

# Kinder Food Finder

## Manual for running the app and server

### Contents

<b>A. System installation guide</b>	3
1. Preparation	3
1.1. Requirements	3
1.2. Files/Documents sent to the client	3
1.3. Source Code on GitHub	3
2. Kinder-Food-Finder Server	4
2.1. Operating System	4
2.2. Device	4
2.3. Software Installation/Preparation	4
2.4. Steps to Install NodeJS	4
2.5. Steps to Install ExpressJS	4
2.6. Steps to Install and Run/Stop/Kill MongoDB	5
2.7. Steps to Install and Run Robo 3T (Optional)	5
2.8. Steps to Run the Server	6
3. Manual for Kinder-Food-Finder Application	12
3.1. Operating System	12
3.2. Device	12
3.3. Software Installation/Preparation	12
3.4. Steps to Open Developer Mode in Your Android Device and Enable USB Debugging	12
3.5. Steps to Close Developer Mode If Not Needed Any More	13
3.6. Steps to Setup Virtual Device in Android Studio If No Android Device (Optional)	14
3.7. Steps to Generate APK File of Application (Optional)	18
3.8. Steps to Find Out Your IP Address in Your Laptop/Desktop Which Will Run Server	21
3.9. Steps to Run Our Application and Install It in Your Android Device/Virtual Device	21
<b>B. Guidance on operating the server-side</b>	25
1. Login to the Server-side	26
2. Import/ Insert data	29

<b>3.</b>	<b>Update/ delete brand data .....</b>	<b>32</b>
<b>4.</b>	<b>Update/ delete store data .....</b>	<b>35</b>
<b>5.</b>	<b>Report .....</b>	<b>40</b>
<b>6.</b>	<b>Publish .....</b>	<b>41</b>
<b>7.</b>	<b>Statistics.....</b>	<b>41</b>

## A. System installation guide

### 1. Preparation

#### 1.1. Requirements

- Laptop/desktop (Mac/MacBook recommended)
- Android mobile device
- Source code of both app and server
- A script to install essential libraries before server setup
- Sample CSV file for brand data (correct format required)

#### 1.2. Files/Documents sent to the client

- A folder called “**KinderFoodFinder**” that consists of source code for application
- A folder called “**KFF\_Server**” that consists of source code for server
- A sample csv file called “**products.csv**”
- A script called “**installExpress**”
- A manual for app and server called “**Kinder Food Finder manual**”

#### 1.3. Source Code on GitHub

- As the source code could be updated later, you will need to download the newer version on our GitHub: <https://github.com/WolfeLeee/comp5703project>

## 2. Kinder-Food-Finder Server

### 2.1. Operating System

- Laptop/desktop: macOS

### 2.2. Device

- Mac/MacBook

### 2.3. Software Installation/Preparation

- **WebStorm** installed in the laptop/desktop (<https://www.jetbrains.com/webstorm/> and free 30-day trial or as Sydney Uni student, you can apply a student account from <https://www.jetbrains.com/shop/eform/students> so you are free to use it until you graduate)
- (Optional) **Eclipse** or **IntelliJ** can replace WebStorm to run the server code
- **NodeJS** installed in your laptop/desktop
- **ExpressJS** installed in your laptop/desktop
- **MongoDB** used as database installed in your laptop/desktop
- (Optional) **Robo 3T** installed in your laptop/desktop so you can see data directly in MongoDB

### 2.4. Steps to Install NodeJS

- Go to the website: <https://nodejs.org/en/download/>
- Download the **macOS Installer (.pkg)**
- Run the installer you have downloaded
- Keep clicking on Next and accept the License Agreement and done

### 2.5. Steps to Install ExpressJS

- Go to the website and install **Homebrew** first: <https://brew.sh/>
- Open the **terminal** in your laptop/desktop
- Type the command brew install npm and enter to **install “npm”** first by Homebrew
- Type the command sh ScriptName and enter to run the script, this will install all the essential elements for Express. The “**ScriptName**” should be the script sent to you

at the beginning called “installExpress”. You can simply type “sh ” and then drag the script to the terminal so that it will give the absolute path for the script

- During running the script, it will ask your username and password for the laptop/desktop to give the permission to install them
- After the installation completed, it is done

## 2.6. Steps to Install and Run/Stop/Kill MongoDB

- Go to the website: <https://treehouse.github.io/installation-guides/mac/mongo-mac.html>
- Follow the instructions (recommended to install MongoDB by Homebrew)
- To run the MongoDB, simply type the command mongod in your **terminal** and enter
- To stop the MongoDB, simply hit **Ctrl-C** (Don’t just close the terminal, it won’t stop)
- If you closed the terminal already, then you will need to type the command **killall - 15 mongod** and enter to kill the MongoDB service

## 2.7. Steps to Install and Run Robo 3T (Optional)

- Go to the website: <https://robomongo.org/download>
- Select the button “**Download Robo 3T**”
- Select the Mac and download the installer (.dmg)
- Double click on the installer to run the installation
- After the installation completed, run **Robo 3T**
- Create a connection to local host
- **Type:** Direct Connection; **Name:** can be anything; **Address:** localhost : 27017
- Click on Save and choose the connection created by you
- Click on Connect
- Now you will see a connection called the name you defined while creating on the left-hand side and double click on it
- There will be a database called “**kff**” after the server runs
- You will see all the collections that server has under the database

## 2.8. Steps to Run the Server

- Complete the installations for **NodeJS**, **ExpressJS**, and **MongoDB**;
- Install **WebStorm** (Other options: Eclipse or IntelliJ) and open it;
- Select “**Open**” (Refer to Figure 1) and open the folder called “**KFF\_Server**” sent to you;
- After then, click on “**Add Configuration...**” at the top-right side (Refer to Figure 2);
- Click on **add (+)** button at the top-left side in the pop-up window (Refer to Figure 3);
- Select “**Node.js**” and it will open the configuration in the right side (Refer to Figure 4 and 5);
- In the “**JavaScript file**”, type “**app.js**” and click on “**OK**” (Refer to Figure 6);
- Now you will see the **arrow icon with green colour** at the top-right side that you are ready to run the code (Refer to Figure 7)
- Before running the server code, make sure **MongoDB** has been run (See the tutorial from above)
- If running **successfully**, it will show you the message “**KFF app listening on port 3000!**” and “**mongodb connected!**” (Refer to Figure 8)
- Now you can prepare and run application by following the **manual for application** in the next page

**Note:** If you are using **Eclipse** or **IntelliJ**, then it is the same that you add the configuration for “**Node.js**”.

**\*\*\*Note:** If you want to **change the port** to whatever you like as **the default port setup is 3000**, then you can go to **app.js** in the folder of “**KFF\_Server**” sent to you and change the port to yours as **Figure 21** below.

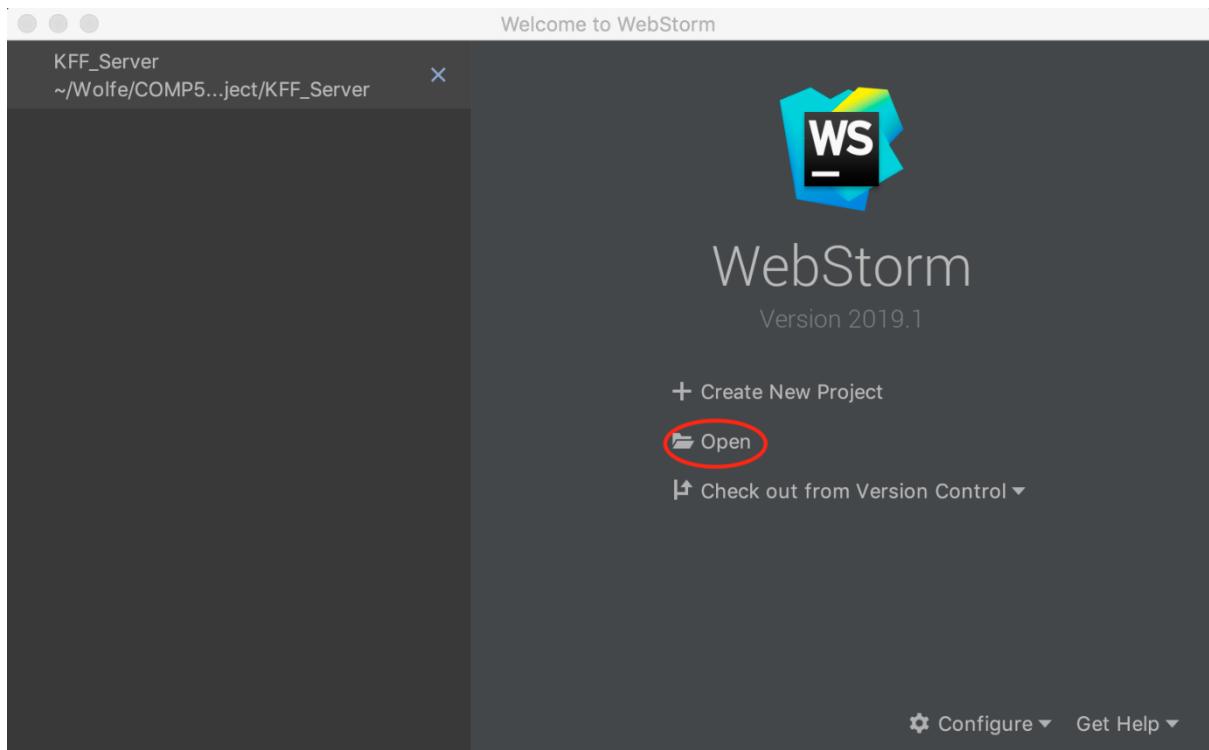


Figure 1. Open the new project

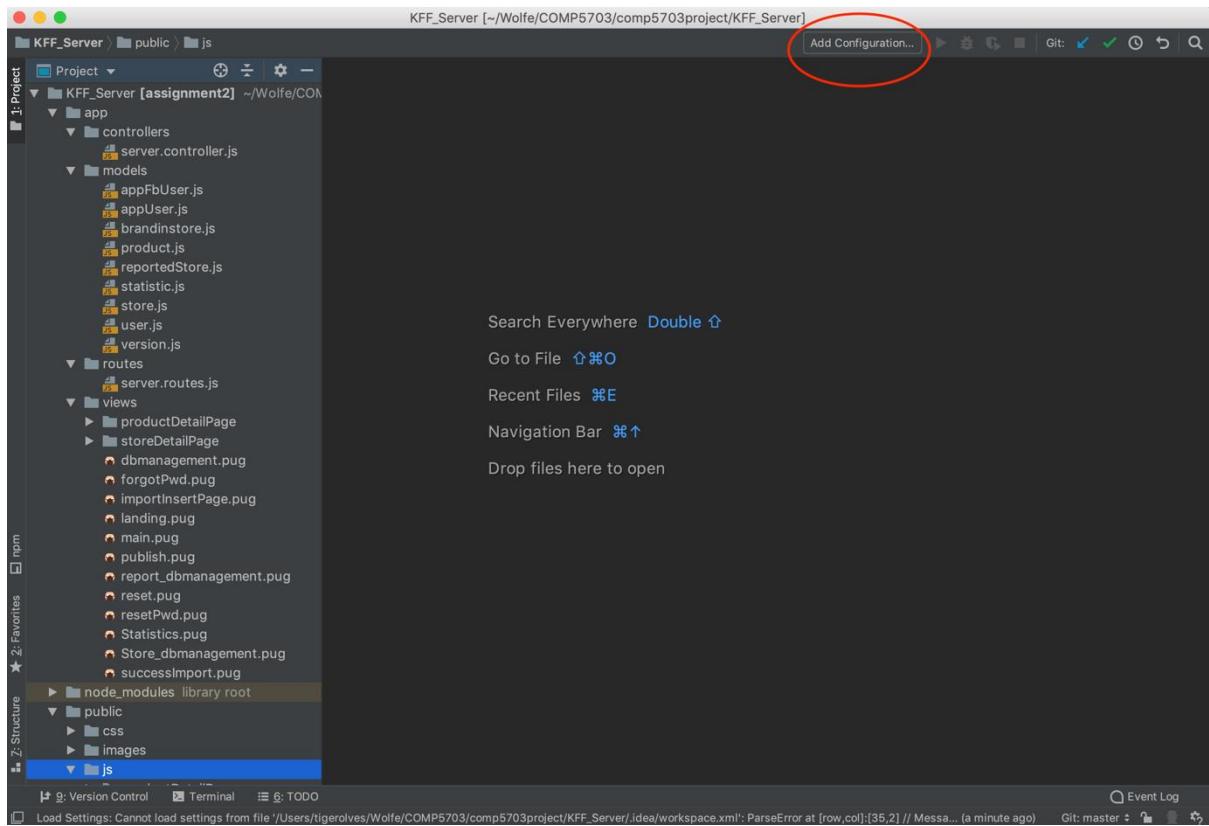


Figure 2. Add configuration

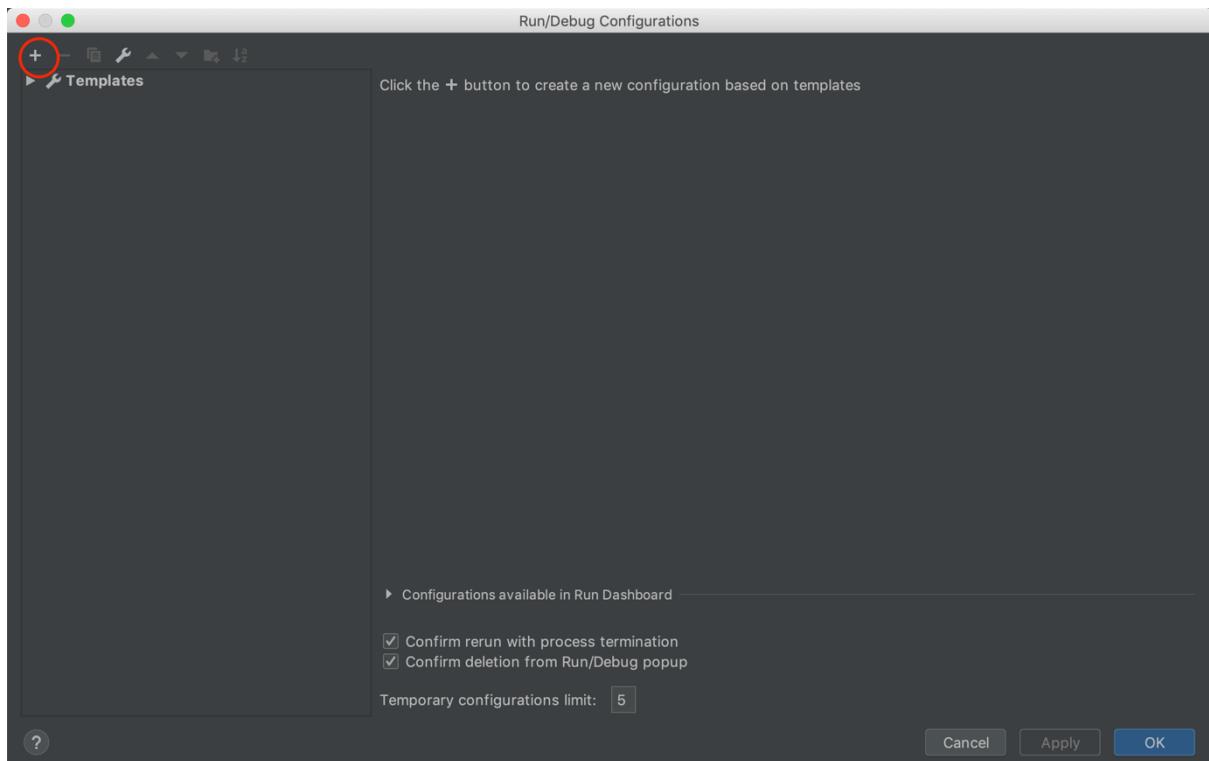


Figure 3. Add (+) button

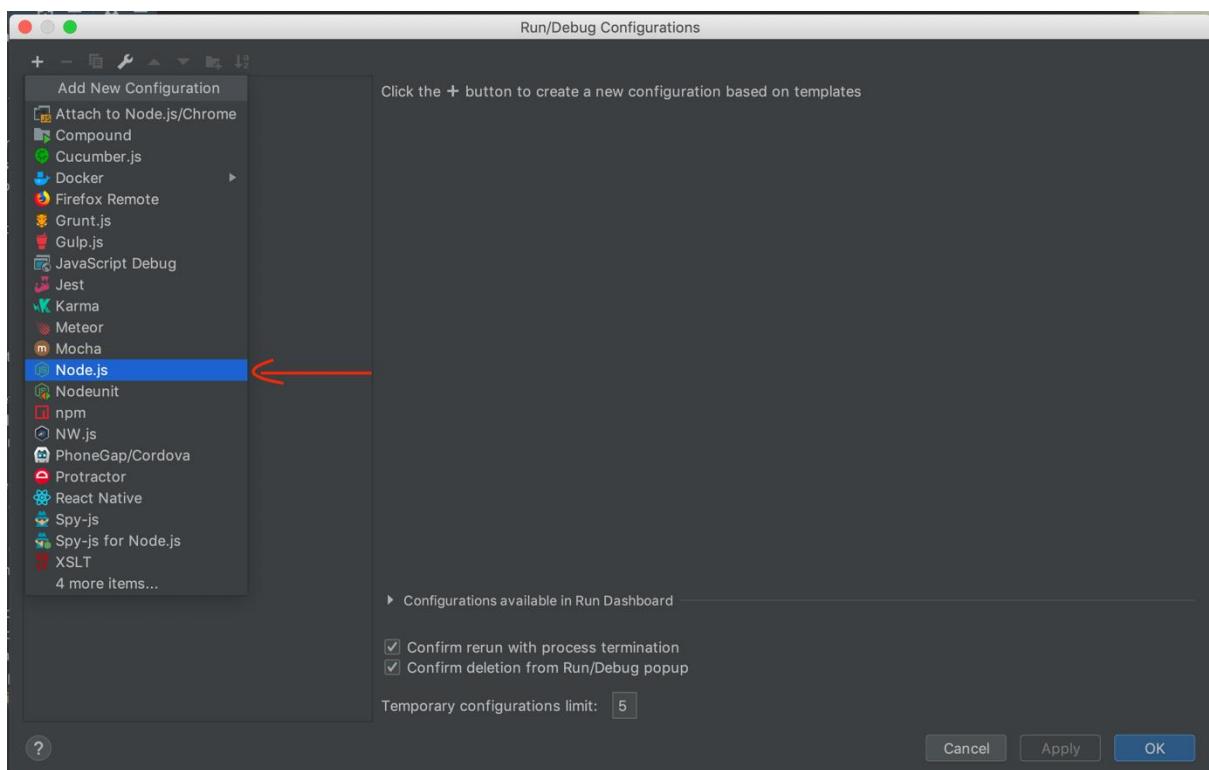


Figure 4. Select Node.js for configuration

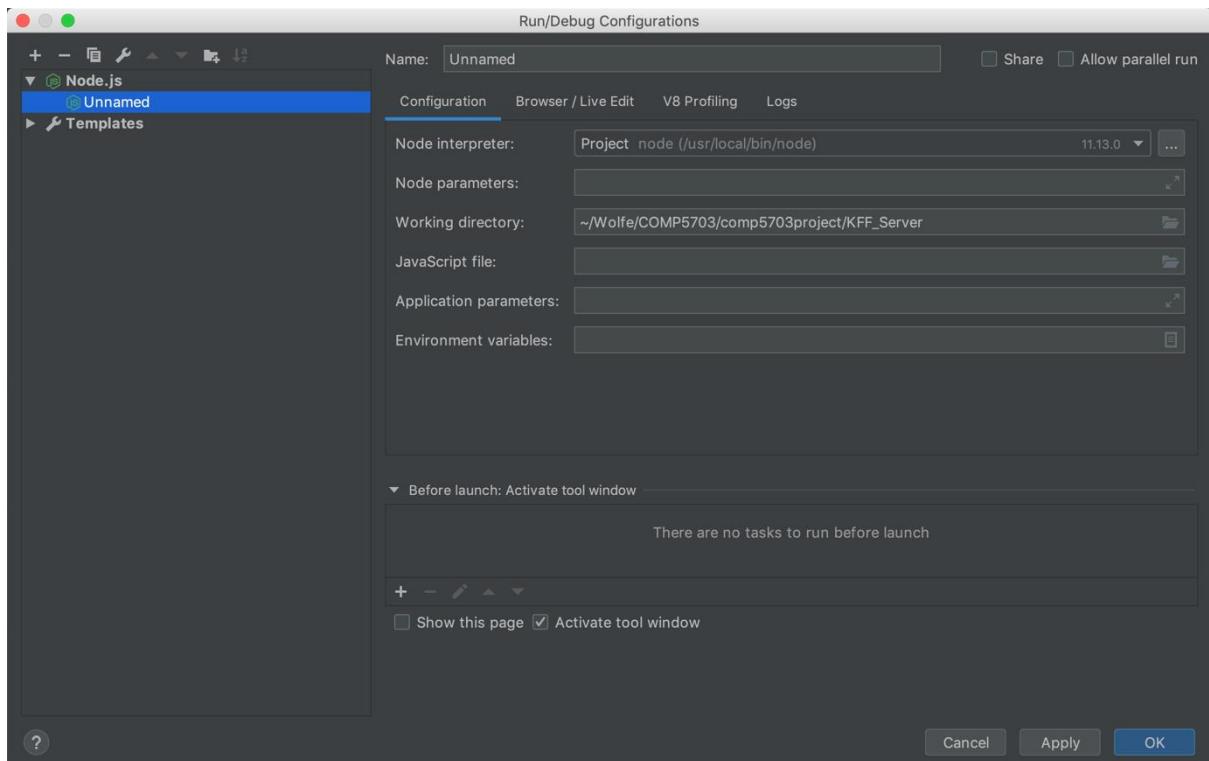


Figure 5. Configuration setup

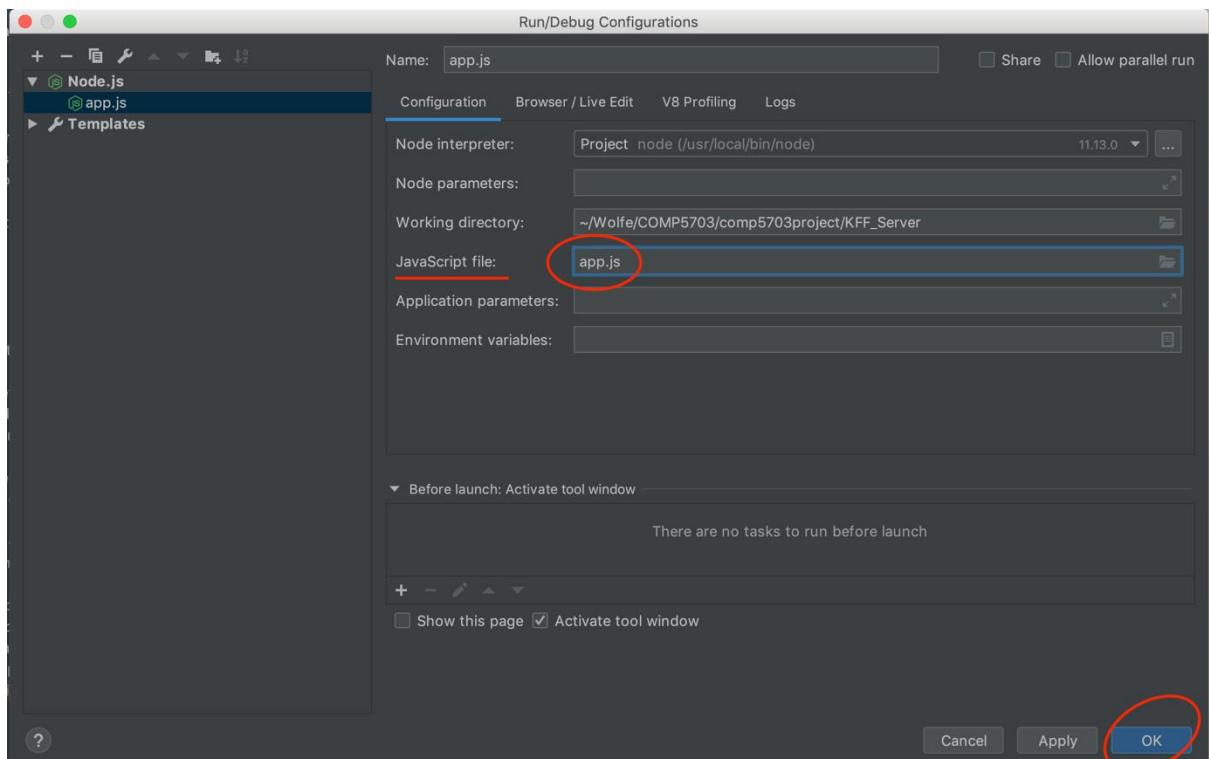


Figure 6. The JavaScript file to run the code

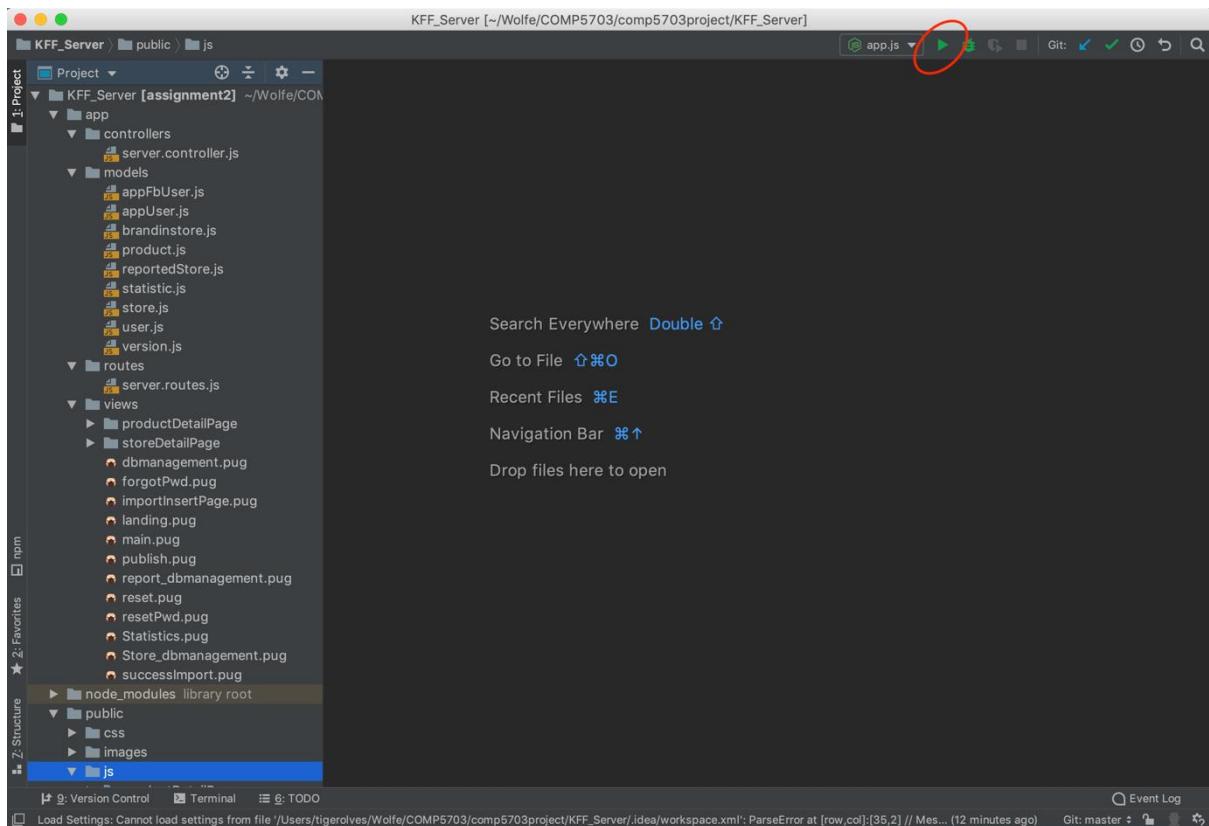


Figure 7. Run the code

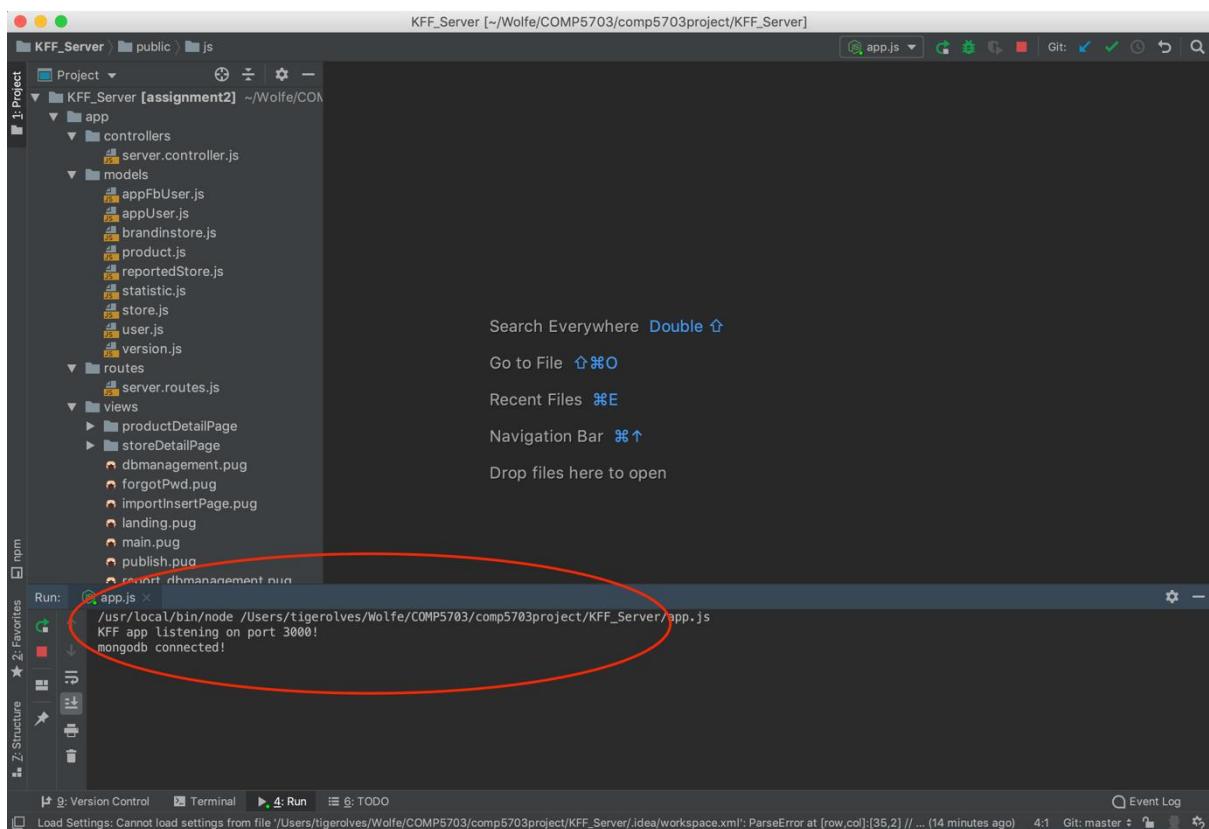


Figure 8. Successfully running message

The screenshot shows a code editor interface with the following details:

- Project:** KFF\_Server [assignment2]
- File:** app.js
- Code Content:**

```
51 app.set('view engine', 'pug');
52
53 // set the routes
54 var routes = require('./app/routes/server.routes');
55 app.use('/', routes);
56
57 // catch 404 and forward to error handler
58 app.use(function (req, res, next) {
59   var err = new Error('File Not Found!');
60   err.status = 404;
61   next(err);
62 });
63
64 // error handler
65 // define as the last app.use callback
66 app.use(function (err, req, res, next) {
67   res.status(err.status || 500);
68   res.send(err.message);
69 });
70
71 // console.log if all ready
72 app.listen(3000, function() {
73   console.log('KFF app listening on port 3000!');
74 });
75
76 module.exports = app;
```
- Run Tab:** Shows the command: /usr/local/bin/node /Users/tigervolves/Wolfe/COMP5703/comp5703project/KFF\_Server/app.js
- Output:** KFF app listening on port 3000!  
mongodb connected!
- Annotations:** The port number 3000 in the app.listen() line is circled in red.

Figure 9. Change the port of server from the default port 3000

### **3. Manual for Kinder-Food-Finder Application**

#### **3.1. Operating System**

- Mobile device: Android;
- Laptop/desktop: macOS.

#### **3.2. Device**

- Any android device (e.g. Nexus 6);
- Mac/MacBook.

#### **3.3. Software Installation/Preparation**

- **Android Studio** installed in the laptop/desktop (<https://developer.android.com/studio>);
- Android device required to open **developer mode**;
- (Optional) Using **Android Virtual Device** in Android Studio to run application if you don't have any Android device.

#### **3.4. Steps to Open Developer Mode in Your Android Device and Enable USB Debugging**

- Go to the **Settings** in your device;
- Scroll down to find the **About/About Phone** and click on it;
- Scroll down again to find **Build number** (Refer to Figure 10);
- There are **few popular devices** and their **location of Build number** (Refer to Figure 11);
- Tap on it **7 times** (After a few taps, you may see a small pop-up alert telling you that “You are now X steps away from being a developer”);
- After the seventh tap, a message will appear telling you that “You are now a developer!”;
- Go back to the **first page of Settings**;
- Scroll down to find the **Developer options** and click on it;
- Scroll down to find **USB Debugging** under **Debugging tag** and turn it on.

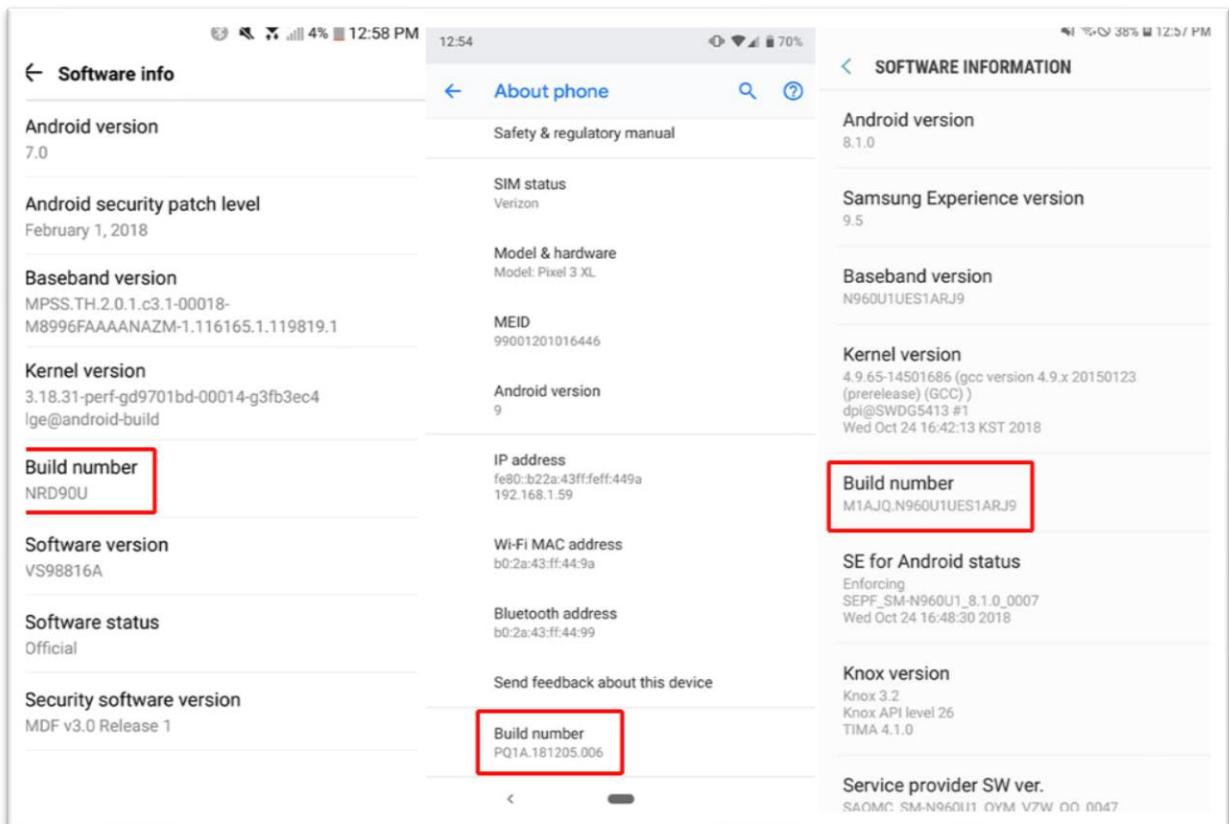


Figure 10. The location of Build number

**Google Pixel:** Settings > System > About phone > Build number

**Samsung Galaxy S8 and later:** Settings > About phone > Software information > Build number

**LG G6 and later:** Settings > About phone > Software info > Build number

**HTC U11 and later:** Settings > About > Software information > More > Build number

**OnePlus 5T and later:** Settings > About phone > Build number

Figure 11. How to find Build number in few popular devices

### 3.5. Steps to Close Developer Mode If Not Needed Any More

- Go to the **Settings** in your device
- Scroll down to find the **Developer options** and click on it

- After you get in, you will see **an on/off toggle** at the top of screen
- Switch it to **Off**

### **3.6. Steps to Setup Virtual Device in Android Studio If No Android Device (Optional)**

- Open **Android Studio** in your laptop/desktop after you installed it;
- Select “**Import project (Gradle, Eclipse ADT, etc.)**” (Refer to Figure 12);
- Choose the **folder** called “**KinderFoodFinder**” sent to you;
- Waiting for it building up and synced successfully (Ignore the warning at the bottom-right side) (Refer to Figure 13);
- On the top-right side, there is an icon called “**AVD Manager**” (Refer to Figure 14);
- Click on it and a window will be opened called “**Android Virtual Device Manager**”;
- Click on the button called “**Create Virtual Device...**” at the bottom-left side (Refer to Figure 15);
- Select **Phone** for category and Choose **Nexus 6** (recommended) for device (Refer to Figure 16);
- After clicking on **Next**, select **Pie** for system image (Download it if you do not have the file) and click on **Next** (Refer to Figure 17);
- In verify configuration page, you can define the **name** for this Android Virtual Device and then select **Portrait** view for Start-up orientation (Refer to Figure 18);
- Click on **Finish**.

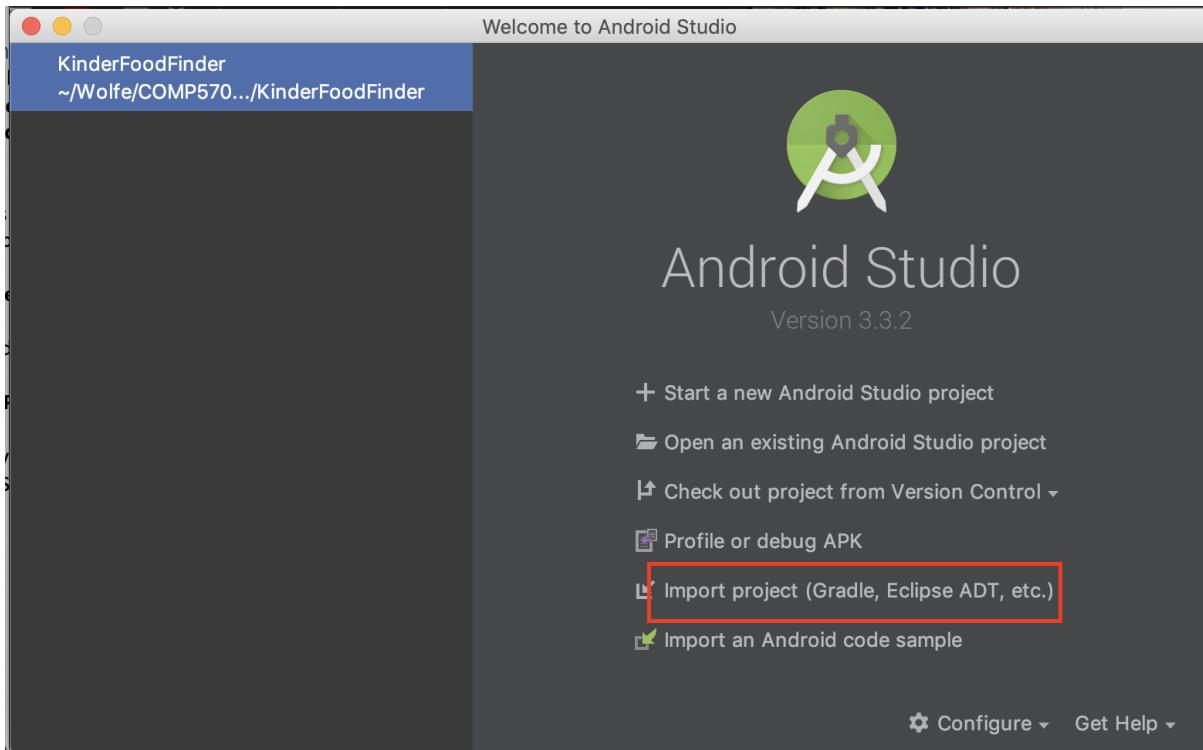


Figure 12. Open Android Studio and import project

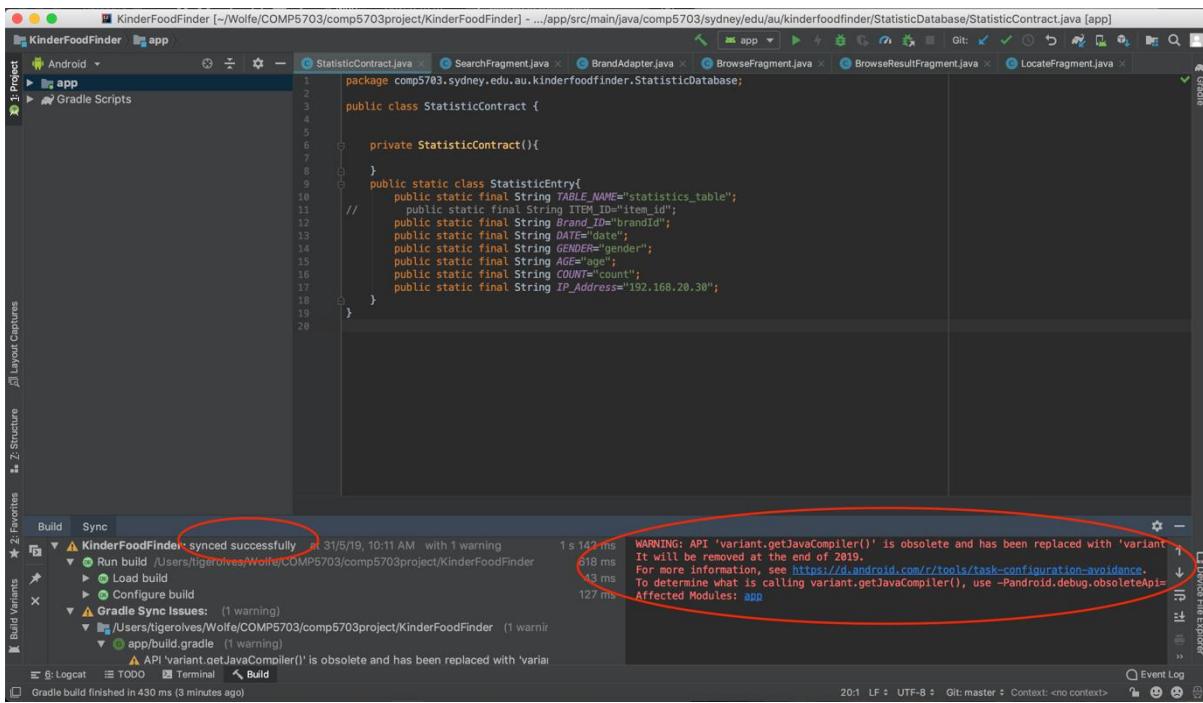


Figure 13. Successfully build up and warning message

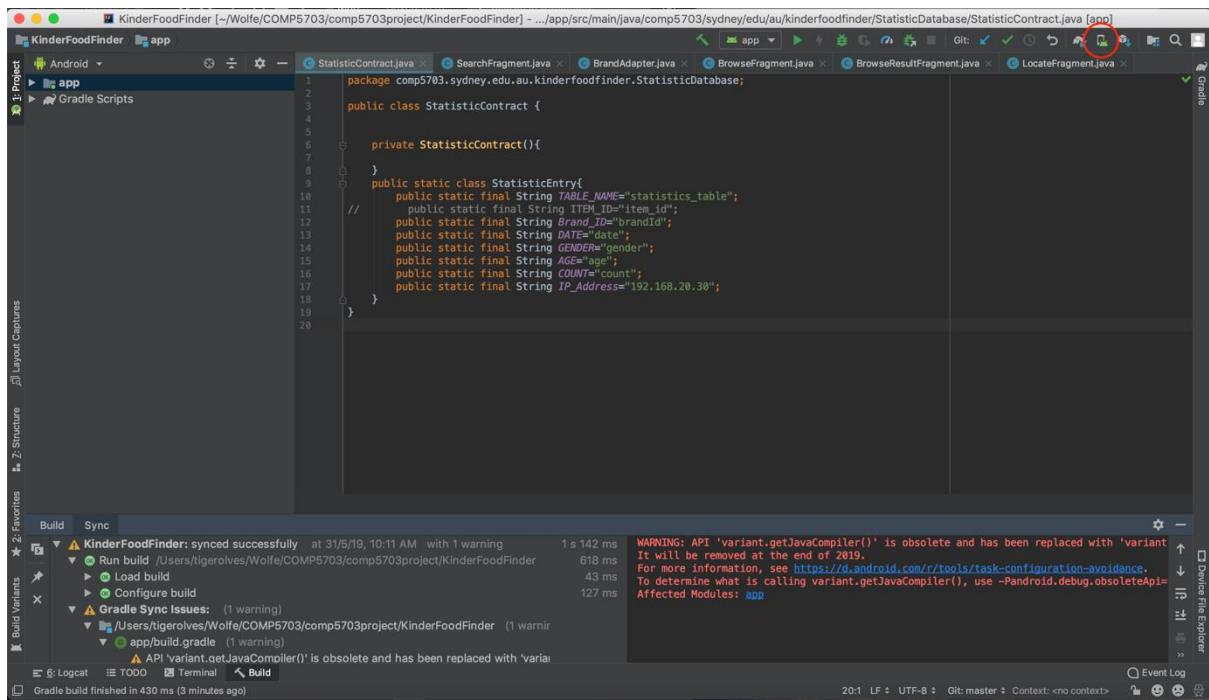


Figure 14. Icon for AVD Manager

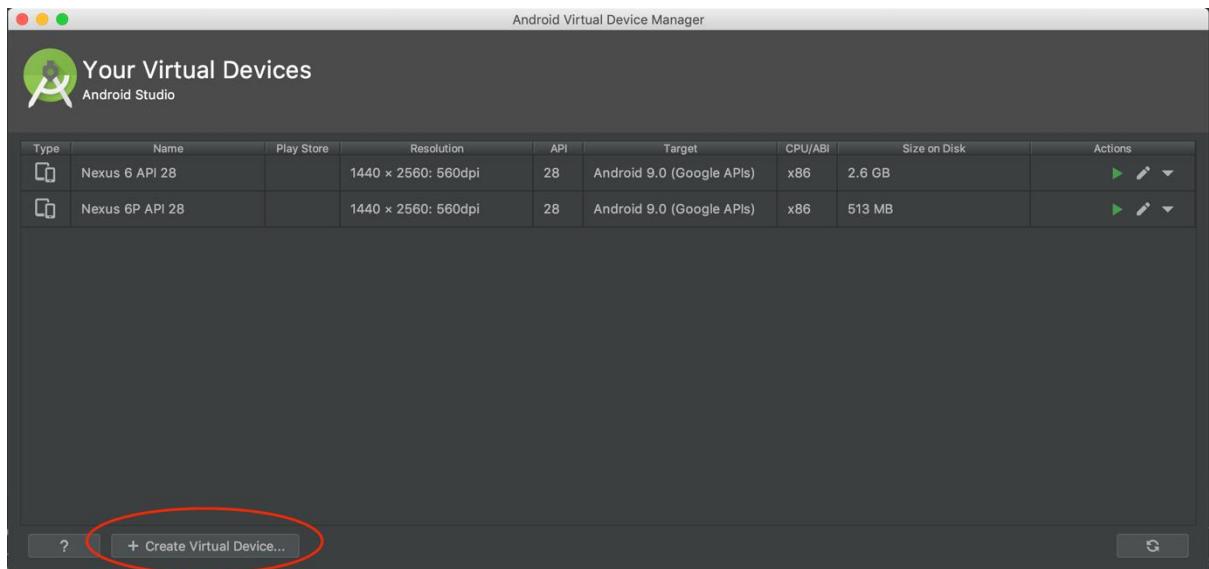


Figure 15. Create new virtual device

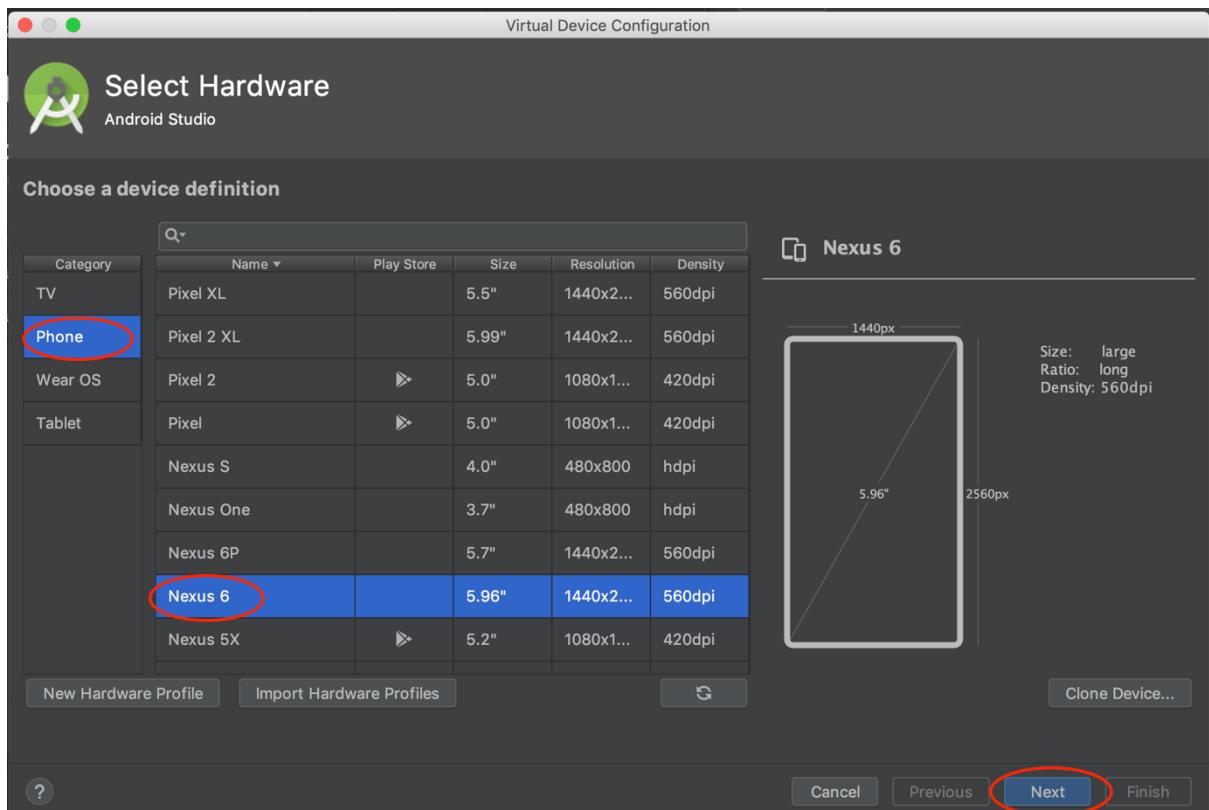


Figure 16. Select category and device

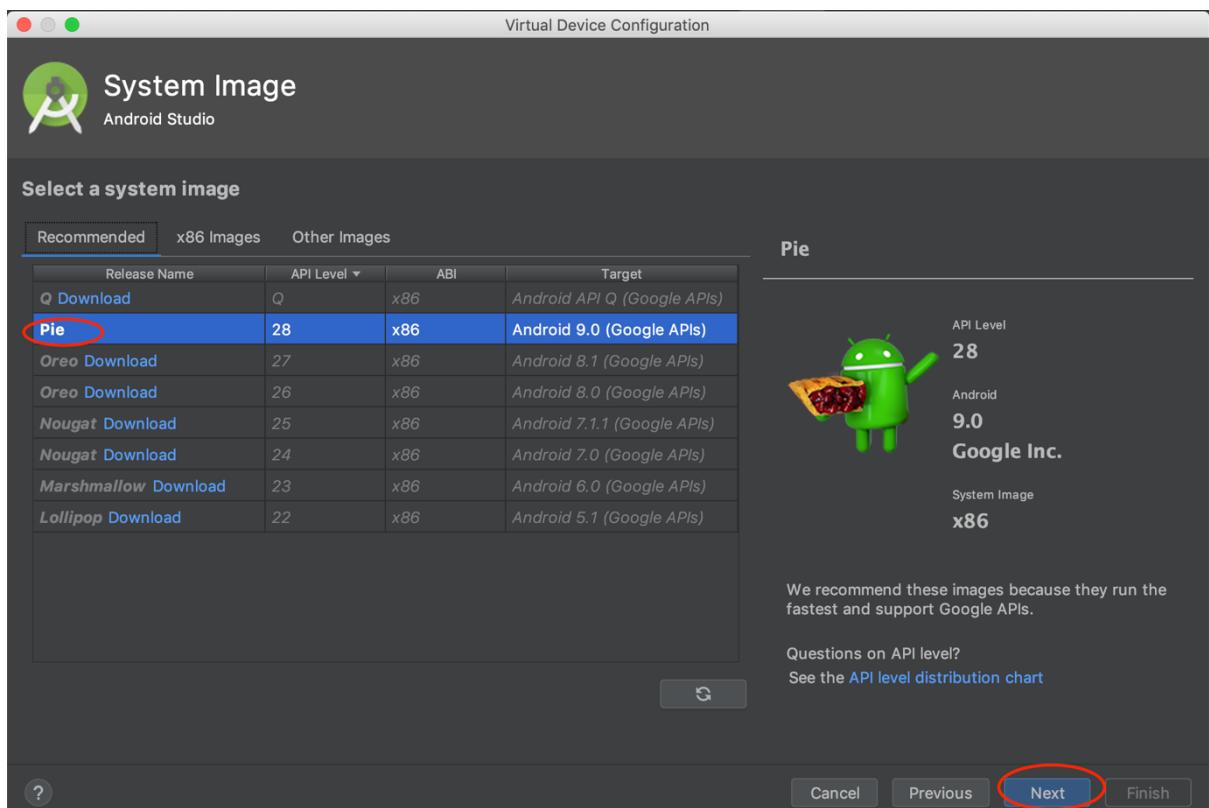


Figure 17. Select a system image

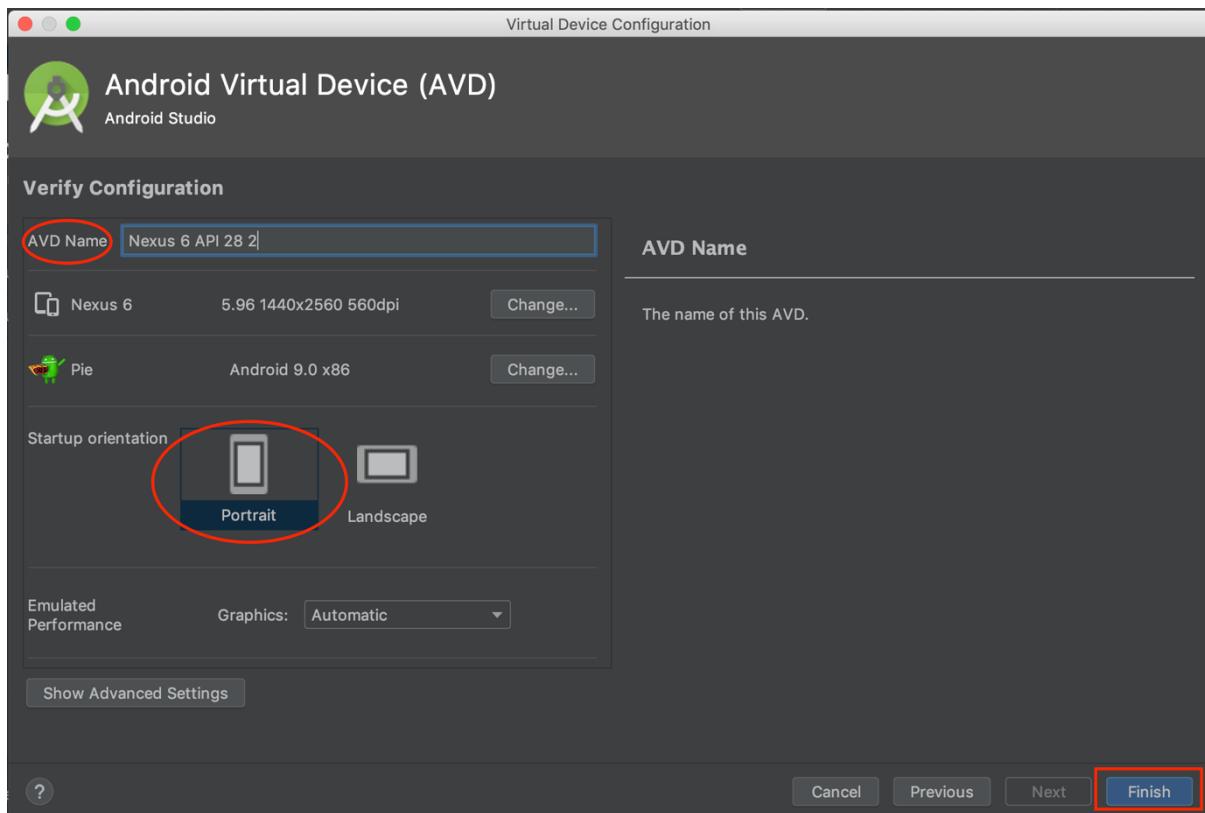


Figure 18. Define the name and start-up orientation

### 3.7. Steps to Generate APK File of Application (Optional)

- Open **Android Studio** in your laptop/desktop after you installed it
- Select “**Import project (Gradle, Eclipse ADT, etc.)**” (Refer to Figure 12 above)
- Choose the **folder** called “**KinderFoodFinder**” sent to you
- Waiting for it building up and synced successfully (Ignore the warning at the bottom-right side) (Refer to Figure 13 above)
- On the top action bar, go through Build -> Build Bundle(s) / APK(s) -> Build APK(s) (Refer to Figure 19)
- Wait until the APK build up being successful (Refer to Figure 20)
- Click on “Event Log” at the bottom-right corner and click on “locate” (Refer to Figure 21)
- It will open the located folder that generates the APK file (Refer to Figure 22)

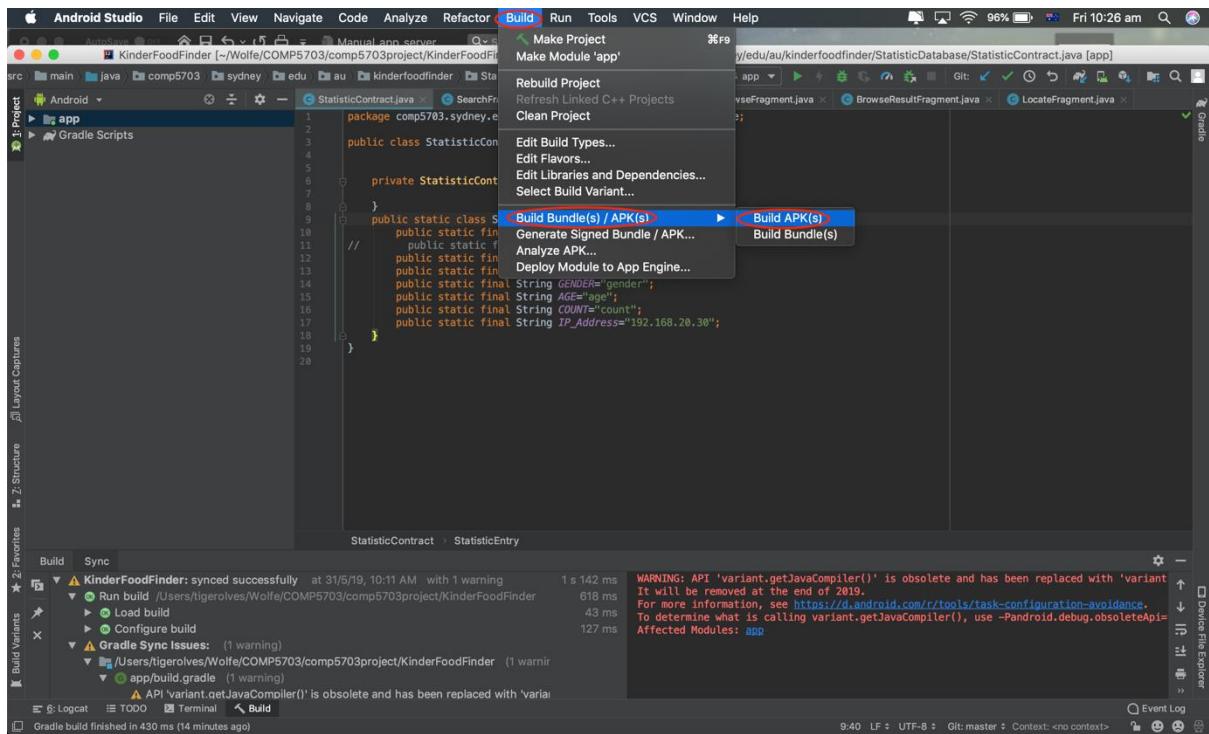


Figure 19. Build-up APK file for application

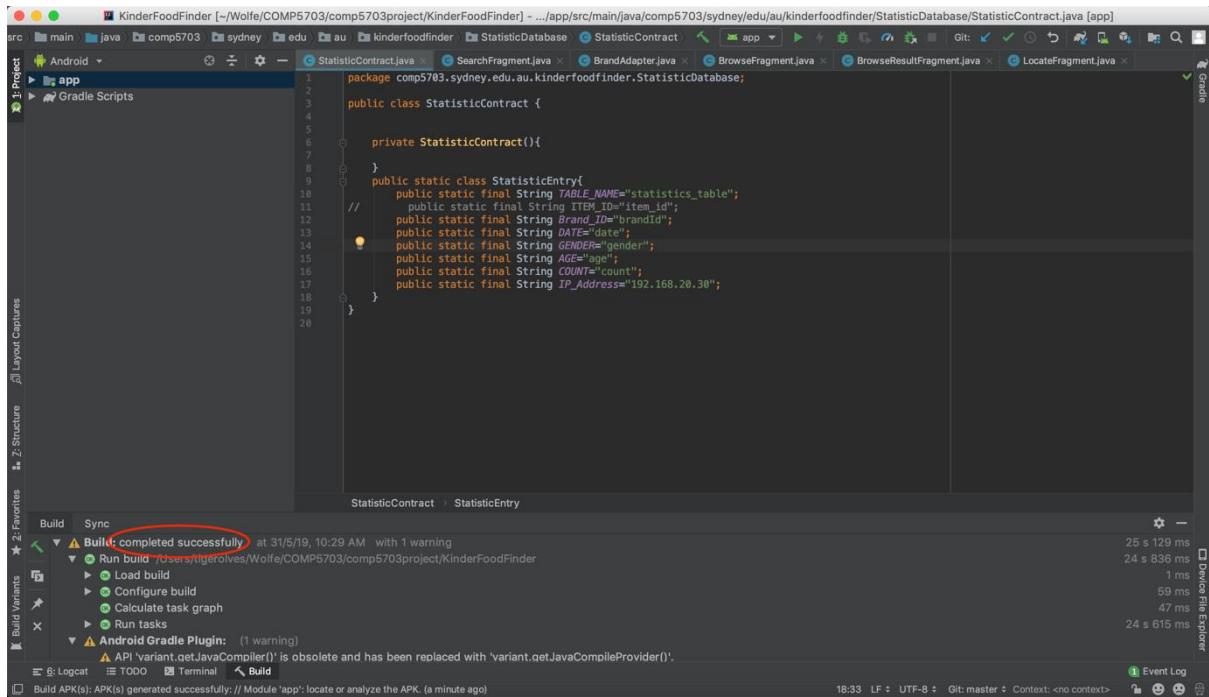


Figure 20. Build-up APK file successfully

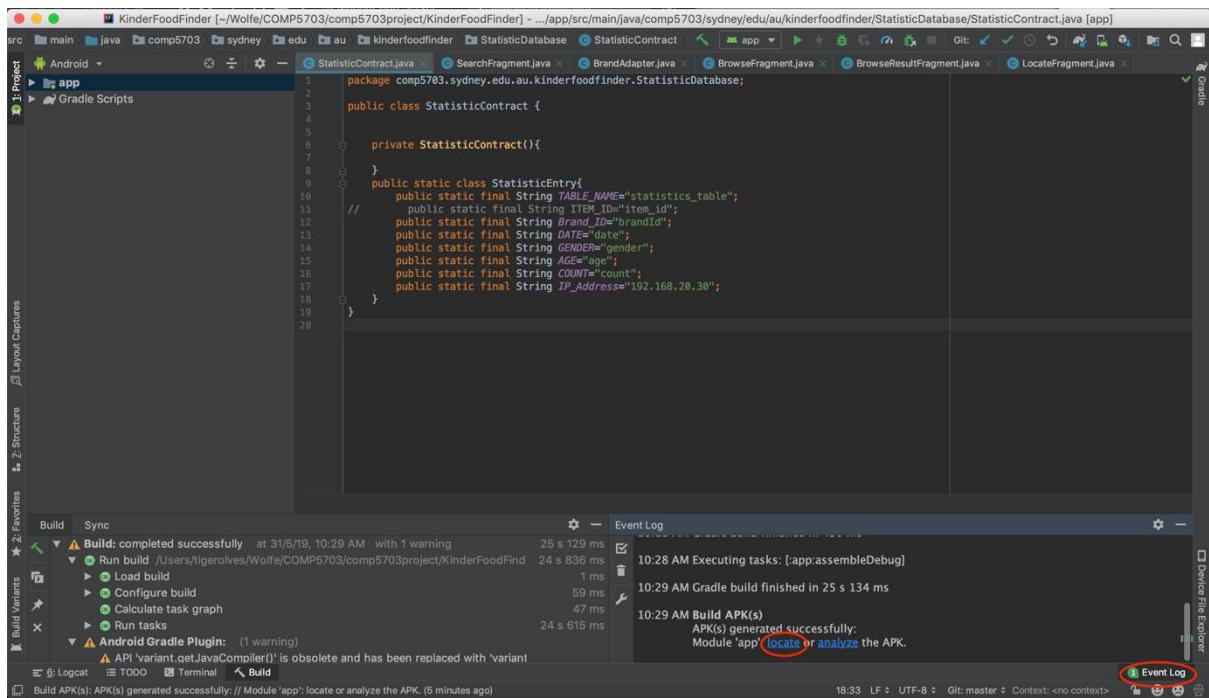


Figure 21. Event Log to see the location of APK file

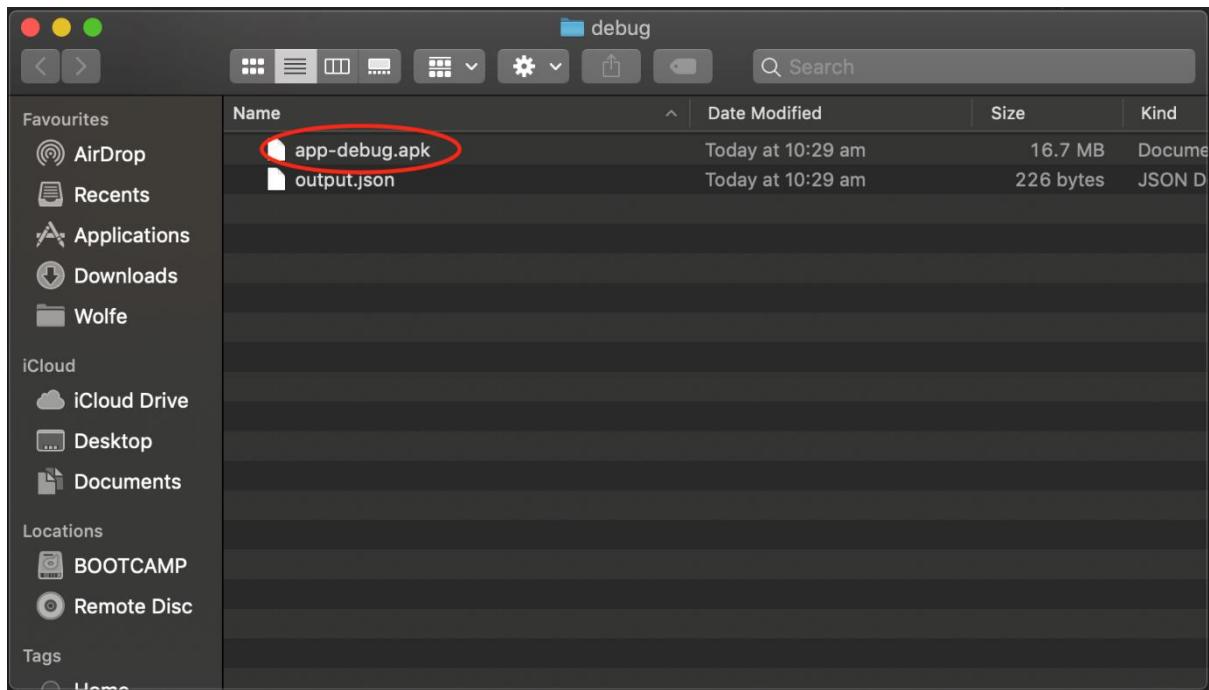


Figure 22. APK file

### **3.8. Steps to Find Out Your IP Address in Your Laptop/Desktop Which Will Run Server**

- Open the **terminal** in your Mac/MacBook
- Type the command **ipconfig getifaddr en1** and enter
- It will show the **IP address** such as 100.101.86.121

### **3.9. Steps to Run Our Application and Install It in Your Android Device/Virtual Device**

- Open **Android Studio** in your laptop/desktop after you installed it
- Select “**Import project (Gradle, Eclipse ADT, etc.)**” (Refer to Figure 12)
- Choose the folder called “**KinderFoodFinder**” sent to you
- Waiting for it building up and synced successfully (Ignore the warning at the bottom-right side) (Refer to Figure 13)
- After building up successfully, you will see **all the related folders and files** included in the folder “**KinderFoodFinder**” sent to you on the left-hand side (Refer to Figure 23)
- Go through the folders: **app -> java -> comp5703.sydney.edu.au -> kinderfoodfinder -> StatisticDatabase** and double click on the file called “**StatisticContract.java**” (Refer to Figure 24)
- You will see the code on the right-hand side and in the last line, there is a variable called “**IP\_Address**” (Refer to Figure 25)
- **Change the value to your IP address** which is found from the tutorial above that teaches you how to find IP address in your computer (Refer to Figure 26)
- Now connect your android device to your computer and the device will pop-up a window to ask you if enable USB debugging or not, then click on “Yes”
- Back to Android Studio, click on the arrow icon with green colour called “Run app” at the top bar (Refer to Figure 27)
- In the pop-up window, you should see your Android device appearing under **Connected Devices** (Refer to Figure 28)
- (Optional) If you don’t have any Android device and choose to use virtual device which is setup from the tutorial above, then you will see the virtual device you created appearing under **Available Virtual Devices** (Refer to Figure 29)
- Select the Android device/Virtual device and click on “OK”

- After setup and installing, the application will be opened automatically in the device

**Note:** Make sure your laptop/desktop running the server and Android device are in the same WIFI network since now they are running locally

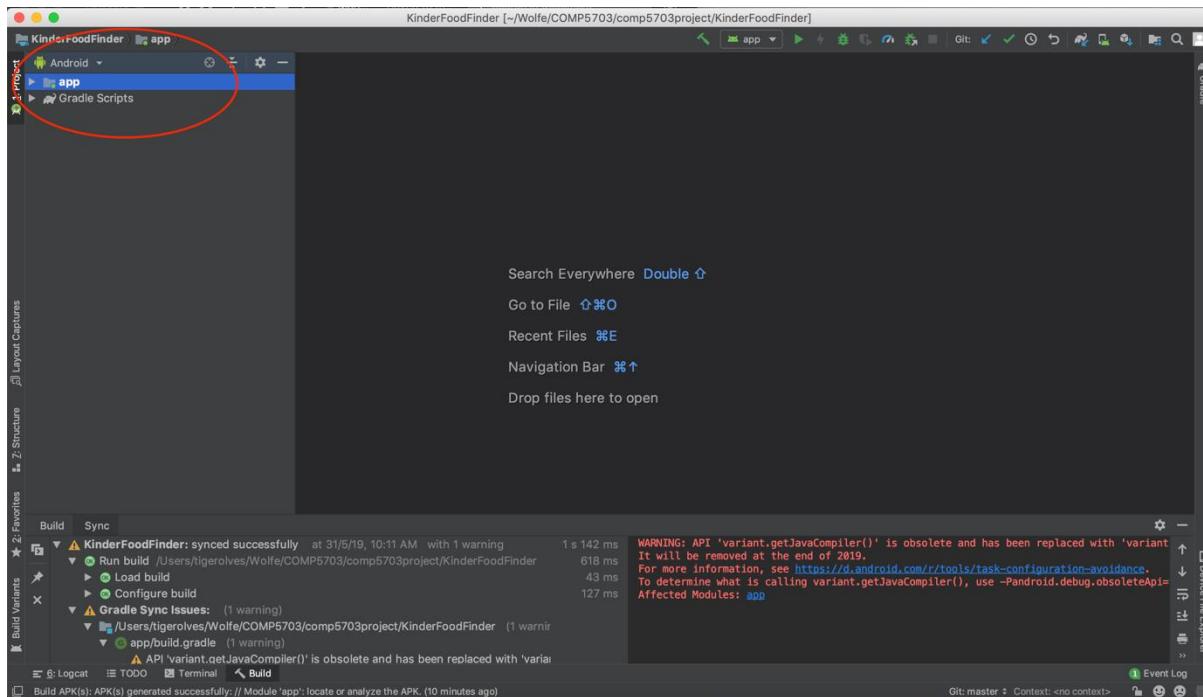


Figure 23. All related folders and files

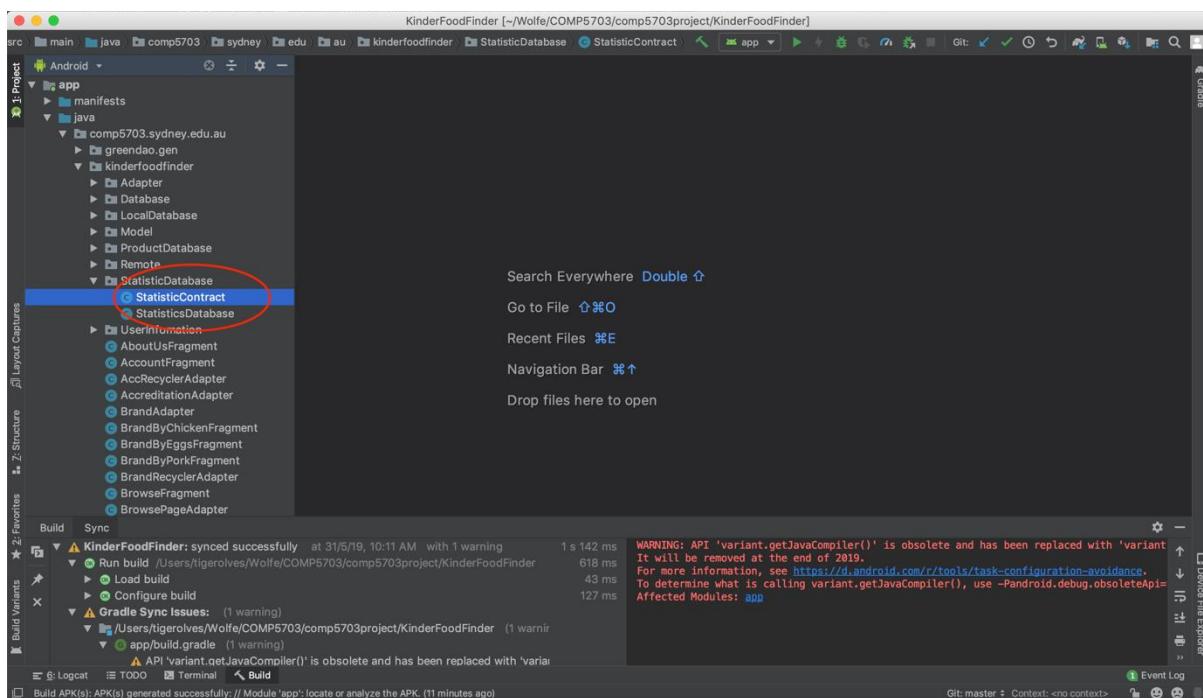


Figure 24. Go to StatisticContract class

The screenshot shows the Android Studio interface with the project 'KinderFoodFinder' open. The code editor displays `StatisticContract.java` from the `StatisticDatabase` package. A red circle highlights the static final string `IP_Address` at line 17, which contains the value `"192.168.20.30"`. The code defines a `StatisticContract` class with a constructor and a static inner class `StatisticEntry` containing various constants like `TABLE_NAME`, `ITEM_ID`, `Brand_Id`, `DATE`, `GENDER`, `AGE`, `COUNT`, and `IP_Address`.

```
package comp5703.sydney.edu.au.kinderfoodfinder.StatisticDatabase;
public class StatisticContract {
    private StatisticContract(){}
    public static class StatisticEntry{
        public static final String TABLE_NAME="statistics_table";
        public static final String ITEM_ID="item_id";
        public static final String Brand_Id="brandId";
        public static final String DATE="date";
        public static final String GENDER="gender";
        public static final String AGE="age";
        public static final String COUNT="count";
        public static final String IP_Address="192.168.20.30";
    }
}
```

Figure 25. Variable of IP Address

This screenshot is identical to Figure 25, showing the same code in `StatisticContract.java`. However, a tooltip appears over the circled IP address value `"192.168.20.30"`, reading "Change to your IP Address". This indicates that the developer has interacted with the highlighted code.

Figure 26. Change IP Address to yours

StatisticContract.java

```

1 package comp5703.sydney.edu.au.kinderfoodfinder.StatisticDatabase;
2
3 public class StatisticContract {
4
5     private StatisticContract(){
6
7     }
8
9     public static class StatisticEntry{
10         public static final String TABLE_NAME="statistics_table";
11         public static final String ITEM_ID="item_id";
12         public static final String Brand_ID="brand_id";
13         public static final String DATE="date";
14         public static final String GENDER="gender";
15         public static final String AGE="age";
16         public static final String COUNT="count";
17         public static final String IP_Address="192.168.20.30";
18     }
19
20 }

```

Build Variants

Build Sync

- KinderFoodFinder:** synced successfully at 31/5/19, 10:11 AM with 1 warning
- Run build /Users/tigerolives/Wolfe/COMP5703/comp5703project/KinderFoodFinder 1 s 142 ms
- Load build 618 ms
- Configure build 43 ms
- Gradle Sync Issues: (1 warning)
  - /Users/tigerolives/Wolfe/COMP5703/comp5703project/KinderFoodFinder (1 warning)
    - API 'variant.getJavaCompiler()' is obsolete and has been replaced with 'variant.getJavaCompiler()'. It will be removed at the end of 2019. For more information, see <https://d.android.com/r/tools/task-configuration-avoidance>. To determine what is calling variant.getJavaCompiler(), use -Pandroid.debug.obsoleteApis=Affected Modules: app

Event Log

Build APK(s): APK(s) generated successfully; // Module 'app': locate or analyze the APK. (17 minutes ago)

Figure 27. Run the code

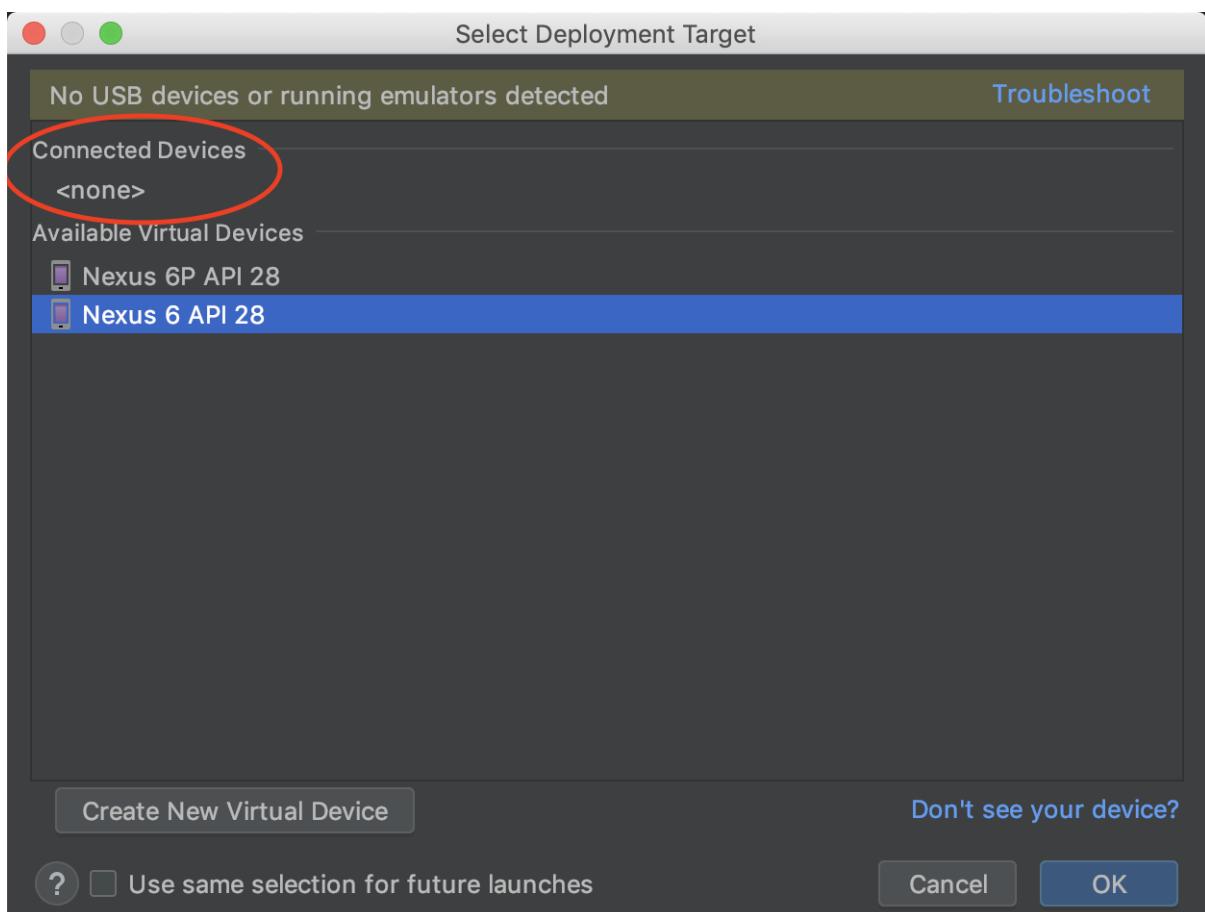


Figure 28. Select connected device to run the code

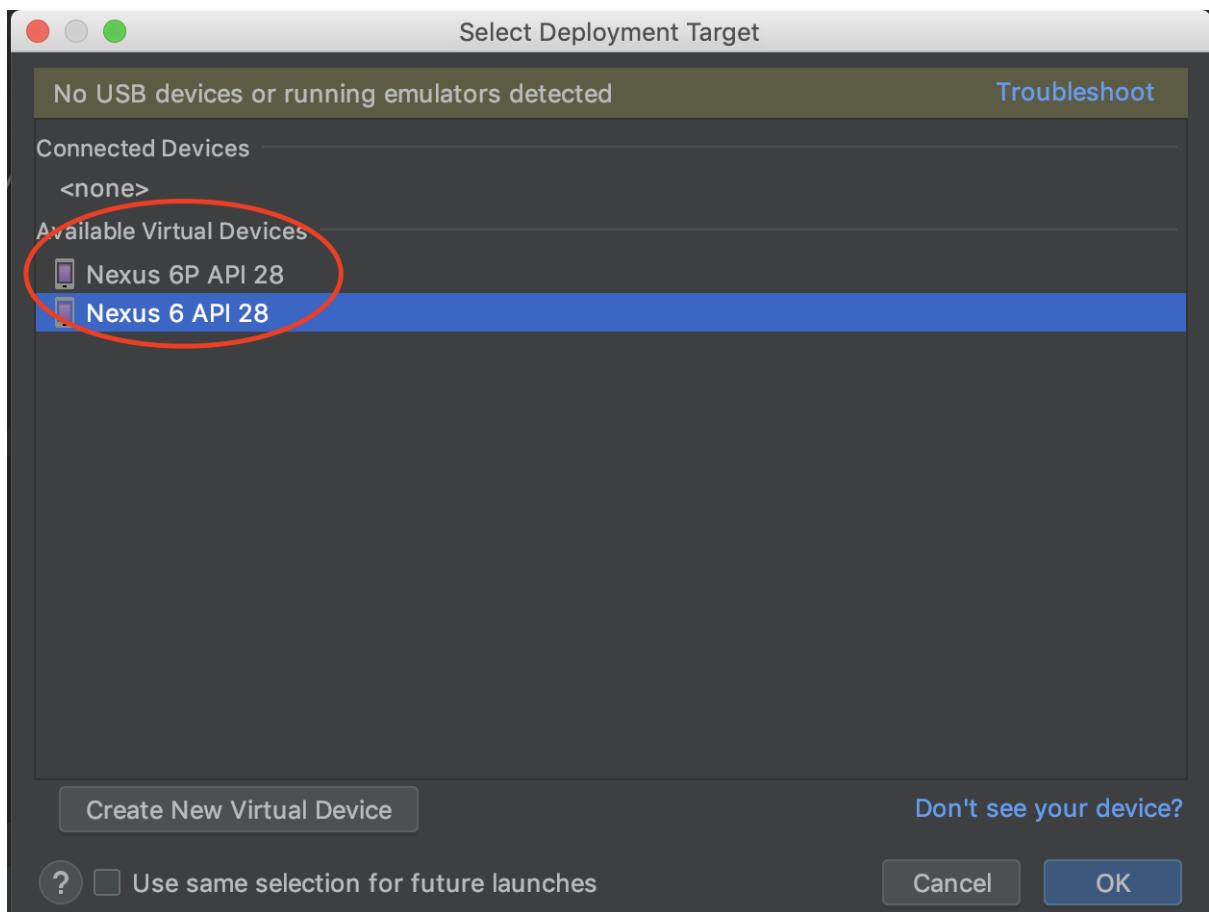


Figure 29. Select virtual device created to run the code

## B. Guidance on operating the server-side

This part of the manual documents how the administrator (Admin) shall use and manage the server-side of the system. Firstly, we want to talk about the architecture of the app using figure 30, which illustrates perfectly the architecture of the Kinder Food Finder system. Firstly, consumers who goes shopping for animal-based products can access information about ethical brands, find location, and inform others where to find a particular good using the mobile application; In turn, depends on the commands given by the end-users, the mobile app will choose to display information or interact with the web server through HTTP connection. Likewise, the server side of the Kinder Food Finder system was developed in the format of a Web application; In short, by accessing to the server using browser and HTTP URL, a web page will be displayed to the administrator and grants him/ or her full control over the data in the database (Which directly impacts the data that the server sends to the Kinder Food Finder app).

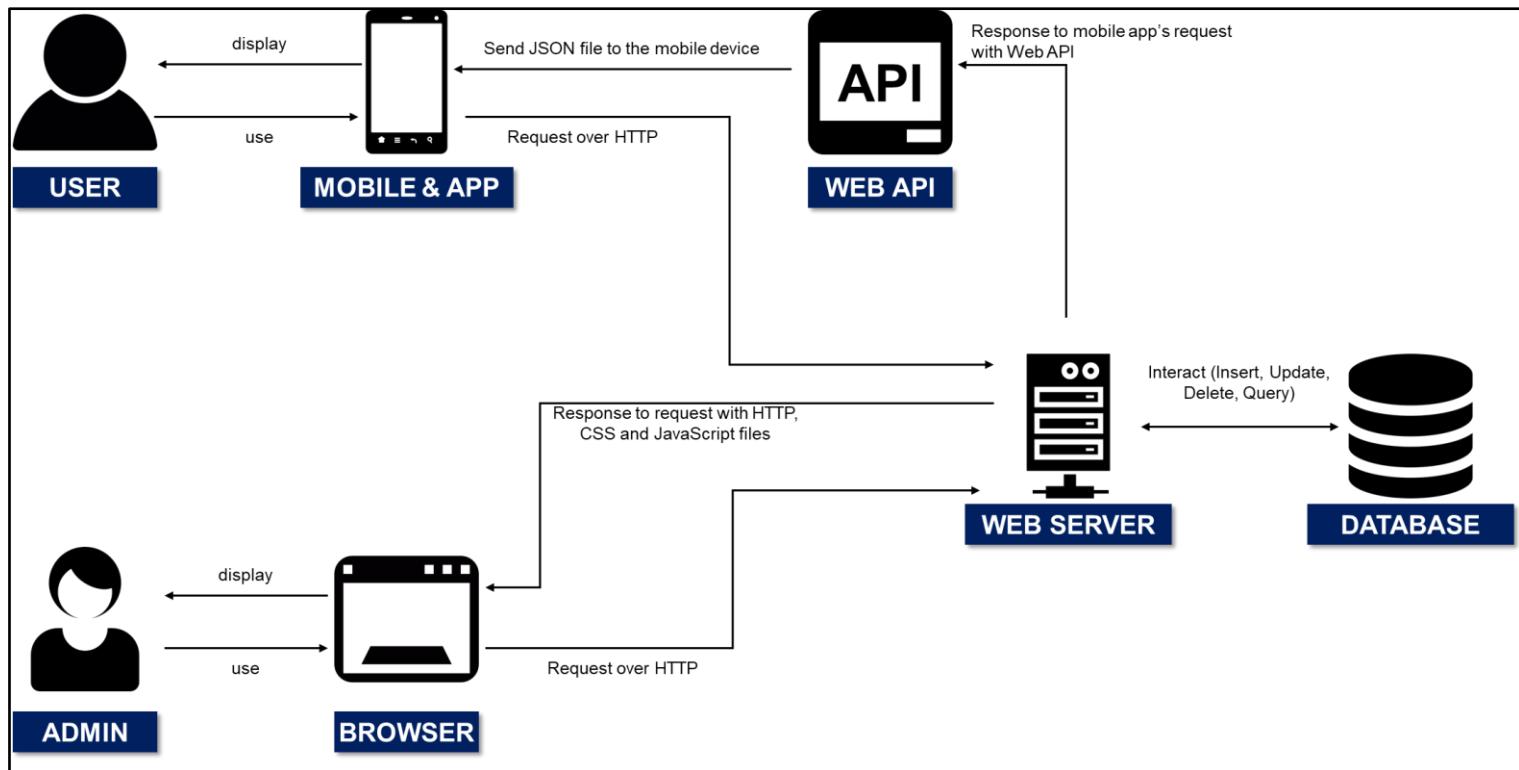


Figure 30. Architecture of the Kinder Food Finder system

In short, the server-side will **control the information displayed** in the mobile application. Hence, the next section will discuss in detail on how you should operate the server, including login, insert new data, update and delete data, and generate statistical data (demographic data of end-users: How many times a brand is searched, filtered by age, timeline and gender).

## 1. Login to the Server-side

Firstly, you should open the default web browser in your PC and type the following URL address: <http://localhost:3000>. The web browser will automatically display the web-page server landing page (Refer to figure 31). Hence, enter the following **default** username and password to access the server: Username – **admin**, Password - **admin**; Please be advised that, the username and password can be changed later. Upon login into the server, you will see a page identical to figure 32.

## Kinder Food Finder - Server Site

Login

Forgot Password



Figure 31. Landing page of the web-page server

**KINDER FOOD FINDER**

Search a brand name...

RESET ACCOUNT

LOGOUT

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

**Statistics**

**BRAND STATISTICS**

Guideline: This function enables the admin to export a csv file contains statistical data collected from the end-users

Timeline (\*)

All

Period

Choose brand

Gender

All ▾

**10 MOST SEARCHED BRANDS**

1. Adelaide Eggs Cage - total searches: 8
2. Grill'd - total searches: 2
3. Budget Mixed Grade Cage eggs - total searches: 0
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.
- 10.

Figure 32. Home page/ statistics page

To help the administrator access to every feature on the server conveniently, the layout is built with a control panel tab of the left side and a navigation bar on the top; Hence, by clicking on the different buttons, you will be able to access other functions (For instance: Import/ Insert data, report, update/ delete brand):

- Report: This feature allows the admin to view report on where to find different brands from the end-users. In this view, the admin can choose to delete or add the report to the database so other users will know where to find a particular brand.
- Import/ Insert data: This feature allows the admin to import brand data (Brand name, accreditation and rating) from a csv files, or insert a brand/ or store using the input box provided on the web-page server.
- Update/ Delete Brand data: This feature enables the admin to delete a brand from the database. Furthermore, the admin can also access the “Brand detail page” from this view to update a particular brand (Add image, change name, add more accreditation and rating).
- Update/ Delete Store data: This feature enables the admin to delete a store from the database. Furthermore, the admin can also access the “store detail page” from

this view to update a particular store (Add address, change name, add or remove product from the store).

- Publish: Publish the new version of the database (Brand database and store database to the mobile application)
- Reset account: This feature allows the admin to change the account's username and password after login.

## 2. Import/ Insert data

As mentioned above, this function allows the admin to import data from a csv file or insert one brand/ store using the input boxes provided on the web-page server. Upon going to this view, you will see a drop-down list that asks for the type of data to insert (Refer to figure 33).

The screenshot shows the Kinder Food Finder Control Panel. At the top, there is a navigation bar with 'KINDER FOOD FINDER' on the left, a search bar 'Search a brand name...', and 'RESET ACCOUNT' and 'LOGOUT' on the right. On the far left, a vertical sidebar titled 'CONTROL PANEL' lists 'Report', 'Import/ Insert Data', 'Update/ Delete Brand', 'Update/ Delete Store', and 'Publish'. The main content area is titled 'Import/ Insert Data' and contains a sub-section titled 'Which data do you want to import'. A dropdown menu is open, showing options: 'Select ▾', 'Select', 'Brand' (which is highlighted in blue), and 'Store'.

Figure 33. Import/ Insert Data

### 2.1. Import/ Insert brand data

Upon choosing to input brand data to the database, the "Import/ insert brand data" view will be displayed to you (Refer to figure 34). On the left of the view, the admin can click on the button "Choose File" to choose the **CSV** file that consists of data about multiple brands (Brand name, accreditation, rating); After that, the admin can click on the "Upload file" button and the server will automatically process that file to import data to the database. Likewise, on the right-side of the view, the admin can choose to insert one brand to the database by filling multiple input boxes (This feature can insert only one data at a time, while the import csv file allows the admin to insert more than one data at a time).

KINDER FOOD FINDER
Search a brand name...
RESET ACCOUNT
LOGOUT

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

**Import/ Insert Brand Data**

**IMPORT CSV FILE**

Guideline: This function enables the admin to upload a csv file to the database which contains multiple brand data; Hence, the data imported will be recorded in the database.

No file chosen

**INSERT NEW BRAND**

Guideline: This function enables the admin to insert a new brand to the database. Likewise, only one accreditation should be recorded at this stage. If the admin wants to input more accreditation to a brand, please access the "Accreditation" tab in the "Detail Brand Page" (Update/ Delete Brand > Detail Brand > Accreditation).

**Enter a Brand Name (\*)**

**Choose an image for this brand**  
 No file chosen

**Accreditation (\*)**

**Category (\*)**

**Rating (\*)**

*Figure 34. Import/ Insert Brand Data*

## 2.2. Insert store data

Figure 35 displays the layout of the insert store data feature; Hence, to insert data, you should fill in all required input boxes on the right-side of the layout. Upon clicking the submit button, the server will process the address that you just insert and compare it with **real** address in the country; After processing the input data, a window will appear that enable you to choose from a list of address (Refer to figure 2). Please be advised that, the feature is designed this way to ensure that: (1) The address is accurate in context of the real world, (2) The server will automatically process the address to obtain the longitude and latitude (Geographical location) of the system, which will be used by the mobile app (Locate function). Lastly, due to the complexity and ad-hoc nature of the import csv function, we decided to remove the “Import store csv file” in the server-side.

## CONTROL PANEL

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

## Import/ Insert Store Data

**INSERT NEW STORE**

Guideline: This function enables the admin to insert a new store to the database. Likewise, if the admin wants to input the brands which this store have, please access the "Detail Store Page". Furthermore, one name can be used for many stores/ supermarkets (Woolworths, Aldi, .etc). Please access the "Detail Store Page" to add more address to a store.

## Store Name (\*)

Woolworth

## State/ Territory

NSW

## Postcode (\*)

2193

## Street Address (\*)

2A Charles Street, Canterbury

Submit

Reset

*Figure 35. Insert store data (1)*

## CONTROL PANEL

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

## Import/ Insert

## Add new store

Address: 2A-E Charles Street, Canterbury  
State: NSW  
Long: 151.1180258

Postcode: 2193  
Lat: -33.9125035

Address: 2 CHARLES ST, CANTERBURY  
State: NSW  
Long: 151.1176673

Postcode: 2193  
Lat: -33.9121823

**NEW STORE**

Admin to insert a new store to the to input the brands which this store page". Furthermore, one name can be Woolworths, Aldi, .etc). Please access address to a store.

Add

Close

Woolworth

## State/ Territory

NSW

## Postcode (\*)

2193

## Street Address (\*)

2A Charles Street, Canterbury

Submit

Reset

*Figure 36. Insert store data (2)*

### **3. Update/ delete brand data**

Upon going to the “Import/ Delete Brand data” feature, the server will display a table that contains all available brand in the database (Refer to figure 37). Hence, to delete a brand, you can check on the delete box of the brand you want to delete on the right-side. Please be noticed that, deleting a brand will also delete related connections that the brand involves in. For instance, in case the database currently records that “Adelaide Eggs Cage” can be found in “Woolworth – 2A Charles Street, Canterbury, 2193, NSW”; By deleting “Adelaide Eggs Cage”, the connection between that brand and “Woolworth – 2A Charles Street, Canterbury, 2193, NSW” will also be removed (This feature is designed this way to ensure that the database is consistent and connection between different factors is available only when those factors exist. Having a connection without the required factors will result in ambiguous data displayed to the administrator). Furthermore, in case you want to update a brand, click on the **brand name** and the server will automatically direct you to the “Brand detail page”.

The screenshot shows the 'Kinder Food Finder' application interface. On the left, there's a 'CONTROL PANEL' sidebar with links for Report, Import/ Insert Data, Update/ Delete Brand, Update/ Delete Store, and Publish. The main area has a dark header with 'KINDER FOOD FINDER', a search bar 'Search a brand name...', and buttons for 'RESET ACCOUNT' and 'LOGOUT'. Below the header is a section titled 'Update/ Delete Data' with a 'Brand' filter. It includes a 'Show 25' dropdown, a 'Select All' checkbox, and a trash bin icon. A search bar 'Search a brand...' is also present. The main content is a table with columns: 'Brand Name', 'Accreditation', 'Category', and 'Delete'. The table lists various egg brands with their accreditation status and category (all listed as 'egg'). The 'Delete' column contains a checkbox for each row, with the first row having its checkbox checked.

Brand Name	Accreditation	Category	Delete
Adelaide Eggs Cage	Cage, Caged - Avoid	egg	<input checked="" type="checkbox"/>
Budget Mixed Grade Cage eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Aussie Pride Multi Grain Eggs Cage Laid	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Black & Gold Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Country Classics Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Country Farmhouse Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Country House Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Days Eggs Country Fresh Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Farm Fresh Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Farm Pride Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Farmer Brown Fresh Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>
Flinders All Grain Cage Eggs	Cage, Caged - Avoid	egg	<input type="checkbox"/>

*Figure 37. Update/ Delete brand data*

### 3.1. Detail brand page – Brand summary

This feature enables you to update the (1) Name of a brand, (2) Category and (3) Image of the brand. Please be advised that the system only allows one image per brand due to the limitation in mobile application storage (The app will download the image and save it to the local database of the smart device; Hence, when user searches for a brand, the app will display that image to user. Likewise, an image file is heavy and having heavy sized files in a smart device is not the best practice in mobile computing).

The screenshot shows a mobile application interface for managing brands. At the top, there is a navigation bar with 'KINDER FOOD FINDER' on the left, a search bar 'Search a brand name...', and 'RESET ACCOUNT' and 'LOGOUT' on the right. On the far left, a vertical sidebar titled 'CONTROL PANEL' lists 'Report', 'Import/ Insert Data', 'Update/ Delete Brand', 'Update/ Delete Store', and 'Publish'. The main content area displays a brand named 'Adelaide Eggs Cage'. A large image of a carton of eggs is shown. Below the image, there are three data entries: 'Brand name' (Adelaide Eggs Cage), 'Category' (egg), and 'Image' (a placeholder button labeled 'Choose File' with the message 'No file chosen'). At the bottom, there are two statistics: '1' under 'Accreditations for this brand' and '3' under 'Stores have this brand'.

Figure 38. Detail brand page – Brand summary

### 3.2. Detail brand page – Accreditation

The Accreditation page displays accreditation and corresponding rating that a brand has. This feature enables the system to store and display **multiple accreditations** for a brand. Furthermore, the admin can delete or add more accreditations to a brand using this feature. To add new accreditation, you shall click on the “Plus” button (Button with the “+” sign) and a new window will pop up enables you to enter the new accreditation (Figure 39)

The screenshot shows the 'Detail Brand Page – Accreditation' for the brand 'Adelaide Eggs Cage'. The left sidebar has a 'CONTROL PANEL' with links: Report, Import/ Insert Data, Update/ Delete Brand, Update/ Delete Store, Publish. The main area shows 'Brand Summary' and 'Accreditation' sections. Under 'Accreditation', there is a search bar 'Please enter the Accreditation ...' and a table with two rows:

Accreditation	Rating
Cage, Caged	Avoid <input checked="" type="checkbox"/>
Accreditation	Rating

At the bottom are 'Previous', '1', and 'Next' buttons.

Figure 39. Detail Brand Page – Accreditation (1)

The screenshot shows the 'Detail Brand Page – Accreditation' for the brand 'Adelaide Eggs Cage'. A modal window titled 'Adding an Accreditation to Adelaide Eggs Cage' is open, containing a dropdown menu with options: Best, Best (highlighted in blue), Good, and Avoid. The main page background is dimmed. The table below the modal shows the same data as Figure 39.

Figure 40. Detail Brand Page – Accreditation (2)

### 3.3. Detail brand page – Store

This feature enables the administrator to identify which store and address is selling the brand and delete the connection between that brand and the store. However, this feature does not allow the user to create a connection between brand and store; to create the connection between these factors, please access the “Detail store page – Address”.

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

**Adelaide Eggs Cage**

**Brand Summary**

**Accreditation**

**Store**

Please enter the store name . . .

Select All

Store name	Address
Aldi	73-87 Jeffrey Street, Canterbury, 2193, NSW
Aldi	4 MAGDALENE TCE, WOLLI CREEK, 2205, NSW
Woolworth	2A-E Charles Street, Canterbury, 2193, NSW

Previous **1** Next

Figure 41. Detail Brand Page – Store

#### 4. Update/ delete store data

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

**Update/ Delete Data**

Store

Show 25  Select All  Search a store....

Store Name	Address	Number of Brand	Delete
Aldi	73-87 Jeffrey Street, Canterbury, 2193, NSW 4 MAGDALENE TCE, WOLLI CREEK, 2205, NSW	1	<input checked="" type="checkbox"/>
Woolworth	2A-E Charles Street, Canterbury, 2193, NSW	3	<input type="checkbox"/>

Showing 2 of 2

Previous **1** Next

Figure 42. Update/ Delete Store data

Similar to the Update/ Delete Brand data, this feature displays all the store available in the database, addresses and the number of brands that the store has. Furthermore, the feature also allows the admin to delete or update information of a store. Lastly, by clicking on the store name, you will go to the “Detail store page”. Please be advised that, deleting a store will remove any connections between that store and the brands which that store has.

#### 4.1. Detail store page – Store Summary

This feature enables the admin to change the name of the store and displays information about the number of addresses and brands that the store has.

The screenshot shows a web-based administration interface for a food store database. The top navigation bar includes links for 'KINDER FOOD FINDER', 'Search a brand name...', 'ABOUT THE SERVER', 'RESET ACCOUNT', and 'LOGOUT'. On the left, a 'CONTROL PANEL' sidebar lists options: Report, Import/ Insert Data, Update/ Delete Brand, Update/ Delete Store, and Publish. The main content area is titled 'Aldi' and contains a 'Store Summary' section. This section displays the store's name ('Aldi'), its address ('2 address(es) for this store'), and the number of brands ('1 brand(s) in this store'). There is also an 'Edit' link next to the store name.

Figure 43. Detail store page – Store Summary

#### 4.2. Detail store page – Address

This feature displays all the addresses that a store has; For instance, Woolworth will have several infrastructures all over the country. Furthermore, you can choose to delete or add new address to the database using the delete checkbox on the right-handed side or the “Plus” button on the top side of the layout (Figure 44). In addition, by clicking on an address, the server will direct the admin to the “Detail store page – brand”.

The screenshot shows the 'Address' section of the Aldi store detail page. It features a search bar with placeholder text 'Please enter the address ...' and a 'SEARCH' button. Below the search bar is a toolbar with 'Select All' and a plus sign (+) button. A table lists two addresses: '73-87 Jeffrey Street, Canterbury' and '4 MAGDALENE TCE, WOLLI CREEK'. Each row in the table includes columns for Address, State, PostCode, Lat, Long, Number of brands, and an 'Add Brand' button. At the bottom of the table, there are navigation buttons for 'Previous', '1', and 'Next'.

Figure 44. Detail store page – Address (1)

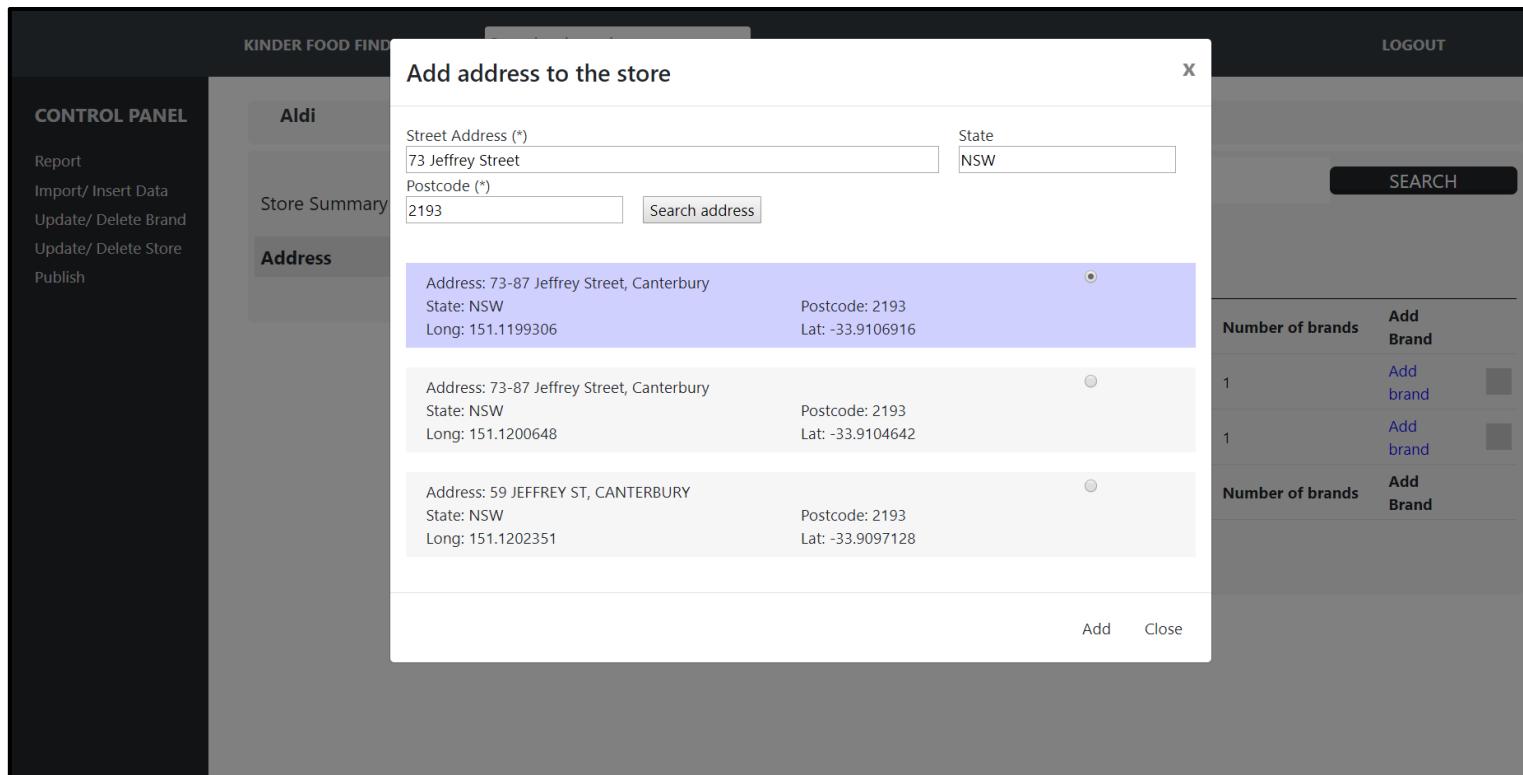


Figure 45. Detail store page – Address (2)

Likewise, you can also add connection between a brand and store address using the “Add brand” button (There are two approaches that the admin can use to create connection between brand and store address, this is one of the two approaches; We will mention another approach in the **Report** section). Upon clicking on the “Add brand button”, a new window will appear that displays all brands that the store address **does not have** (After you added a connection and reset the page, the brand you just added will be removed from this list).

KINDER FOOD FINDERS

LOGOUT

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

Aldi

Store Summary

**Address**

Add brand(s) to the store

Search for a brand...

Brand Name	Category	Add
Aussie Pride Multi Grain Eggs Cage Laid	egg	Add
Black & Gold Cage Eggs	egg	Add
Budget Mixed Grade Cage eggs	egg	Add
Country Classics Cage Eggs	egg	Add
Country Farmhouse Cage Eggs	egg	Add
Country House Cage Eggs	egg	Add
Days Eggs Country Fresh Cage Eggs	egg	Add
Farm Fresh Cage Eggs	egg	Add
Farm Pride Cage Eggs	egg	Add
Farmer Brown Fresh Cage Eggs	egg	Add
Flinders All Grain Cage Eggs	egg	Add
Flinders Ranges Cage Eggs	egg	Add
Foodland South Australian Farm Fresh Eggs Cage Eggs	egg	Add
Golden Egg Cage Eggs	egg	Add
Gourmet Breakfast Cage Free Eggs	egg	Add
Homebrand cage eggs	egg	Add
IGA Signature Cage Free Eggs	egg	Add
Jacobs Well Egg Farm Cage Eggs	egg	Add
Just 4 You Cage Eggs	egg	Add
Kalbarri Eggs Fresh Laid Cage Eggs	egg	Add
Pace Farm Cage Eggs	egg	Add
Swan Valley Cage Eggs	egg	Add
Tassie's Own Cage Eggs	egg	Add
The Good Egg Cage Eggs	egg	Add

Number of brands Add Brand

1	Add brand
1	Add brand

Number of brands Add Brand

Close

Figure 46. Detail store page – Address (3)

#### 4.3. Detail store page – brand

This layout displays all brands that a store address has. Furthermore, it also enables you to delete or add more connection between brands and store address to the database using the delete checkbox on the right-handed side and the “Plus” button on the top side of the layout.

KINDER FOOD FINDER

Search a brand name...

ABOUT THE SERVER

RESET ACCOUNT

LOGOUT

**CONTROL PANEL**

Report  
Import/ Insert Data  
Update/ Delete Brand  
Update/ Delete Store  
Publish

Aldi - 73-87 Jeffrey Street, Canterbury, 2193, NSW

Store Summary

Address

Brand

Please enter the Brand ...

Select All

Brand Name	Category
Adelaide Eggs Cage	egg
	Category

Previous 1 Next

SEARCH

Figure 47. Detail store page – Brand (1)

KINDER FOOD FINDER

LOGOUT

**CONTROL PANEL**

Report  
Import/ Insert Data  
Update/ Delete Brand  
Update/ Delete Store  
Publish

Aldi - 73-87

Store Summary

Address

Brand

Add brand(s) to the store

Search for a brand...

Brand Name	Category	Add
Budget Mixed Grade Cage eggs	egg	<input type="button" value="Add"/>
Aussie Pride Multi Grain Eggs Cage Laid	egg	<input type="button" value="Add"/>
Black & Gold Cage Eggs	egg	<input type="button" value="Add"/>
Country Classics Cage Eggs	egg	<input type="button" value="Add"/>
Country Farmhouse Cage Eggs	egg	<input type="button" value="Add"/>
Country House Cage Eggs	egg	<input type="button" value="Add"/>
Days Eggs Country Fresh Cage Eggs	egg	<input type="button" value="Add"/>
Farm Fresh Cage Eggs	egg	<input type="button" value="Add"/>
Farm Pride Cage Eggs	egg	<input type="button" value="Add"/>
Farmers Brown Fresh Cage Eggs	egg	<input type="button" value="Add"/>

Close

SEARCH

Figure 48. Detail store page – Brand (2)

## 5. Report

This feature records reports from mobile app users about which store address has a brand. For instance, when a mobile app user report where they find “Adelaide Eggs Cage”, his/ or her report will be sent to the server and recorded in this feature; The administrator can choose to delete the report or approve it and add the information to the database (Be noticed that this is the second feature that helps create connection between a brand and store address). Hence, by clicking on the “Add report” button, a new window will appear that lets you choose the correct address from a list of addresses (Similar to the insert store data).

The screenshot shows a web-based application interface for managing reports. At the top, there's a navigation bar with 'KINDER FOOD FINDER', 'Search a brand name...', 'RESET ACCOUNT', and 'LOGOUT'. On the left, a 'CONTROL PANEL' sidebar lists options: Report, Import/ Insert Data, Update/ Delete Brand, Update/ Delete Store, and Publish. The main content area is titled 'Reported stores and brands from users (Waiting for approval from the admin)'. It contains a table with one row of data:

Brand Name	Store Name	Address	Email	Delete	Add report
Adelaide Eggs Cage	Woolworth	2A Charles Street, 2193, NSW	long.lythien@gmail.com	<input type="checkbox"/>	<a href="#">Add report</a>

Below the table, it says 'Showing 1 of 1'. At the bottom right, there are 'Previous' and 'Next' buttons, with '1' in the middle indicating the current page.

Figure 49. Report page (1)

The screenshot shows a modal dialog box titled 'Add address to the store'. It contains two address entries:

Store Name (*) Woolworth Street Address (*) 2A Charles Street	Postcode (*) 2193 State: NSW
Address: 2A-E Charles Street, Canterbury State: NSW Long: 151.1180258	Postcode: 2193 Lat: -33.9125035
Address: 2 CHARLES ST, CANTERBURY State: NSW Long: 151.1176673	Postcode: 2193 Lat: -33.9121823

At the bottom of the dialog, there are 'Add' and 'Close' buttons.

Figure 50. Report page (2)

## 6. Publish

Upon adding, deleting, updating data in the database, to ensure that the mobile application could receive the most updated version of the database, the admin should use this feature to publish information.

The screenshot shows the Kinder Food Finder Control Panel. At the top, there is a navigation bar with 'KINDER FOOD FINDER' on the left, a search bar 'Search a brand name...', and 'RESET ACCOUNT' and 'LOGOUT' on the right. Below the navigation bar is a sidebar titled 'CONTROL PANEL' containing links: Report, Import/ Insert Data, Update/ Delete Brand, Update/ Delete Store, and Publish. The main content area is titled 'Publish/ Release Brand Data' and contains a section titled 'PUBLISH NEW DATA'. It includes a guideline: 'Guideline: This function enables the admin to publish the new version of data to the app; Thus, the data will be updated while the app detects there's newer version on server.' Below this, there is a 'Brand:' dropdown set to 'Publish' and a 'Version:' input field set to '3'.

Figure 51. Publish page

## 7. Statistics

This feature enables the admin to generate a CSV file that contains statistical data collected from mobile application users (Number of times a brand is searched by gender, age and timeline). In specific, by filling all the required input boxes on the left-handed side of the layout and click “Generate csv file” button, the browser will automatically download a statistical file (Refer to figure 52). In specific, the Timeline section enables the admin to choose the timeline of the data that will be generated, the admin can choose to generate data from all the time using the “All” button or data from a particular time range by defining the start date and end date of the timeline; Likewise, the Brand section allows you to choose to generate statistical data of all the brands available or just one specific brand in the database. Lastly, the gender enables you to choose which gender the statistics file should have.

KINDER FOOD FINDER      Search a brand name...      RESET ACCOUNT      LOGOUT

**CONTROL PANEL**

- Report
- Import/ Insert Data
- Update/ Delete Brand
- Update/ Delete Store
- Publish

**Statistics**

### BRAND STATISTICS

Guideline: This function enables the admin to export a csv file contains statistical data collected from the end-users

**Timeline (\*)**

All  
 Period  -

**Choose brand**

**Gender**

**10 MOST SEARCHED BRANDS**

1. Adelaide Eggs Cage - total searches: 8
2. Grill'd - total searches: 2
3. Budget Mixed Grade Cage eggs - total searches: 0
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.
- 10.

Figure 53. Statistics page

AutoSave Off      StatisticalReport (2) - Excel

File Home Insert Page Layout Formulas Data Review View Help Team      Tell me what you want to do

Cut Copy Format Painter      Font Alignment Number Styles

F10      X ✓ fx

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Brand	Gender	Age	Timeline	Count								
2	Grill'd	Male	Not Disclose	2019-05-16	2								
3	Adelaide Eggs Cage	Female	Not Disclose	2019-03-15	7								
4	Adelaide Eggs Cage		Not Disclose	2019-05-17	1								
5	Budget Mixed Grade Cage eggs		Not Disclose	2019-05-24	1								
6	Adelaide Eggs Cage	Male	18-29	2019-05-28	1								
7													
8													
9													
10													
11													
12													
13													
14													
15													

Figure 54. Statistics file