COMP3900-9900 – capstone-project-waitless

User Documentation / Manual

Author: Kavitha Narayanan (z5190588)

Table of Contents

Python Installation

Python can be downloaded from the following location;

https://www.python.org/downloads/

This user manual assumes that any version of Python >= 3.6.5 is installed and available in the System.

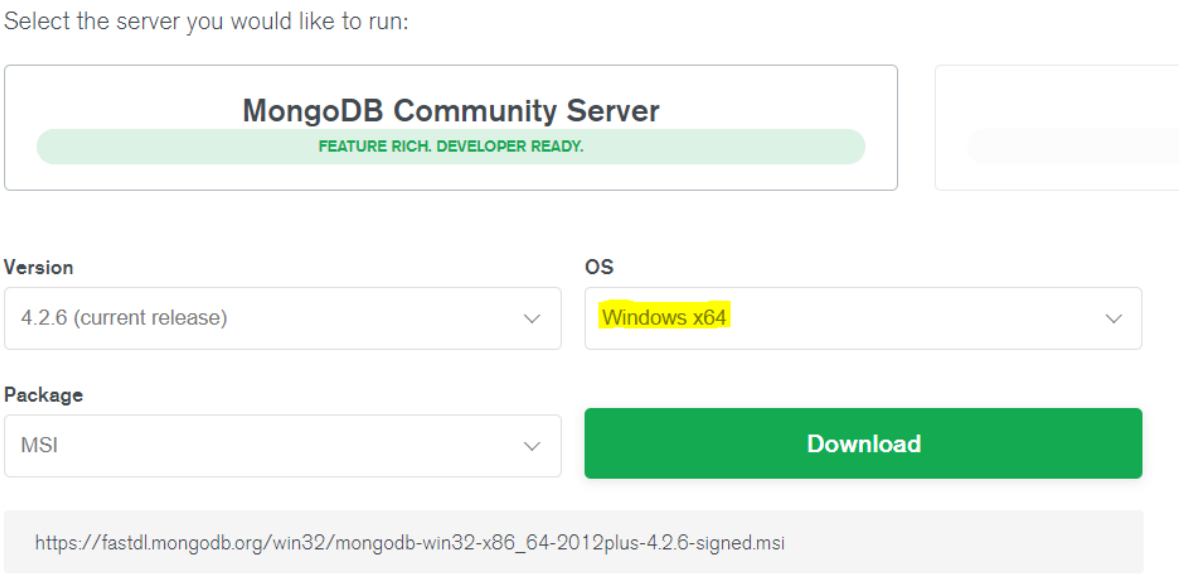
MongoDB Community Server

Download and Install

Download and Install MongoDB Community Server latest version (4.2.6 as of 28th April, 2020) from the following location

https://www.mongodb.com/download-center/community

Select the appropriate OS from the dropdown.



Running MongoDB Server

The Server 'bin' folder will be automatically set in 'PATH' during installation. If not, open 'Command Prompt' (in Windows Operating System) and navigate to the folder where MongoDB community server is installed.

```
c:\Program Files\MongoDB\Server\4.2\bin>
```

Execute 'mongod.exe' which will start the MongoDB Server.

```
c:\Program Files\MongoDB\Server\4.2\bin>mongod.exe
```

The server will be running on localhost (127.0.0.1) and listening for incoming connections on port 27017.

```
2020-04-28T12:35:32.794+1000 I  NETWORK  [listener] Listening on 127.0.0.1
2020-04-28T12:35:32.794+1000 I  NETWORK  [listener] waiting for connections on port 27017
```

Project Code Setup

Project Location in GitHub

https://github.com/unsw-cse-comp3900-9900/capstone-project-waitless

Cloning a local copy

Create a local folder for cloning the project source code from GitHub and navigate to that folder in 'Command Prompt' (Windows).

```
c:\>mkdir 9900-project
```

```
c:\>cd 9900-project

c:\9900-project>
```

Clone the project source code with the following command.

git clone https://github.com/unsw-cse-comp3900-9900/capstone-project-waitless.git

```
c:\9900-project>git clone https://github.com/unsw-cse-comp3900-9900/capstone-project-waitless.git
Cloning into 'capstone-project-waitless'...
remote: Enumerating objects: 973, done.
remote: Counting objects: 100% (973/973), done.
remote: Compressing objects: 100% (555/555), done.
remote: Total 4297 (delta 496), reused 709 (delta 291), pack-reused 3324
Receiving objects: 100% (4297/4297), 8.51 MiB | 3.75 MiB/s, done.
Resolving deltas: 100% (1412/1412), done.
Checking out files: 100% (1890/1890), done.
```

Navigate to 'capstone-project-waitless' folder and make sure you are in 'master' branch.

```
c:\9900-project>cd capstone-project-waitless

c:\9900-project\capstone-project-waitless>git branch
* master
```

Installing Python package dependencies

Run the following command to install the Python library packages required for running this project.

pip install –r requirements.txt

```
c:\9900-project\capstone-project-waitless>pip install -r requirements.txt
```

The following are the dependency packages in 'requirements.txt'.

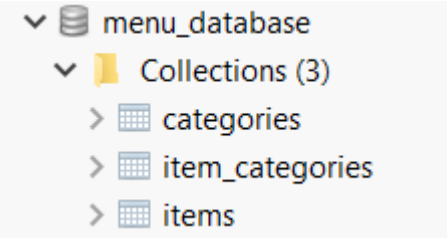
Flask==1.1.1
Flask-PyMongo==2.3.0
flask-restx==0.1.1
Werkzeug==0.16.1
Flask-WTF==0.14.3
WTForms==2.2.1
pymongo==3.10.1

Setup Sample Restaurant Menu

Execute the following command to create a sample Restaurant Style Menu items in the MongoDB.

```
c:\9900-project\capstone-project-waitless>python setup.py
```

This command will create a database called 'menu_database' in MongoDB and create a set of Collections as below;



Also, a sample list of categories and menu items with pictures will be loaded into the database.

Running Flask RESTPlus Application

Run the following command to start the Flask RESTPlus server.

```
c:\9900-project\capstone-project-waitless>python run.py
 * Serving Flask app "server" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 188-797-197
 * Running on http://127.0.0.1:8080/ (Press CTRL+C to quit)
```

The Server will be running on localhost (127.0.0.1) and listening on port 8080.

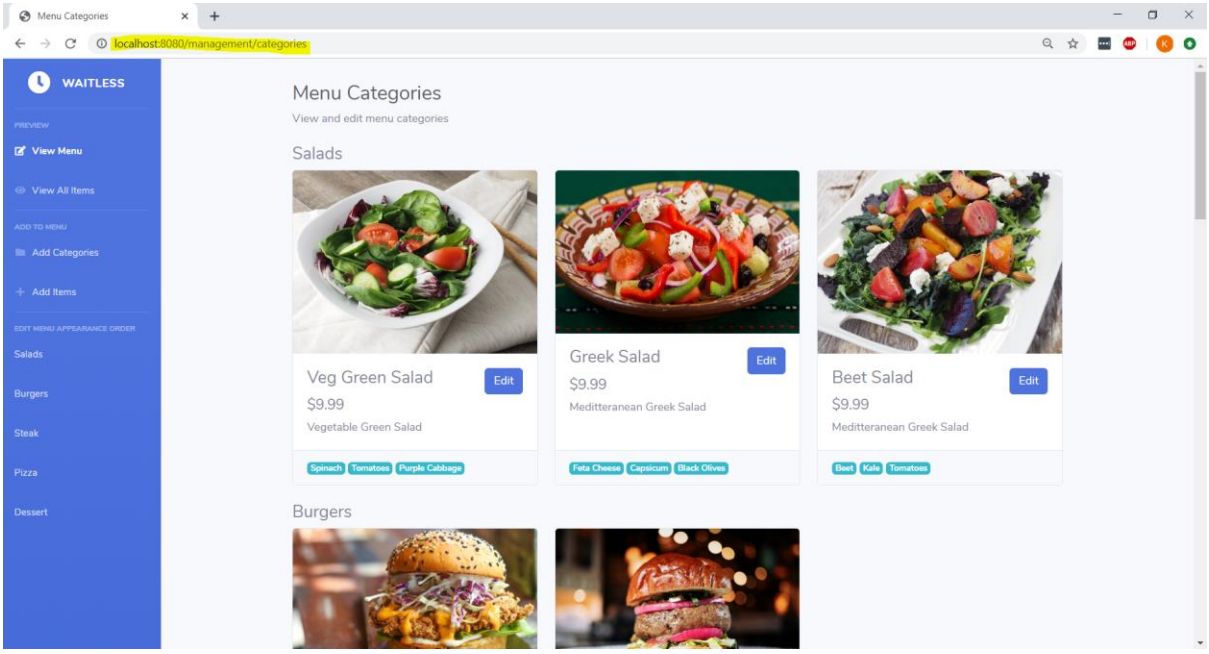
This Python - Flask RESTPlus application serves as the Backend API for all the applications in this Project.

1. Waitless Web Application
2. CustomerApp Android Tab App
3. KitchenStaffApp Android Tab App
4. WaiterApp Android Tab App

Waitless Web Application

The Waitless web application is used by the Restaurant Manager to setup / maintain the Restaurant Menu. The application is developed using Bootstrap UI library. The application can be accessed by navigating to the following URL in browser.

http://localhost:8080/management

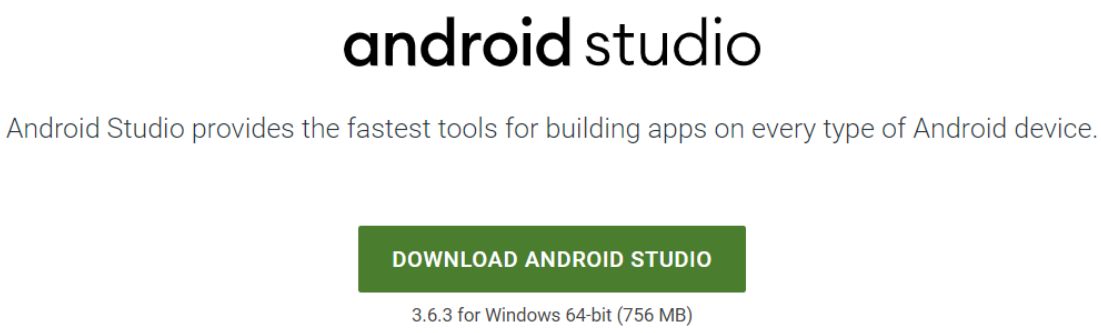


The details on how to use the application will be available in the 'Functionalities and Implementation Challenges' section. Setting up and running the remaining modules of this Project (CustomerApp, KitchenStaffApp and WaiterApp) requires installation of Android Studio.

Android Studio

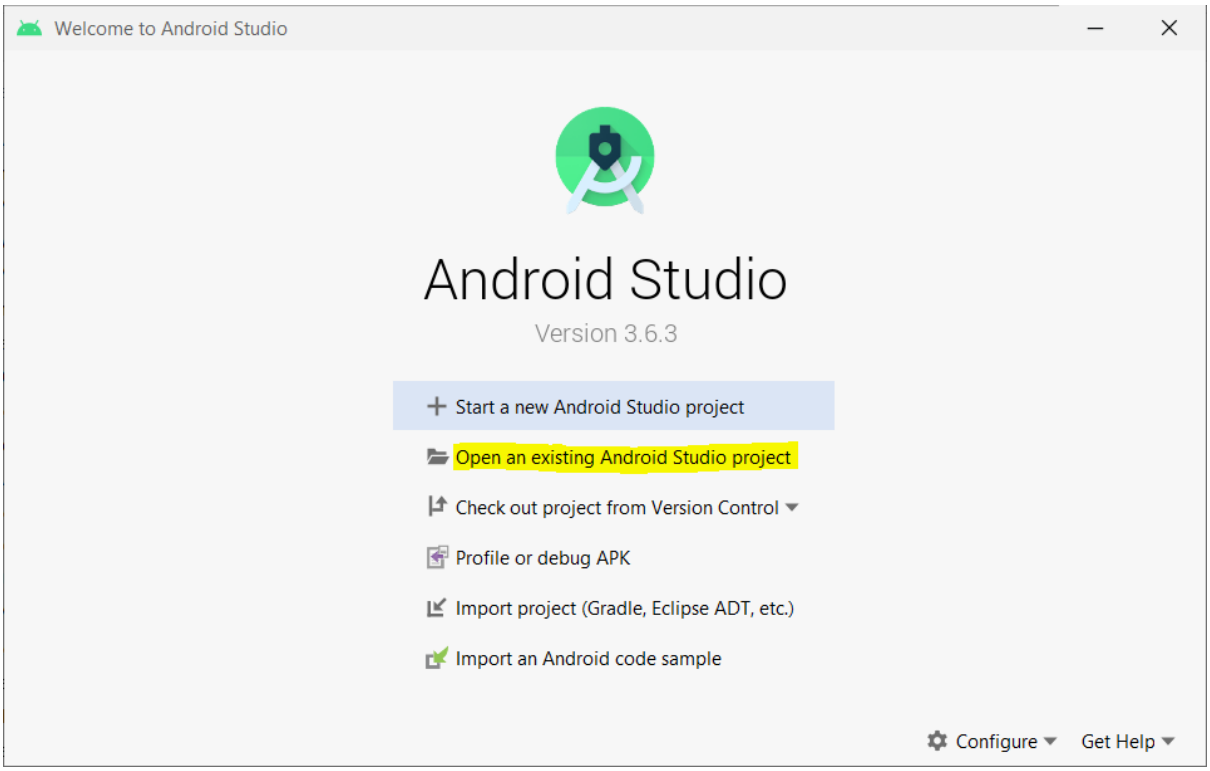
Download & Installation

Download latest version of Android Studio from https://developer.android.com/studio and install with default options.

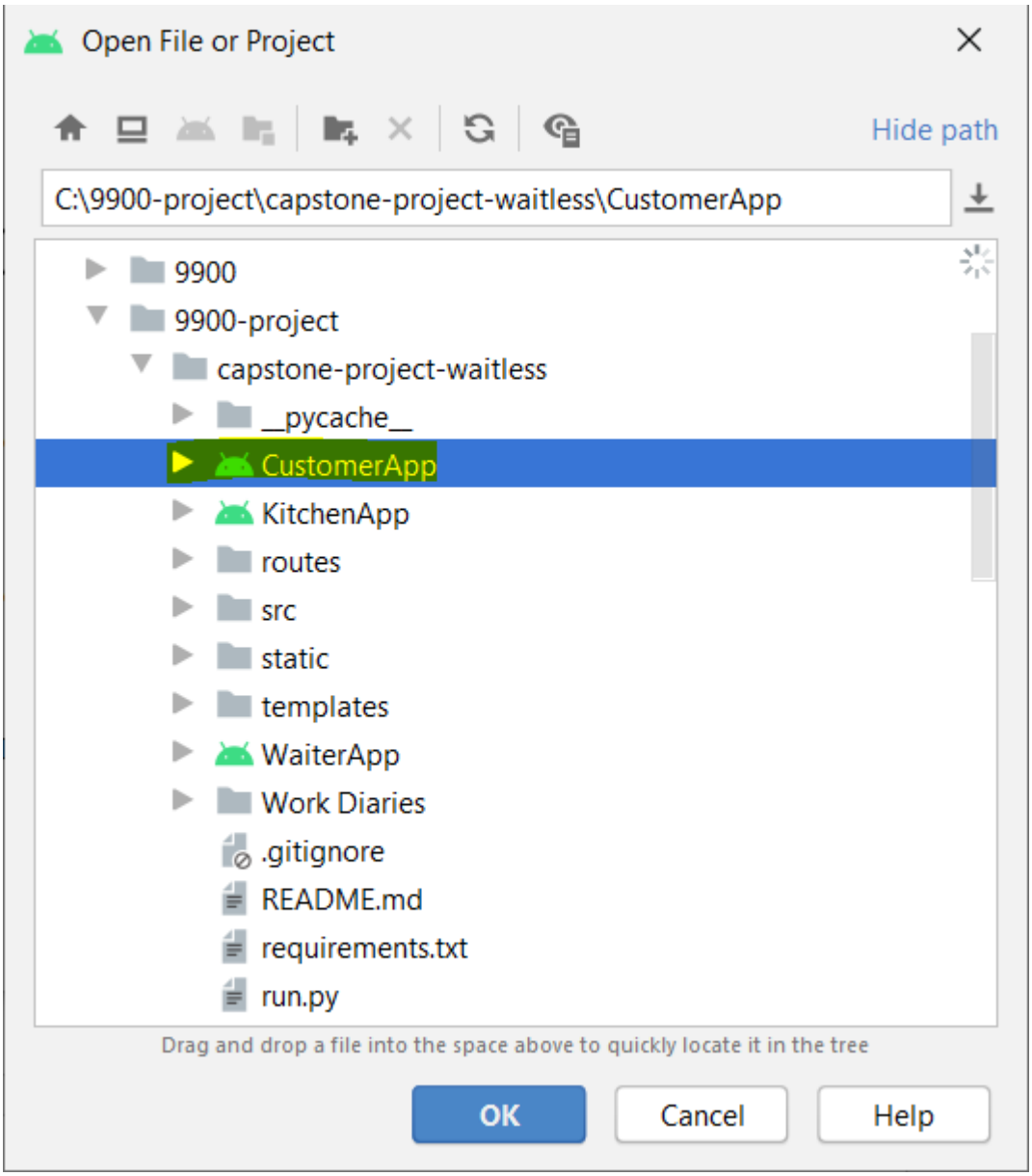


Importing AndroidApp Project

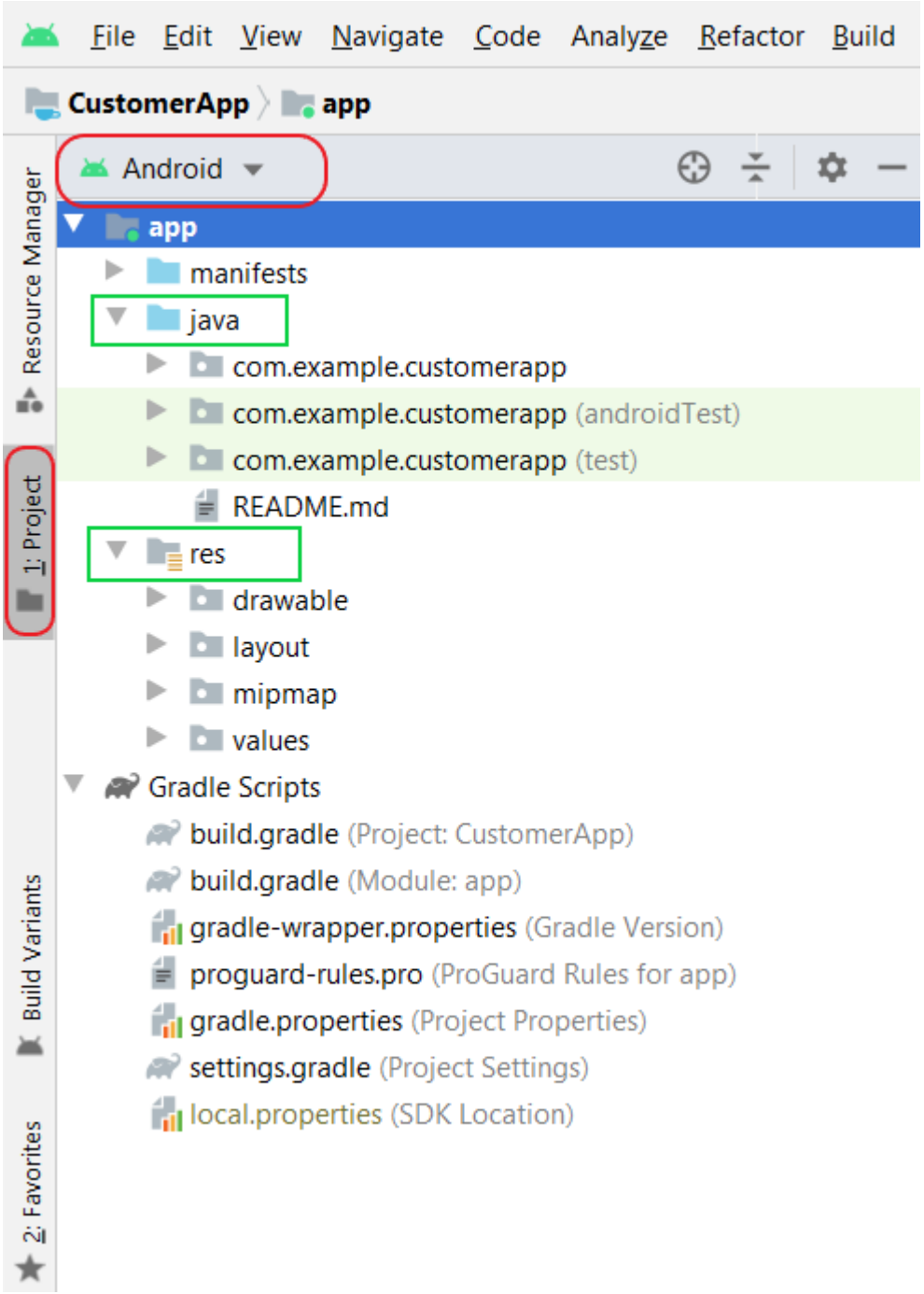
Start Android Studio and select 'Open an existing Android Studio project'



Select ‘CustomerApp’ from the checked-out Project folder.



Once the project is imported, select ‘Android’ view from the ‘Project’ navigator.

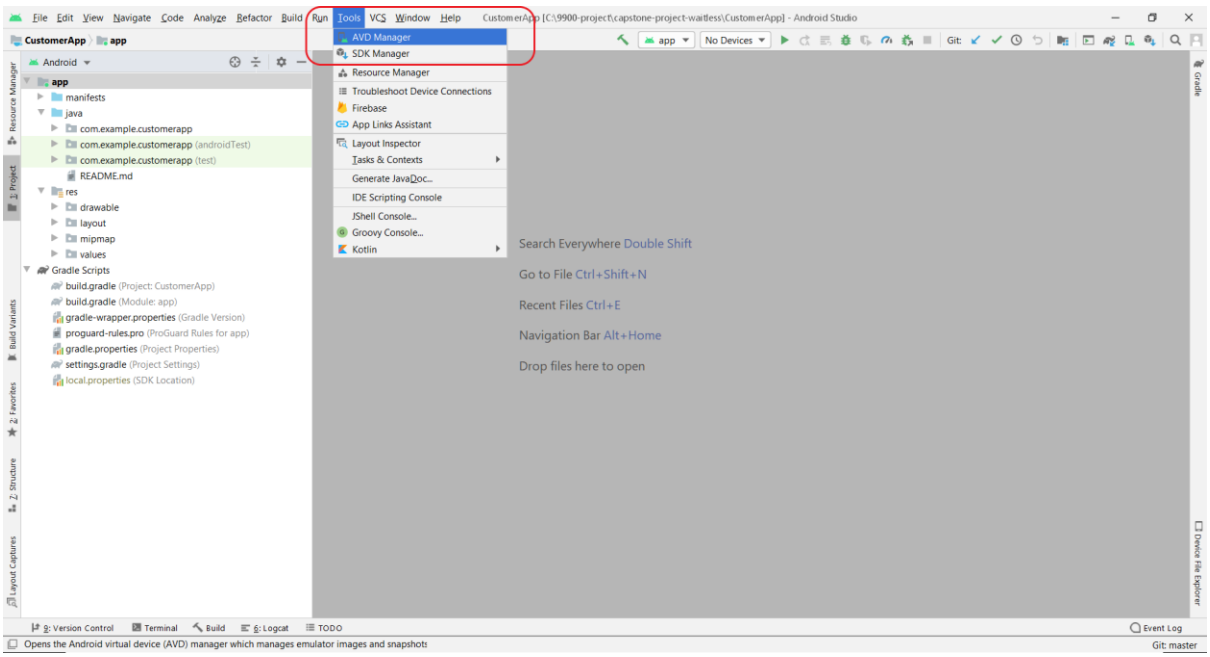


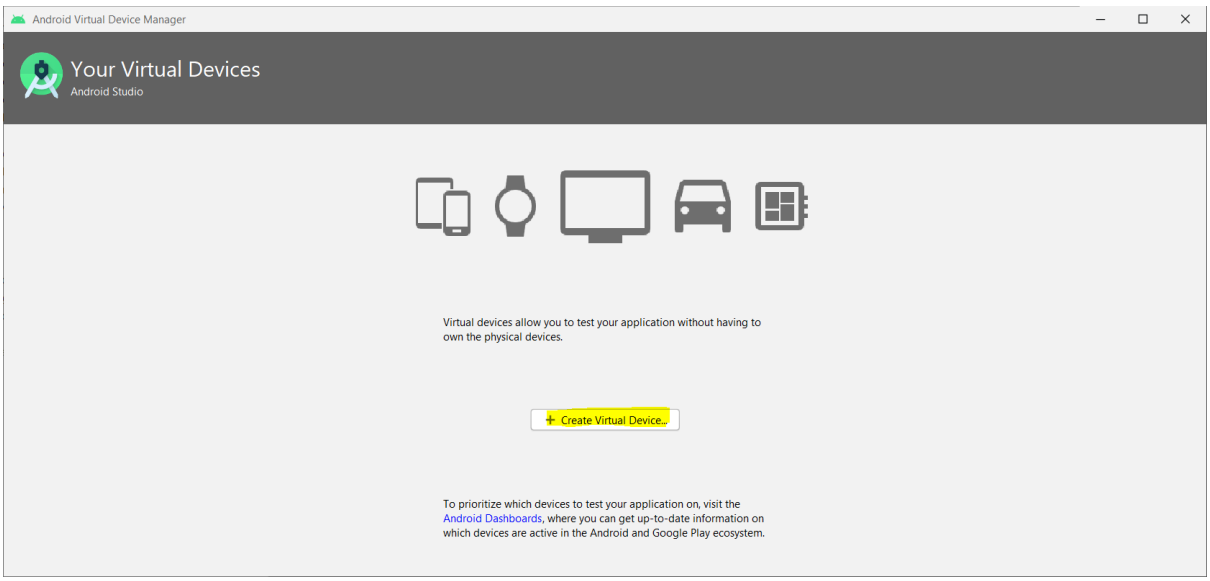
The ‘java’ folder will contain all the Java files and ‘res’ folder will contain all the resources needed for the App (Layout files, Image files etc.).

Follow instruction below to setup Android Virtual Device.

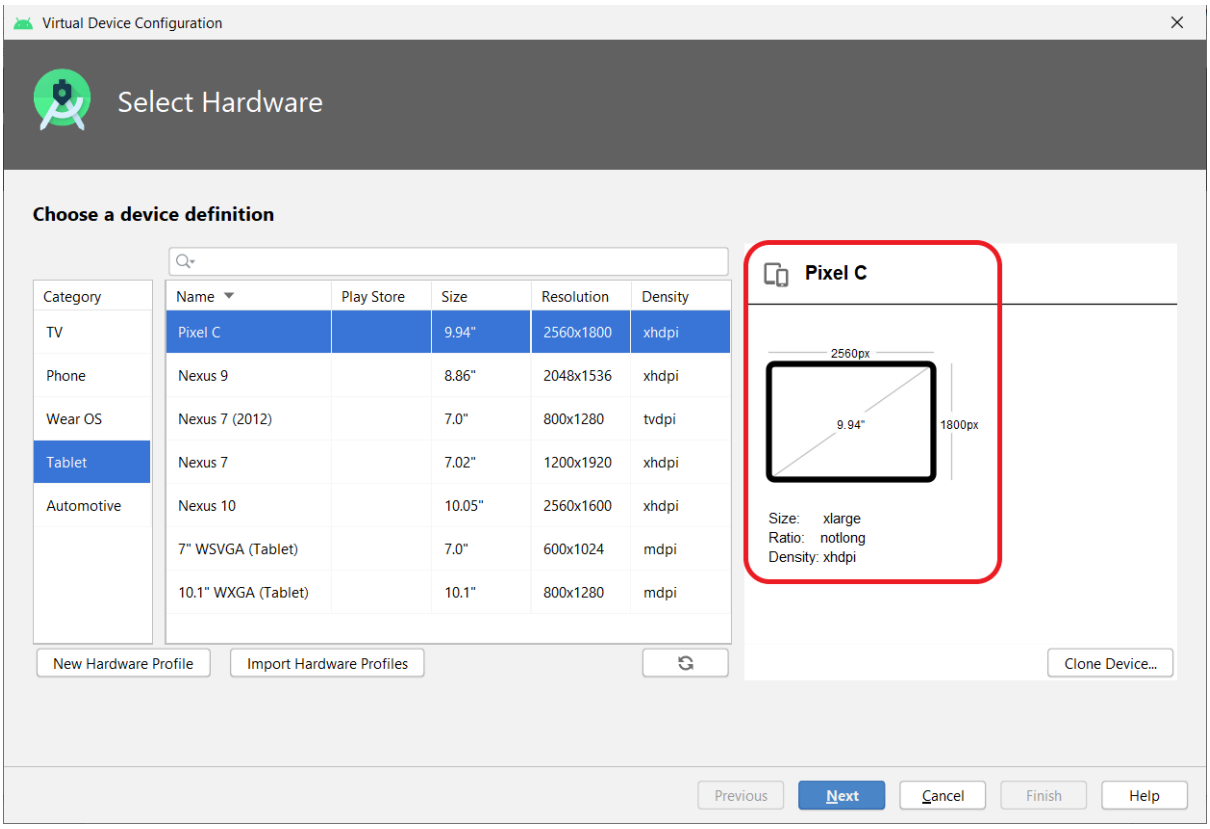
Setting up Android Virtual Device

Select ‘AVD Manager’ from ‘Tools’ menu.

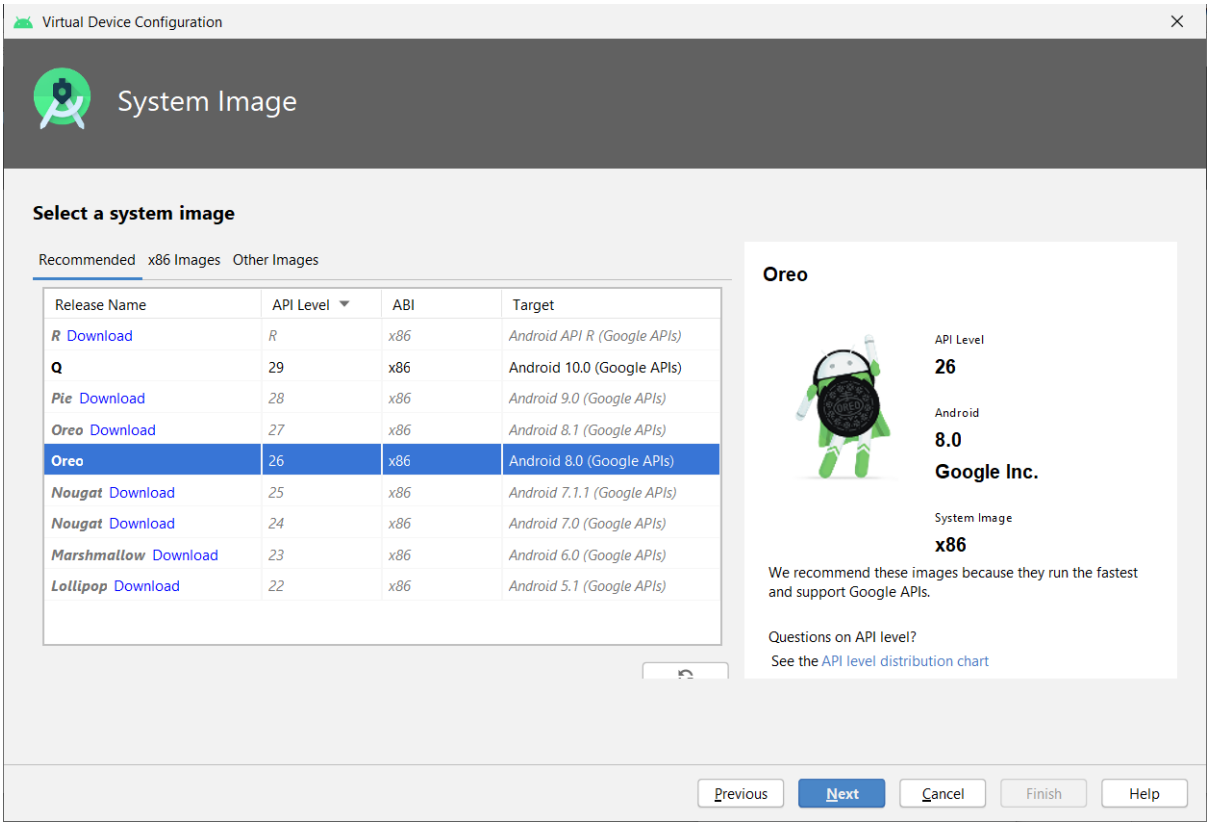




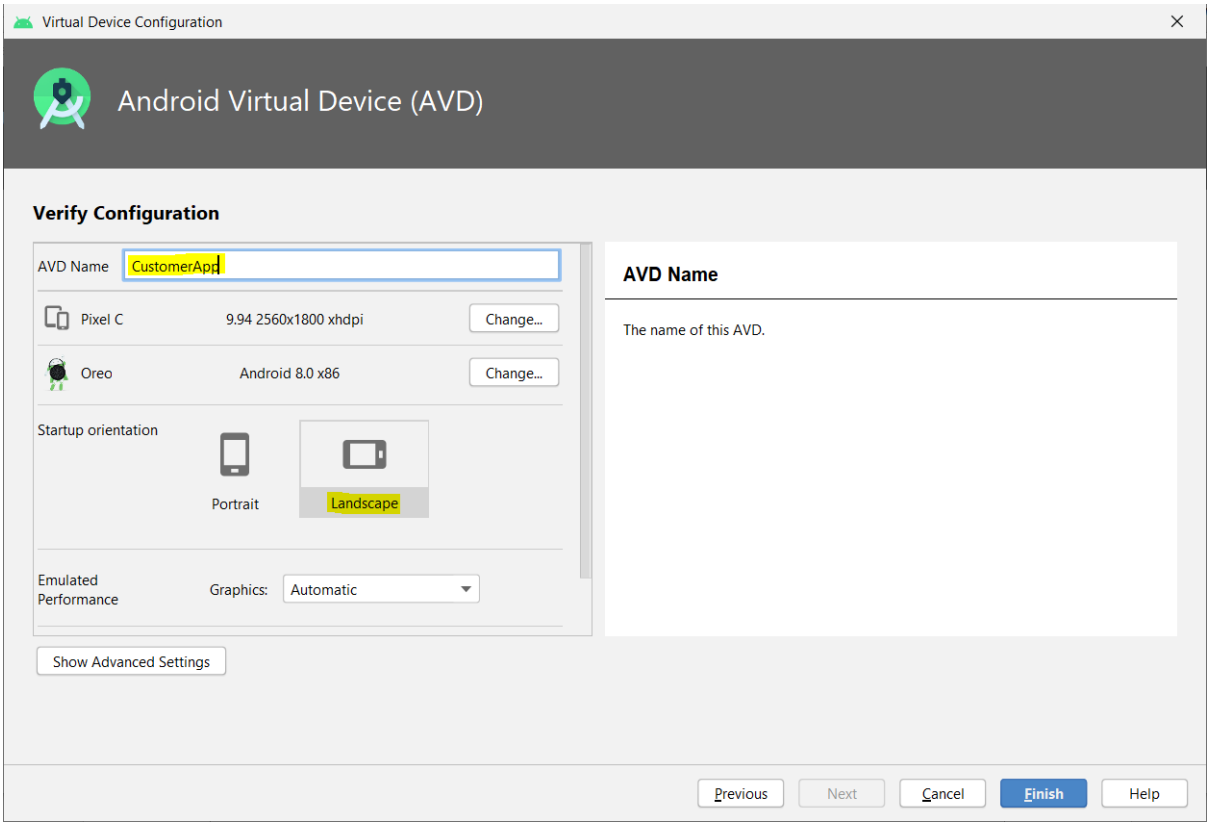
The Android Apps used in this Project are developed for a ‘Pixel C’ type of Tablet. It has the following configuration.



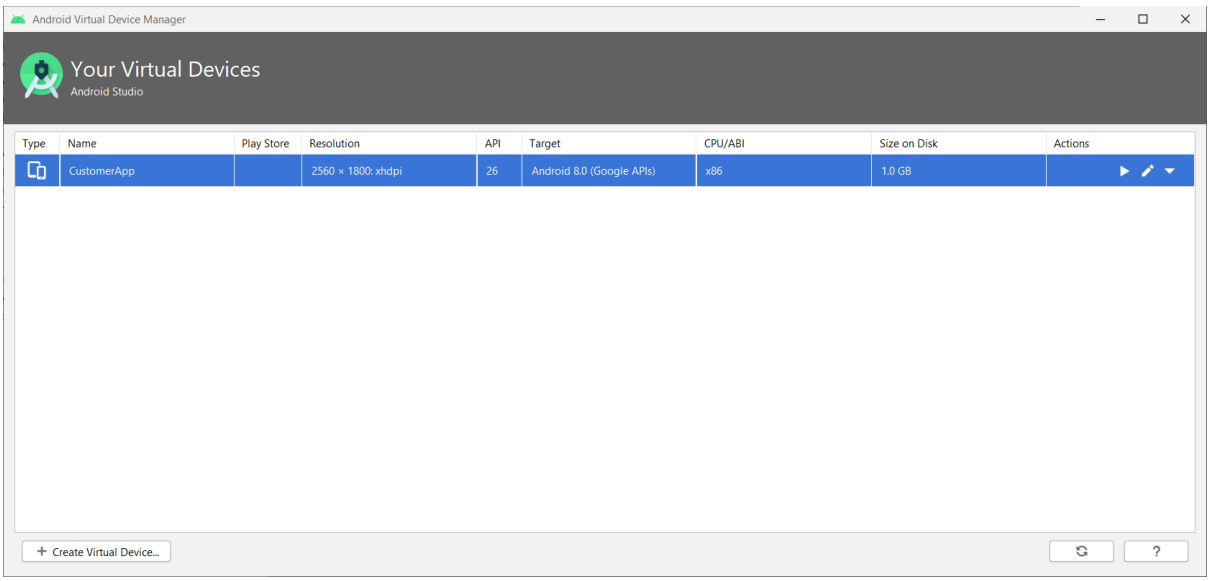
Click ‘Next’ and select ‘Oreo’ API Level 26.



Provide ‘CustomerApp’ as the AVD Name and set ‘Startup orientation’ as ‘Landscape’ and click ‘Finish’.

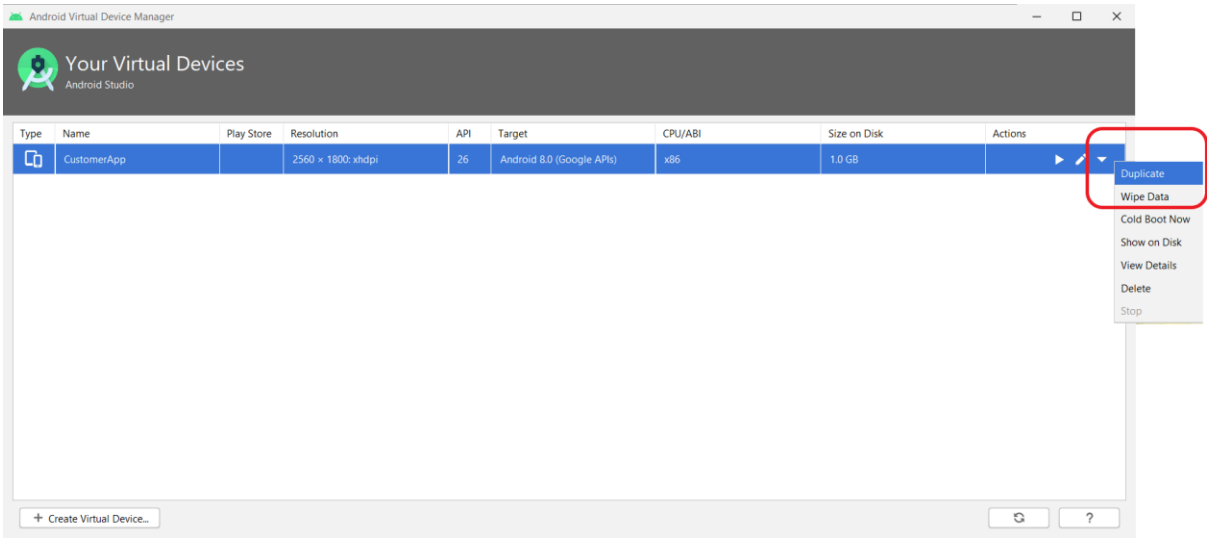


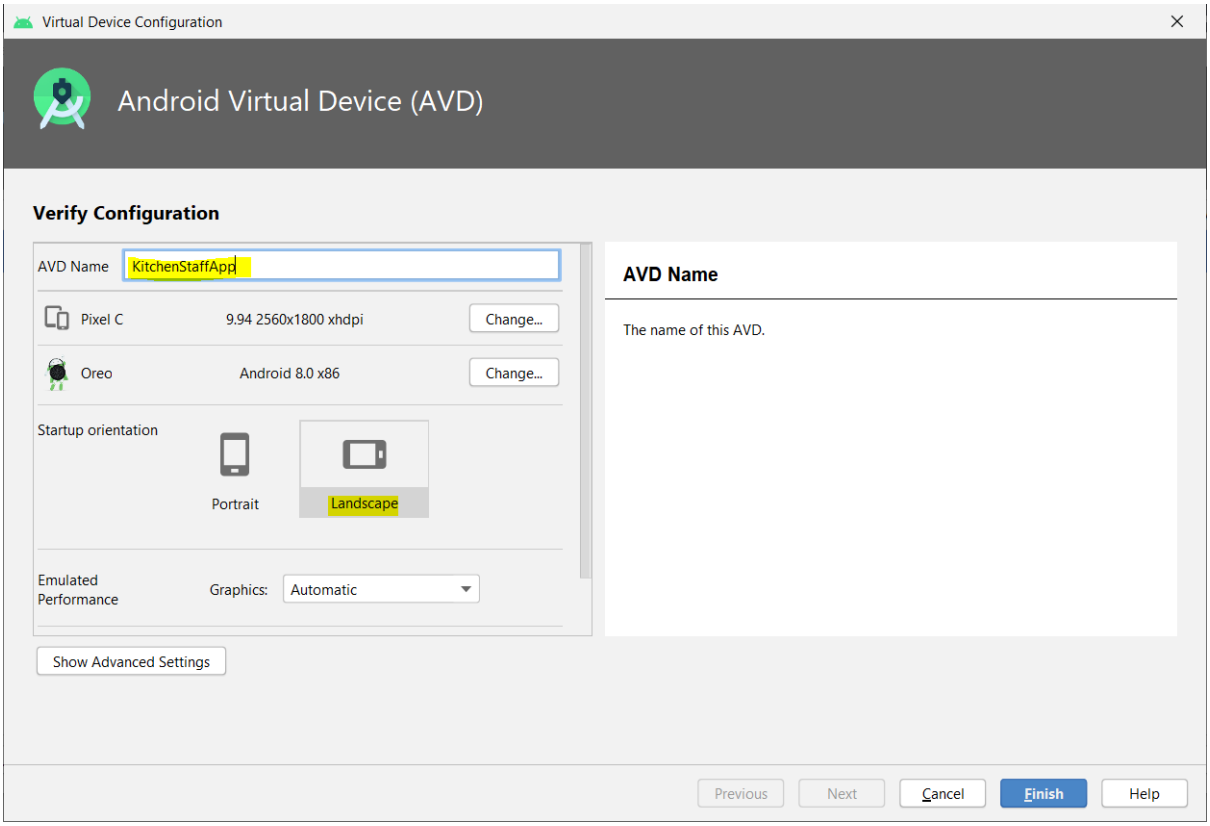
This will create the ‘CustomerApp’ AVD.

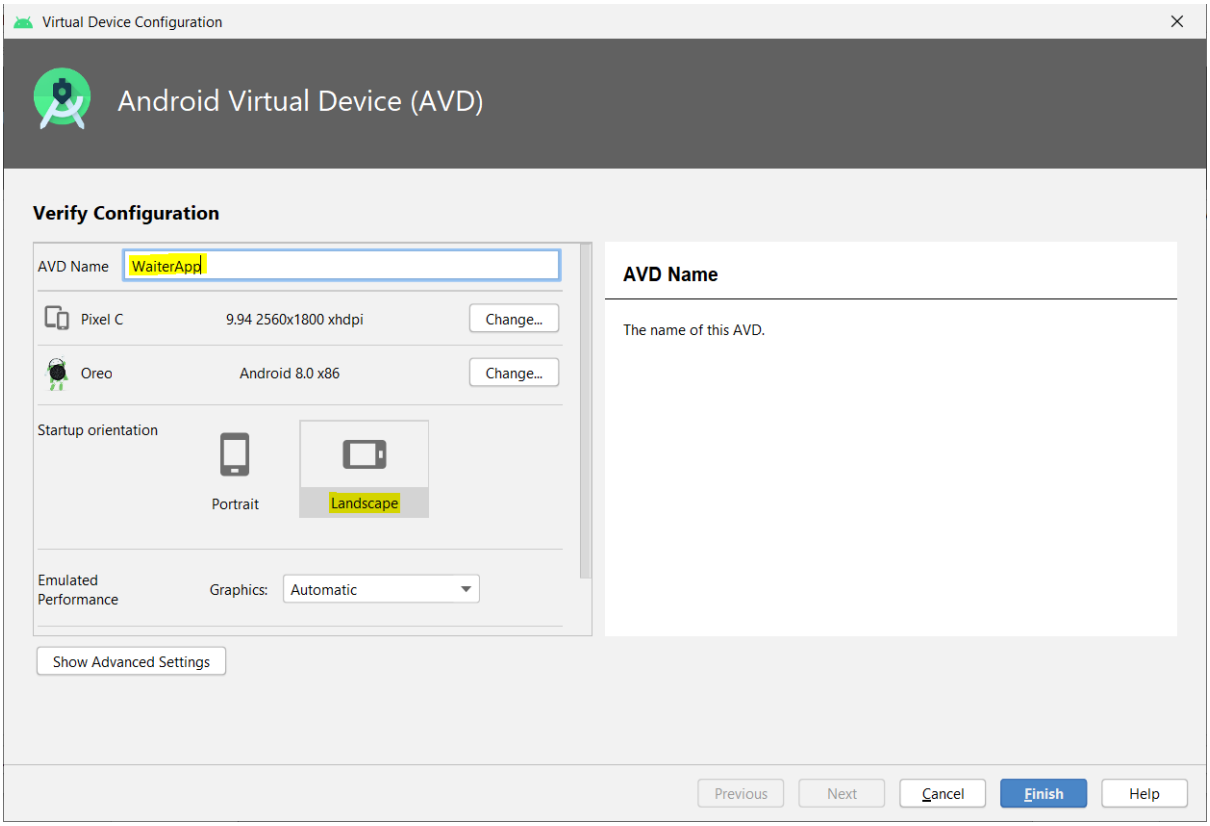


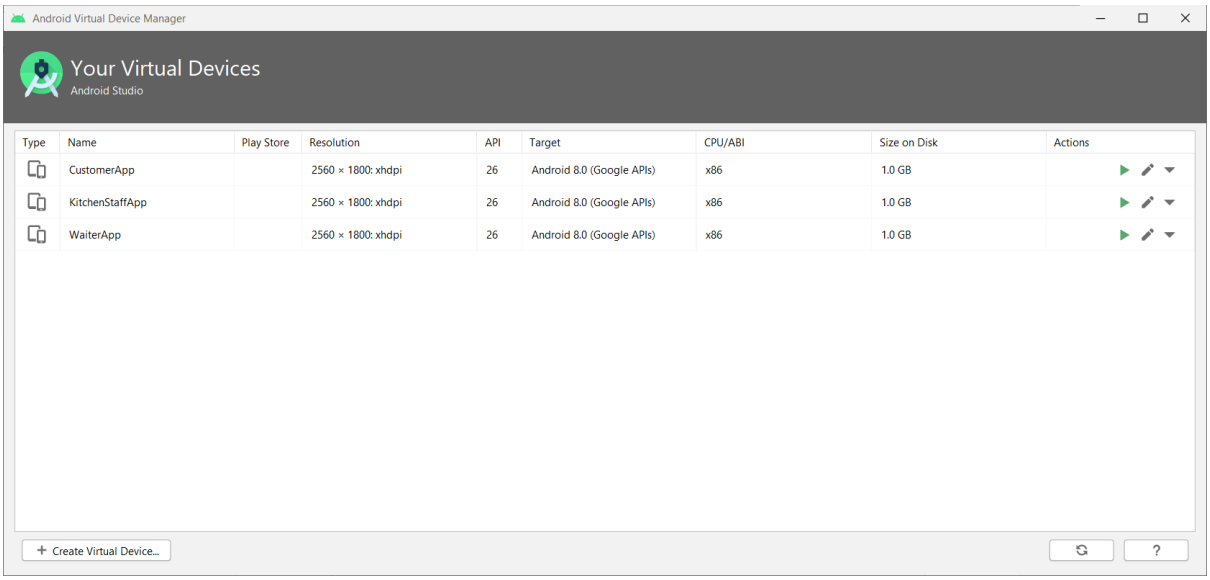
Creating Duplicate AVDs

Now select 'Duplicate' to create Android Virtual Devices for 'KitchenStaffApp' and 'WaiterApp'.



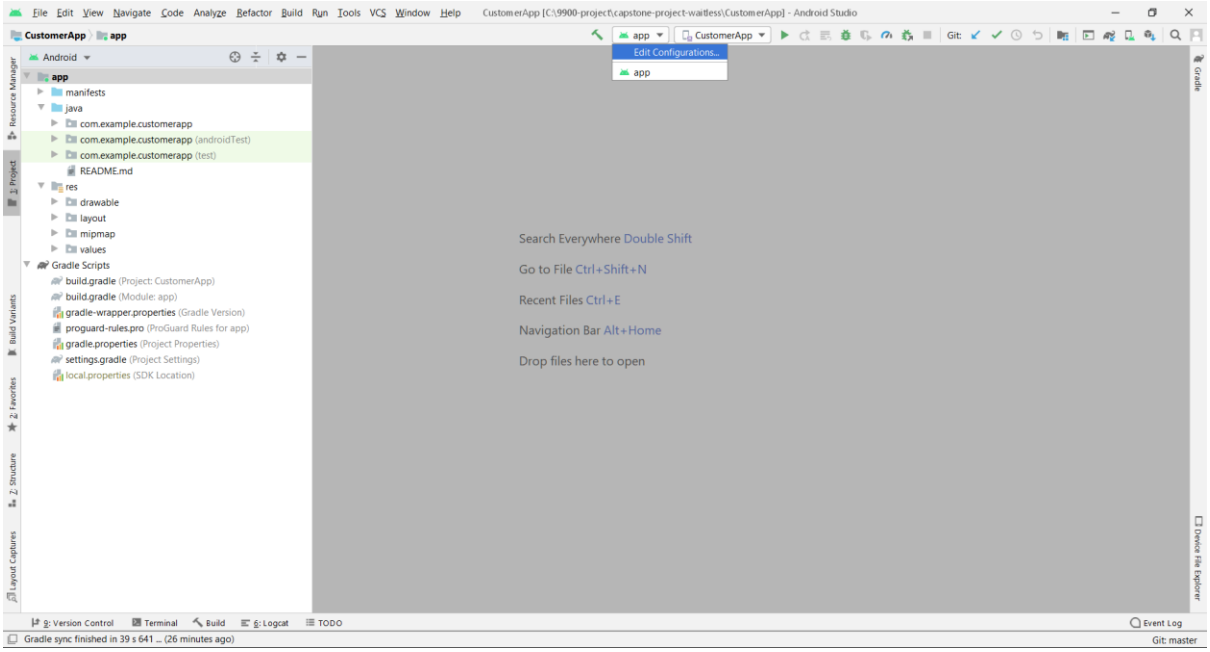



Virtual Device Configuration
Android Virtual Device (AVD)
Verify Configuration
AVD Name
WaiterApp
Pixel C
9.94 2560x1800 xhdpi
Change...
Oreo
Android 8.0 x86
Change...
Startup orientation
Portrait
Landscape
Emulated Performance
Graphics:
Automatic
Show Advanced Settings
AVD Name
The name of this AVD.
Previous
Next
Cancel
Finish
Help

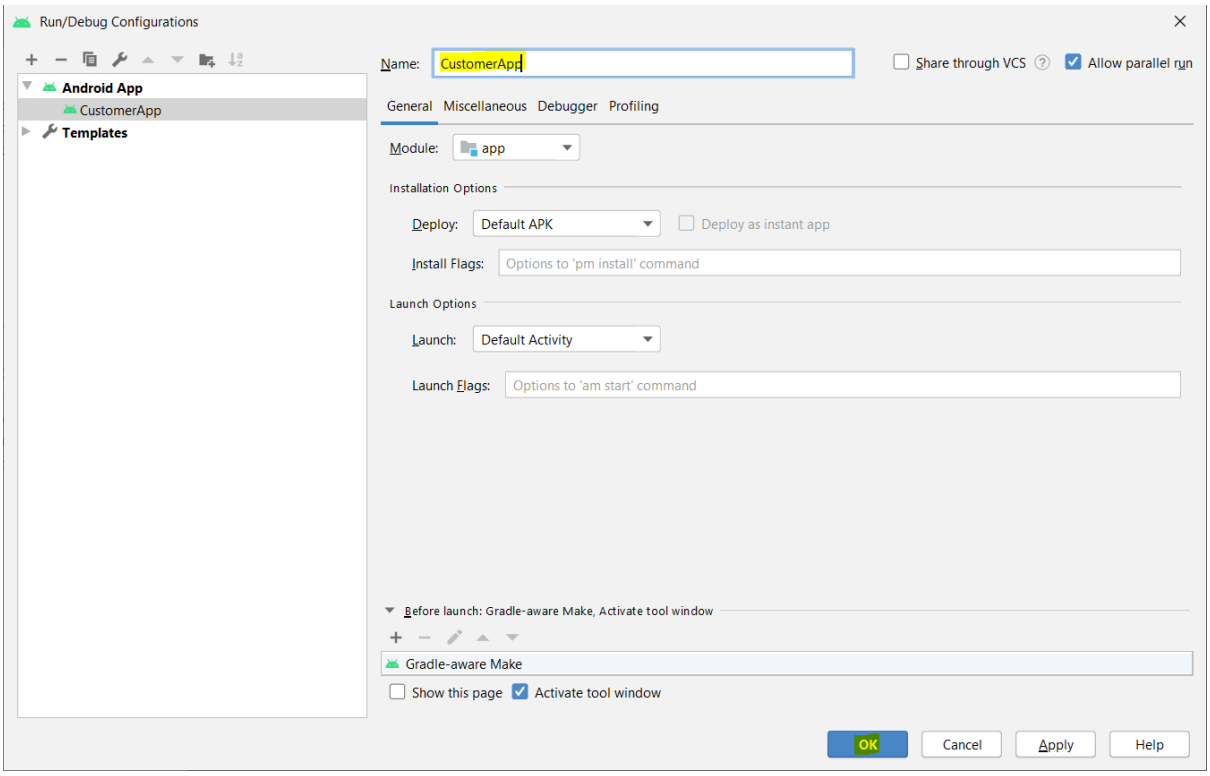

Android Virtual Device Manager
Your Virtual Devices
Android Studio
Type
Name
Play Store
Resolution
API
Target
CPU/ABI
Size on Disk
Actions
CustomerApp
2560 × 1800: xhdpi
26
Android 8.0 (Google APIs)
x86
1.0 GB
KitchenStaffApp
2560 × 1800: xhdpi
26
Android 8.0 (Google APIs)
x86
1.0 GB
WaiterApp
2560 × 1800: xhdpi
26
Android 8.0 (Google APIs)
x86
1.0 GB
Create Virtual Device...

Running CustomerApp Android App

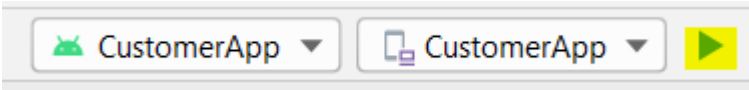
Click on the dropdown next to 'Hammer' icon and select 'Edit Configurations…'



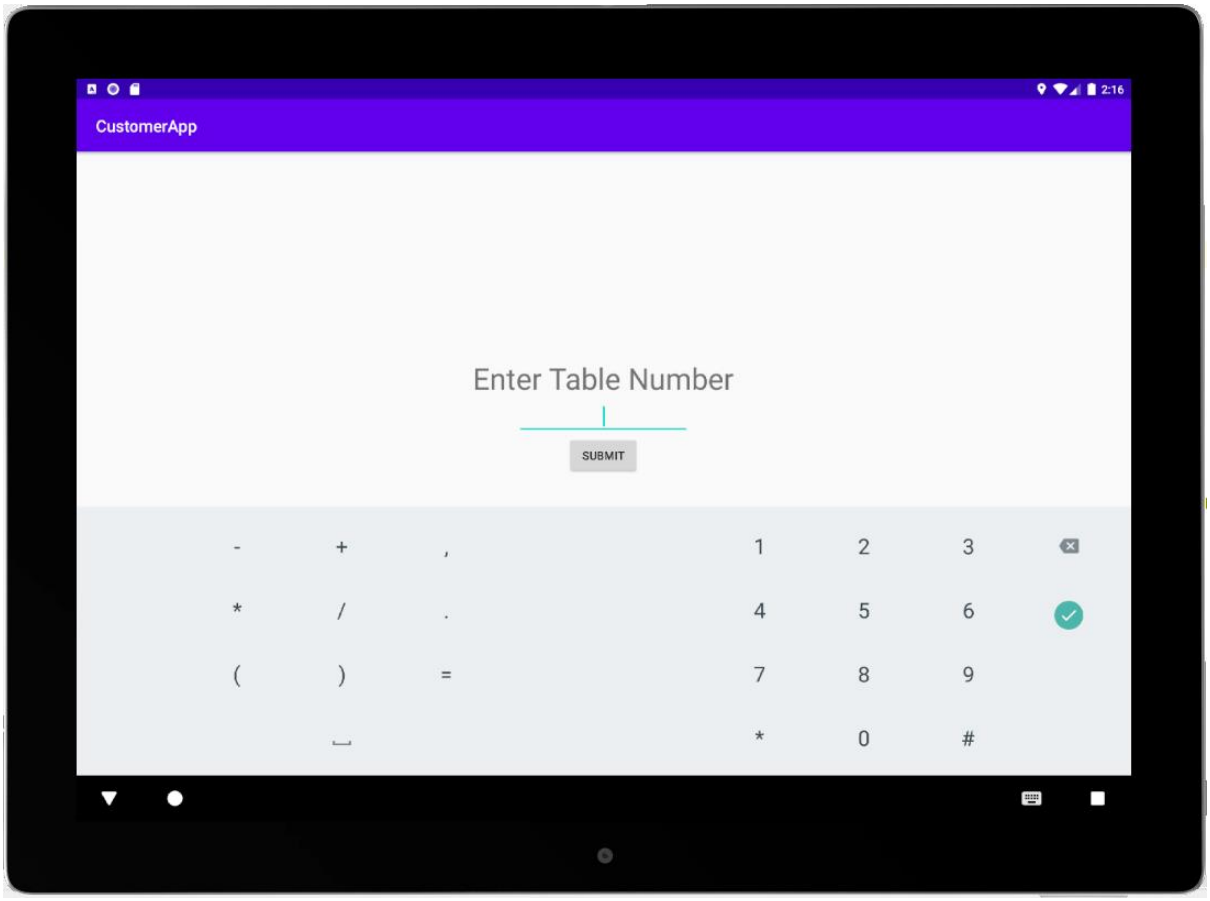
Give name as 'CustomerApp' and click 'OK'.



Now the CustomerApp can be run in the 'Pixel C' android Tab emulator (nicknamed as 'CustomerApp') by clicking on the green 'Play' button.



This will execute the 'Gradle' build and start the Emuator with the 'CustomerApp'.

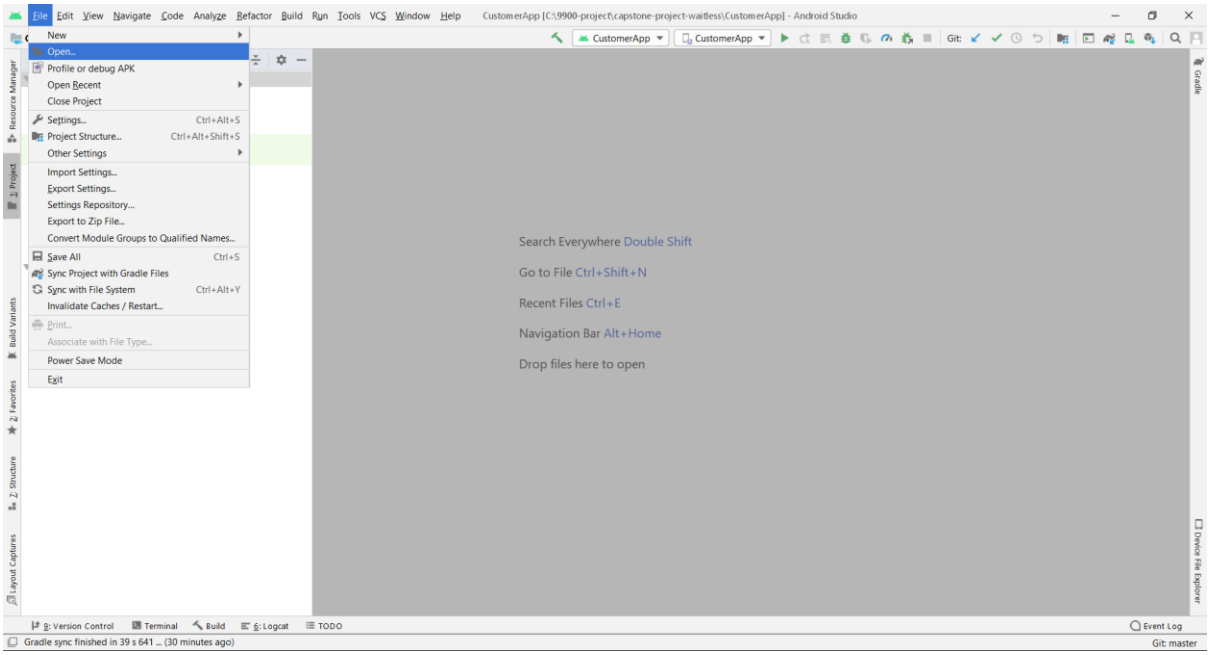


The details on how to use the application will be available in the 'Functionalities and Implementation Challenges' section.

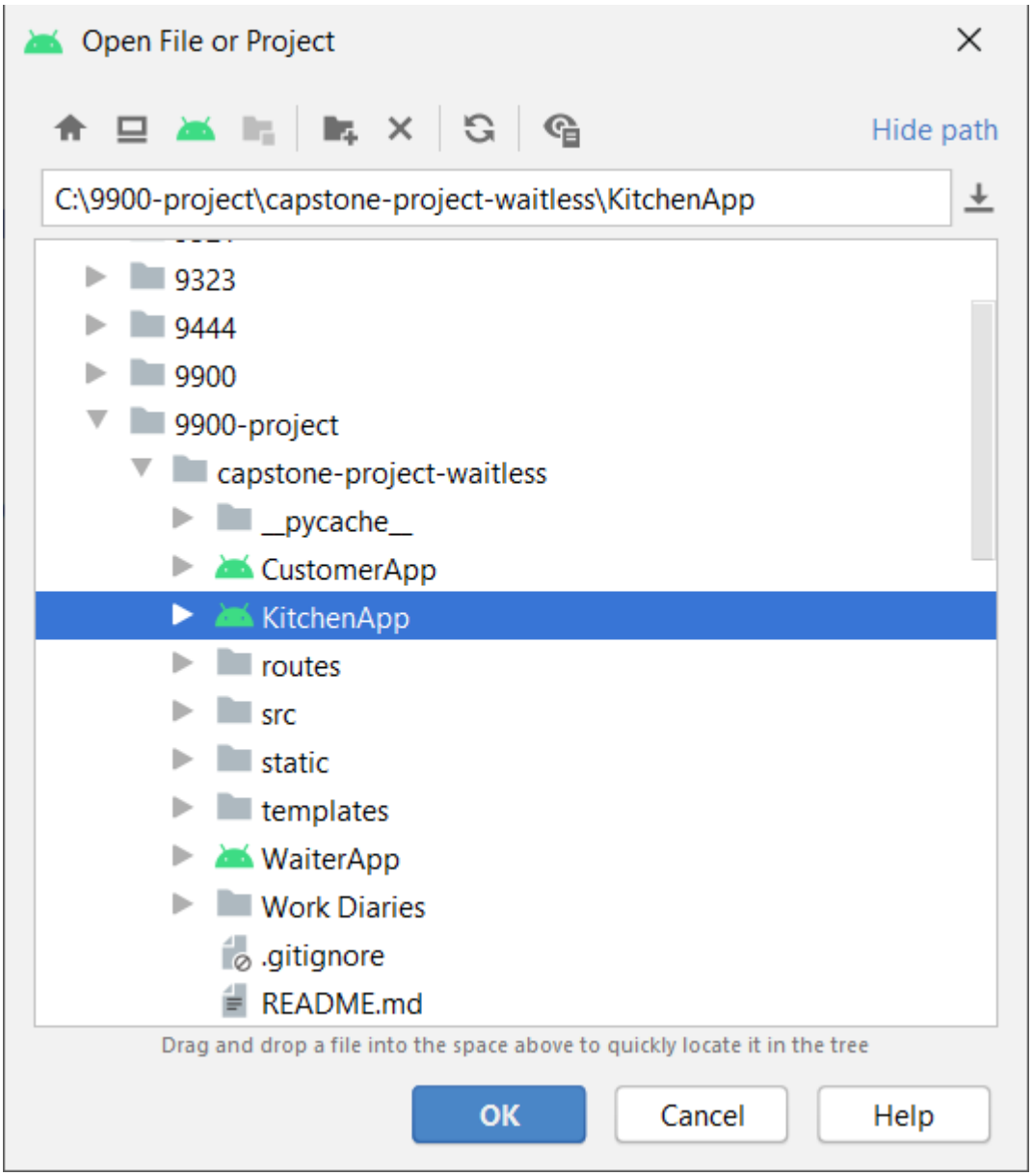
Running KitchenStaff App and Waiter Apps

Similarly, to execute the Kitchen App and Waiter Apps the following steps are to be done;

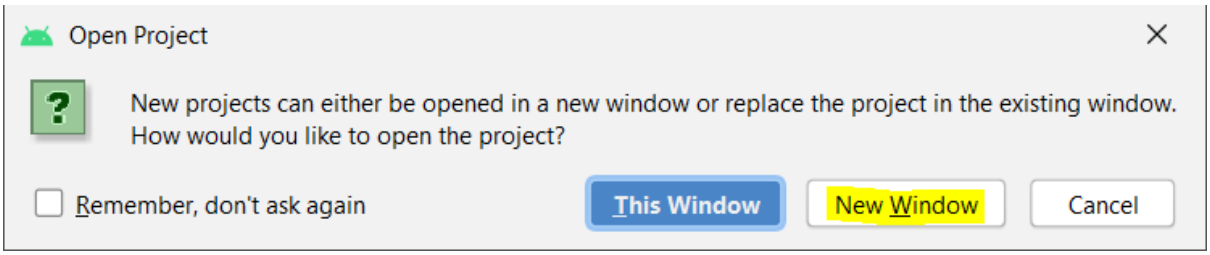
Click 'Open' from 'File' menu;



Select ‘KitchenApp’ and click ‘OK’

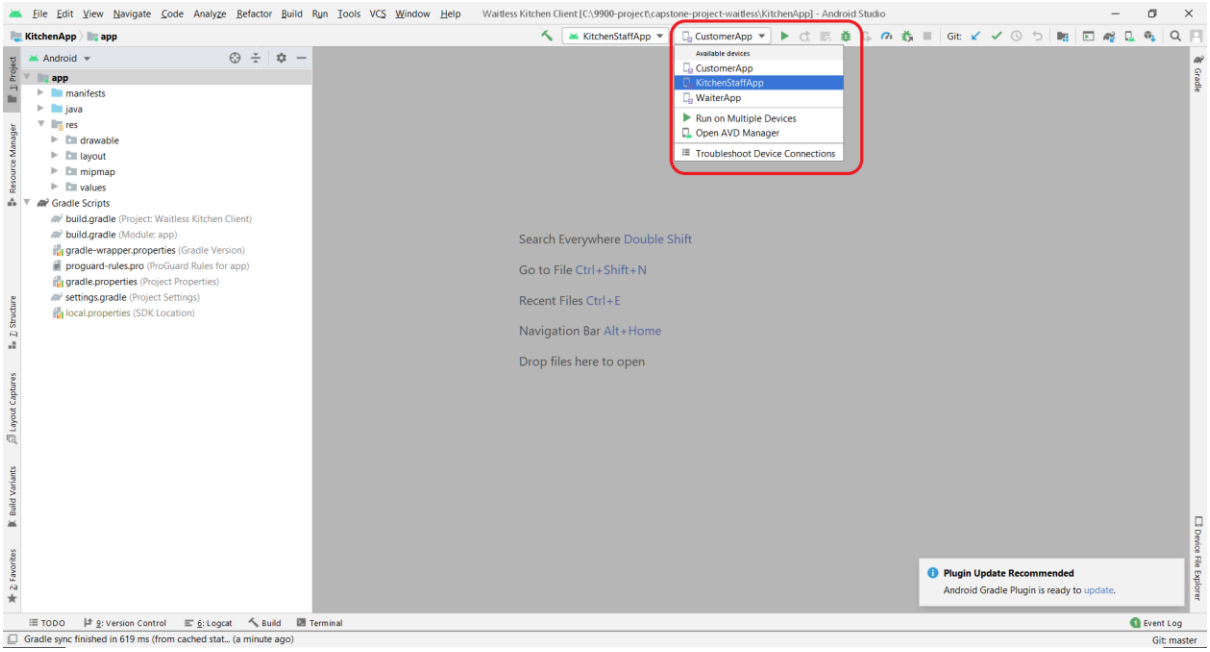


Select ‘New Window’.

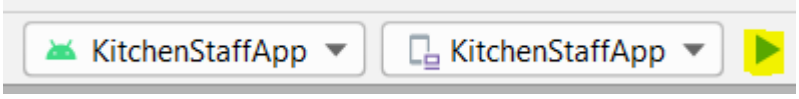


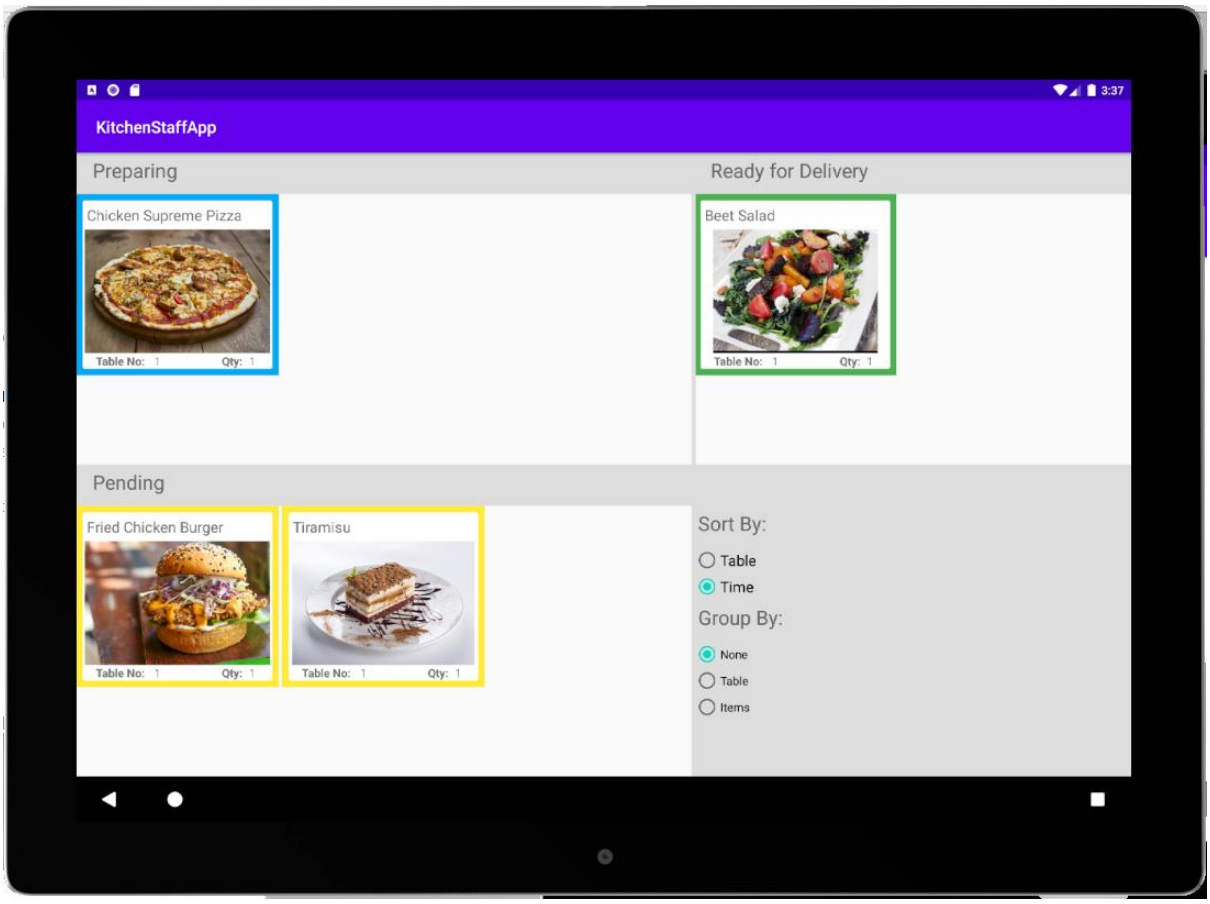
Click on the dropdown next to ‘Hammer’ icon and select ‘Edit Configurations…’ and give name as ‘KitchenStaffApp’ (like how it was done for CustomerApp).

In the AVD dropdown, select ‘KitchenStaffApp’.

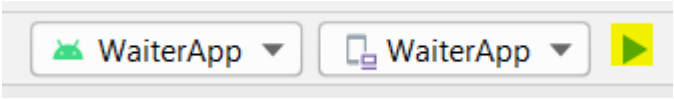


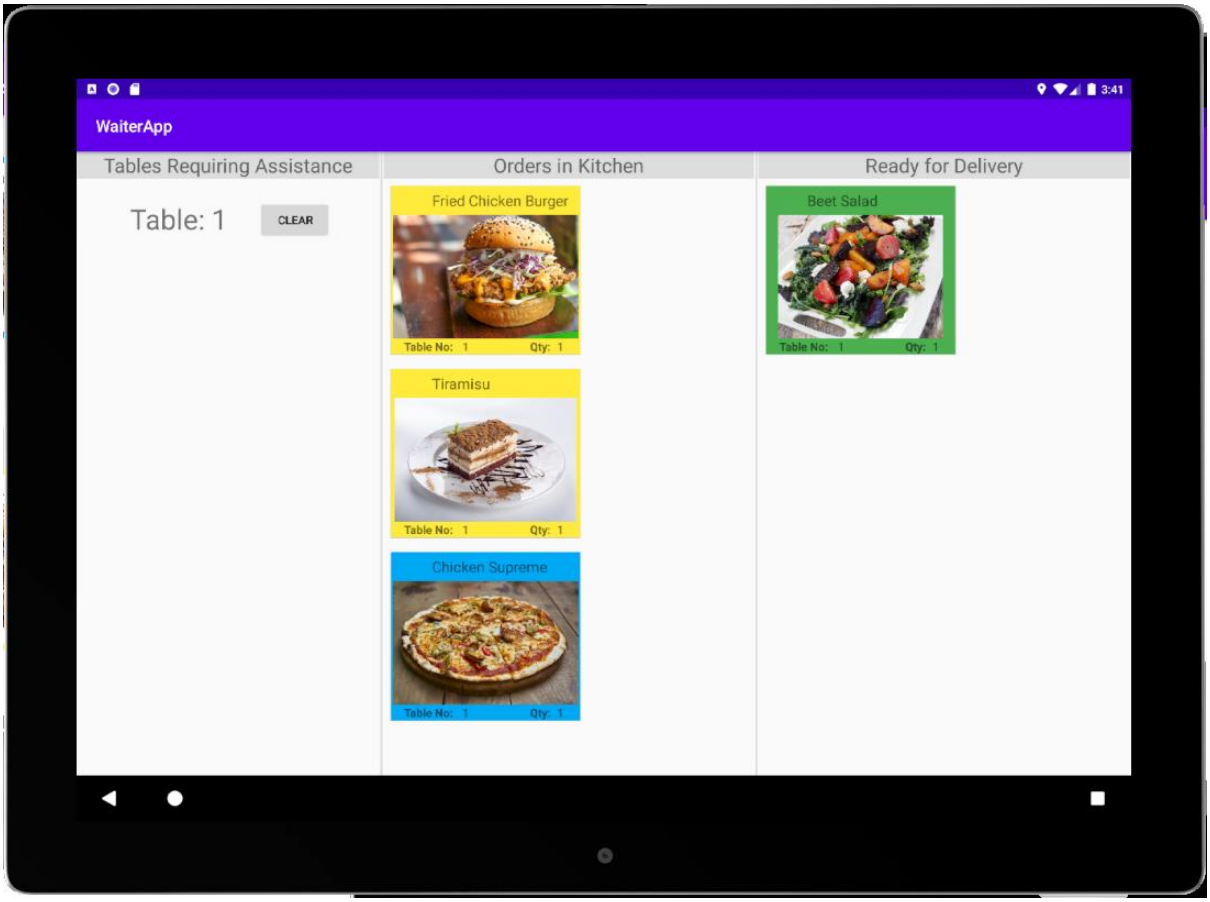
Now, the KitchenStaff App can be executed on the Pixel C device emulator as shown below;





Following the exact similar steps will enable the WaiterApp to be run on the WaiterApp AVD.





The details on how to use the KitchenStaff and Watier Apps will be available in the 'Functionalities and Implementation Challenges' section.

This completes the Install / Setup and Running of all the different applications in this Project.