Assignment 2

instructions and Grading Criteria

- This is an **individual** assessment.
- Assignments must be completed using the approaches and techniques demonstrated in class.
- The majority of grades are assigned based on the correct completion of the required functionality.
- The user interface of your application must be reasonably polished, easy to understand, and readable. Use reasonably pleasant colors and typography
- In addition to the required functionality, learners are expected to use the coding conventions demonstrated in class, meaningful variable naming, and clearly organized code. Comments are helpful but not required.

Submission Checklist

For your submission to be graded, you must provide

1. a **zip** file of your project, and
2. a **screen recording** demonstrating the functionality you implemented.

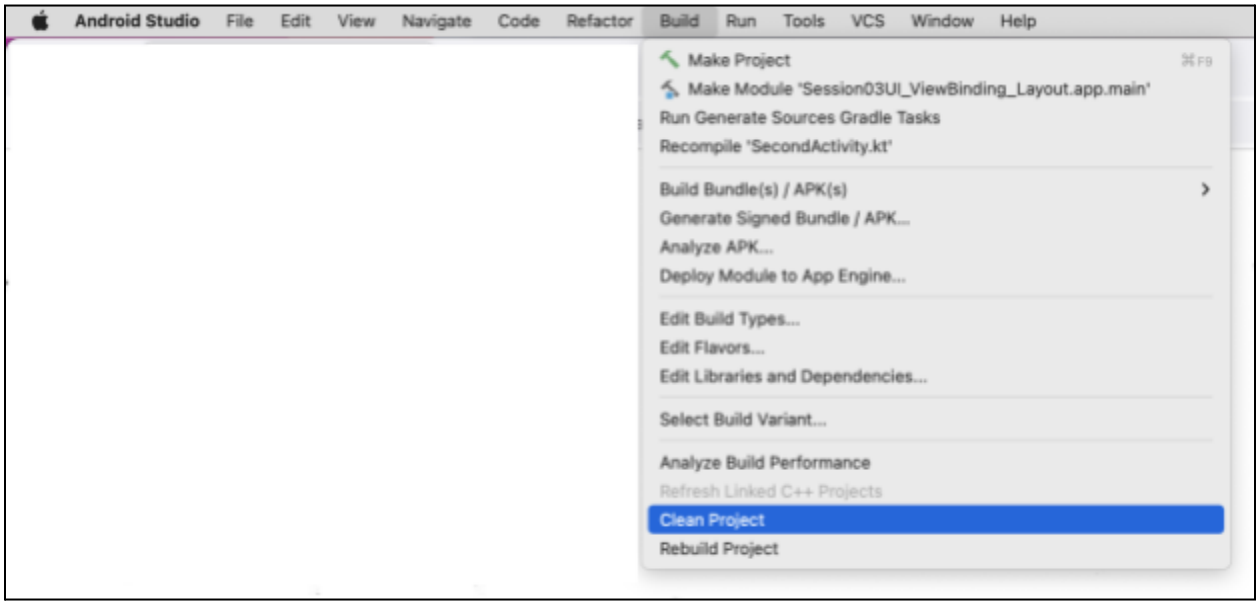
How to create your project:

1. Create an ANDROID project called: **PARKING_FIRSTNAME**. Replace FIRSTNAME with your first name.

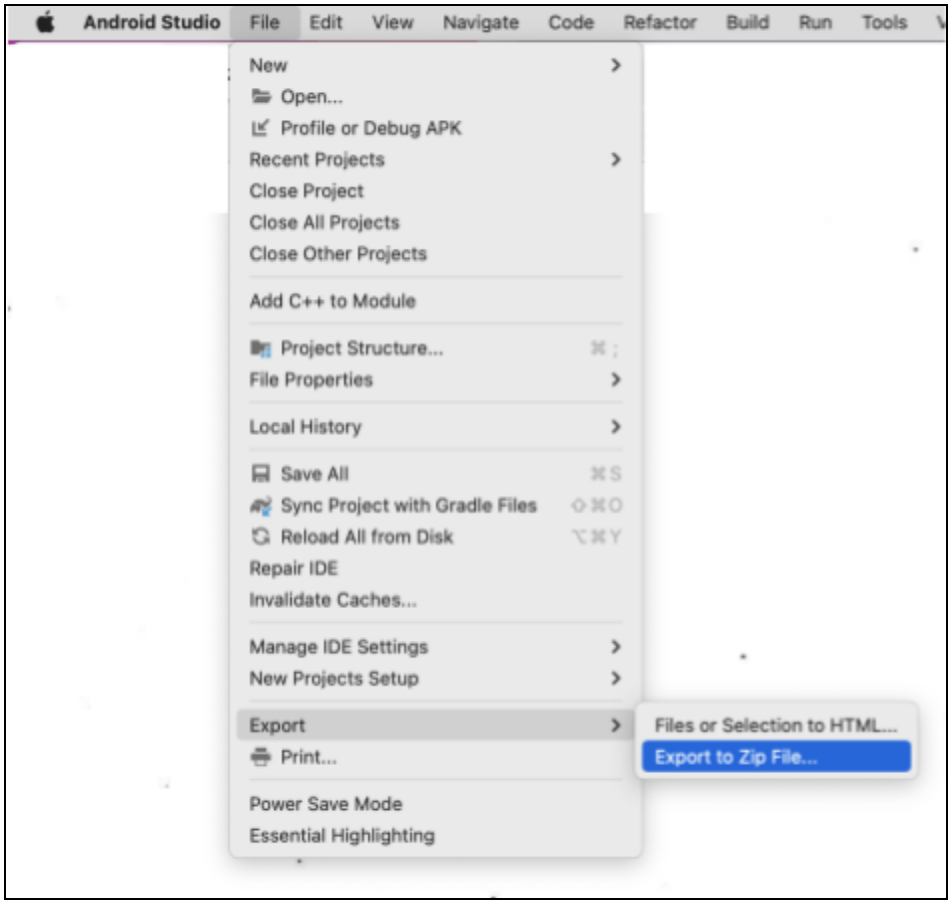
2. In the project, create and test your solution

3. When you are ready to submit, run Build > Clean. After running Build > Clean, DO NOT RERUN YOUR PROJECT!!!

- Build > Clean remove unnecessary configuration files from your project and make your final project smaller in size.



4. Choose File > Export to zip file



5. Ensure the zip file is named **PARKING_FIRSTNAME**, where you replace FIRSTNAME with your first name.

6. Submit the zip file to the dropbox.

Academic Integrity

- This is an individual assessment.
- Permitted activities: Usage of Internet to search for syntax only; usage of course materials
- Not permitted:
  - Communication with others (both inside and outside the class)
  - Discussion of solution or approaches with others; sharing/using a "reference" from someone
  - Searching the internet for full or partial solutions
  - Sharing of resources, including links, computers, accounts

Problem Description

Using Android, build a 2 screen application that lets users pay for parking.

Screen 1: Pay for Parking Screen

This screen must display:

1. A form for entering parking information
2. A options menu with menu item that displays link to Screen #2

Entering Parking Information Form

Show a form with the following fields:

- Radio group for selecting a parking lot. Display two parking lots and their hourly rates. Each parking lot must have a different hourly rate. The user can select only 1 parking lot.
- Textbox for entering the number of hours to park
- Textbox for entering license plate
- PAY NOW button

When the PAY NOW button is pressed, calculate the cost of parking and output a receipt to the screen. After displaying the receipt, clear all form fields and prepare for new input.

- The receipt should show license plate, hours purchased, and total paid.

- For example, assume the user wants to park in Parking Lot A for 3 hours. Parking Lot A costs $2.50 per hour. When the PAY NOW button is pressed, the receipt should show:

```
RECEIPT
Parking Lot A
License Plate: AMHG 100
Hours: 3
Total Paid: $7.50
```

Save the user's purchase to a *list of strings*. For the above example, the string should be:
"`AMHG 100; Parking Lot A; 3 hours; $7.50`"

Options Menu

Screen 1 should also display an options menu. The options menu must have 1 menu link. When the link is clicked, navigate the user to Screen 2. Send the list of purchases to Screen 2.

Screen 2: History of Receipts

Screen #2 should receive the list of purchases from Screen #1.

In a recycler view, display the list of purchases

HINT: Sending and receiving a list of strings via Intents

1/ Assume you have a list of strings:

```
private var todosList:MutableList<String>
       = mutableListOf("Walk the dog", "Do homework", "Eat dinner")
```

2/ Convert the list of strings to an *array*, then send the array to via an intent:

```
intent.putExtra("TODOS_EXTRA", todosList.toTypedArray())
```

3/ To receive the list of strings, get the array from the intent and attempt to convert the array back to a list.

```
// get the array from the intent
val receivedData = intent.getStringArrayListExtra("TODOS_EXTRA")
// attempt to convert the array back to a list
var myList:MutableList<String> = mutableListOf()
// if data from intent contains an array, then:
if (receivedData != null) {
   // convert the array back to a mutable list.
   myList = receivedData.toMutableList()
}
```

END OF ASSESSMENT