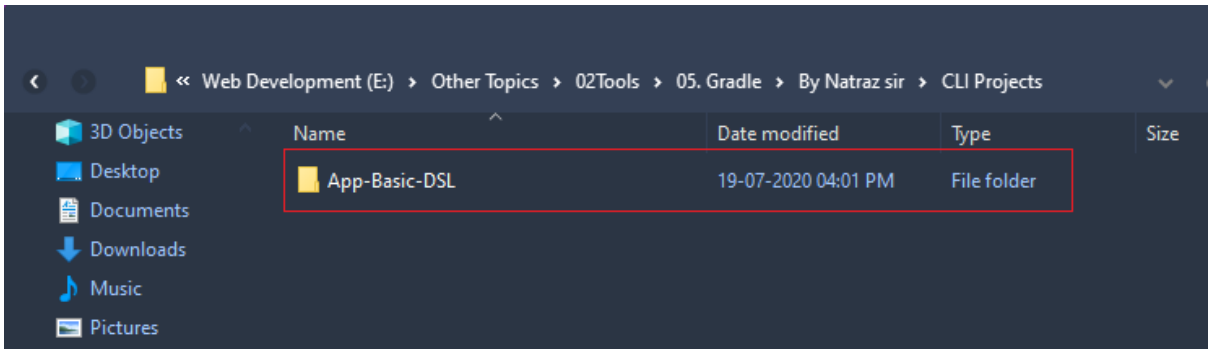
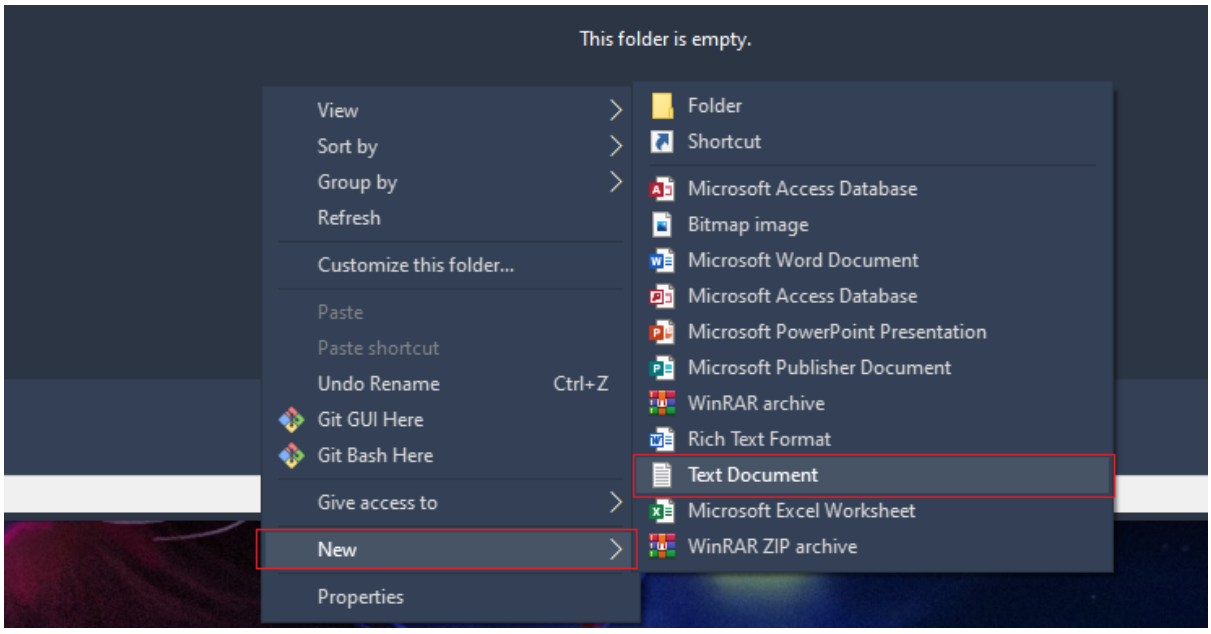
# Gradle CLI Projects: -

## Project 1: Basics of DSL [Domain Specific Language]

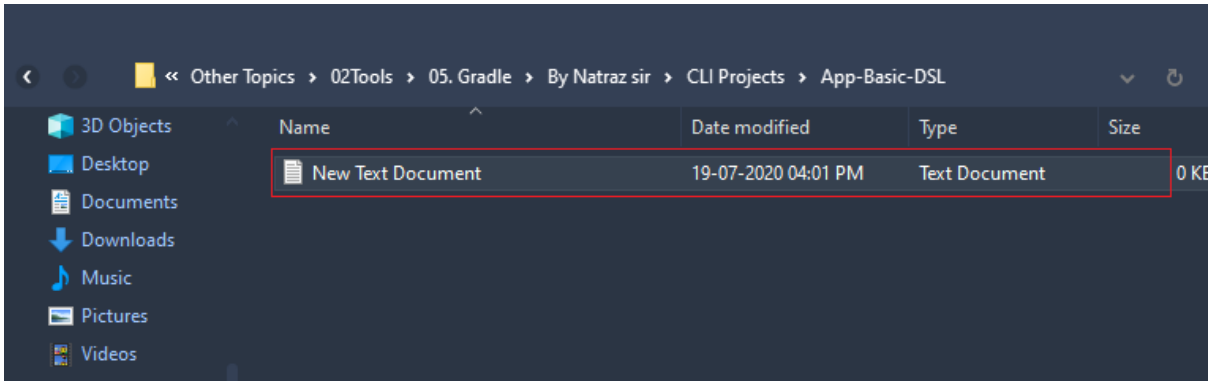
Step #1: Create a folder for the application, like App1-Basic-DSL



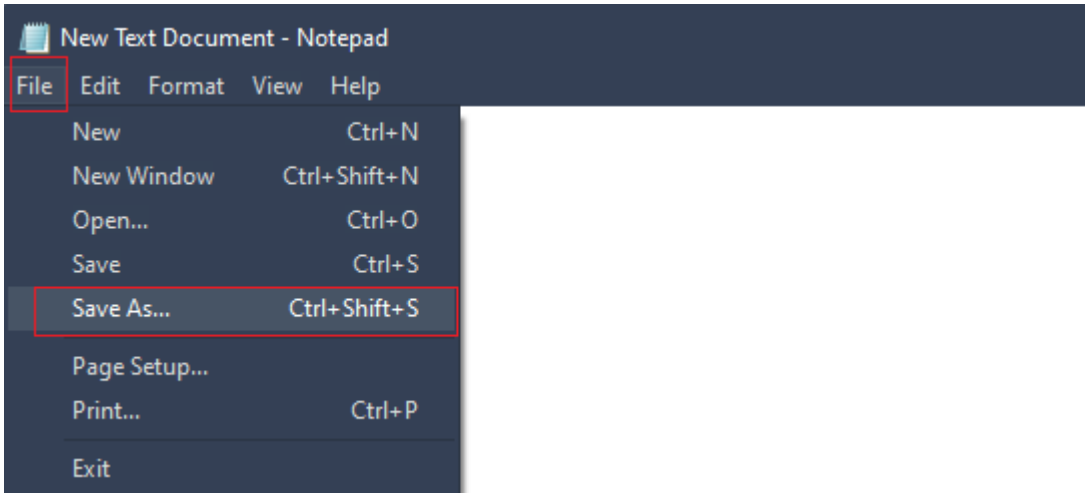

Step #2: Go inside the folder create a Text Document, Right click then new choose Text Document



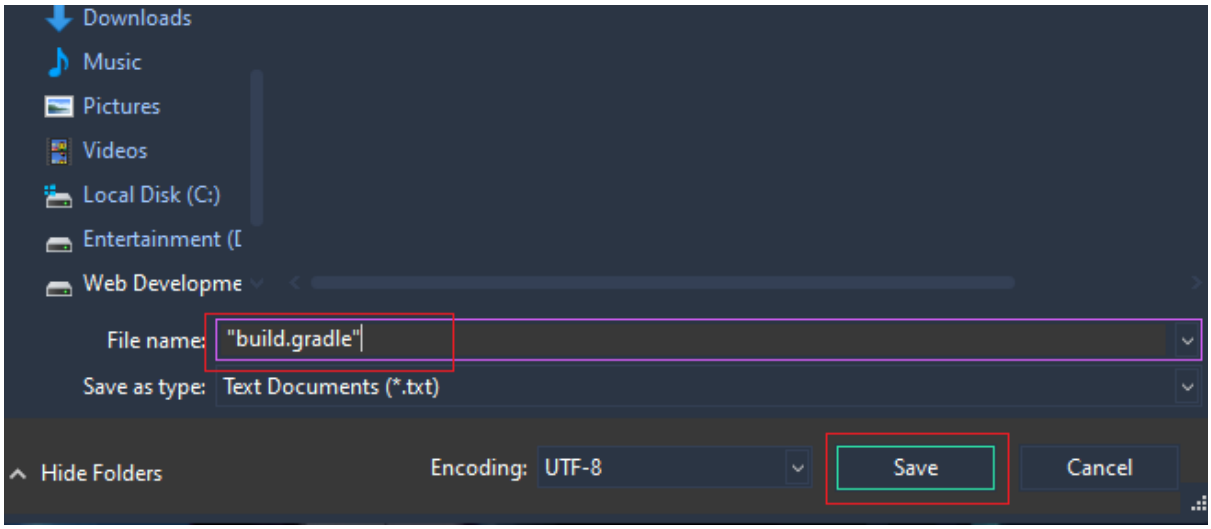

Step #3: Open the New Text Document



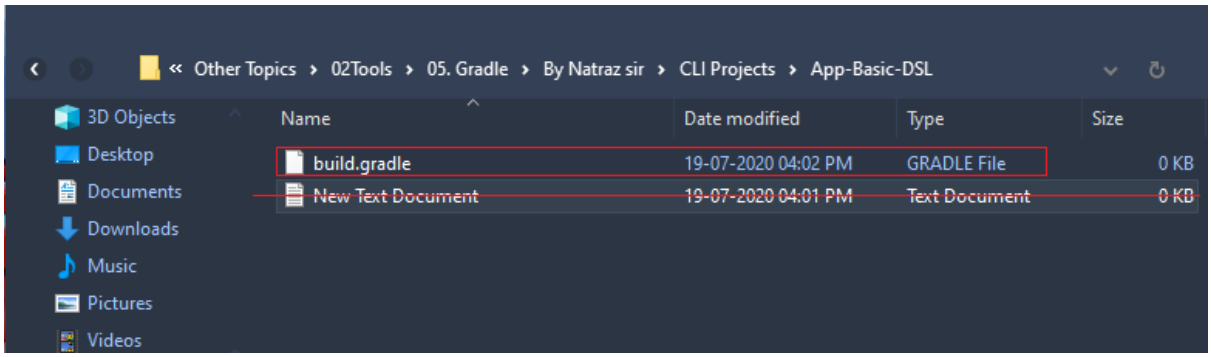
Step #4: Click on file go to Save As option



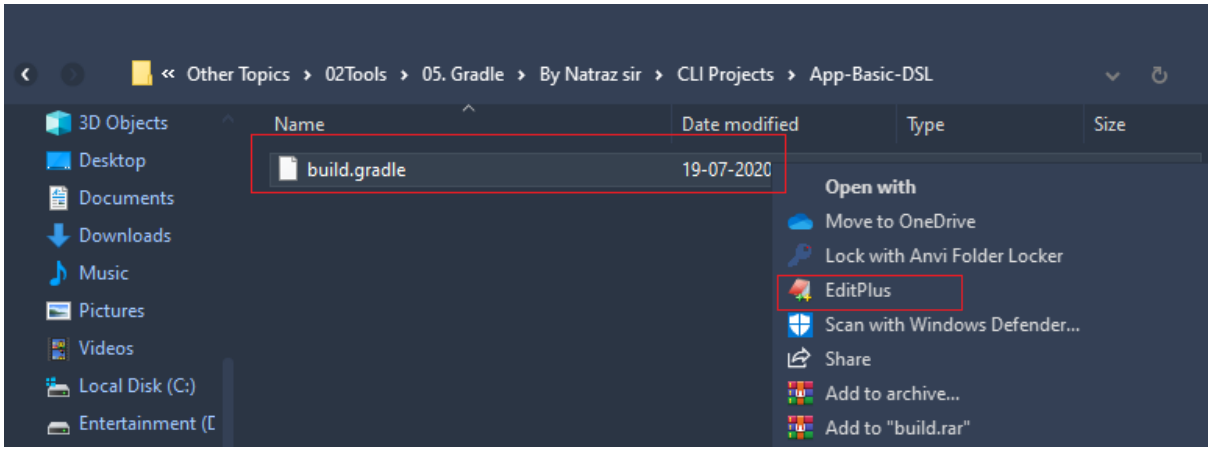

Step #5: Give the file name as “build.gradle” then click on Save [give the name in double cot then the default extension will change to gradle]




Step #6: Now you can see the build.gradle file, Delete the New Text Document file

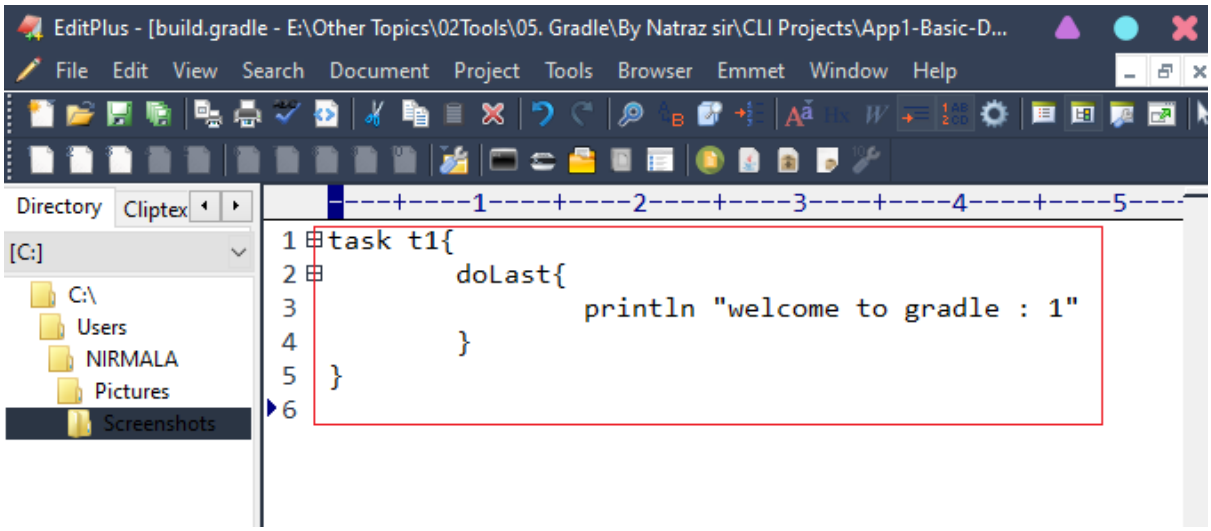


Step #7: Open the build.gradle file with any editor [like Edit Plus]. Right click then Edit Plus

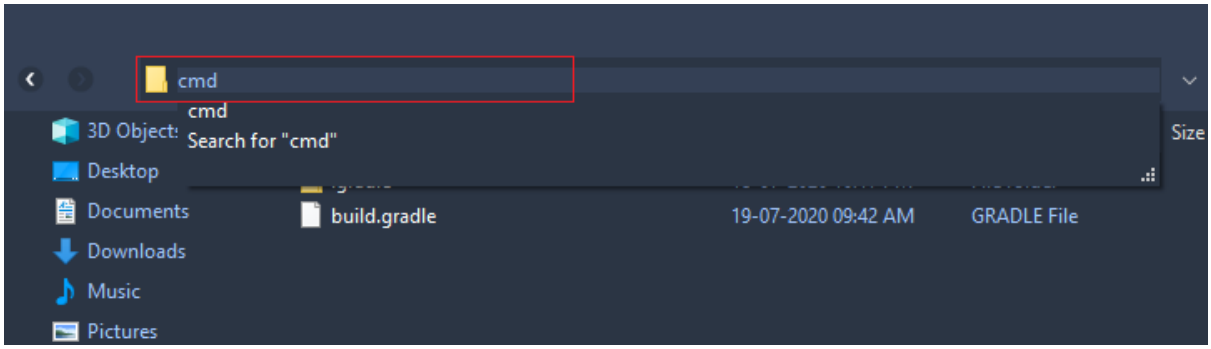



Step #8: Create a task like below

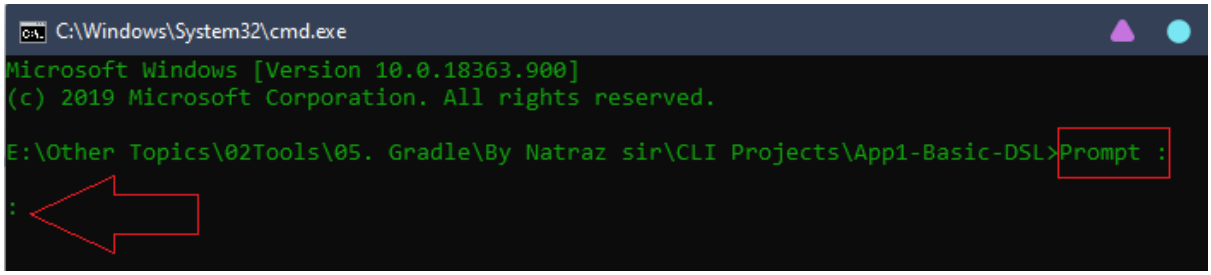
Here “t1” is the task name and the “doLast” is a method having a print statement to print, then save it



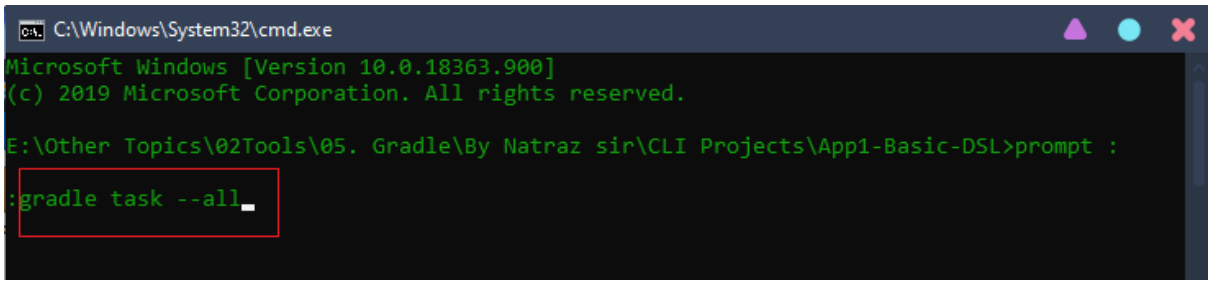

Step #9: Then go to the location where you build.gradle file is present, open Command prompt by type cmd in the address bar of the directory then hit enter



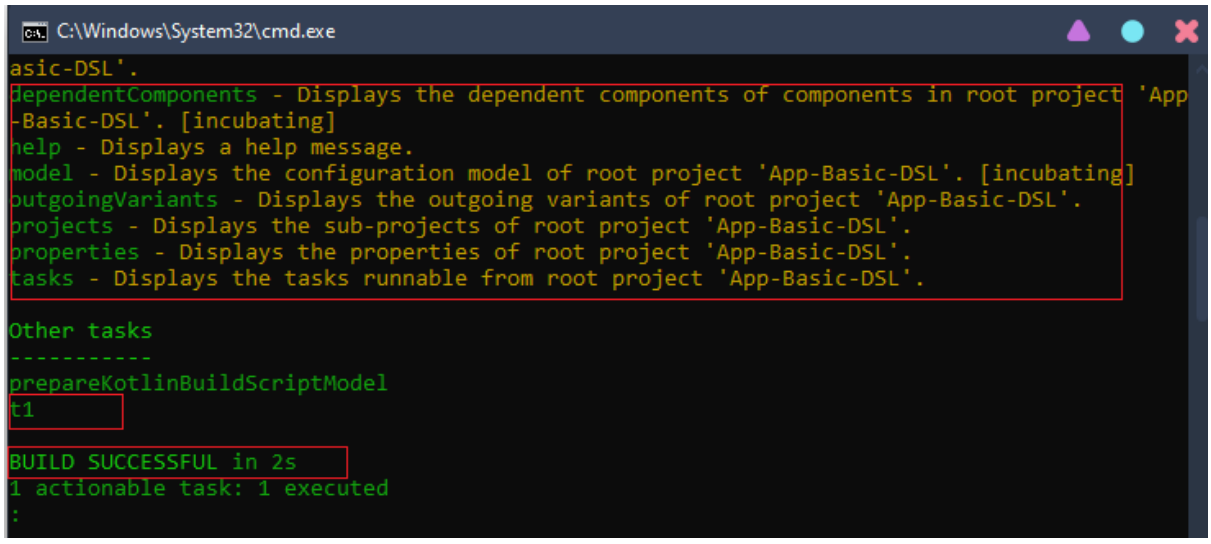
Step #10: Now Command Prompt is open, use "prompt:" to minimize the address like below you will get only ":" symbol



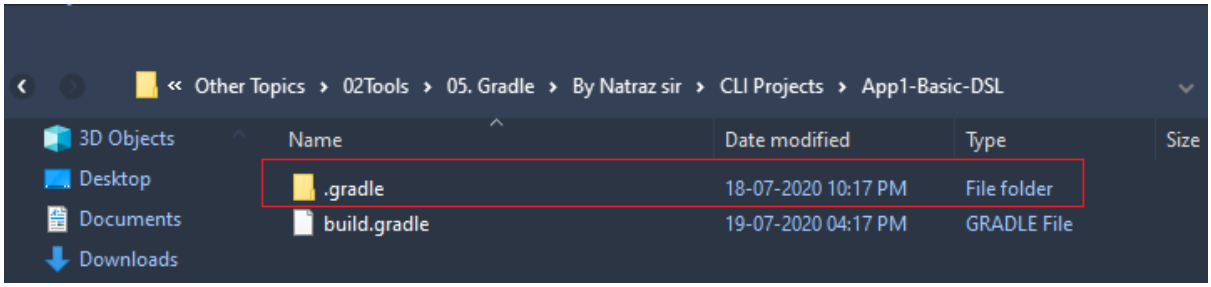

Step #11: Type "gradle task --all" then hit enter to get all the tasks




Step #12: Now you can see your task t1 is appear with all the default task and you also get a message BUILD SUCCESSFUL




Step #13: After executing this command in your directory a .gradle folder will be created



Step #14: Then execute your task, by typing "gradle t1" in your command prompt

```
-Basic-DSL'. [incubating]
help - Displays a help message.
model - Displays the configuration model of root project 'App-Basic-DSL'. [incubating]
outgoingVariants - Displays the outgoing variants of root project 'App-Basic-DSL'.
projects - Displays the sub-projects of root project 'App-Basic-DSL'.
properties - Displays the properties of root project 'App-Basic-DSL'.
tasks - Displays the tasks runnable from root project 'App-Basic-DSL'.

Other tasks
-----------
prepareKotlinBuildScriptModel
t1

BUILD SUCCESSFUL in 2s
1 actionable task: 1 executed
:gradle t1
```

Step #15: Now you can see your task contain message is executed and also get the message BUILD SUCCESSFUL

```
BUILD SUCCESSFUL in 2s
1 actionable task: 1 executed
:gradle t1

> Task :t1
welcome to gradle : 1

BUILD SUCCESSFUL in 1s
1 actionable task: 1 executed
:
```

```
task t1{
        doLast {
                println "welcome to gradle1"
        }
}
```

```
task t1 {
        println "welcome to gradle2"
}
```

```
task t1
t1{
        println "welcome to gradle3"
}
```

```
task t1
        t1{
                doLast {
                        println "welcome to gradle4"
                }
        }
}
```

Execute all the task one by one by commenting others

/* */

CMD> gradle task –all

CMD> gradle <task_name>

## Project 2: Basics of DSL with Dependency

Step #1: Follow Step-1 to Step-7 from Project-1 to create second application named as App2-Basic-DSL-Dependency

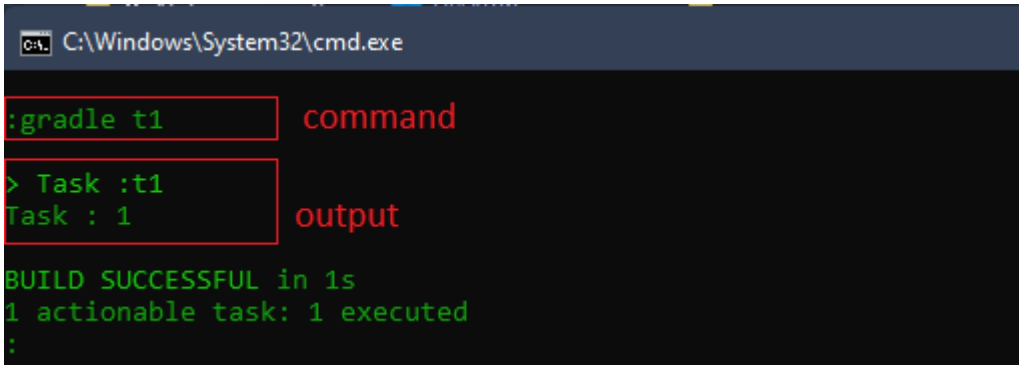
Step #2: Type the below codes in the build.gradle file and save it

```
task "t1"
task ("t2")
task t3
task t4
t1 {
        doLast {println "Task : 1"}
}
t2 {
        doLast {println "Task : 2.a-Last"}
}
t2 {
        doLast {println "Task : 2.b-Last"}
        doFirst {println "Task : 2.b-First"}
}
t3 {
        dependsOn t2, t1
        doLast {println "Task : 3"}
}
t4 {
        doLast {println "Task : 4"}
}
t4.dependsOn t3
defaultTasks 't4'
```

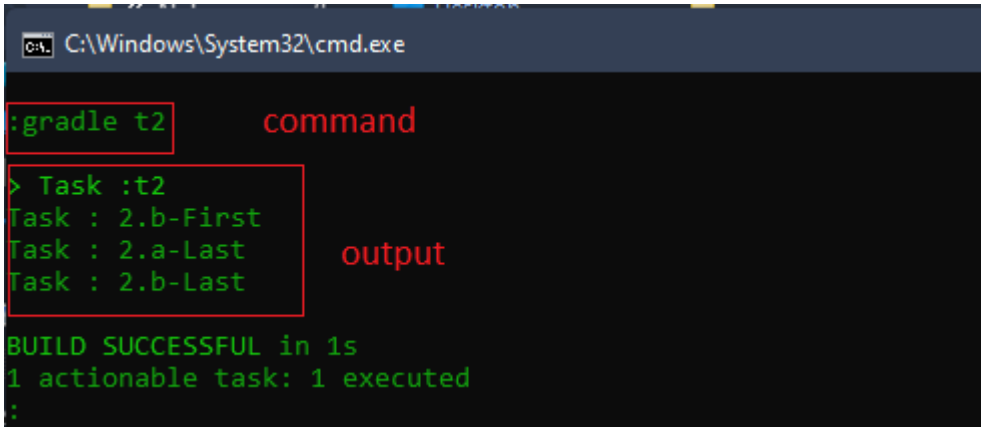
Step #3: Follow Step-9 to Step-11 from Project-1 to ready you project for execution

Step #4: Execute tasks one by one t1, t2, t3, t4 then you will get the following outputs

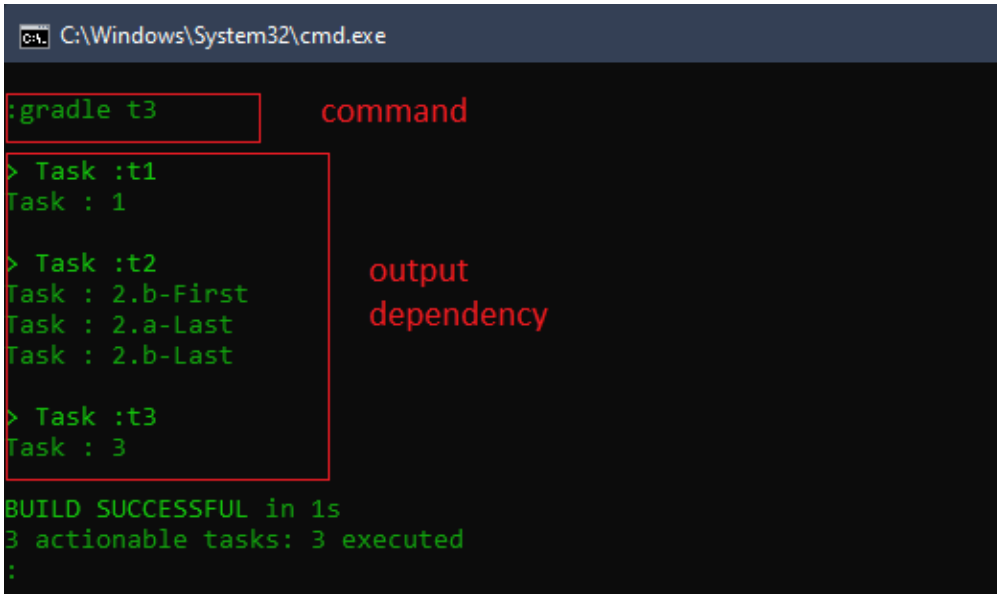
Task 1: t1



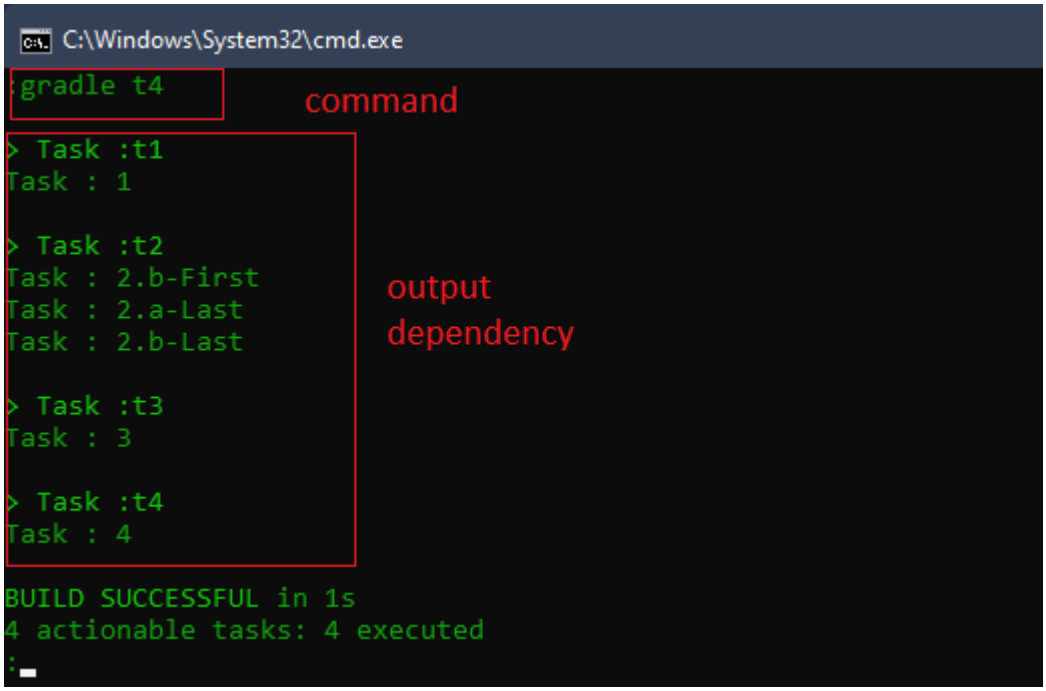

Task 2: t2




Task 3: t3



Task 4: t4




## Project 3: Basics of DSL using Java plugin

Step #1: Create the following Directory structure for application named as App3-java, it is a standalone project using Java plugin

```
E:\GradleProject
        |→ App3-Java
                |→ src
                |      |→main
                |      |      |→ java
                |      |              |→ com.nt.basics
                |      |                          |→Arithmetic.java
                |      |→ test
                |              |→ java
                |→ build.gradle
```

Step #2: Open Arithmetic.java with edit plus and write the following code

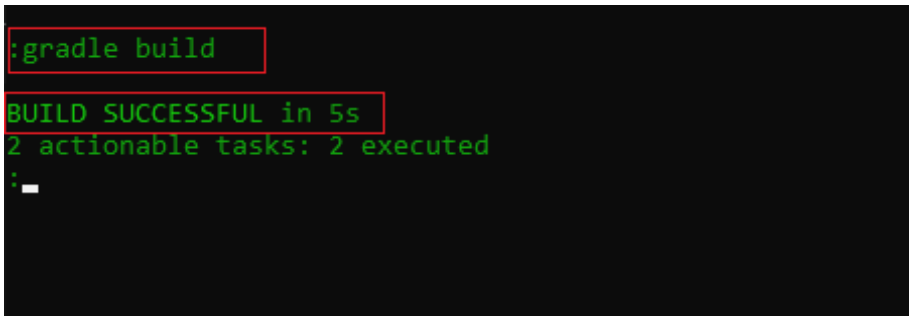
```
//Arithmetic.java
package com.nt.basics;
public class Arithmetic {
	public int sum(int a, int b) {
		return a+b;
	}
	public static void main(String[] args) {
		System.out.println("Welcome to Gradle : java");
		Arithmetic ar = new Arithmetic();
		System.out.println("Result is : "+ar.sum(100,200));
	}
}
```

Step #3: Open build.gradle with edit plus and write the following code

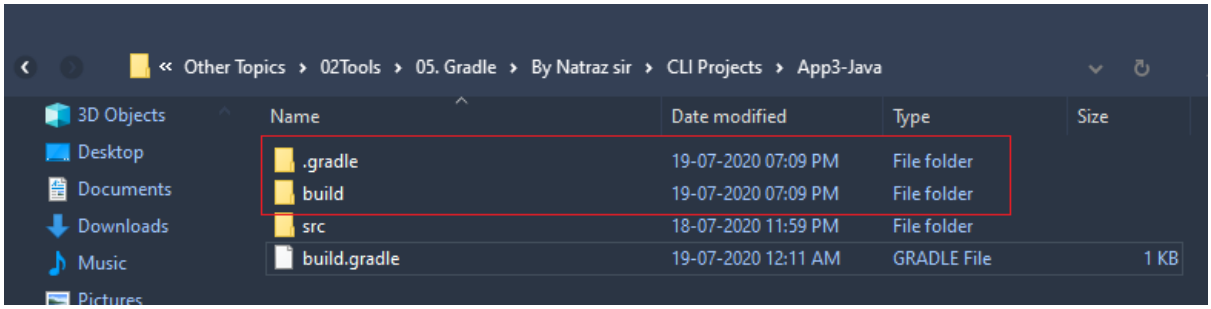
```
apply plugin:'java'
version="1.0"
```

Step #4: Follow Step-9 to Step-11 from Project-1 to ready you project for execution [optional here]

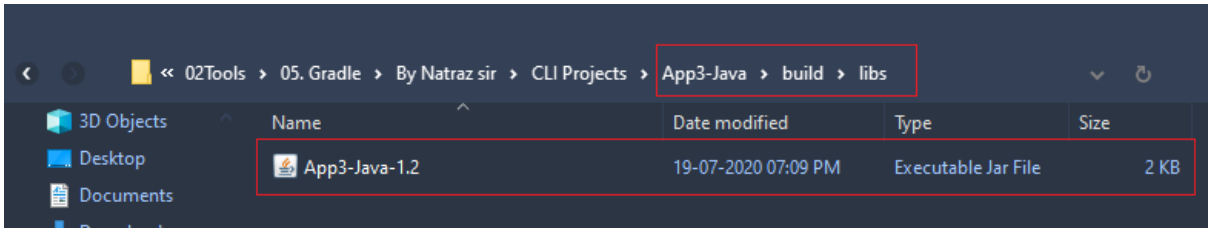
Step #5: To execute your code use “gradle build” command prompt to ready your project for execution and you get also BUILD SUCCESSFUL message [internet connection is required Here]



Step #6: You can see in your Project directory two folder ".gradle and build" is create after the gradle build



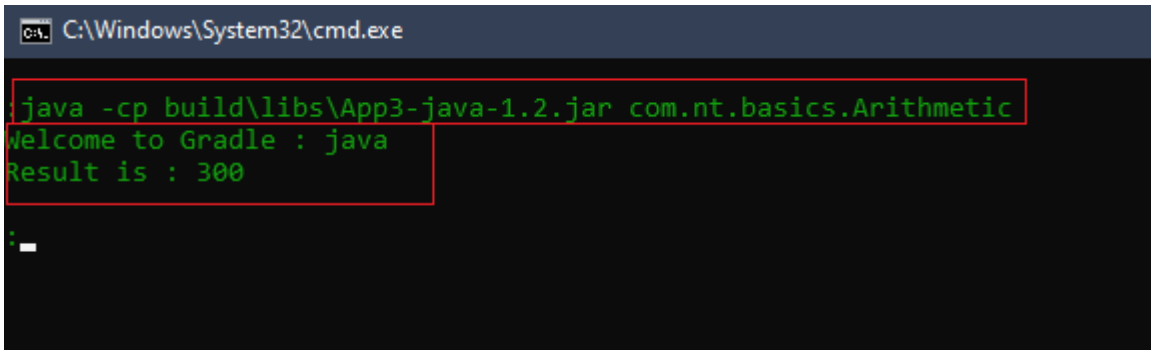

Step #7: Go to build folder inside go to libs then you will get the application jar file



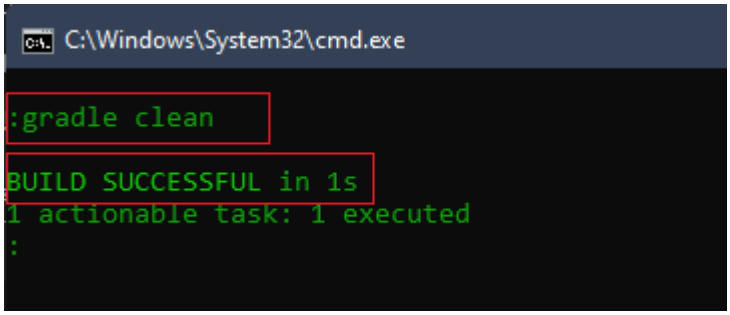

We execute our application through jar file because there is no by default "run" tasks for execute the application, here we run manually

Step #8: Go to command prompt of the project directory home and type the following command and you get the output

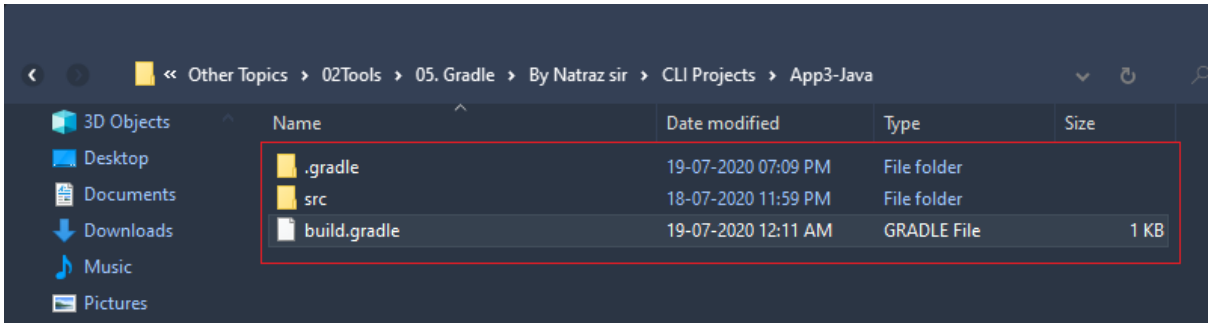
CMC> java -cp build\libs\<Jar_file_name>.jar <Package>.Java_File_Name




Step #9: To clean your gradle project use "gradle clean" command



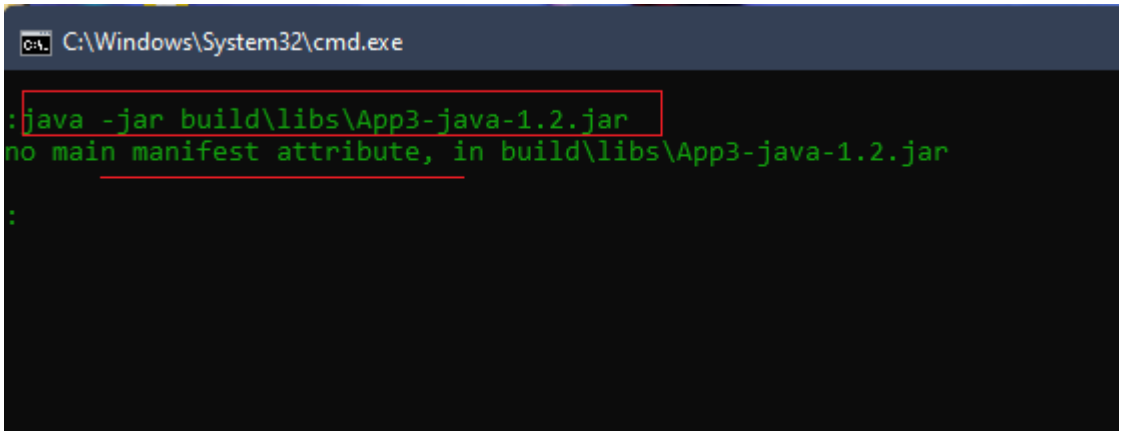
Step #10: It delete the build folder only




Step #11: To make our Jar is executable jar

Executable jar means we can execute the jar using following command

CMD> java -jar build\libs\<Jar_Name>.jar




we have to add manifest attribute having main class name.

To Make our Jar is executable the following details should be available in MANIFEST.MF file which is available inside META-INF folder of jar file like below

MANIFEST.MF

---

Manifest-Version: 1.0

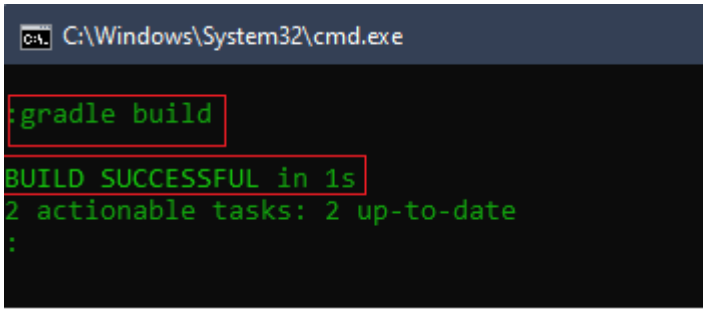
Main-class: com.nt.basics.Arithmetic

We get all the setup directly in gradle by using the following steps

Step #12: Open build.gradle and write the following code and save it

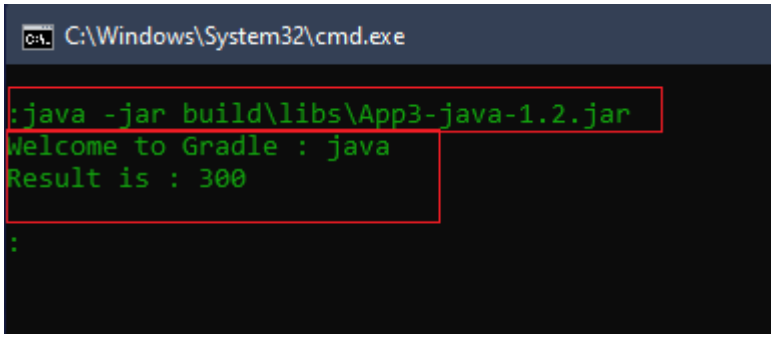
```
apply plugin:'java'

version="1.2"

jar {

        manifest {

                attributes 'Main-Class': 'com.nt.basics.Arithmetic'

        }

}
```

Step #13: Then go to the Command prompt of the directory and done the "gradle build" and you get BUILD SUCCESSFUL message with up-to-date




Step #14: Now the jar is become executable jar, type the following command to execute the jar

CMD> java -jar build\libs\<Jar_Name>.jar

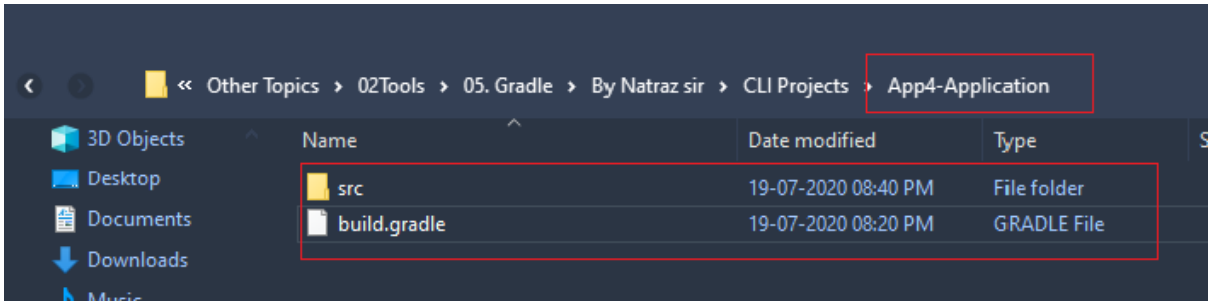



Now the jar is executed and also get the output.

But in "java" plugin there is not "run" task but in "application" plugin there is support of "run" task

## Project 4: Basics of DSL using application plugin

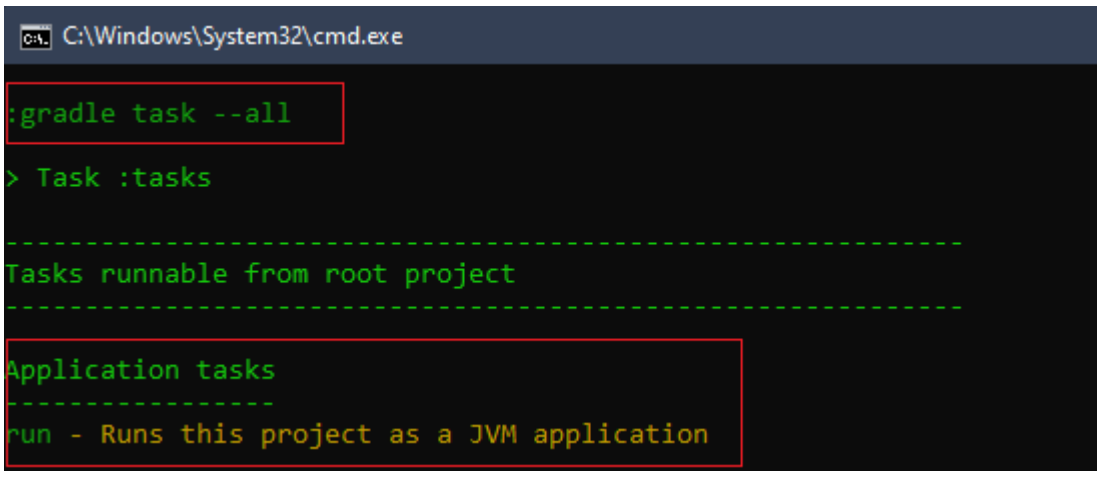
Step #1: Create a application having the same code and directory structure like Project-3, application named as App4-Application




Step #2: Open the build.gradle file and modify with the following code and save it

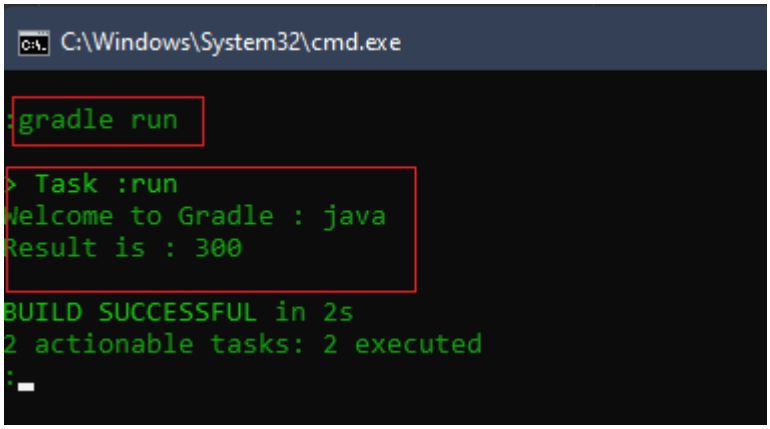
```
apply plugin:'application'

version="1.2"

mainClassName="com.nt.baiscs.Arithmetic"
```

Step #3: Then go to the Command prompt of the directory and type “gradle task --all” you can see with all task, and “run” task is also added

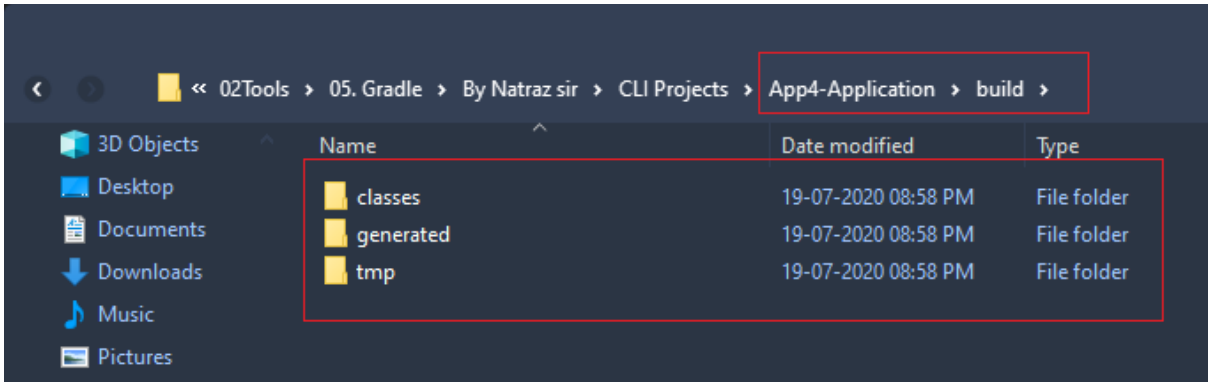



But this command is optional we are check that the task is add or not otherwise use can directly use the “run” task as like below

Step #4: Execute application by using “gradle run”

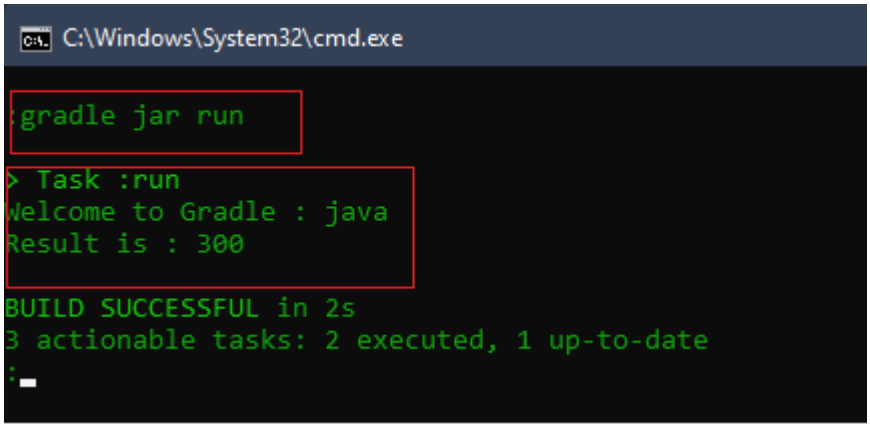



Step #5: But this time build folder is coming but libs\<jar_name>.jar file is not coming with build folder, because this is executed as independent task, it is not having dependency with any other jars



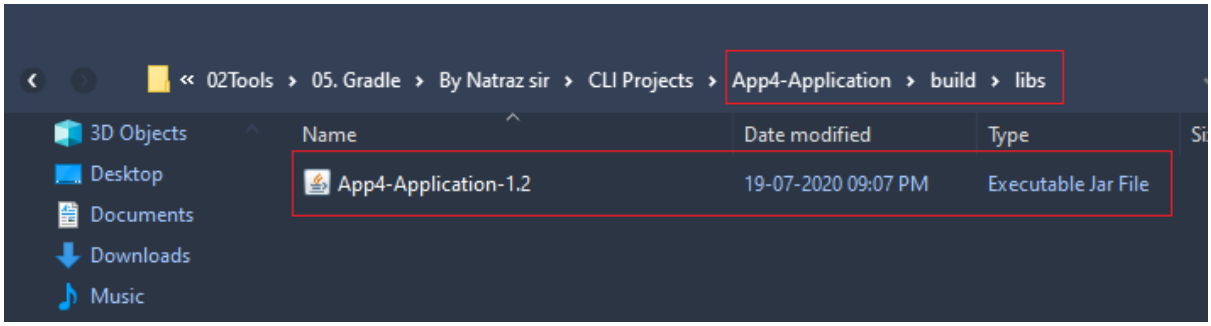

Step #6: If you want jar also then use the following command to get jar

CMD> gradle jar run




It will give the jar file in build\libs folder and also the output

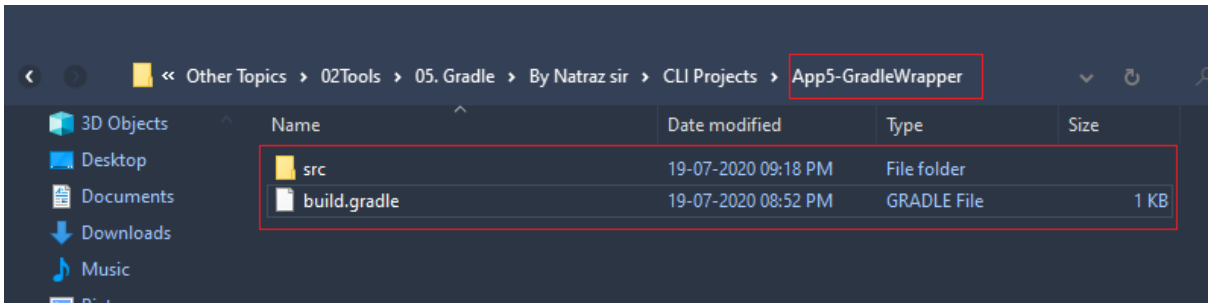
Step #7: Go to build folder you can see the lib folder and go inside the lib folder you can get the jar file also




Step #8: You can add manifest attribute like above project to make you jar as executable jar, though client don't have gradle they execute directly the executable jar file by using "java -jar build\libs\<Jar_Name>.jar" command.

## Project 5: Basics of DSL using gradle wrapper [gradlew] feature

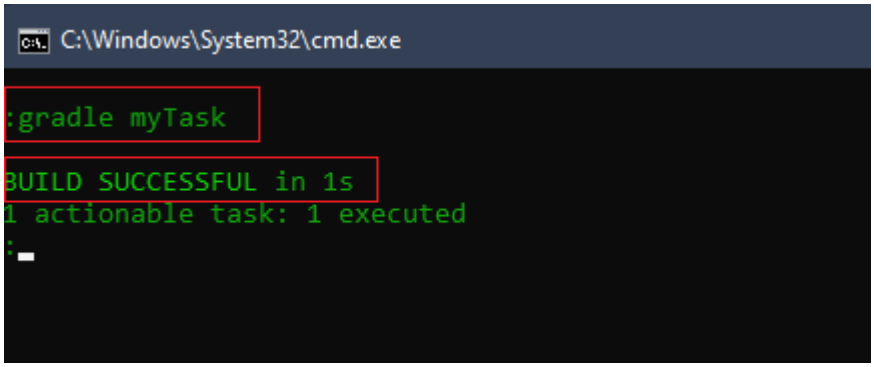
Step #1: Create a application having the same code and directory structure like Project-4, application named as App5-GradleWrapper



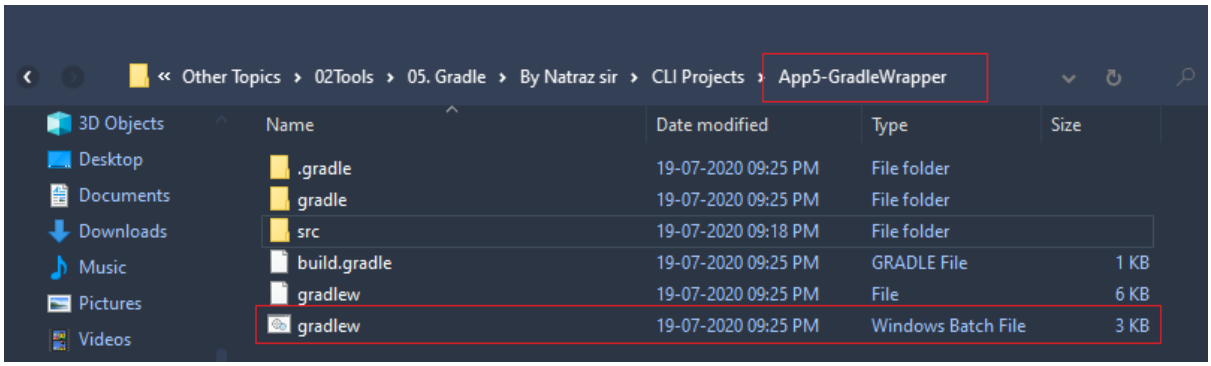

Step #2: Open the build.gradle file and modify with the following code to get gradle wrapper support and save it

```
apply plugin:'application'
version="1.2"

mainClassName="com.nt.basics.Arithmetic"
task myTask(type:Wrapper)
```

Step #3: Then go to the Command prompt of the directory and type “gradle myTask” then you will get a message BUILD SUCCESSFUL

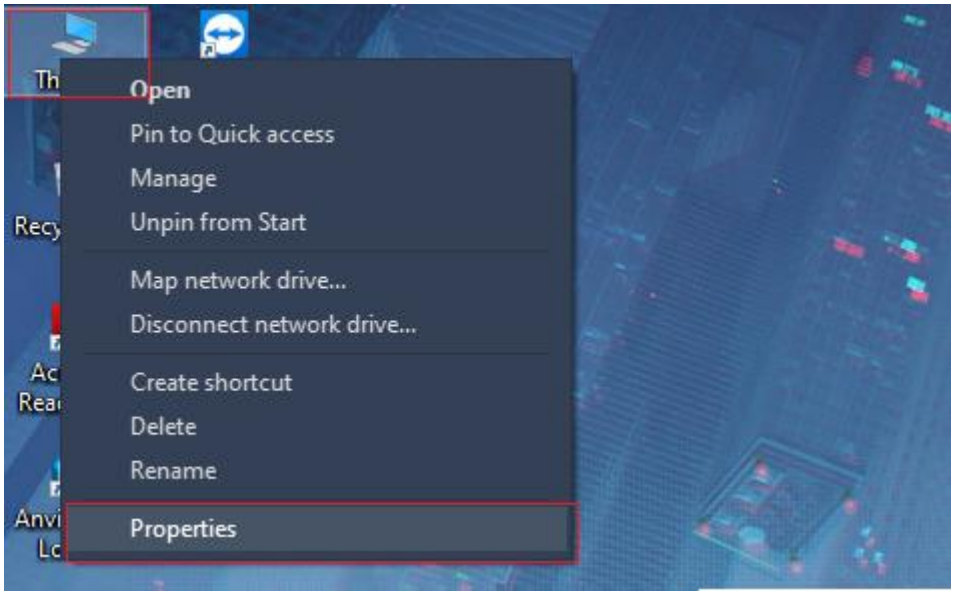



Step #4: The go to the Project directory structure you can see a gradlew Windows Batch file

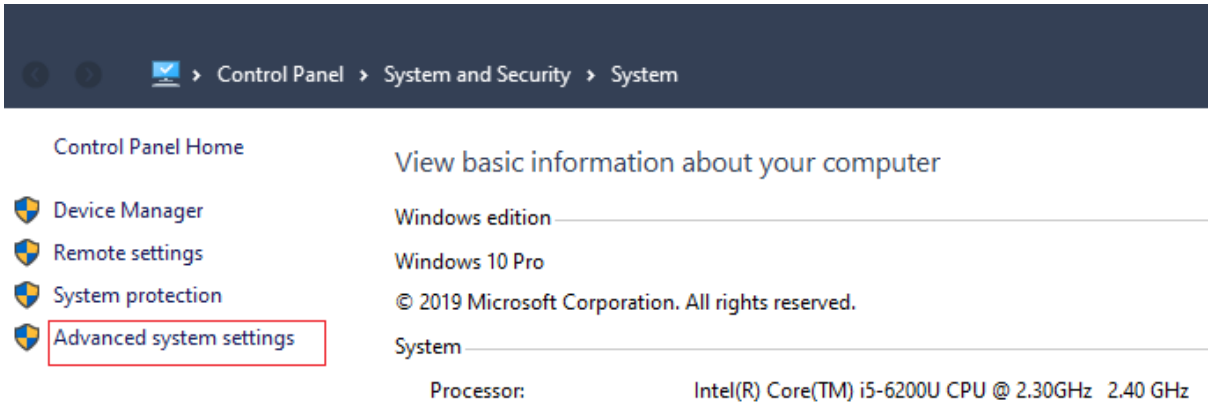



Step #5: To feel the Gradle wrapper feature remove the gradle bin directory value from the Path environment variable. To do This follow the below steps

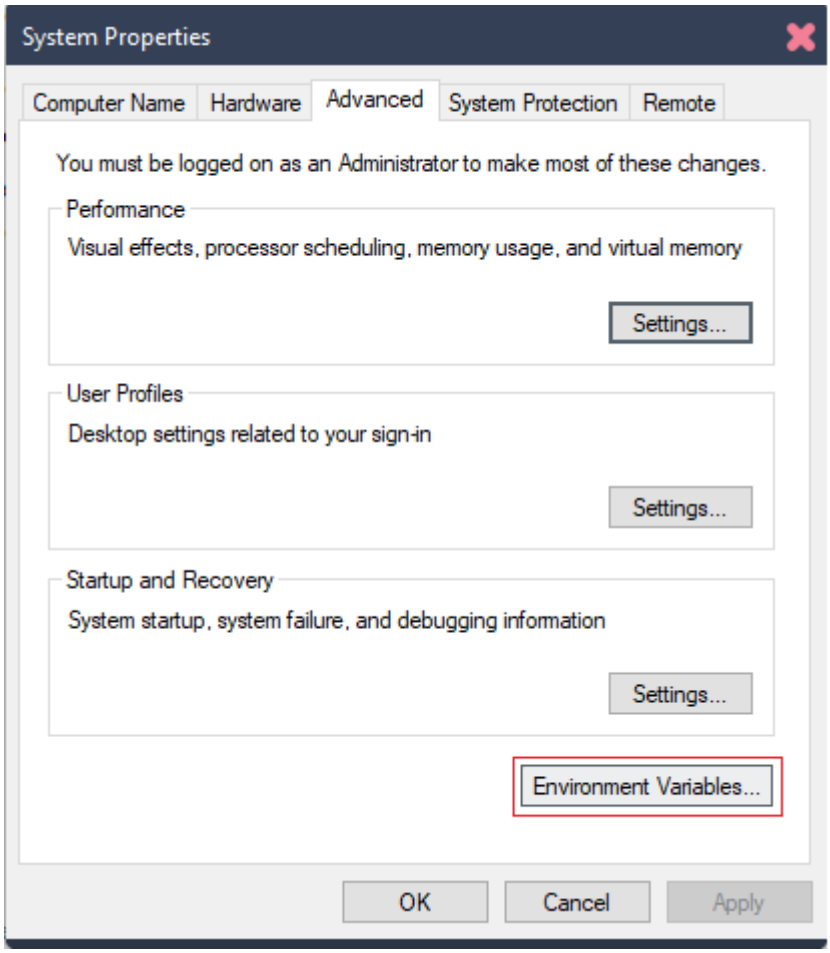
Step #6: Right click on your This PC, then click on Properties



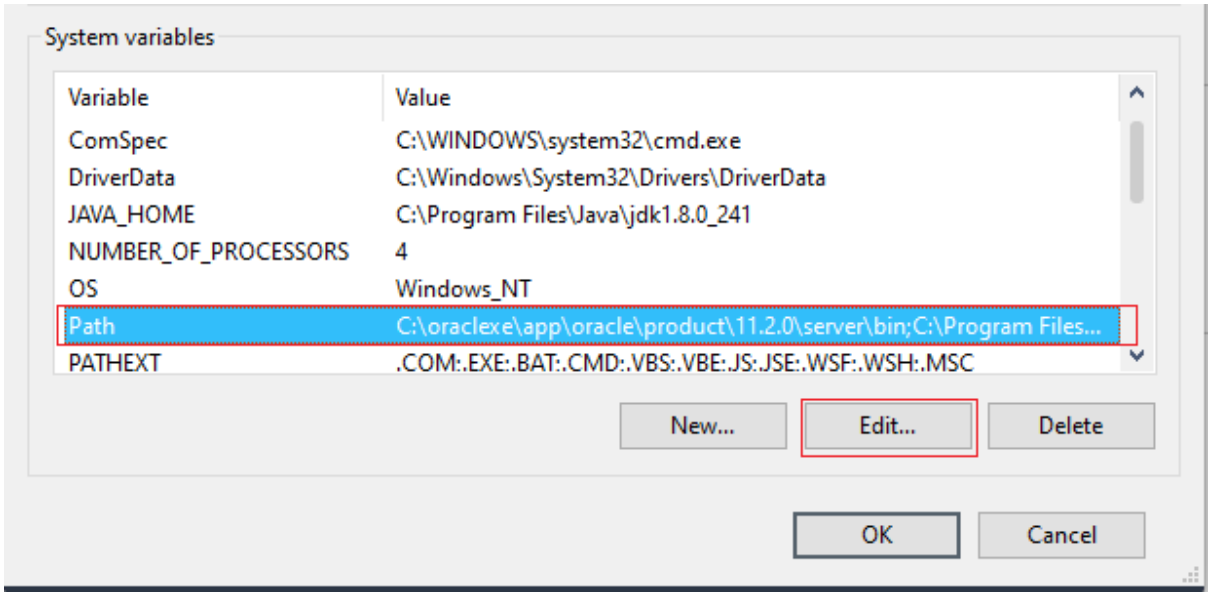
Step #7: Click on the Advanced system settings



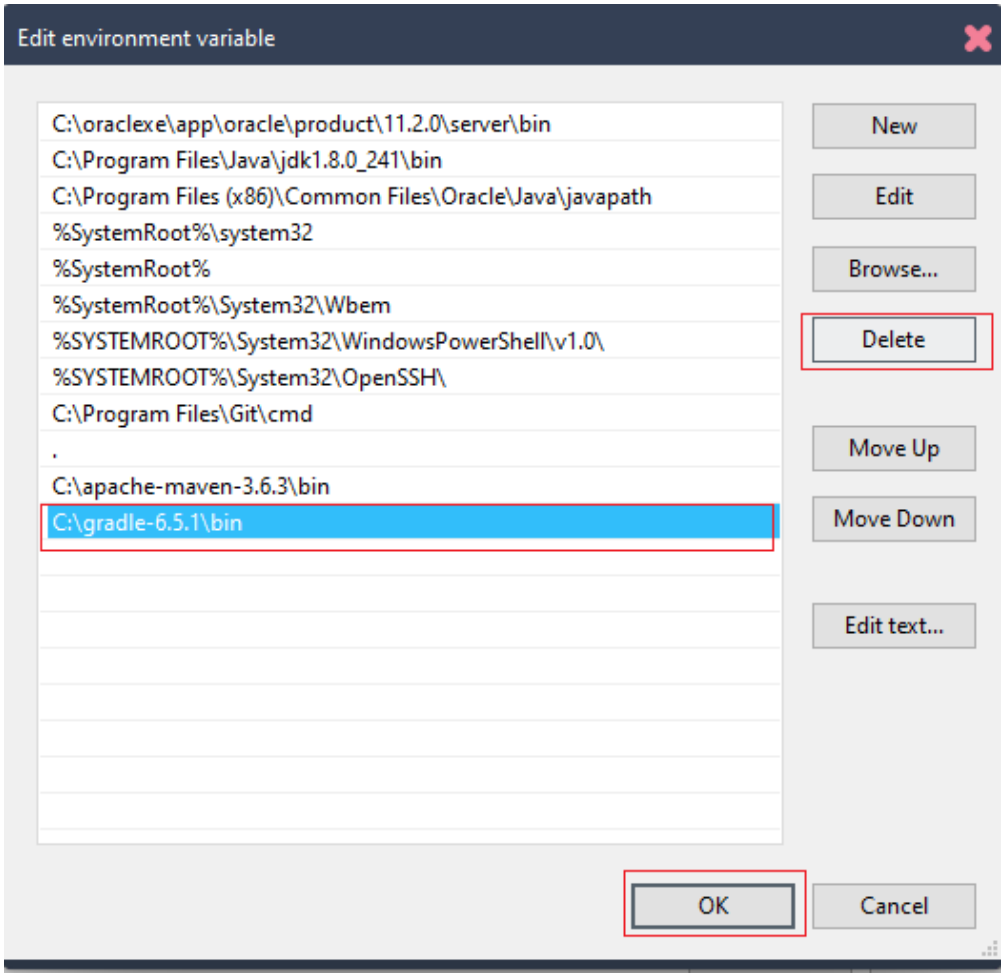

Step #8: Then click on Environment Variables



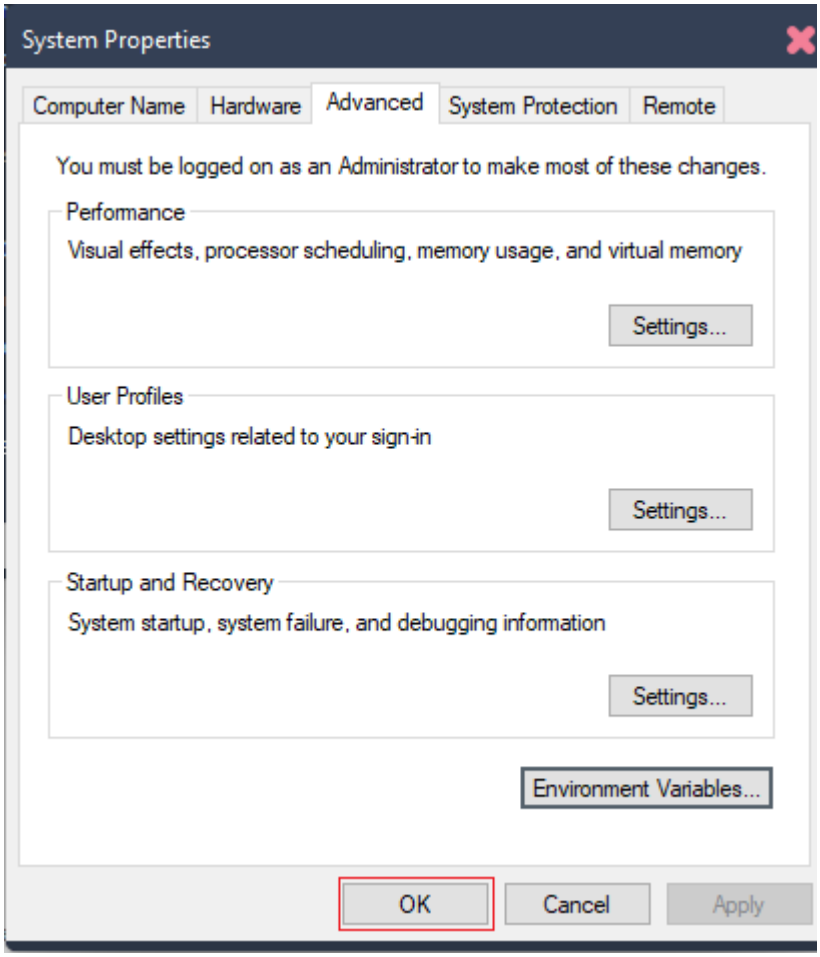
Step #9: Click on you Path variable from System variables section, then click on Edit



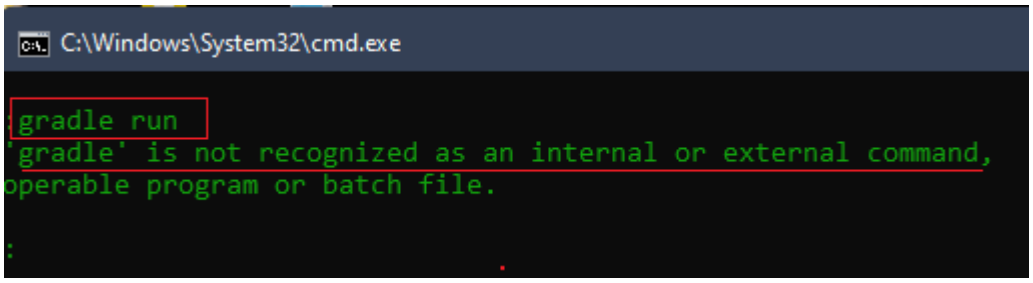

Step #10: Click on “<Gradle Home>\bin" address click on Delete, then click on OK and OK



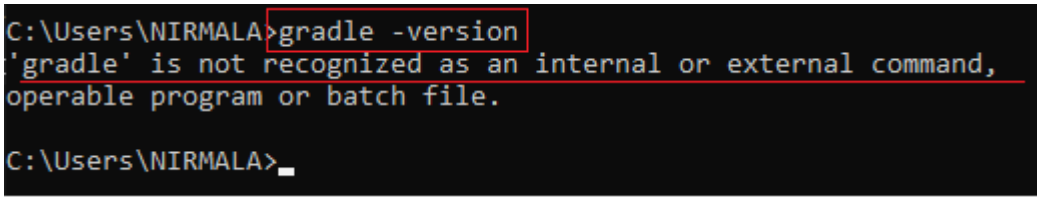
Step #11: Again, click on OK




Step #12: Using “gradle run” confirm no active gradle in our system by opening new Command prompt on the directory because new values are not reflected to old command prompt

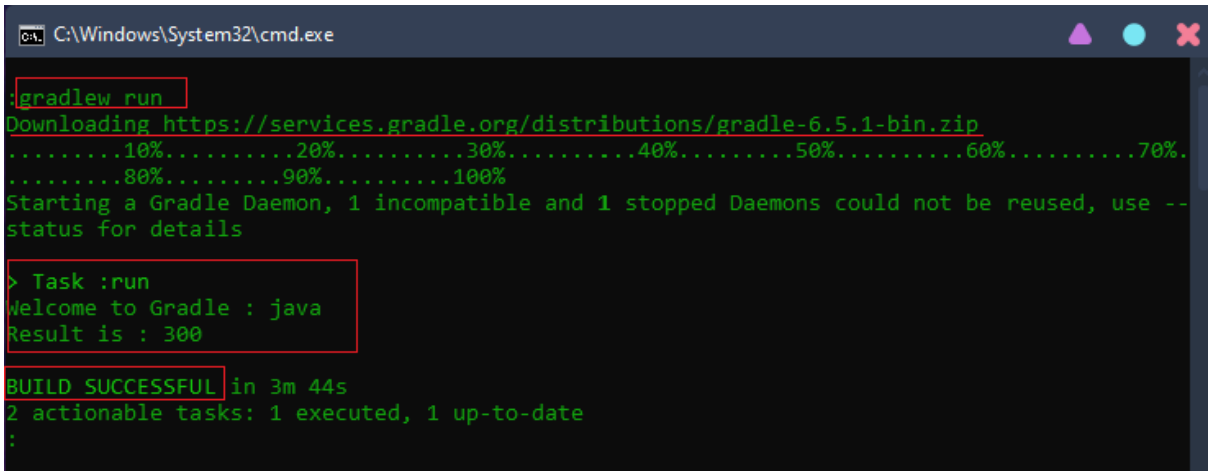



Or by confirming using “gradle -version” in command prompt



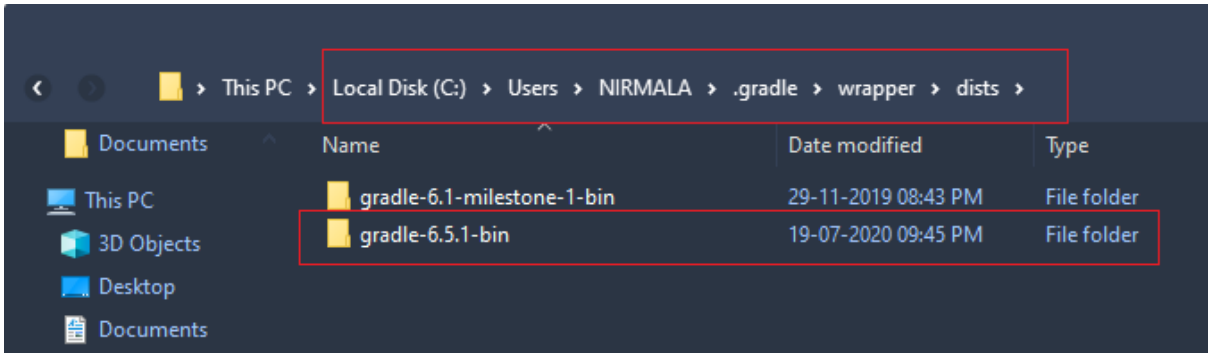
Step #13: Type the below command to execute your application by download the gradle from internet and gives the output and you get BUILD SUCCESSFUL message

CMD> gradlew run




First time it takes time. It is helpful when we give the project to client having gradlew file without arranging gradle in client system, client can run the application but downloading the gradle from internet dynamically

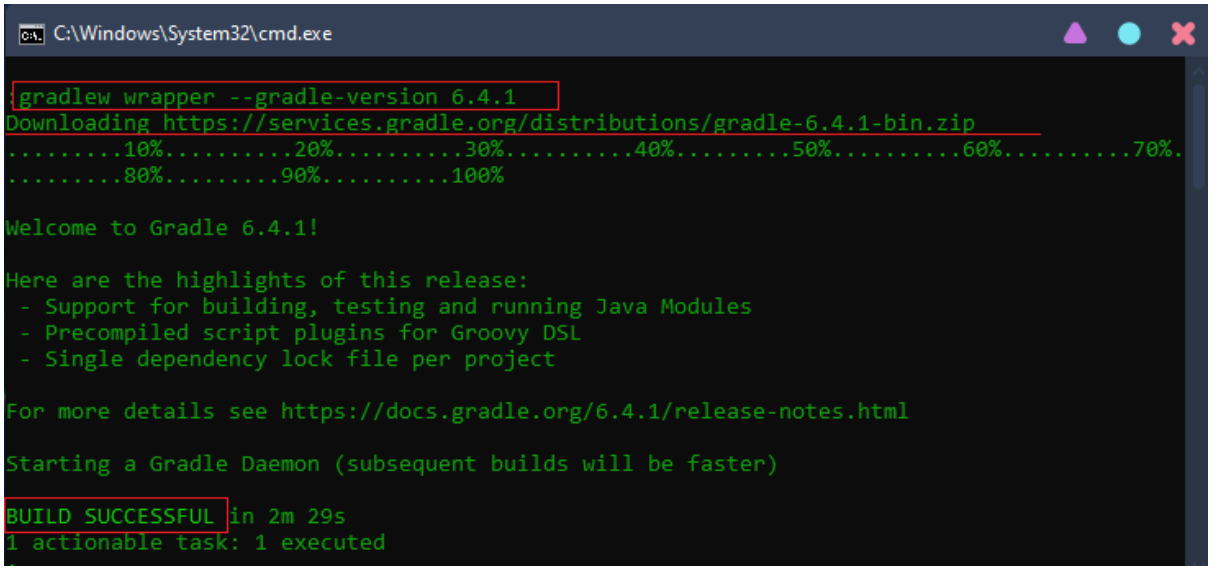
Step #14: All the download goes to your C:\Users\<user_name>\.gradle\wrapper\dists folder




Step #15: Mentioning proper version we can download the particular gradle version using below command.

It will download the particular version jar and kept in your C:\Users\<user_name>\.gradle\wrapper\dists folder and you have to done “gradlew run” again then only it will run the application again with the recent downloaded version

CMD> gradlew wrapper --gradle-version <version_no.>




Step #16: Perform “gradlew run” this time you will get direct output