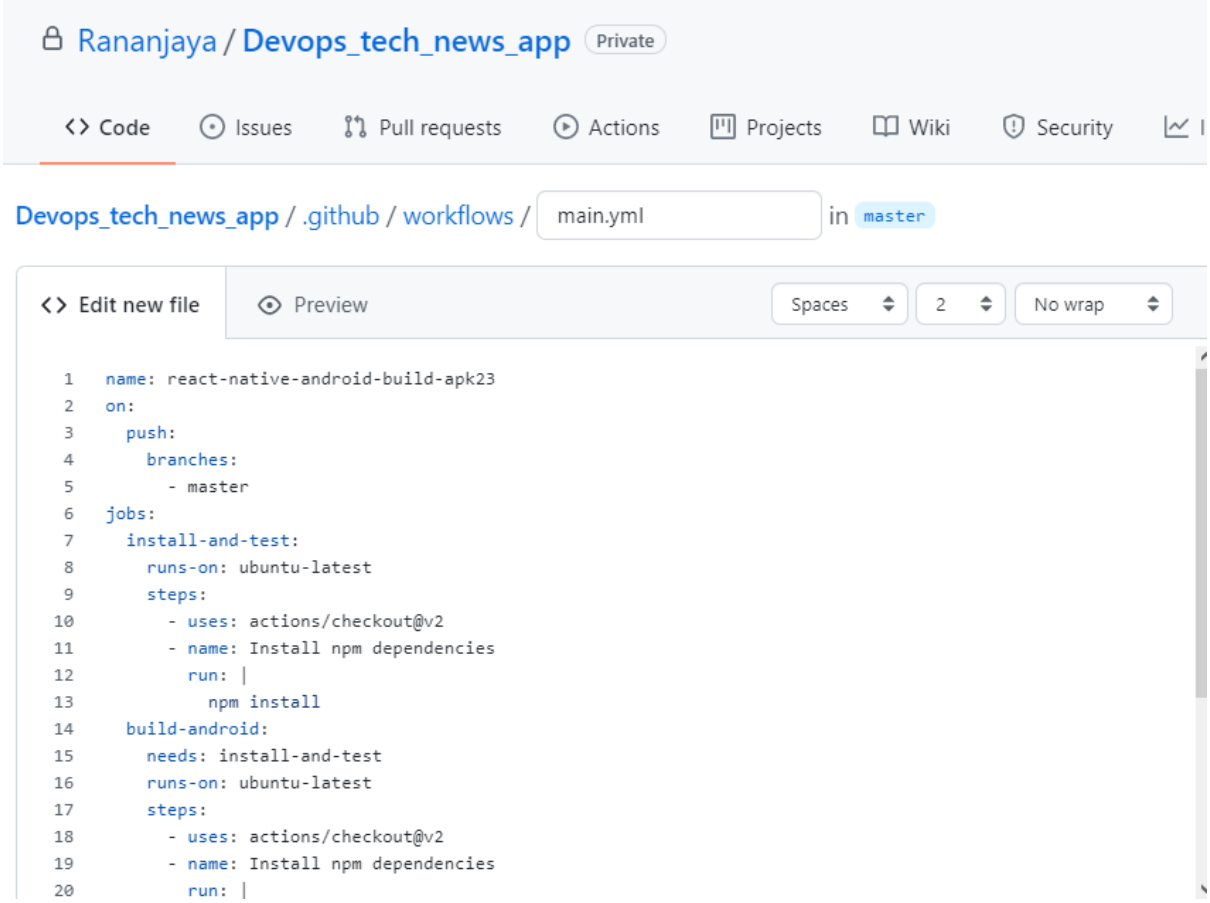


Image 9.8 – workflow defines using YML

After the finished the write workflow code you need to commit it. For that you need to click commit button top left corner.

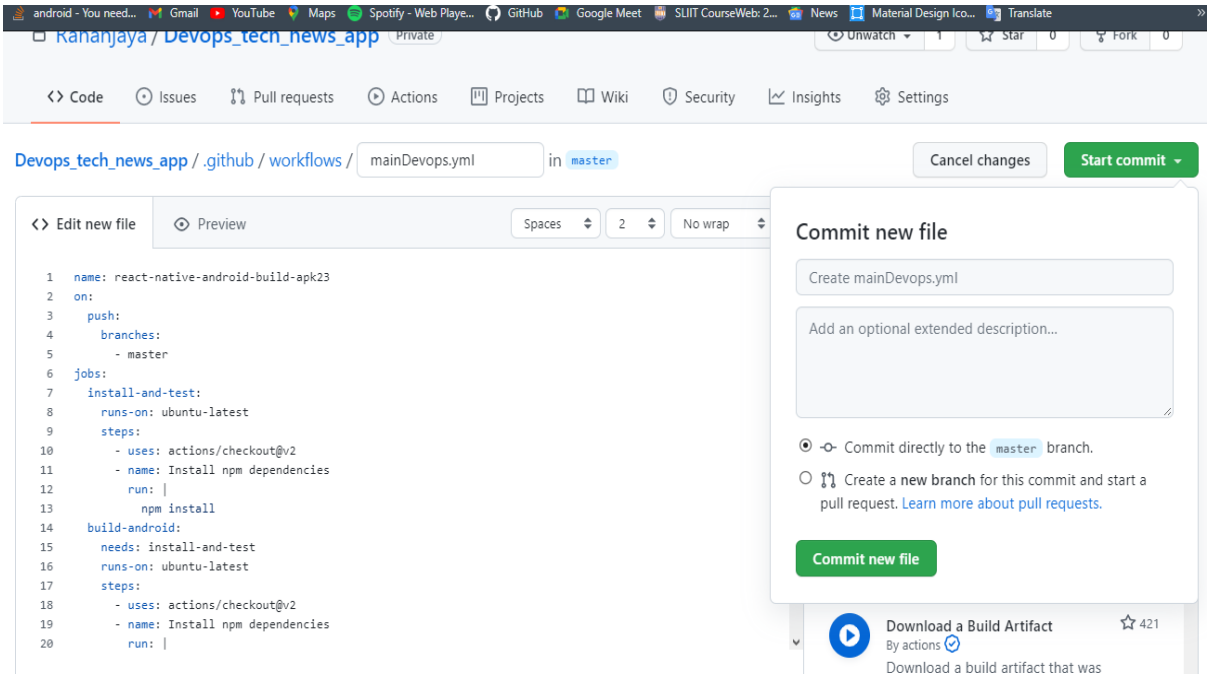


Image 8.3 – commit the changers.

You need to give commit message and description and need to press commit new file button. When you commit the changers, GitHub will create new **main.yml** file.

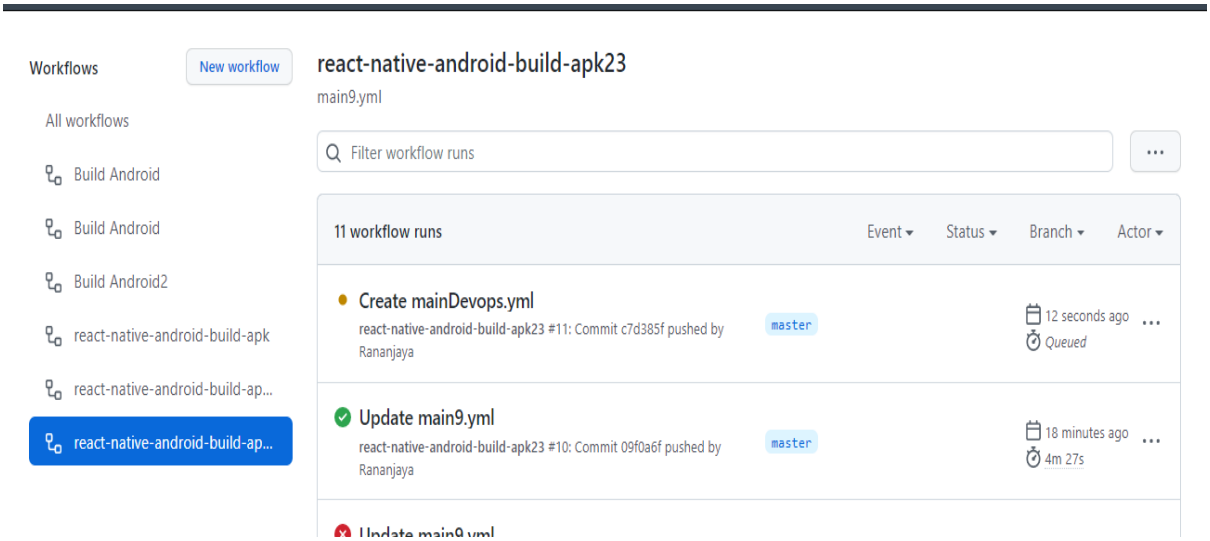


Image 8.4 – create main.yml file

New we need to push the code again to run the automation. Now we will show the step-by-step process of the GitHub workflow here.

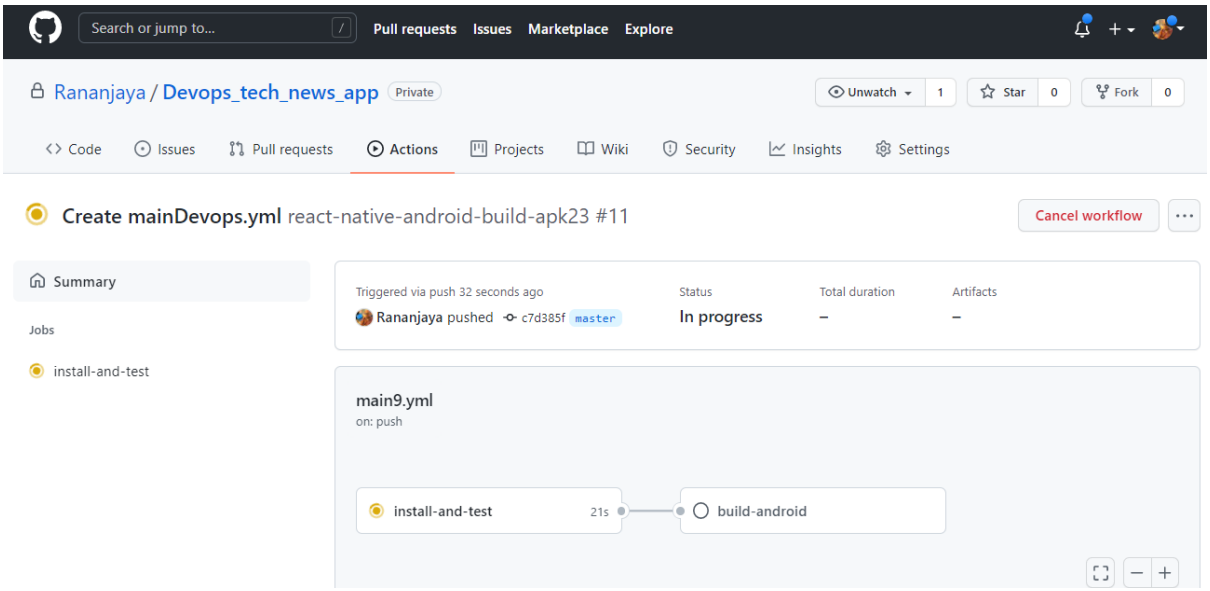


Image 8.5 – workflow running.

According to Image 8.5 he is working begin to run and install and test next image show install and test passed and run other actions.

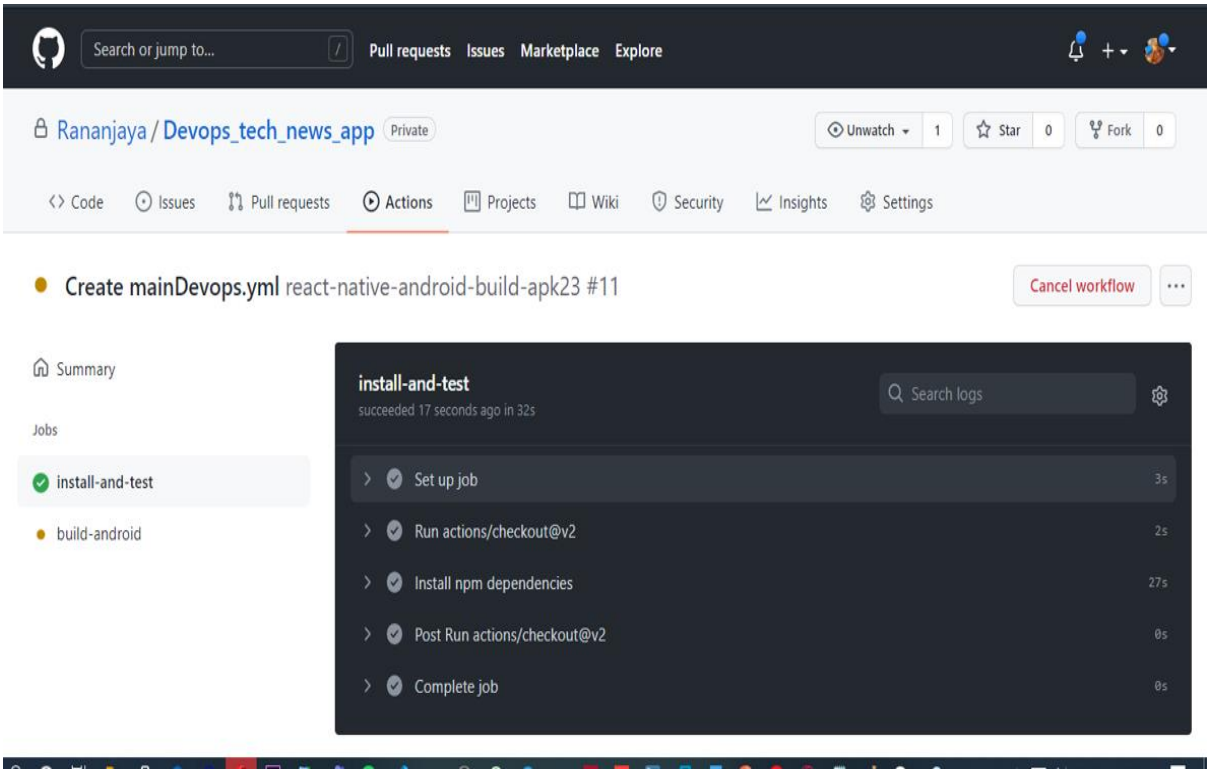


Image 8.6 – workflow running.

In Image 8.6 android build is running. So, you will she in left side all defined jobs are running shows there.

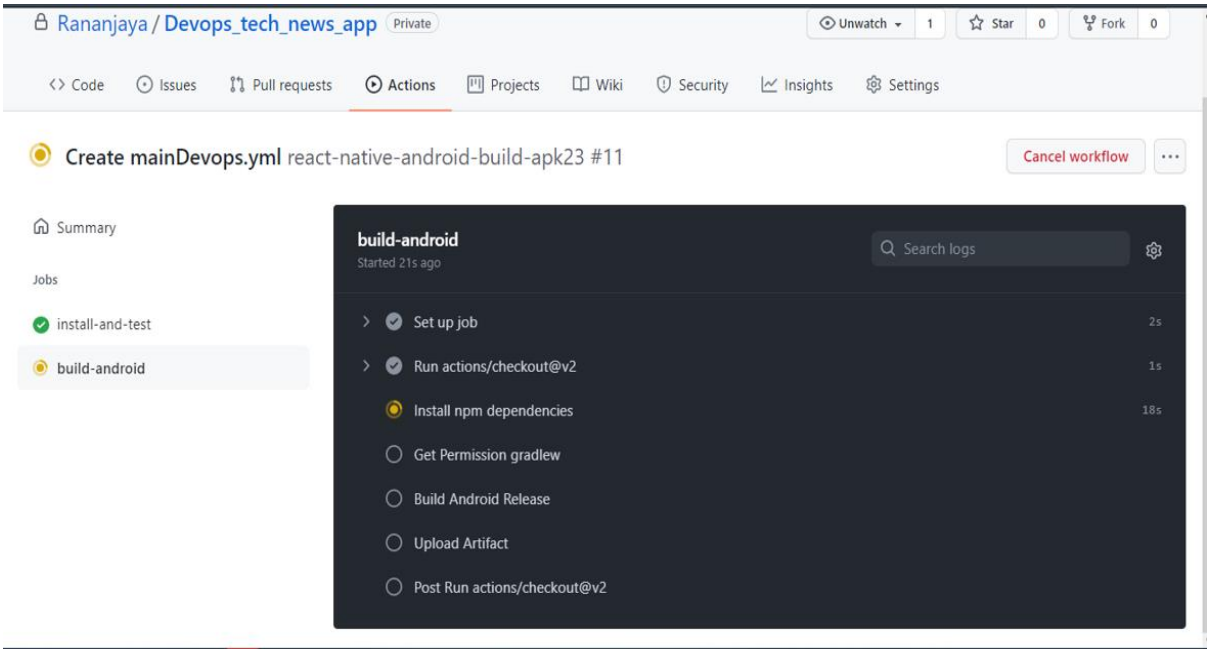


Image 8.7 – workflow running.

According to the Image 8.7 workflow is running for install npm dependencies.

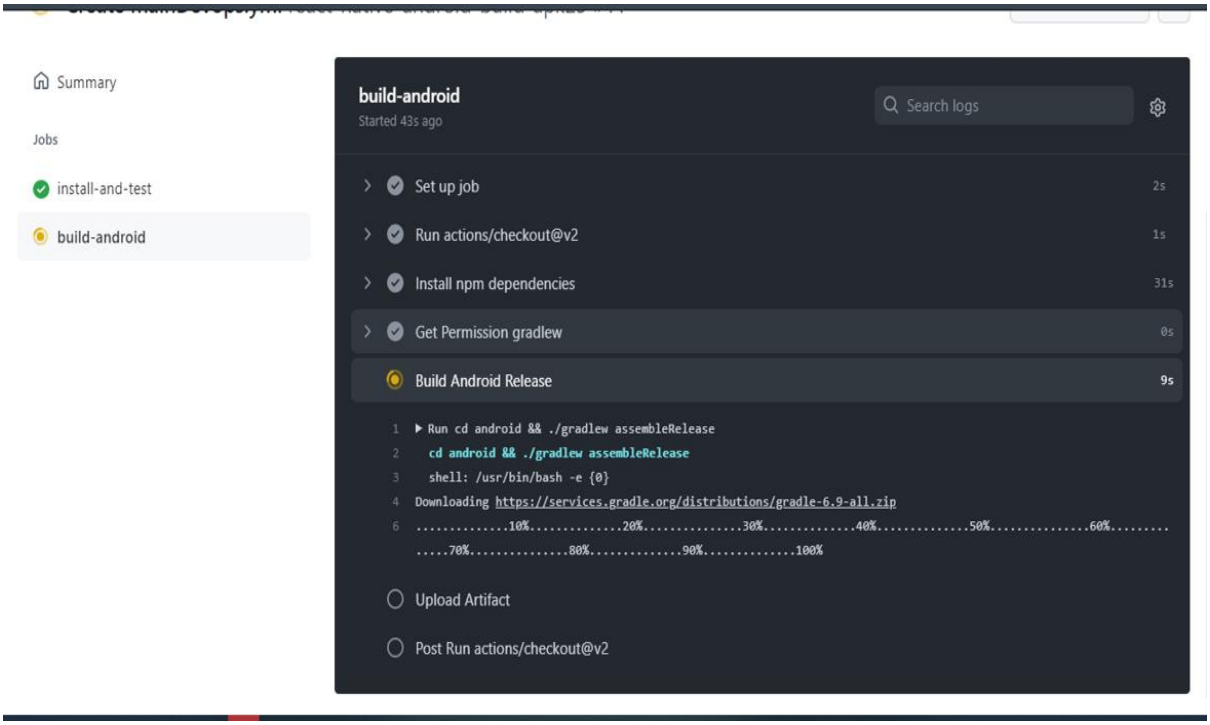


Image 8.8 – workflow running.

According to Image 8.8 workflow is running for build android release. So it will show the process of creating release apk.

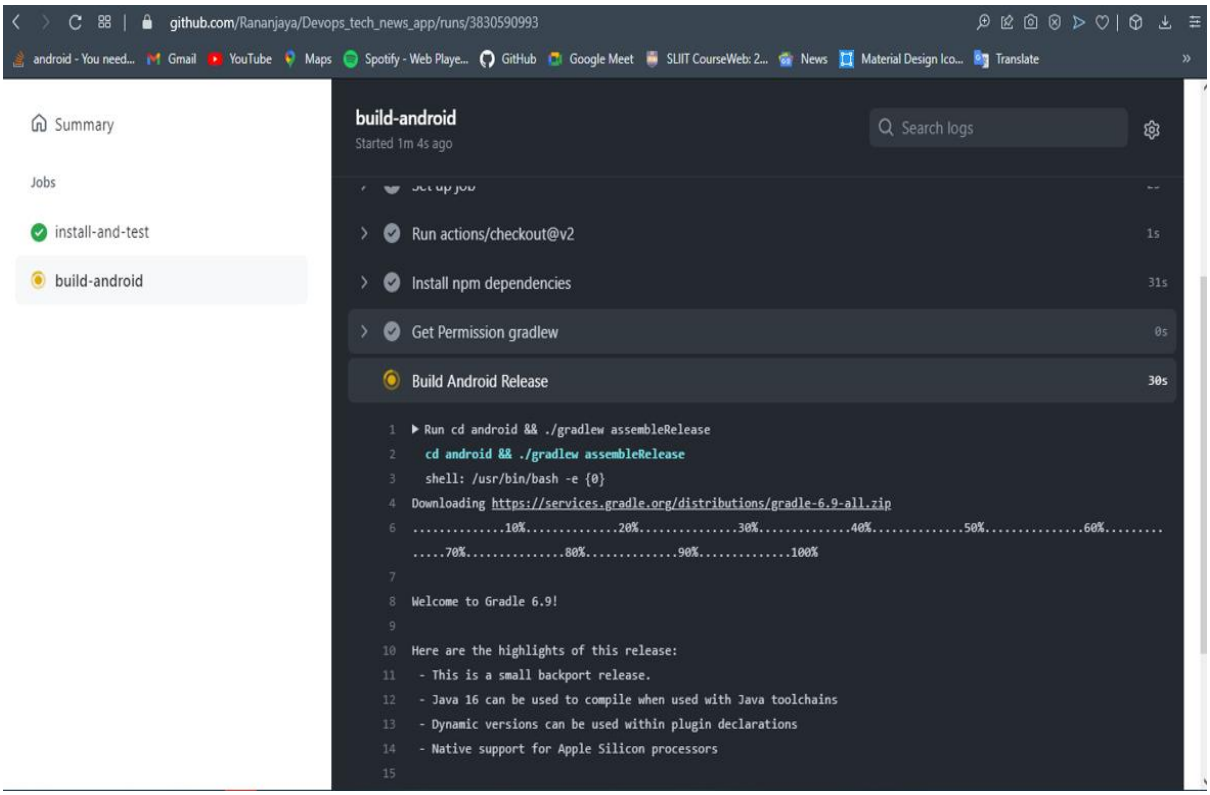


Image 8.9 – workflow running.

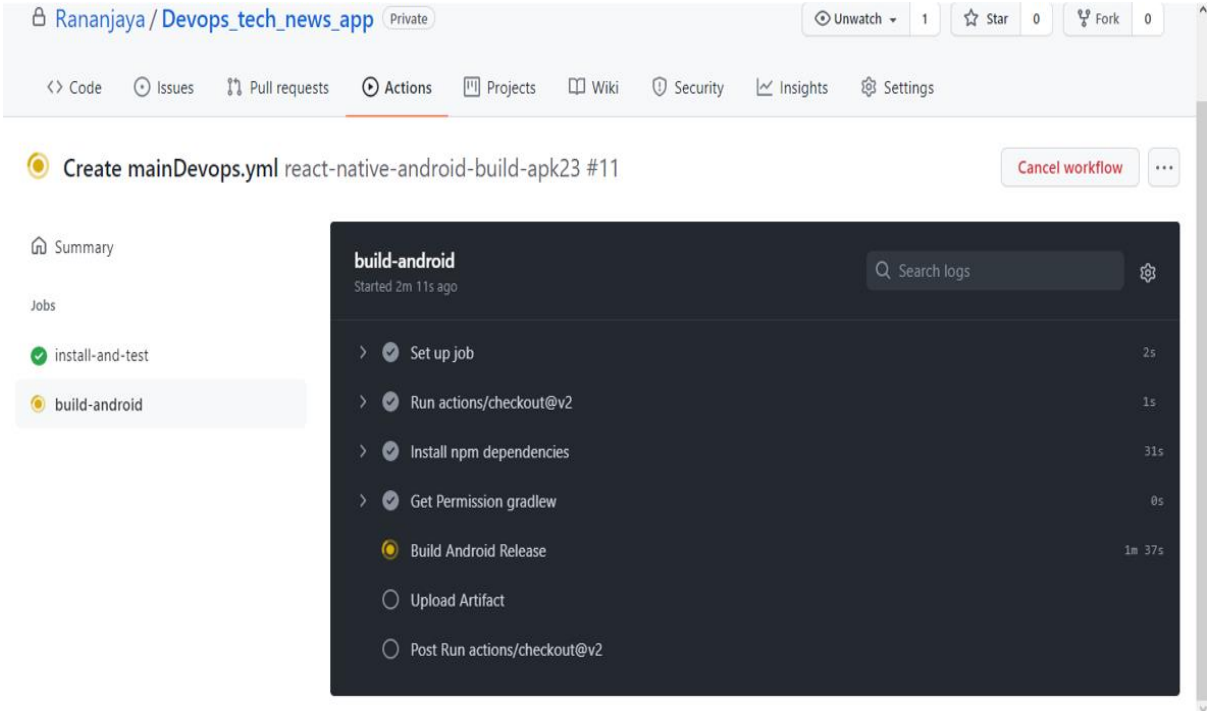
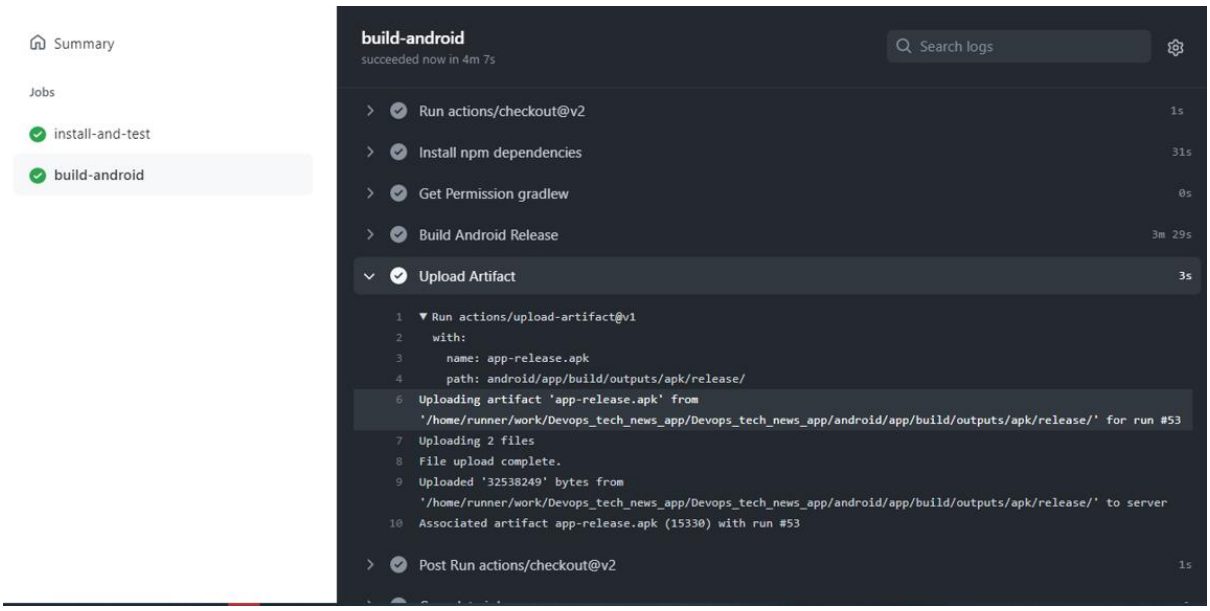


Image 9.0 – workflow running.

Image 9.1 – workflow running.



According to Image 9.1 workflow running for uploading the artifact.

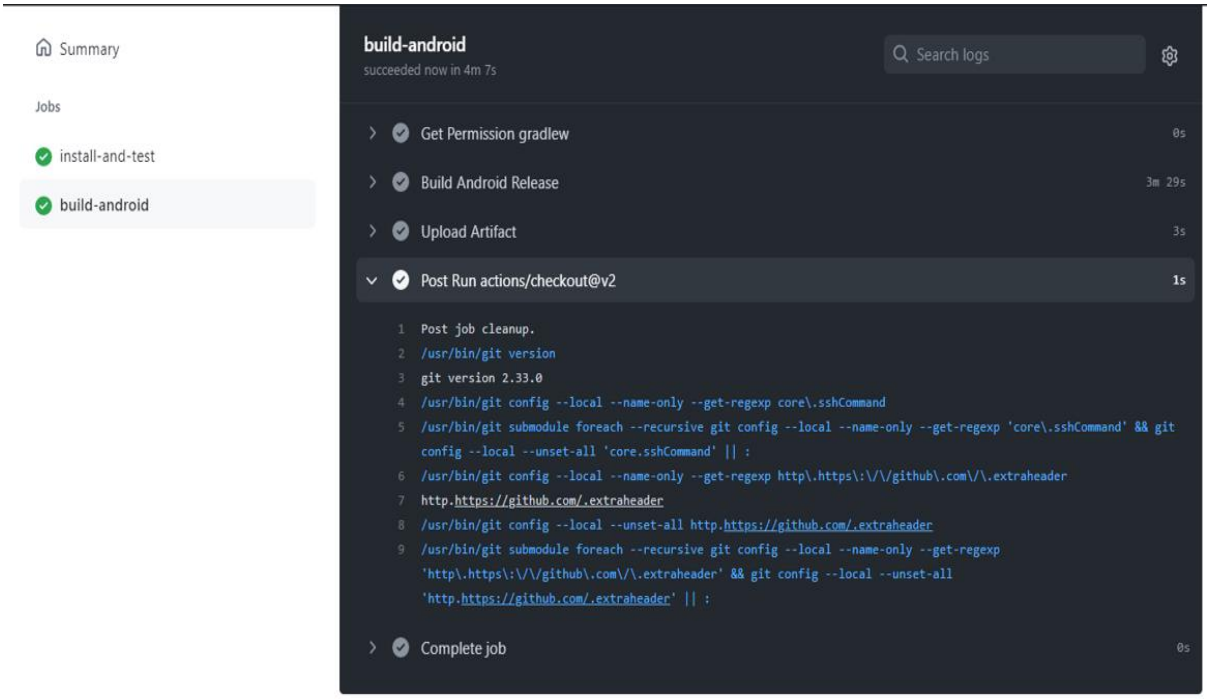


Image 9.2 – workflow running.

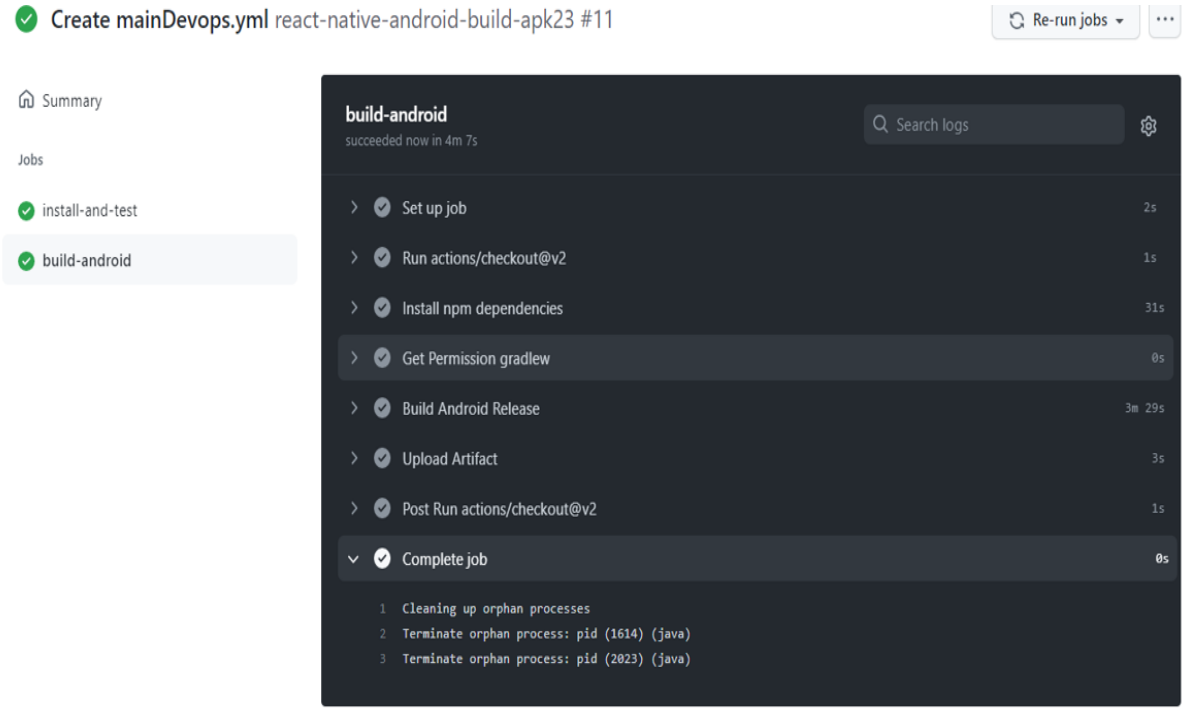


Image 9.3 – workflow running.

Image 9.3 show the completed job process.

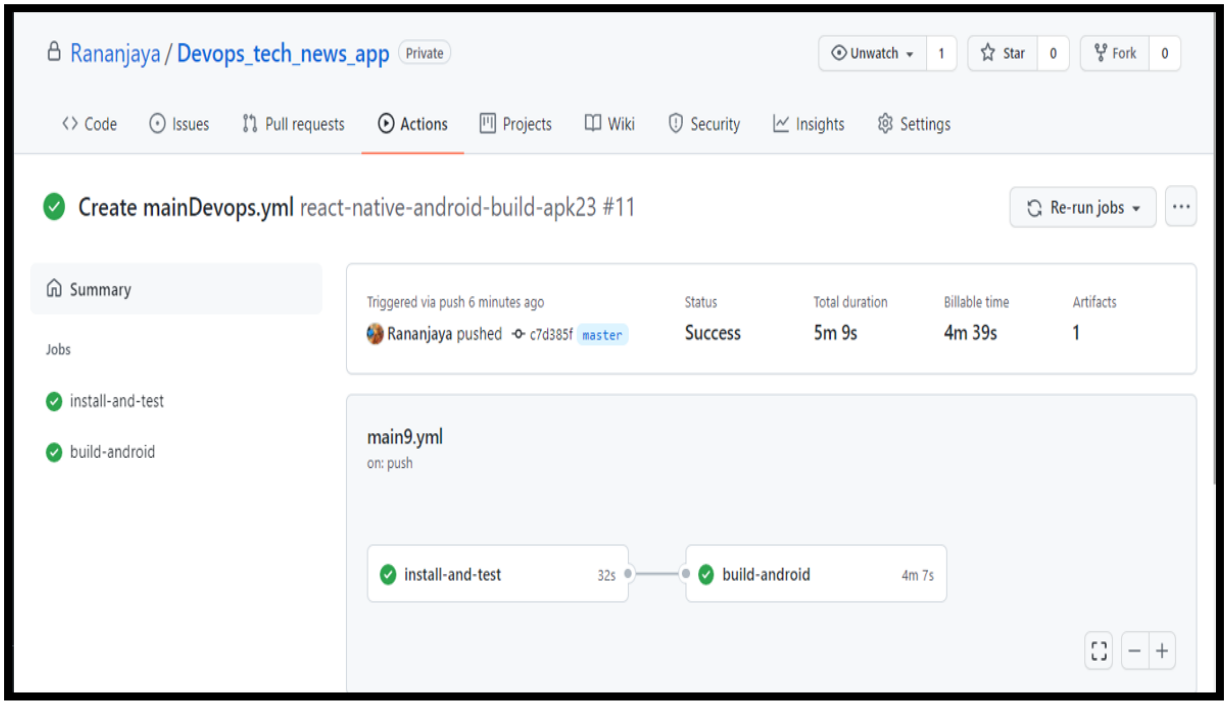


Image 9.4 – workflow running.

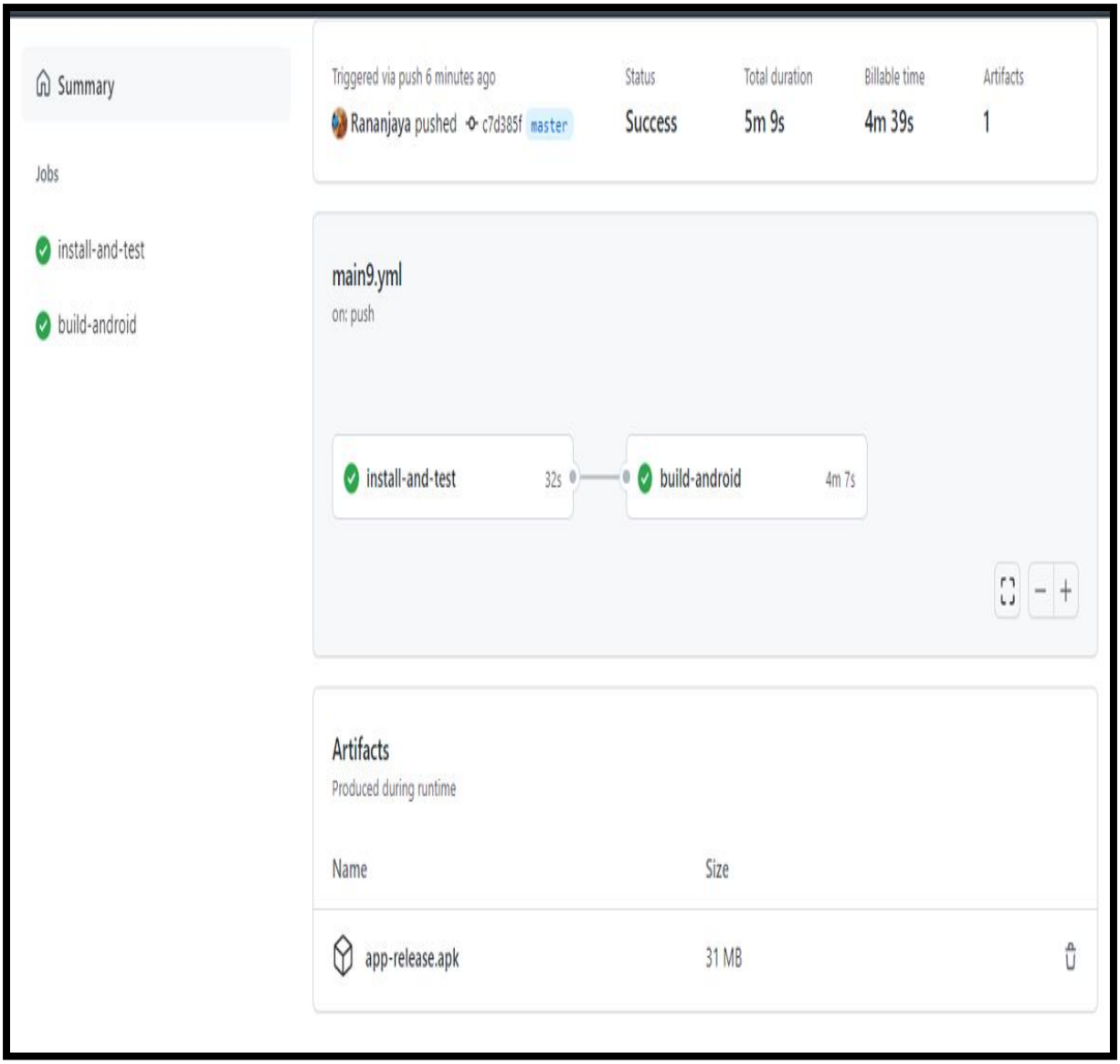


Image 9.6 – workflow running.

According to the image 9.6 you will see our app is build and apk now can be download. So, our book is covering all the necessary parts how to develop mobile app using react native and some basic understanding GIT and how to instigate CICD to the react native mobile app.

References

[1] GitHub. 2021. GitHub: Where the world builds software. [online] Available at: <https://github.com> [Accessed 26 November 2021].

[2] GitHub Docs. 2021. Workflow syntax for GitHub Actions - GitHub Docs. [online] Available at: <https://docs.github.com/en/actions/learn-github-actions/workflow-syntax-for-github-actions> [Accessed 26 November 2021].

[3] Reactnative.dev. 2021. React Native · Learn once, write anywhere. [online] Available at: <https://reactnative.dev> [Accessed 26 November 2021].