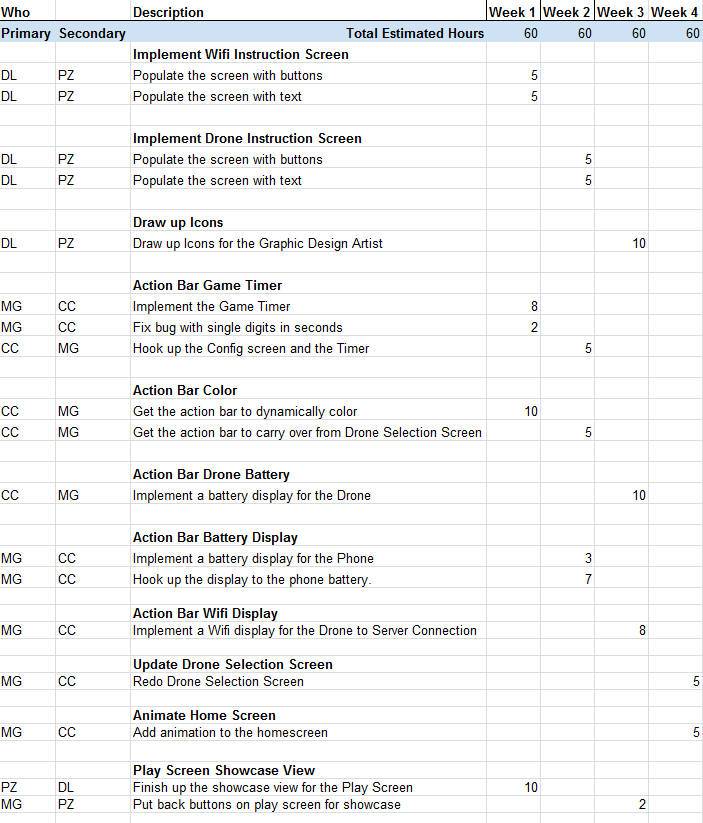
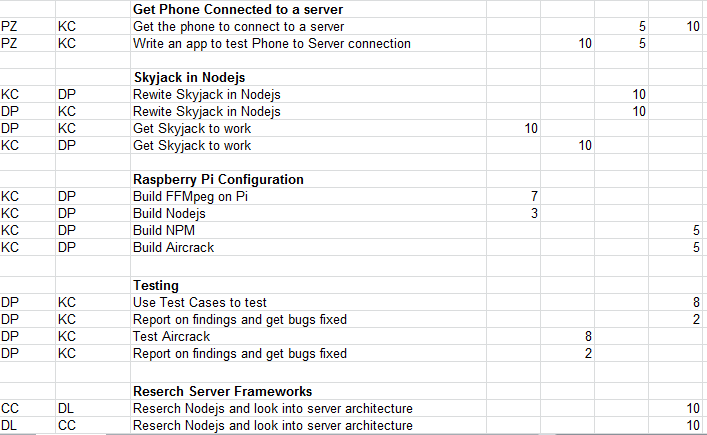
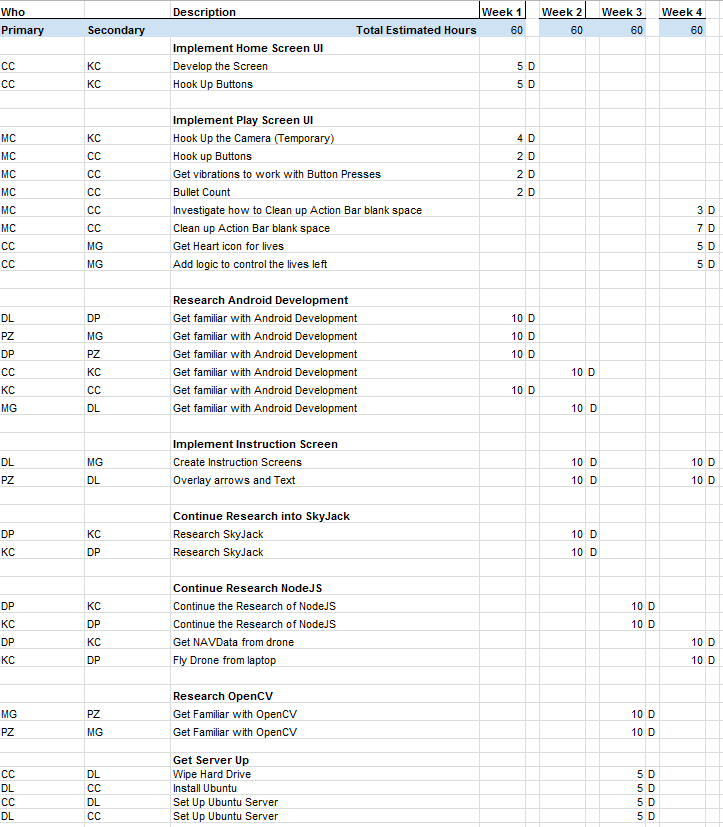
Team A

**Section A - DP:**

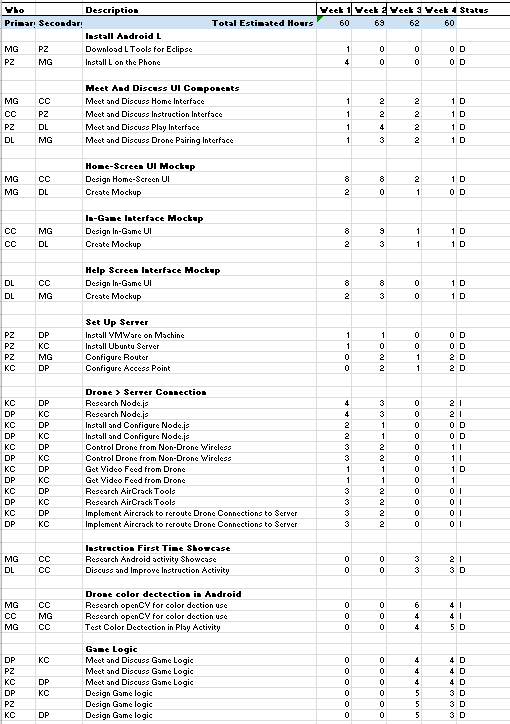
**Sprint 3 Backlog**



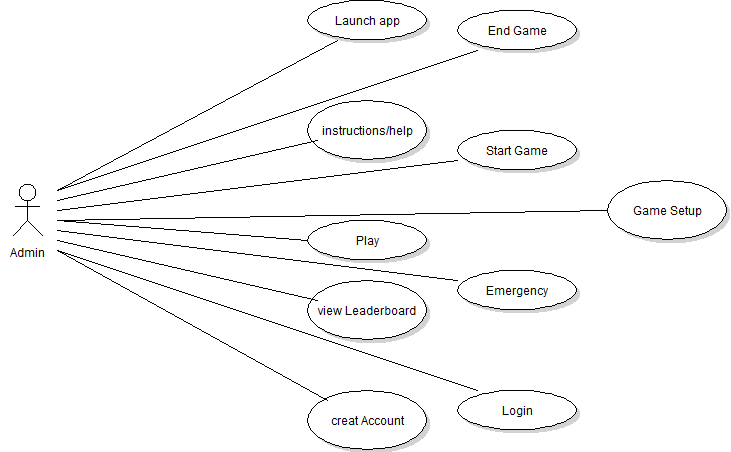
**Sprint 2 Backlog:**



**Sprint 1 Backlog:**

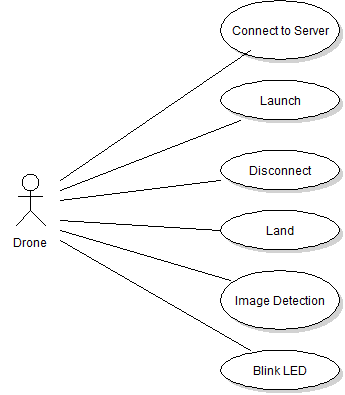


**Section B: Use Case Diagrams - MG**



**Figure B-1 The Player Actor Use Case diagrams**

The Drone Wars application has two different users, which are represented by the actors; they include the **Drone** and **Player**. In figure B.1 the player actor is shown with the use-cases Launch App, Instructions/Help, Leaderboard and Play Game. These user cases allow the user to control the **drone** actors and register their previously played game to the DWA leaderboard server by either creating an account or logging in. The **player** also has the ability to start and end their own involvement in the game.



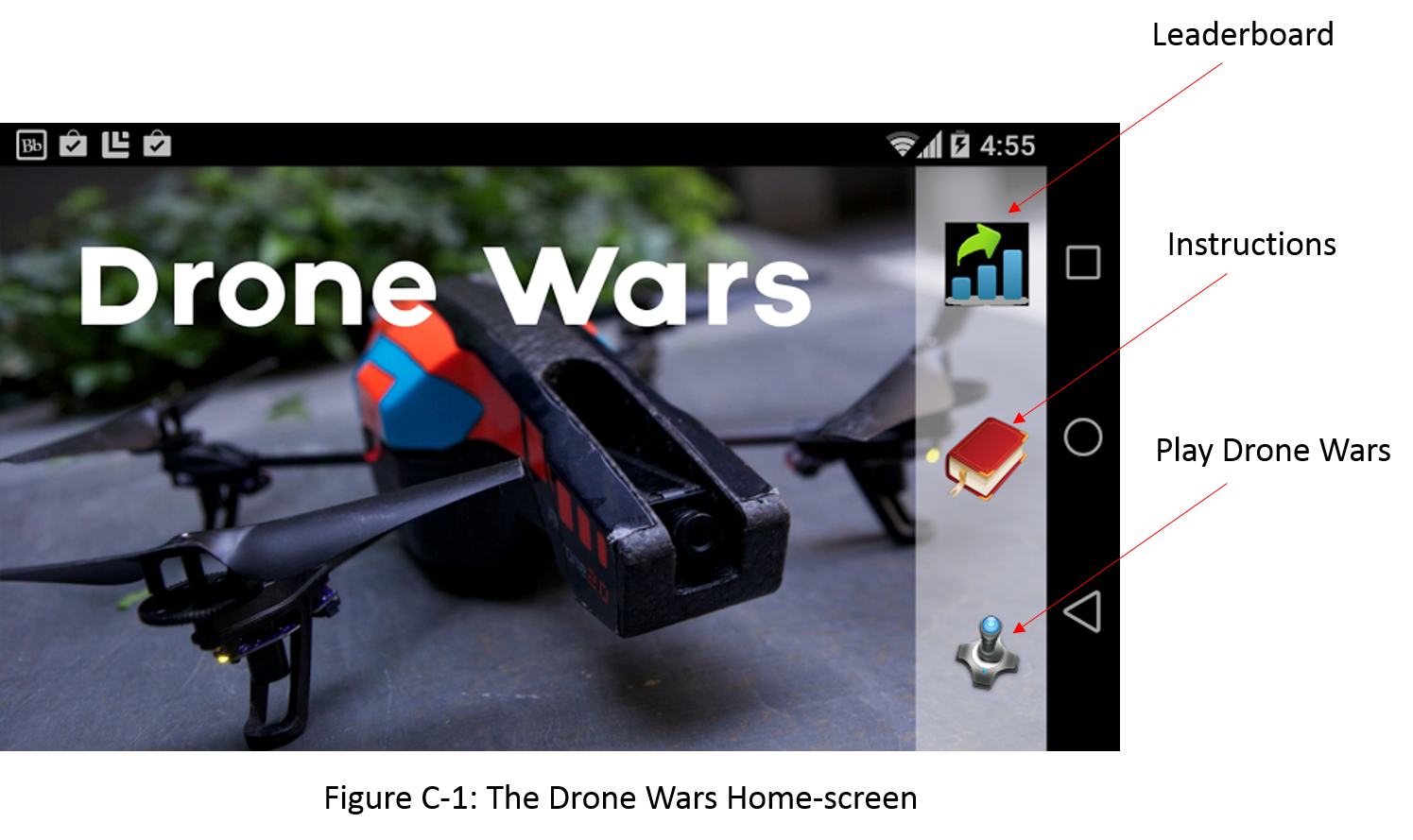
**Figure B.2 The Done Actor Use Case.**

From figure B.2, it is apparent that the **drone** actor has the ability to connect to the game server. The game server connects the **Drone**, to the game server and the **player** by IP. The **Drone** actor will be able to disconnect itself from the game server if its lives run out and it enters the auto land feature and it can disconnect if the **player** actor flies out of the designated games boundary. The **drone** actor also has the ability to blink its LED and image Detections.

**Section C: User Based Specifications/Interfaces - Team**

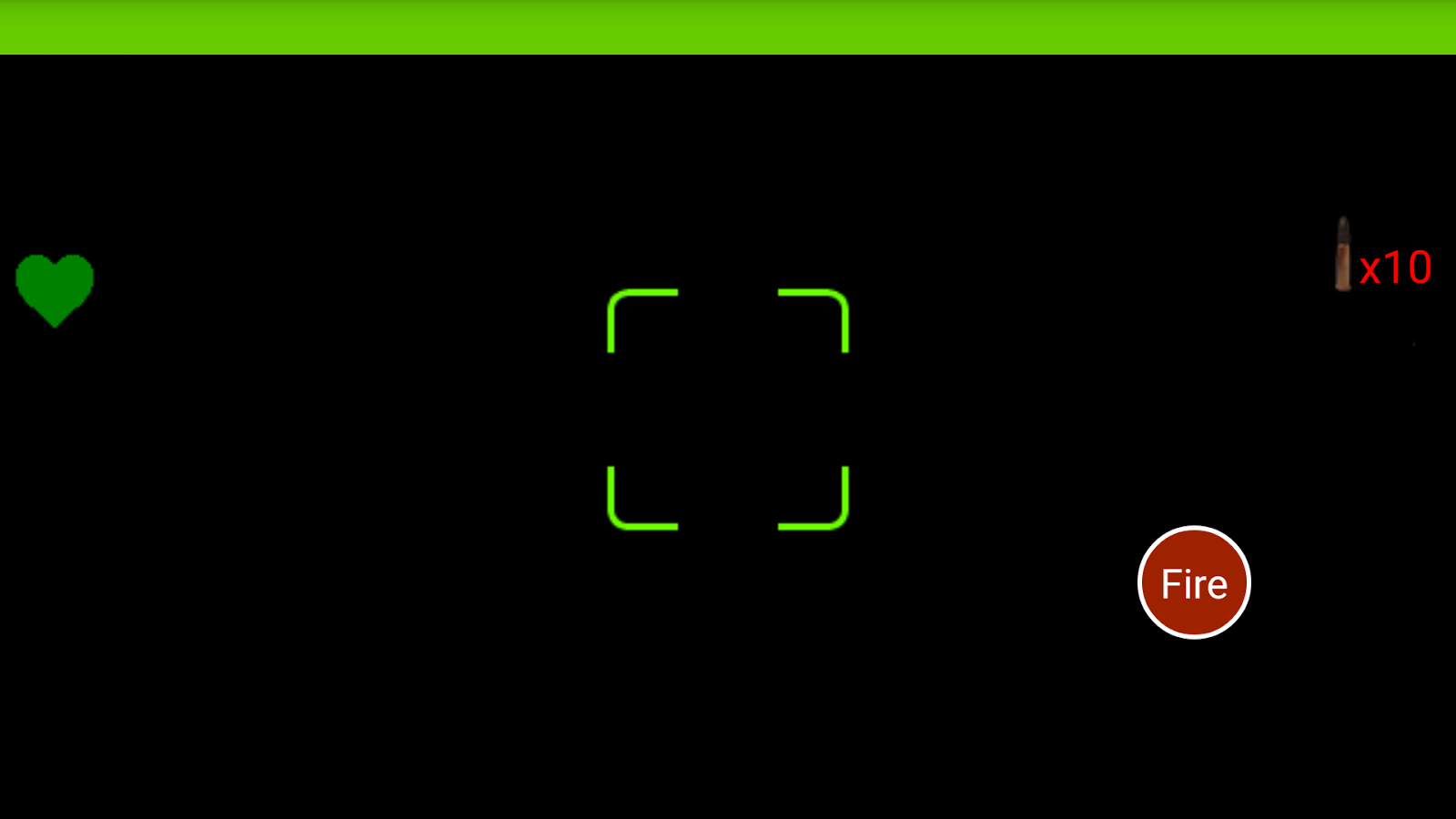
**C.1 Home-screen**

When the app first opens the user is brought to the home screen. Figure C-1 shows the Home-screen and the three buttons available to the user.



From the Home-screen the user will be able to choose from 3 options: the first is Leaderboard, the second is Instructions, and the last is Play Drone Wars. If the user selects the Play Drone Wars button for the first time then the user will be brought a help screen where the Play Screen Instructions will be shown. The Play Screen Instructions label and explain the various parts of the Play Screen. If it is not the user’s first time selecting Play Drone Wars then the user will go immediately to the Drone Selection Screen (See section **C.4**) and then from there to the Play Screen (See section **C.2**). When the user selects the Instructions button they are brought the Instructions Screen (See section **C.3**). If the user selects Leaderboard they are brought the Leaderboard Screen (this is an additional feature and will be implemented if time allows).

**C.2 Play Mode**



**Figure C.2.1** The interface for play mode

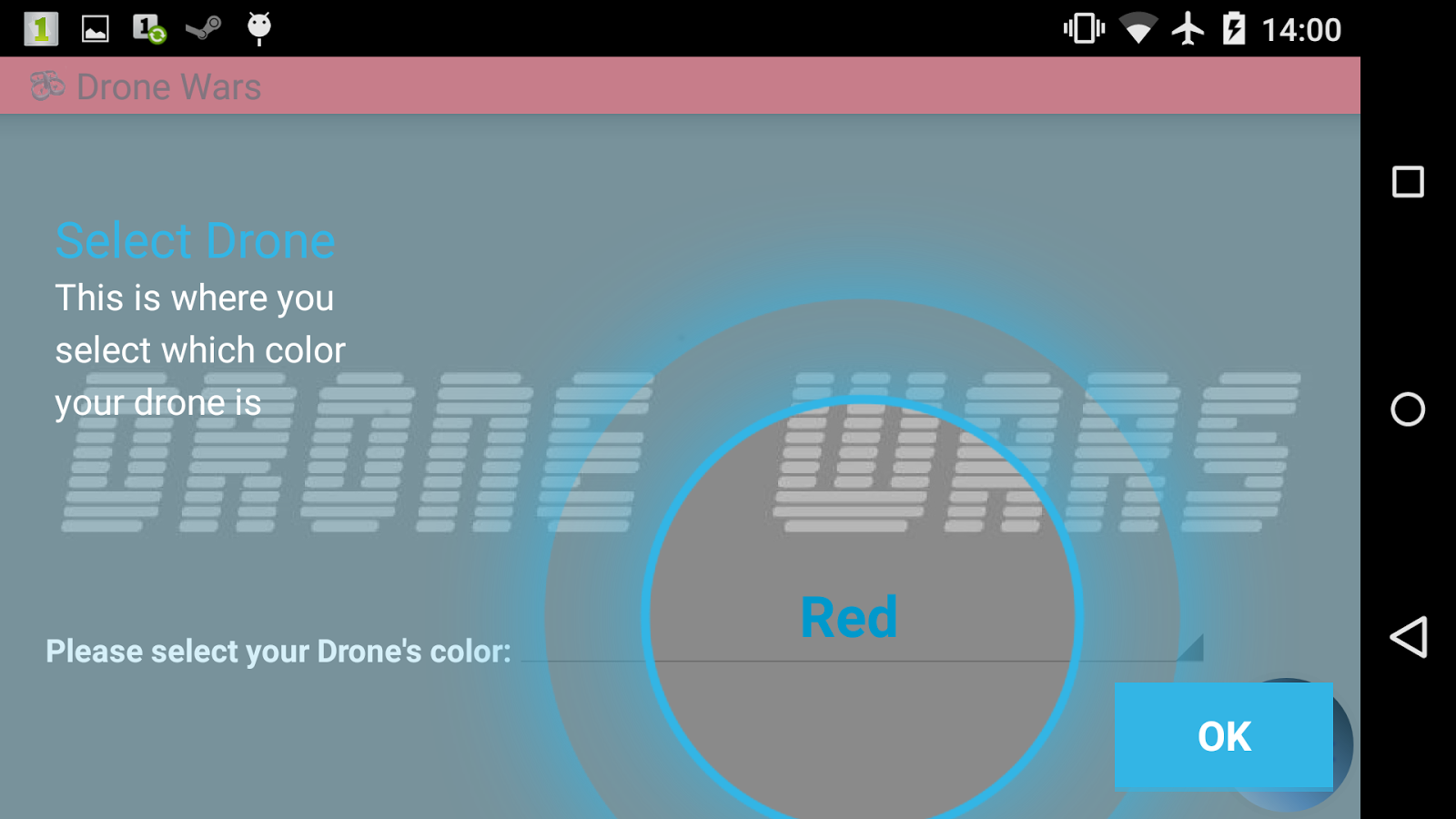
The mockup screen of the play mode is shown in Figure C.2.1. There are two main areas of the screen: the status bar and the game interface.

As a multi-player game, different drones will have different colors. The background of the status bar will be the color of the drone that you picked before playing. In the center of the status bar, is the game timer; it shows the time remaining of the current game. On the left of the bar, there is a wifi signal strength sign. This shows the Wi-Fi signal of the drone from the router of the server. Keeping this signal strength will allow us to detect if a player is too far away from the play area, so that the application can notify the play to go back. To the right, it is the battery of the drone. This allows the player and the application to keep track of the battery on the drone. If the drone’s battery reaches 5%, it will notify the player and go to emergency landing mode automatically. On the right side of the status bar, there is a battery life of the phone.

The background of the game interface will be the video feed from the drone that is being controlled. In the upper left corner of game interface, there is remaining life of the drone. In the upper right corner, there is the remaining ammunition of the player. On the left side of the screen, it shows the drones that are still in the game. There are three controls on the bottom of the screen: the accelerometer activator, the joystick/fire button and the land button. The accelerometer activator allows the player to move the drone. As the accelerometer activator is pressed, tilting the phone to control the motion. The joystick will control the drone’s facing direction. We will let the joystick be able to detect double tap, which will be the fire button. The red box on the screen is the indicator of the enemy drone, it will show whenever an enemy drone is in sight; and the red box will follow the enemy drone. In the middle of the screen, it is the target area. Whenever the player make a shot, the server will computes to see if a red box is in the target area. If so, the enemy is hit.

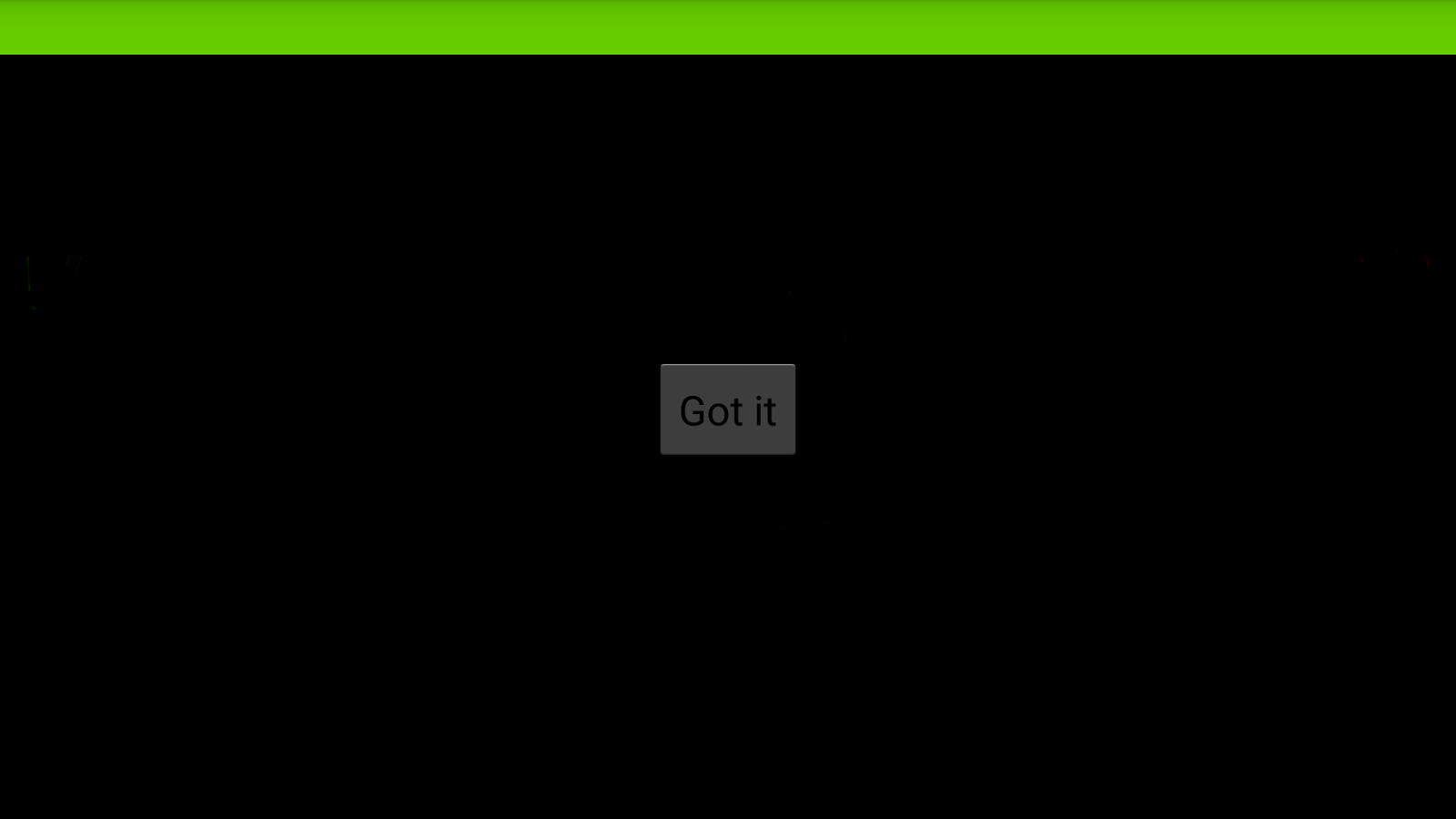
**C.3** **Instructions Screens**

The Instructions Screen will enable the user to read and review the instructions for different parts of the app.



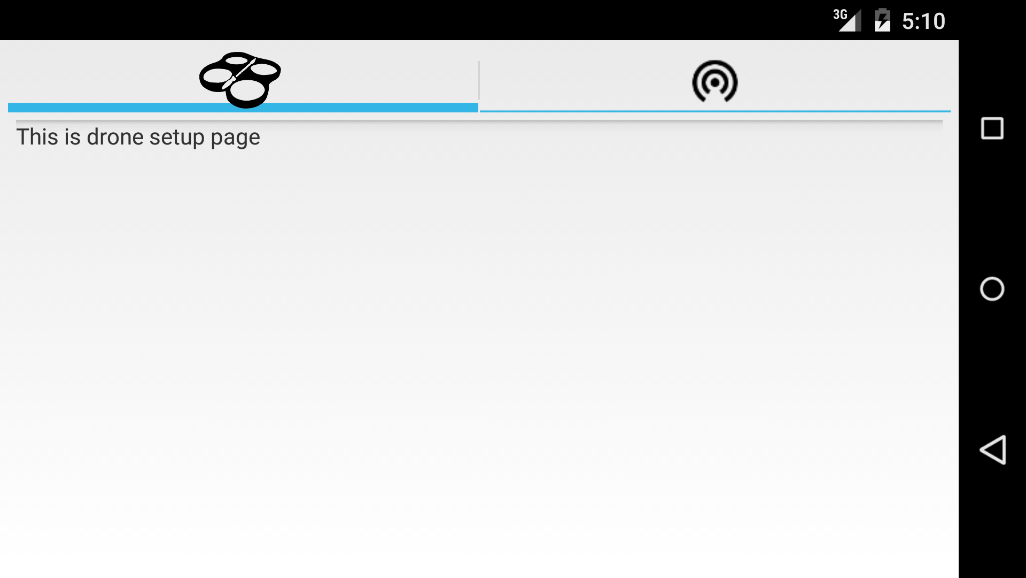
**Figure C.3.1:** Drone Selection Instruction Screen

Figure C.3.1 is the first screen that the user will see when they enter the play mode for the first time. The showcase viewer will overlay the screen to provide the user with the information needed to understand what is going on. The viewer will allow the user to participate in selecting the drone color. Once the selection has been made, the user can select the next screen.



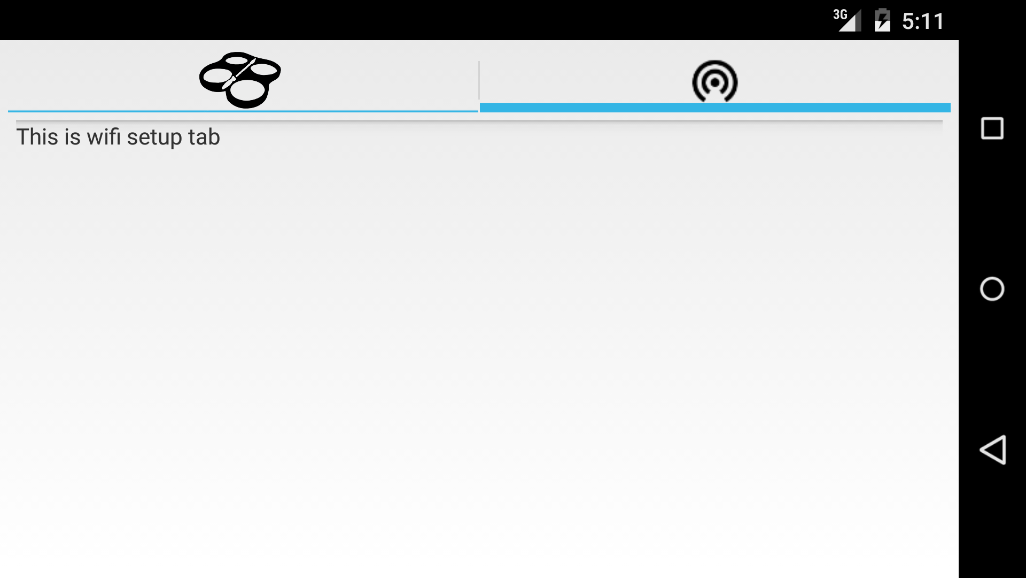
**Figure C.3.2** The drone setup page Instruction Screen

Figure C.3.2 is a limited mockup of the Play Screen Instruction Screen. This screen will show up when the user first launches the app and goes through all the other instruction screens. This screen is going to be populated with the showcase view and all of the icons will have explanations of what they do. There will be a different view for each control. This will encompass all the action bar items, all drone control buttons as well as HUD items, and life/bullet displays.



**Figure C.3.3** The drone setup page Instruction Screen

Figure C.3.3 shows the drone setup page instruction screen. In this screen the user will be able to read instructions regarding drone setup. These instructions will help the user connect to their drone to the server so that they can then control their drone in game.

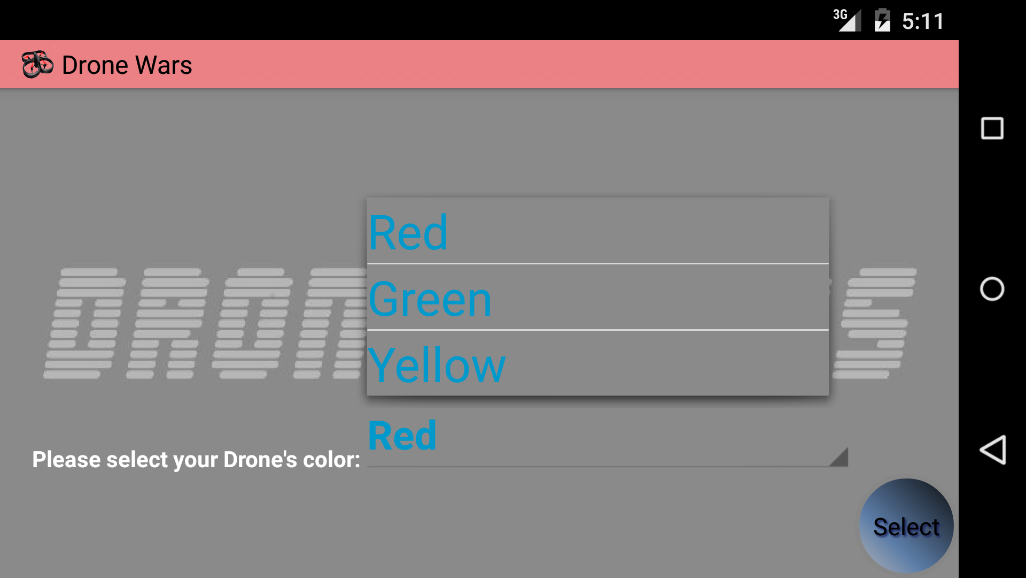


**Figure C.3.4** The Wi-Fi setup page Instruction Screen

Figure C.3.4 shows the Wi-Fi setup page instruction screen. In this screen the user can learn how to connect their phone to the server via Wi-Fi. These instructions will help make clear how to configure their Android device so that they can control their drone.

**C.4 Drone Selection Screen**

The Drone Selection Screen is very minimal and will allow the user to select their respective drones.

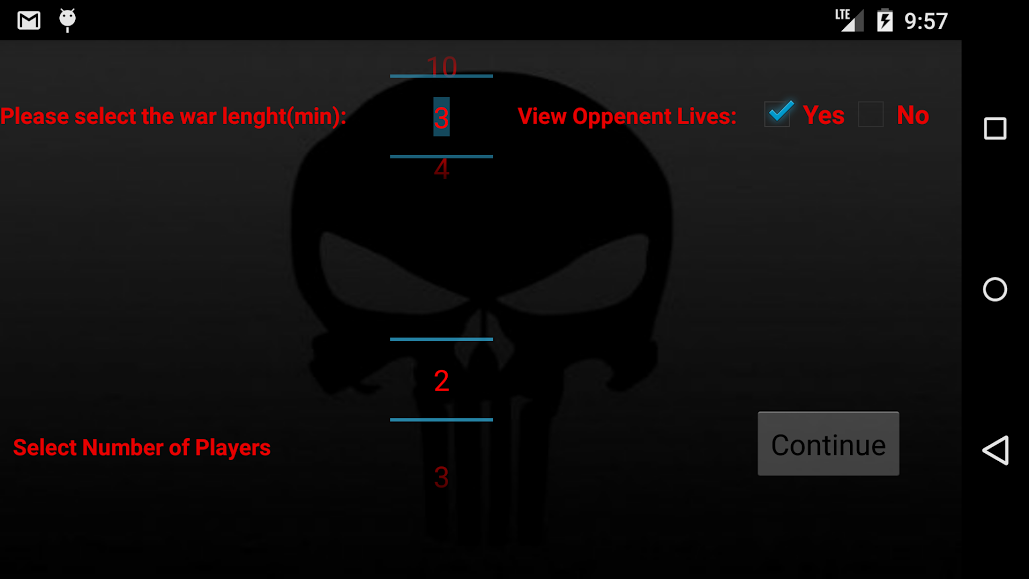


**Figure C.4.1** The Drone Selection Screen

Figure C.4.1 shows the Drone Selection Screen where the user will select which drone is theirs. Depending on the color the user has chosen for their drone determines which drop down they will select. The list will be dynamically populated based on the number of drones connected. The action bar will reflect the current displayed color on the page. The user can change colors before hitting the select button.

**C.5 Game Configuration Screen**

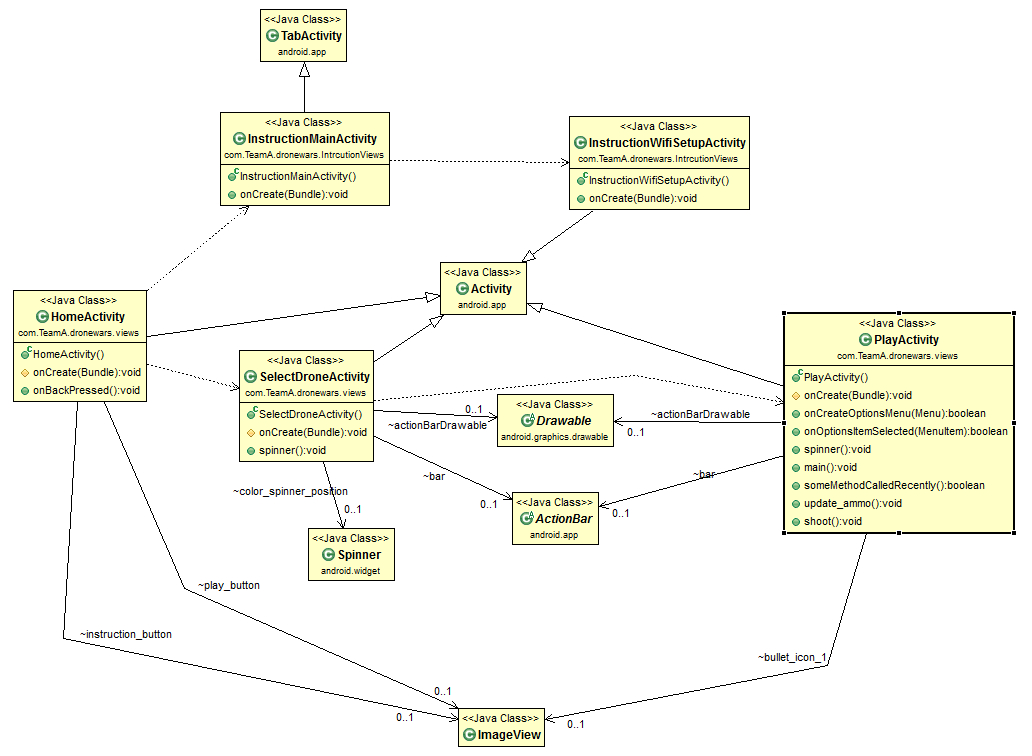
The Game configuration screen is where the user who selects red will set the match parameters like game time, lives, etc.



**Figure 5.1:** Game Configuration Screen

Figure 5.1 shows the Game Configuration Screen. If the user elects to be the red drone, by default they will be elected into the administrative position for the match. The interface that they will see is basic parameter selection spinners. This user will be responsible for all the game properties like time of match, lives, type of flight (future) and whatever configurations pertain to the match. There will be another

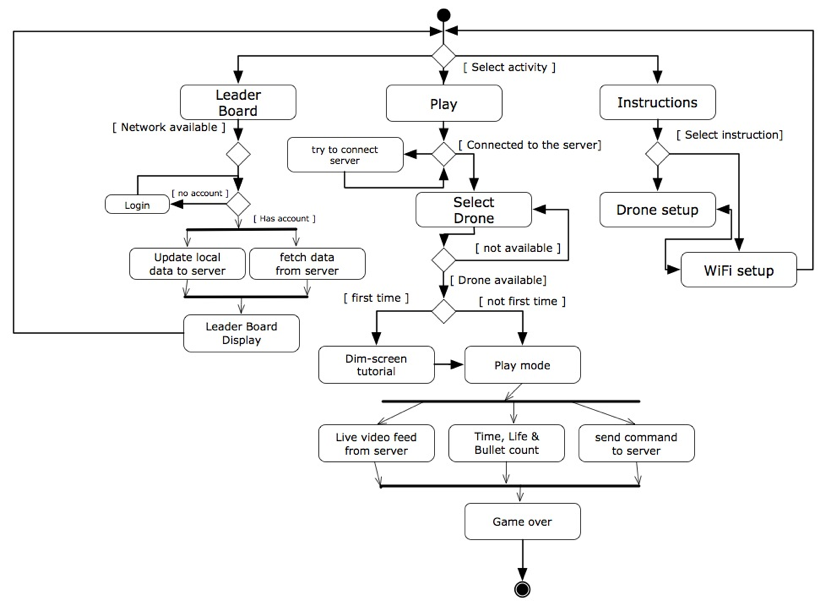
**Section D:**



**Figure D.2** UML Diagram of the program.

**UML Class Diagram - C.C.**

Figure D.2 shows the UML diagram of our application so far. We have partially implemented two parts of the app: instruction and play. As it is shown in the figure above, the instruction has a main activity, InstructionMainActicity, which extends TabActicity, and each tab is also an activity. All other classes extend Activity as well. The play part contains two activities so far: SelectDroneActivity and PlayActivity. A user will go to SelectDroneActivity before PlayActivity. SelectDroneActivity and PlayActivity have custom action bars, so they contain ActionBar class in order to be able to customize the action bar. SelectDroneActivity contains a spinner, so that a user can pick a color to represent his drone. The Drawable class is included to modify the action bar. Both HomeActivity and PlayActivity have ImageViews.



**Figure D.3** Activity Diagram

**Activity Diagram - D.L.**

The activity diagram in *Figure D.3* illustrate the different states of the App that user may be in and the transition between the states. When launching the app, user will be able to select next step within 3 options.

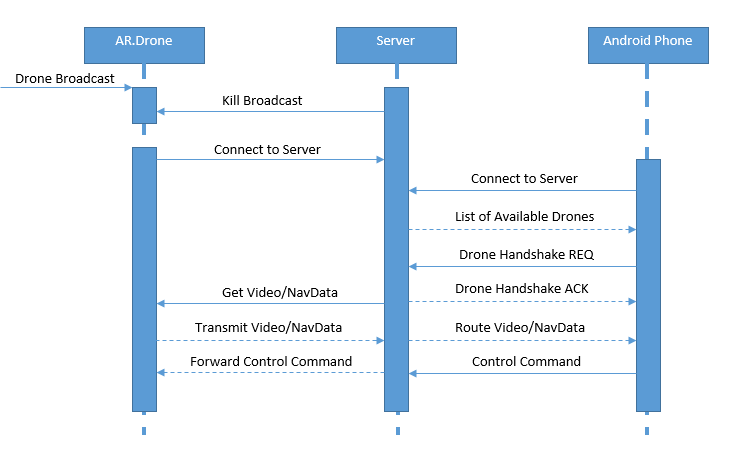
The first option is leaderboard activity in which the user and check his/her rank on the Google Play server. In the leaderboard activity, the app will first check if there is already an account that can be used to login into the Play server, if not, the user will be asked to provide an valid account and password with which the App can fetch out user data from the server and otherwise there will be not leaderboard information available.

The second option is play activity, which is the main activity of the App that allows user to play the drone wars. In this activity, the app will first check if the Android device is connect to the local server, if no, it will ask the user the concept to the server in order to go to next step. If successfully connect to the server, the App will allow user to select a available, the user will be asked to pick one of drones that hasn't been pick by other player. The server will send to information to the App that which drone is whether available or not. After picking the drone, the user will enter the play mode, which allows user to control his drone and fight against other player. If it’s the first time entering the play mode, there will be a dim-screen tutorial (The device screen will be dimmed a little bit while some of the buttons or parts on the screen will be highlighted with description in order to help user get familiar with the drone control interface). In this mode the App will keep communicating with the server for getting live video feed and sending out the command that controls the drone. The video feed will be displayed as the background of drone control interface and through the video user can see the picture captured by the front camera on the drone. When the user presses the button on the drone control interface, an command will be sent to server accordingly and then be transferred to drone. Meanwhile the app will count the game time, bullets and health points of the drone and will terminate the game when certain condition are met (such as the depletion of health points or running out of time).

The third option is the instruction activity. There will be instruction for how to setup the drone and server in order to play the game. There will also be an exhaustive tutorial about how to control the drone using the drone control interface in the play mode.

All of the three activity can goes back to the first screen of the App where all three options are display. One game session is completed when the game ends in the play mode.

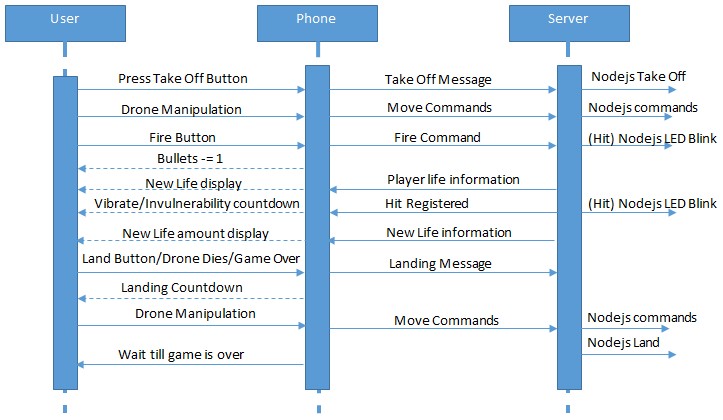
**Server Sequence Diagram - K.C.**



**Figure D.4** Server Sequence Diagram

Figure D.4 shows that initially when the AR.Drones are started they are in broadcasting mode and the server actively scans the surrounding area to find available AR.Drones based on their MAC address. When a broadcasting AR.Drone is found the server sends it a request to kill the broadcast and connect to the server as a client. At this time the AR.Drone performs a soft reset upon which it starts up in client mode instead of broadcasting mode and it connects to the server’s network. The server maintains a list of all connected drones. When an Android device connects to the server, it receives a list of all the available drones that have not been paired yet. At that time the user can select given drone to pair to which initiates a handshake request to the server that maps the current users Android device to the selected drone. When the handshake is successful the server sends a handshake acknowledgment message to the Android device as well as sending a request to the AR.Drone to get its Video and Navigation Data which is then routed to the appropriate Android device. Lastly any control commands from the Android device are sent to the Server which then forwards those commands to the appropriate AR.Drone.

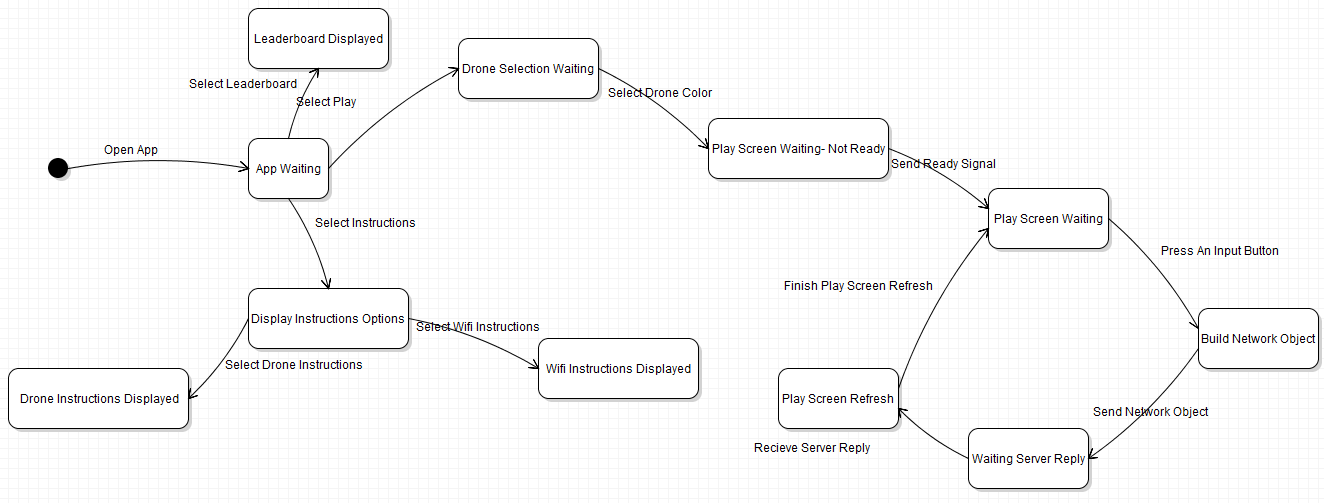
**App Sequence Diagram - D.P.**



**Figure D.5** App Sequence Diagram

Figure D.5 shows how the app will handle certain commands and actions coming from the server and user. The first thing that the user will do is issue the takeoff command. The game will not start until all drones are flying. The user will select the takeoff command and the phone will send the command to the server which will relay a NodeJs command to the drone. Once in the air, the user can use the drone manipulation buttons like left, right, up down, etc. to fly the drone. These commands will be again sent to the server and then relayed to the drone as NodeJS commands. If the user hits the fire button, the phone will see if they indeed have a shot remaining. If they do, the command is sent to the server which will see if the enemy drone is in the target area. At the same time, the phone will update the bullet display to reflect the shot fires. If a hit is registered, the server sends a NodeJS LED animation command to visually show that the drone has been hit and entered an invulnerability state. At the same time, the server will not allow the affected drone to take any damage while in this state. The server will send the new life data to the phone and the phones display will update to reflect the data. When the user hits the land button, loses all their lives, or the game ends, the phone will send a land command to the server. The server will start a timer and show it to the user. The user will have the specified amount of time to get to a safe location to land. During this time, the player can only move the drone; all other game functions are disabled. Once the drone has landed safely, all commands are disabled until all drones are landed and the playing field is safe and the game is over. The player must wait until the game is over.

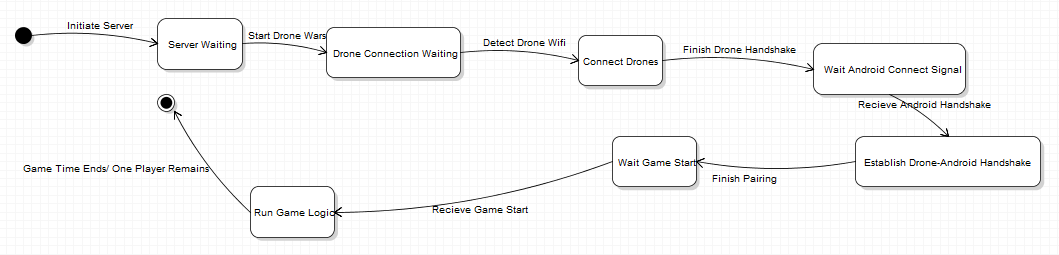
**Statechart UML Diagrams - P.Z.**



**Figure D.6** Android App Statechart UML Diagram

Figure D.5 shows the statechart diagram UML for the Android application. The machine starts when the user opens the app. From there the application will be in a waiting state where it is expecting input from the user. If the user selects the Leaderboard section of the app the machine moves into the Leaderboard Displayed state. As the name suggests, this state is where the user will view the leaderboard. The user can also leave the App Waiting state by selecting the Instructions. This will bring the user into the Display Instructions Options state where the user will leave by choosing either the Drone Instructions or the Wi-Fi Instructions. Depending on what they select, the user will move into the Display Drone Instructions or Display Wi-Fi Instructions, respectively.

The last way that the App Waiting state can be left is if the user selects the Play option. Once the user selects the Play option, the machine moves into the Drone Selection Screen Waiting state. This state can only be left once the user selects which Drone they wish to pair with. Upon leaving, the Play Screen Waiting - Not Ready state will be entered. This can only be left when all players select that they are ready to begin the game. From there the Play Screen Waiting state is entered. This state is constantly looped to. The state is left when the user presses an input button on the screen to control the drone. The machine will then enter the Build Network Object state. This state will build the network object to send to the server. Once the object is sent, this state is left and the Waiting Server Reply state is entered. This state will be left when the server sends back a response. At this point the Play Screen Refresh is entered and the visuals on the screen are updated. This state is left and the Play Screen Waiting is re-entered when the Play Screen is finished refreshing. Every state is able to the App Waiting state by transitioning on the back button of the device being pressed. Likewise, every state is able to end the state machine by transitioning on the event of the home button being pressed. In order to simplify the diagram these were left out.



**Figure D.7** Server Statechart UML Diagram

Figure D.7 show the statechart UML diagram for the server. The state machine begins when the server is booted up. At this point it has entered the Server Waiting state where it waits for the Drone Wars application to start. When the Drone Wars application starts the machine enters the Drone Connection Waiting state. When the server detects Drone Wifis then the machine will enter the Connect Drones state. This state is left when the Drones finish the handshake with the server. At this point the machine will be in the Wait Android Connect Signal state. This state will be left when the server receives the Android handshakes. The machine will then enter the Establish Drone-Android Handshake state. This state can only be left when the pairing between the Android devices and the drones is complete. At this point the machine will be in the Wait Game Start state. This state is left when the server receives the game start signal. The server will then be in the Run Game Logic state. Finally, the machine will finish when either the game time ends or only one player remains.

**Section E: Test Plans - Team**

**Test Plans Sprint 2 With Results**

Test #1 **Ammunition limitation**

Ensuring that the Players ammunition is regenerated 15 seconds from when the clicked the fire button in sequence. The ammunition regeneration timer was taken of from the server and implemented in the PlayActivity of DroneWars. A boolean is used to keep track of when the reloading process is finished so the handler can start on the next bullet. The Ammunition limitation test will also compare the drone average flight time and compare how many shots can be fired within that time frame to ensure that enough ammunition regenerates within that time frame to make the game enjoyable and fair.

*Test Results* - PASS

Test #2 **Acquiring Target**

Acquiring target test involves checking whether the enemy drone can be recognized when they are within the target area in the mobile app. This test will include testing whether the applications crosshairs x and y coordinates will have to be mapped, or the servers image detection could be used to determine if an enemy drone enters the players crosshairs with OpenCV. Both methods will be tested to determine which is the best solution for us to use.

*Test Results* - Not Applicable

Test #3 **Home Screen Logic**

So far we have implemented two of three buttons on the home screen: instruction and play buttons. Base on whether the application is first time open or not, the button will direct the user to different activities. For example, if the play button is selected the first time, the user will go to instruction activity automatically to ensure that the user know how to use our app before he actually play the game. And there will be a check box to indicate “Don’t show this again”. This test will focus on the flow of the app, to make sure that the design is as friendly as possible.

*Test Results* - Failed - Not Implemented - Planned for Sprint 3

Test #4 **Action Bar In Play**

In our play activity, we made a custom action bar to hold all the information we need to show to the users. First, when a user selects a color for his drone in select drone activity, the action bar will dynamically change to that color. After selecting the color, the action bar in the play mode will also have the same color. And the action bar in both activities will be 50% transparent. This test case will focus on testing how the action bar reacts when a different color is selected. It should dynamically change to correspond color and to be 50% transparent.

*Test Results* - Pass

Test #5 **Aircrack-ng**

Initially when AR.Drones are turned on, they broadcast a local Wi-Fi network and the server uses Aircrack-ng to scan the local area for available access points within the registered sequence of MAC Addresses for AR.Drones. Once an access point is found the server connects to the AR.Drone and executes a sequence of commands causing the AR.Drone to stop broadcasting the local WiFi network and instead connect to the Wi-Fi network being broadcasted by the server. Will need to make sure that server can find broadcasting AR.Drones and that they will appropriately connect to the server as clients.

*Test Results* – Pass - Aircrack works fine, the script needs to be rewritten in node to add enhancements and make access point work

Test #6 **Android Device/Drone Handshakes**

When an Android Device connects to the server it receives a list of all available AR.Drones. Once an AR.Drone is selected, the handshake sequence is initiated and a Handshake Request is sent to the server to map the current device to the requested AR.Drone. When the handshake is successful a Handshake Acknowledgment is sent to the Android device. At the same time the server sends a request to the AR.Drone to forward its video stream and navigation data through the server to the paired Android device. Test will include verifying list of available AR.Drones and success rate of pairing to AR.Drones.

*Test Results* - Not Applicable - Planned for Sprint 3

Test #7 **Server Latency**

The server will be a centralized location through which the video stream and navigation data from all connected AR.Drones as well as control signal from all connected Android devices will be routed through. The tests will include measuring the video latency from the drone to the Android device and the latency of control signals from the Android device to the AR.Drone. Need to make sure both latencies are as small as possible to create a smooth play experience.

*Test Results* - Not Applicable - Planned for Sprint 3

Test #8 **Home Screen Button Responses**

The home screen will house three different buttons. Each one will do something different. The Play button will have two different behaviors depending how many times the game has been accessed. If the user is accessing the game for the first time, they are forced into the walkthrough instructions. This will walk the user through the process of selecting drones and what all the buttons and icons mean on the screen. If it is not the first time, the button will take the user straight to the drone selection screen and allow an unhindered access to the game. The Instruction Button will bring the user to the instruction screen. The leader board button will bring the user to the sign in page if they are not signed in. Once signed in, they will be brought to the leaderboard interface.

*Test Results* - Pass

Test #9 **OpenCV**

During gameplay if an enemy AR.Drone is seen in the video crosshairs, the enemy needs to be identified by OpenCV using its colored tag. OpenCV will either run on the server preprocessing the video streams identifying which drones are visible on the given video stream before streaming it to the Android device, alternatively OpenCV will run natively on the Android device analyzing the received video stream. OpenCV performance running on both the server and Android device will be evaluated and the one that provides best performance and least amount of implementation complexity will be used in the final product.

*Test Results* - Android Phone not successful running OpenCV - Test on Server Sprint 3

Test #10 **Drone Control**

The Drone is expected to behave a predetermined way with predetermined inputs. If the user provides a takeoff command while the drone is landed, the drone is expected to take off. If a land command is used, the drone is expected to land, if flying. The drone must respond as expected with all combinations of user input.

*Test Results* - Pass

Test #11 **Tutorial**

When the user first opens selects the Play option they will be guided through a tutorial. In this tutorial they will be shown the different features of the Drone Selection Screen and the Play Screen. This tutorial will be deemed successful if all supported features are shown in the tutorial and explained regardless of current application conditions.

*Test Results* - Pass

Test #12 **Router**

The router will be tested to ensure that connections are maintained throughout the entirety of the game. The range and signal strength to the Drones as well as the range and signal strength to the phones will be tested. Furthermore, a stress test to ensure that all video feeds are smooth will be performed.

*Test Results* - Successful Connection on both 2.5 and 5 GHz bandwidth

**Test Plan Sprint 3:**

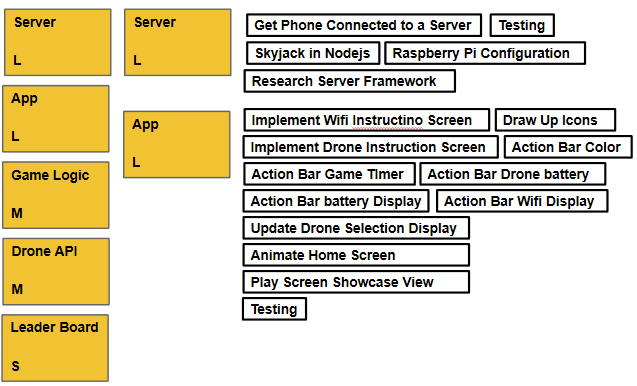
The plan for sprint 3 is to reuse the test plan for Sprint 2 when more gets implemented. Some additional test cases:

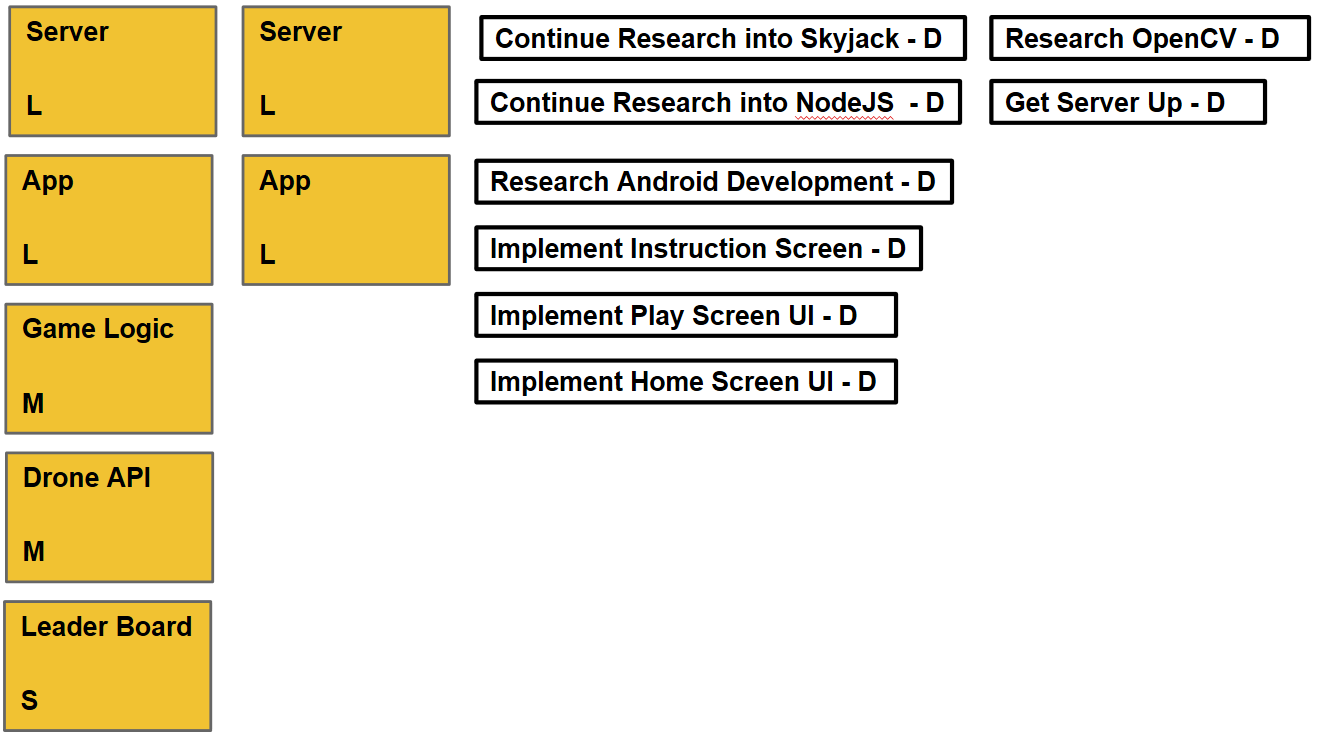
Test #13 **Router Access Point**

The router will be tested to ensure that connections are maintained throughout the entirety of the game. The range and signal strength to the Drones as well as the range and signal strength to the phones will be tested. Furthermore, a stress test to ensure that all video feeds are smooth will be performed.

**Product Backlog**

**Sprint 3**



**Sprint 2**

**Sprint 1**