Student Name: Aldalton Choo Chien Khin
Student ID: 101212783

Week 4 Assignment 4 Report

Task 1 – Latitude and Longitude Coordinate Unit Tests

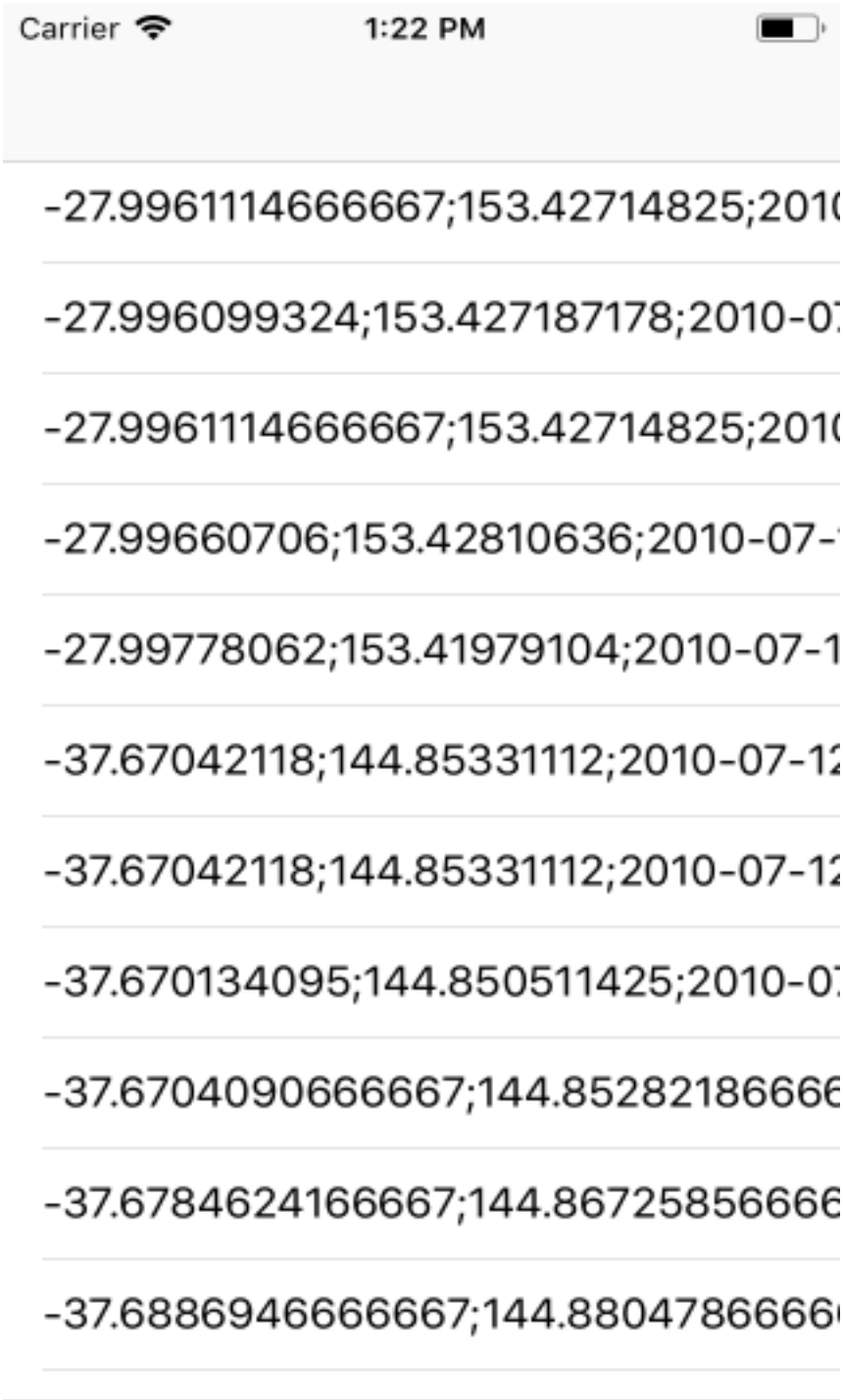


Figure (1) Successfully loaded coordinates from csv files into app

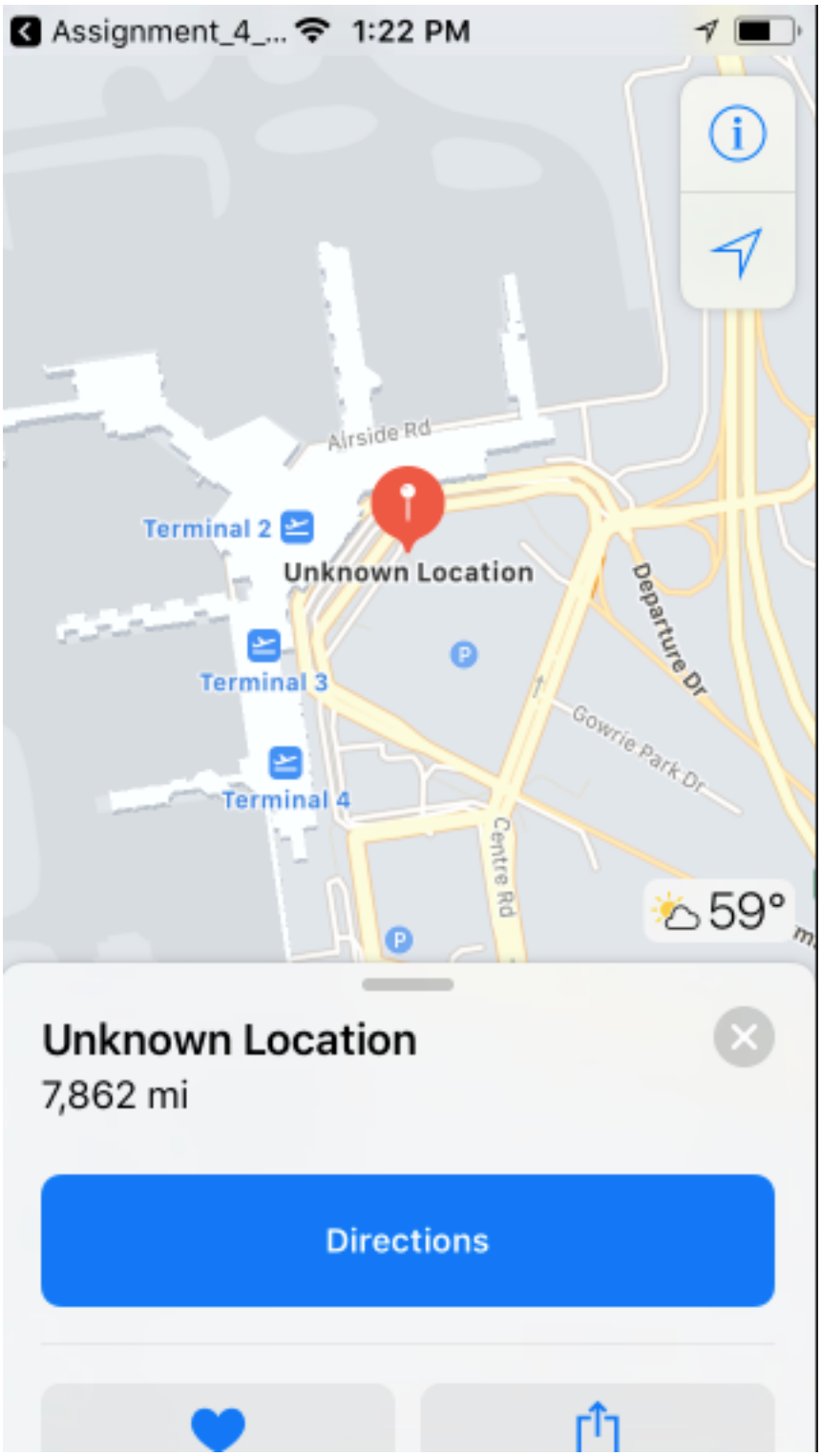


Figure (2) When one of the coordinates is clicked it will be loaded into maps app and taken to the location automatically.

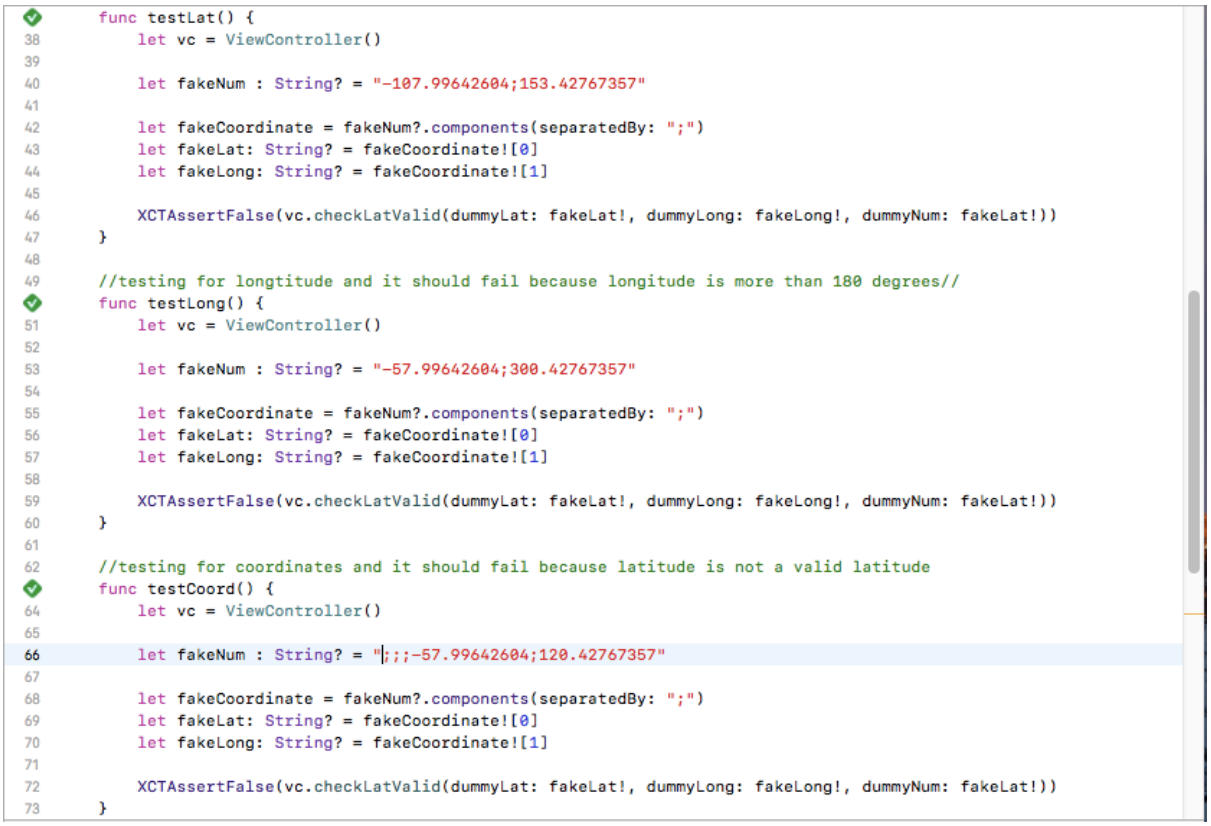
```
30      func externalCoords(fileName: String) -> [String]{
31
32          if let coords = Bundle.main.path(forResource: "gps_coords", ofType: "csv"){
33
34              do{
35                  let coordinates = try String(contentsOfFile: coords)
36                  coorArray = coordinates.components(separatedBy: "\n")
37                  return coorArray
38              }
39              catch let error as NSError{
40                  print("File not found:\(error)")
41              }
42          }
43          return coorArray
44      }
45
46      func tableView(_ tableView:UITableView, numberOfRowsInSection section: Int) -> Int{
47          let numOfCoords = externalCoords(fileName: "gps_coords")
48          return numOfCoords.count
49      }
50
51      func tableView( _ tableView: UITableView, cellForRowAt indexPath: IndexPath) -> UITableViewCell{
52          var gpsCoords = externalCoords(fileName: "gps_coords")
53          let cell = UITableViewCell(style:UITableViewCellStyle.default, reuseIdentifier:"cell")
54          cell.textLabel?.text = gpsCoords[indexPath.row]
55          return cell
56      }
57
58      func tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath) {
59          //self.performSegue(withIdentifier: "mapTransfer", sender: self)
60          print(coorArray[indexPath.row])
61          performSegue(withIdentifier: "mapTransfer", sender: coorArray[indexPath.item])
62          let newCoords = coordinates(string: coorArray[indexPath.row])
63          latitude = Double(newCoords[0])
64          longitude = Double(newCoords[1])
65          time = newCoords[2]
66      }
```

Figure (3) These are the functions created to load coordinates.csv file into the tableview in the app and using the segue method to link the coordinates and the map kit.

```
74      func checkLatValid (dummyLat:String, dummyLong:String, dummyNum: String) -> Bool
75      {
76          let Num = Double (dummyNum)
77          let Lat = Double (dummyLat)
78          let Long = Double (dummyLong)
79
80          if (Num != nil){
81              if (Lat! < Double(-90.0) || Lat! > Double(90.0)){
82                  return false
83              }
84              else if (Long! < (-180.0) || Long! > Double(180.0)){
85                  return false
86              }
87              else{
88                  return true
89              }
90          }
91          else{
92              return false
93          }
94
95      }
96
```

Figure (4) Test conditions to be used in test cases later on.

checkLatValid is created to set a certain condition to be used in the test cases later on to check if the app is working as intended. The variable Num is created to make sure that the coordinates entered are in numeral form and not in other forms else it will return false. Lat variable is created to make sure the latitude entered is more than -90 degrees and less than 90 degrees so that the latitude can be verified as a valid latitude else it will return false. Long variable is created to make sure the longitude entered is more than -180 degrees and less than 180 degrees so that the longitude can be a verified as a valid longitude else it will return false.



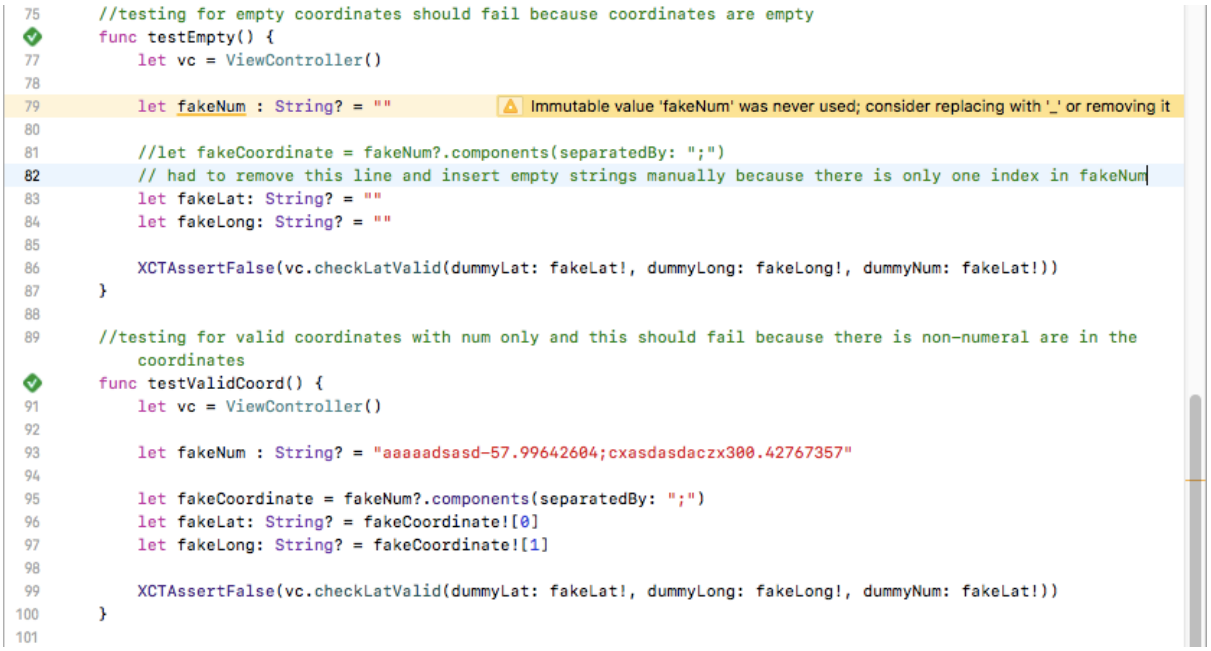
```
func testLat() {
    let vc = ViewController()

    let fakeNum : String? = "-107.99642604;153.42767357"

    let fakeCoordinate = fakeNum?.components(separatedBy: ";")
    let fakeLat: String? = fakeCoordinate![0]
    let fakeLong: String? = fakeCoordinate![1]

    XCTAssertFalse(vc.checkLatValid(dummyLat: fakeLat!, dummyLong: fakeLong!, dummyNum: fakeLat!))
}

//testing for longtitude and it should fail because longitude is more than 180 degrees//
func testLong() {
    let vc = ViewController()

    let fakeNum : String? = "-57.99642604;300.42767357"

    let fakeCoordinate = fakeNum?.components(separatedBy: ";")
    let fakeLat: String? = fakeCoordinate![0]
    let fakeLong: String? = fakeCoordinate![1]

    XCTAssertFalse(vc.checkLatValid(dummyLat: fakeLat!, dummyLong: fakeLong!, dummyNum: fakeLat!))
}

//testing for coordinates and it should fail because latitude is not a valid latitude
func testCoord() {
    let vc = ViewController()

    let fakeNum : String? = ";;;-57.99642604;120.42767357"

    let fakeCoordinate = fakeNum?.components(separatedBy: ";")
    let fakeLat: String? = fakeCoordinate![0]
    let fakeLong: String? = fakeCoordinate![1]

    XCTAssertFalse(vc.checkLatValid(dummyLat: fakeLat!, dummyLong: fakeLong!, dummyNum: fakeLat!))
}
```

Figure (5) Upper 3 test cases that is used to test if the app is working as intended.

5 test cases are created to make sure that the app is working as intended. testLat () is to make sure that the latitude entered is a valid latitude based on the conditions set in the checkLatValid function in the ViewController.swift file. If the latitude is less than -90 or more than 90 degrees then it will return false. XCTAssertFalse is used to help validate that the result will return false. Since the latitude entered is “-107.99642684” which is less than -90 degrees the result will return false. Since the result is false then the test is a success.

testLong () is to make sure the longitude entered is a valid longtitude based on the conditions set in the checkLatValid function in the ViewController.swift file. If the longitude entered is less than -180 degrees or more than 180 degrees then it will return false. XCTAssertFalse is used to help validate that the result will return false. Since the longitude entered is 300 degrees and more than 180 degrees the result will return false. Since the result is false than the test is a success.

testCoord() is to make sure the coordinates entered are all in numerical value. As you can see the fakeNum coordinates are “;;;-57.99642604;120.42767357 ” Since the coordinates entered are non-numerical the result will return false. Since XCTAssertFalse is used and false result is returned the test case is a success.



```
75      //testing for empty coordinates should fail because coordinates are empty
        func testEmpty() {
77          let vc = ViewController()
78
79          let fakeNum : String? = ""          Immutable value 'fakeNum' was never used; consider replacing with '_' or removing it
80
81          //let fakeCoordinate = fakeNum?.components(separatedBy: ";")
82          // had to remove this line and insert empty strings manually because there is only one index in fakeNum
83          let fakeLat: String? = ""
84          let fakeLong: String? = ""
85
86          XCTAssertFalse(vc.checkLatValid(dummyLat: fakeLat!, dummyLong: fakeLong!, dummyNum: fakeLat!))
87      }
88
89      //testing for valid coordinates with num only and this should fail because there is non-numeral are in the
            coordinates
        func testValidCoord() {
91          let vc = ViewController()
92
93          let fakeNum : String? = "aaaaadsasd-57.99642604;cxasdasdaczx300.42767357"
94
95          let fakeCoordinate = fakeNum?.components(separatedBy: ";")
96          let fakeLat: String? = fakeCoordinate![0]
97          let fakeLong: String? = fakeCoordinate![1]
98
99          XCTAssertFalse(vc.checkLatValid(dummyLat: fakeLat!, dummyLong: fakeLong!, dummyNum: fakeLat!))
100     }
101
```

Figure (6) Bottom 2 test cases to make sure the app is working as intended.

testEmpty is created to make sure that the coordinates entered is not empty. If the coordinates entered is empty the test result will return false and this is based on the conditions set in checkLatValid where if num = valid return false. Since the fakeNum coordinate, fakeLat and fakeLong are empty strings the result will return false. Since XCTAssertFalse is used and the result returned is false the test case is a success.

testValidCoord is create to make sure the coordinates entered are valid numeral coordinates only and no alphabets in it. Since the fakeNum coordinate is “aaaaadsasd-57.99642604;cxasdasdaczx300.42767357 ” the test result should return a false. Since XCTAssertFalse is used and the result returned is false the test case is a success.

Task 2 – Test your assumptions

Unit test are important as they can help to test the app whether it is ready to be released into the market. Several test cases are created to test the created app whether it is working as the developers wants it to be. If one of the test cases is not returning the results that the developers wants that means there is an error in the app. Other than that, unit test makes it much easier for developers to figure which area of the app needs fixing.

An example of how a unit testing would be useful is testing whether the user has access to internet connections. The scenario would be a user trying to load an item from the Lazada application and the test condition would be the user has to have internet connection or the test case will fail. If the user clicked on an item on the Lazada app but the user has not internet connection meaning that the user failed to meet the test condition, the Lazada app will print out an error message saying "connect to an internet source and reload the page after". But if the user has internet connection then the user will be redirected to the item page that he clicked on the Lazada app.

Another example of a test case would be the user has internet connection but the Lazada app database server is not responding and the condition of this test would be the Lazada app will be able to connect to the database within 30 seconds else the test case is a failure. If the user has selected an item from the Lazada app, the Lazada app will try to connect to the database and if it is able to connect to the database within 30 seconds it will retrieve the data required and show it to the user on his screen and this is considered a success but if the app fails to connect to the database after 30 seconds, the Lazada app will print out a message to the user saying " server is not responding please try again later or reload the page" and this is means the test case failed.

A last example of a test case would be when the user requested for a list of items on sales and whether or not the Lazada app is able to retrieve the list of items on sales is the condition. If the Lazada app is able to show the user the list of items on sales then the test case is successful based on the conditions. Else if the Lazada app is not able to show the user the list of items on sale and it will print out an error message then the test case is a failure based on the conditions and in real life situation the Lazada app will print out a message saying "Items on promotion is not available please try again later."