THE UNIVERSITY OF WESTERN AUSTRALIA

Grand Traffic Auto 2 (GTA2)
User Manual
CITS5552 - Groups 3 & 4

Name	Student Number
Michael Dorrell	21989175
Joshua Goh	21978677
Jarrod Greene	21963609
Max Evans	21970246
Samuel De La Motte	21720063
Dina Jaber	21812873
Hanadi Jaber	21815003
Larry Huynh	21516059
Daniel Jacobs	21952584
Mackenzie Jones	21968754

1. Introduction

Transportation options have appreciably increased over past years with alternatives to driving such as autonomous vehicles, rideshare options (Uber, Didi etc.) and public transport (buses, trains etc.). This project was designed so that researchers at UWA can conduct experiments to assess how different individuals respond and react in numerous transport-based events. Grand Traffic Auto was developed to reproduce these events in order to gather information regarding the player's transportation choices throughout the game. Grand Traffic Auto assembles data on the player's choices and stores it in a Mongo dB database for easier accessibility. This gives researchers a strong foundation to manage transport experiments in a controlled environment whilst providing players with an appealing game to enjoy.

2. Users

There are two types of users in the Grand Traffic Auto application. The first is the *Administrator* (admin) who configures the game settings to create different experiments in which they would like to observe and collect data. The second type of user is the *Player* who plays the game and has their responses collected for the administrator.

2.1 Administrators

The administrator can change the information regarding the drivers in the game by following these steps:

1. Go to the *Scripts* folder under the *Assets* folder in the GTA2 unity game (figure 1).

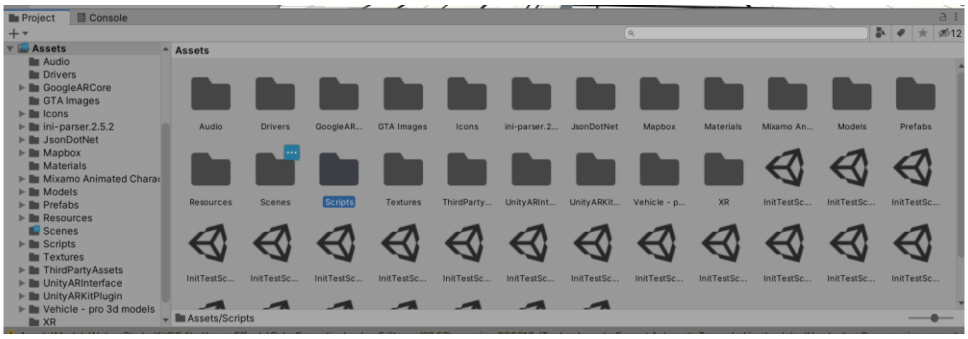
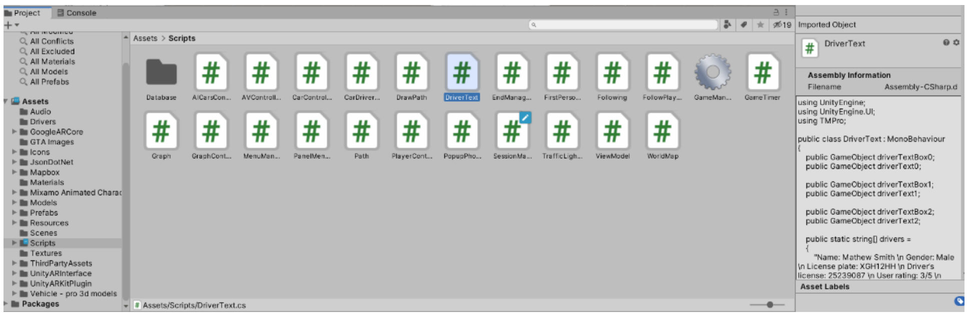


Figure 1 – Scripts folder

2. Inside the *Scripts* folder, open the script file labelled *DriverText.*(figure 2).

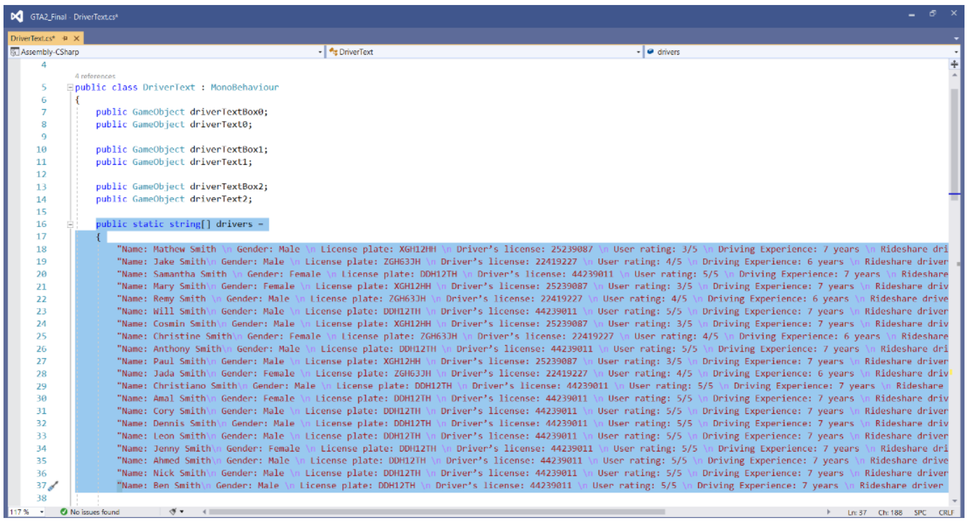


```
using UnityEngine;
using UnityEngine.UI;
using TMPro;

public class DriverText : MonoBehaviour
{
   public GameObject driverTextBox0;
   public GameObject driverText0;

   public GameObject driverTextBox1;
   public GameObject driverText1;

   public GameObject driverTextBox2;
   public GameObject driverText2;

   public static string[] drivers =
   {
      "Name: Mathew Smith \n Gender: Male
\n License plate: XGH12HH \n Driver's
license: 25239087 \n User rating: 3/5 \n
```

Figure 2 - DriverText script

3. The *DriverText* script has a string array labelled *‘drivers’* with the details of all the drivers separated by a comma (figure 3). These details can be changed as needed based on the experiments desired by the admin. Ensure to only edit the red text and do not make any changes to the **\n** character which symbolizes a new line or the comma at the end of every line. Every driver’s description is within double quotations. The admin can change the information regarding the driver’s name, gender, license plate number, driver’s license number, user rating, driving experience and rideshare driver experience. An example can be seen in the text below:

"Name: Mathew Smith \n Gender: Male \n License plate: XGH12HH \n Driver’s license: 25239087 \n User rating: 3/5 \n Driving Experience: 7 years \n Rideshare driver experience: 2 years"**,**

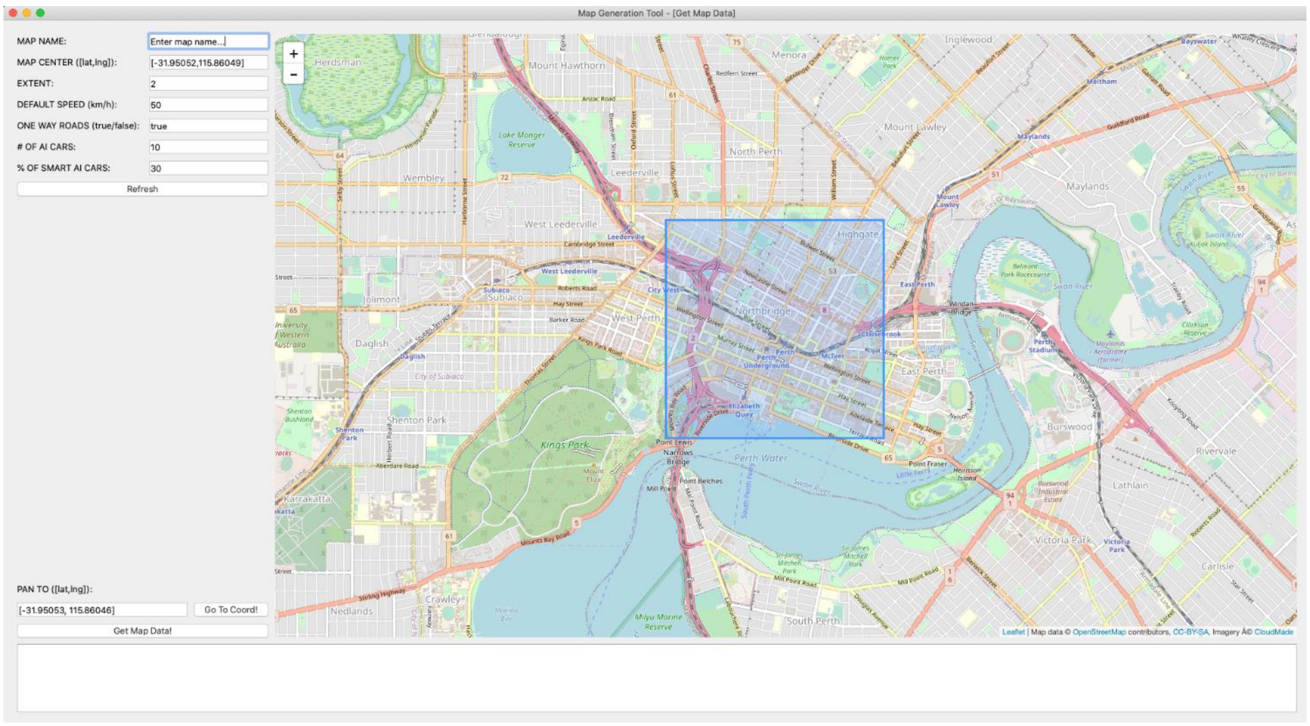


```

public class DriverText : MonoBehaviour
{
    public GameObject driverTextBox0;
    public GameObject driverText0;

    public GameObject driverTextBox1;
    public GameObject driverText1;

    public GameObject driverTextBox2;
    public GameObject driverText2;

    public static string[] drivers =
    {
        "Name: Mathew Smith \n Gender: Male \n License plate: XGH12HH \n Driver’s license: 25239087 \n User rating: 3/5 \n Driving Experience: 7 years \n Rideshare dri
        "Name: Jake Smith\n Gender: Male \n License plate: ZGH63JH \n Driver’s license: 22419227 \n User rating: 4/5 \n Driving Experience: 6 years \n Rideshare driver
        "Name: Samantha Smith \n Gender: Female \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare
        "Name: Mary Smith\n Gender: Female \n License plate: XGH12HH \n Driver’s license: 25239087 \n User rating: 3/5 \n Driving Experience: 7 years \n Rideshare driv
        "Name: Remy Smith \n Gender: Male \n License plate: ZGH63JH \n Driver’s license: 22419227 \n User rating: 4/5 \n Driving Experience: 6 years \n Rideshare drive
        "Name: Will Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driver
        "Name: Cosmin Smith\n Gender: Male \n License plate: XGH12HH \n Driver’s license: 25239087 \n User rating: 3/5 \n Driving Experience: 7 years \n Rideshare driv
        "Name: Christine Smith\n Gender: Female \n License plate: ZGH63JH \n Driver’s license: 22419227 \n User rating: 4/5 \n Driving Experience: 6 years \n Rideshare
        "Name: Anthony Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare dri
        "Name: Paul Smith\n Gender: Male \n License plate: XGH12HH \n Driver’s license: 25239087 \n User rating: 3/5 \n Driving Experience: 7 years \n Rideshare driver
        "Name: Jada Smith\n Gender: Female \n License plate: ZGH63JH \n Driver’s license: 22419227 \n User rating: 4/5 \n Driving Experience: 6 years \n Rideshare driv
        "Name: Christiano Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare
        "Name: Amal Smith\n Gender: Female \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driv
        "Name: Cory Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driver
        "Name: Dennis Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driv
        "Name: Leon Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driver
        "Name: Jenny Smith\n Gender: Female \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare dri
        "Name: Ahmed Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare drive
        "Name: Nick Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driver
        "Name: Ben Smith\n Gender: Male \n License plate: DDH12TH \n Driver’s license: 44239011 \n User rating: 5/5 \n Driving Experience: 7 years \n Rideshare driver

```

Figure 3 - String array labelled ‘drivers’

2.1.1 MapTool and Pathing

The MapTool utility (figure 4) can be used to create new maps and pre-generate the regular traffic’s paths from spawn to destination.

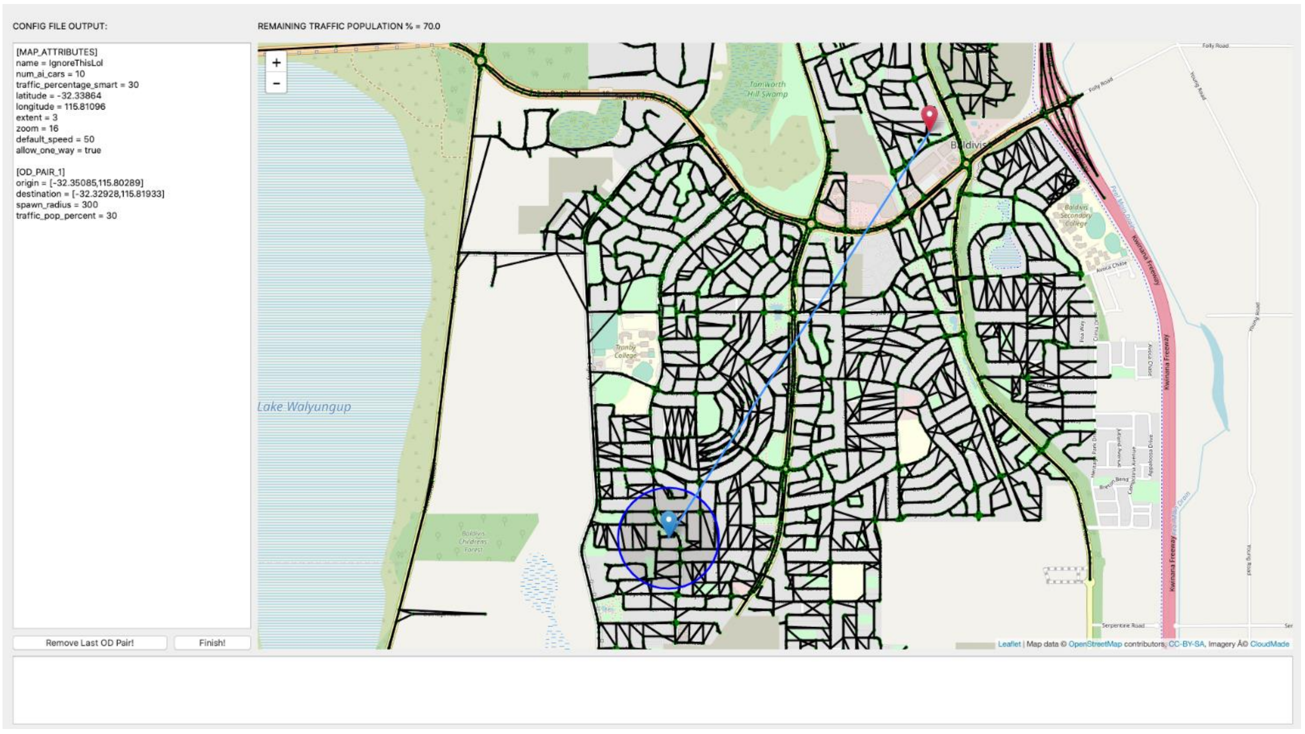


MAP NAME: Enter map name...
MAP CENTER ([lat,lng]): [-31.95052,115.86049]
EXTENT: 2
DEFAULT SPEED (km/h): 50
ONE WAY ROADS (true/false): true
OF AI CARS: 10
% OF SMART AI CARS: 30
PAN TO ([lat,lng]): [-31.95053, 115.86046]

Figure 4 – MapTool area selection

The first screen of MapTool allows you to select the area you want the map to cover and general background traffic options. The admin can move the map selection by panning around the map with the mouse or can enter the latitude and longitude coordinates of the area desired. Besides area selection the other options available on this screen are:

- Map Name
- Extent
    - Determines the size of the selection area
        - One-way roads
    - Determines whether the selected roads are one or two way
- # of AI Cars
    - The number of background traffic vehicles to spawn
    - Includes regular traffic and smart traffic
- % of Smart AI Cars
    - The percentage of “# of AI Cars” that will be smart traffic


CONFIG FILE OUTPUT:

REMAINING TRAFFIC POPULATION % = 70.0

```
[MAP_ATTRIBUTES]
name = IgnoreThisLol
num_ai_cars = 10
traffic_percentage_smart = 30
latitude = -32.33864
longitude = 115.81096
extent = 3
zoom = 16
default_speed = 50
allow_one_way = true

[OD_PAIR_1]
origin = [-32.35085,115.80289]
destination = [-32.32928,115.81933]
spawn_radius = 300
traffic_pop_percent = 30
```

Remove Last OD Pair!
Finish!
Lake Walyungup
Baldivis
Leaflet | Map data © OpenStreetMap contributors, CC-BY-SA, Imagery Â© CloudMade

Figure 5 – MapTool OD pair definitions

The second MapTool screen (figure 5) lets the admin choose the origin-destination (OD) pairs for the background traffic. These pairs determine the routes along which the background traffic will flow, which allows simulation of major roads. To define an OD pair click anywhere on the map to drop a blue pin representing the origin. You will be asked to provide a spawn radius, which defines the area around the origin the traffic can spawn. You will then be asked to enter the percentage of background traffic that should use this OD pair. The percentage of traffic still available to work with it stated above the map. After entering these values click on the map again to drop a red pin, the destination for this OD pair. If you make a mistake you can click the 'Remove Last OD Pair' button.

Note you do not have to define OD pairs for all the traffic, any unassigned traffic will be spawned randomly across the map and be given random destinations.

When all the desired OD pairs have been defined, click finish to export the map, pre-calculate the regular traffic paths, and generate the start and end nodes for the smart traffic, whose pathing is done on the fly in the game to take into account congestion.

When the paths have finished calculating the map will be exported into a folder with the name defined on the first MapTool screen, which can be moved into the game assets folder to make it available.

2.1.2 Configuration File Parameterization

Once a map has been generated, it can be set in the game through a configuration (config) file, allowing with several other metadata parameters. This file can be found in the path 'Resources/config.ini' and edited using any text editing program. The general structure of the config file is as follows:

[“Section1”]

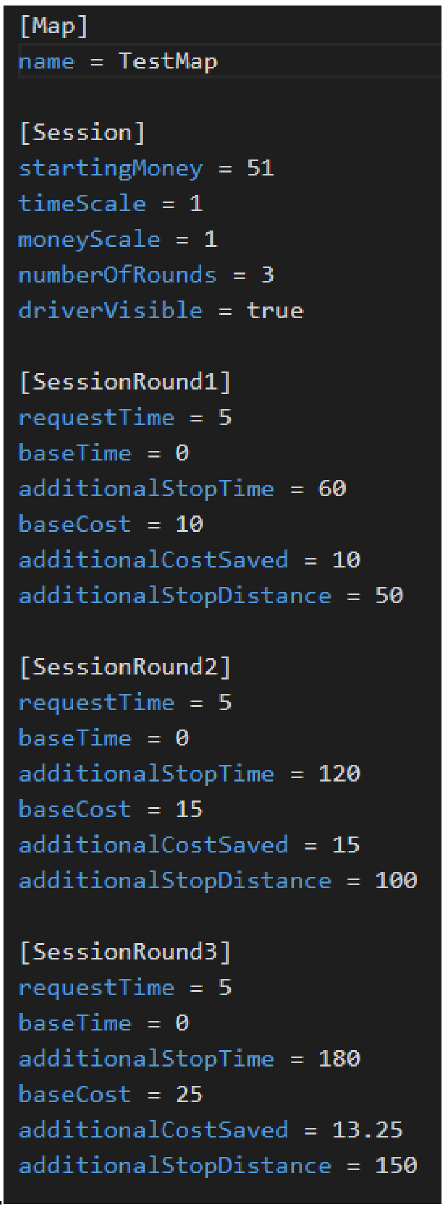
Variable1 = Value1

Variable2 = Value2

….

[“Section2”]
….

A section defines a grouping of related variables, each variable contains a value. Figure 6 demonstrates a working config file for a map called “TestMap”.



```
[Map]
name = TestMap

[Session]
startingMoney = 51
timeScale = 1
moneyScale = 1
numberOfRounds = 3
driverVisible = true

[SessionRound1]
requestTime = 5
baseTime = 0
additionalStopTime = 60
baseCost = 10
additionalCostSaved = 10
additionalStopDistance = 50

[SessionRound2]
requestTime = 5
baseTime = 0
additionalStopTime = 120
baseCost = 15
additionalCostSaved = 15
additionalStopDistance = 100

[SessionRound3]
requestTime = 5
baseTime = 0
additionalStopTime = 180
baseCost = 25
additionalCostSaved = 13.25
additionalStopDistance = 150
```

Figure 6 – Config File Structure

2.1.2.1 Loading a Map

Of main importance is “name” in the [“Map”] section. This value refers to the name of a generated map from Section *2.1.1*. Changing this value to match the file name of a generated map loads that map into the game once it is run.

2.1.2.2. Other Parameters.

The [“Session”] section denotes variables that affect the game session. Below is a list of parameters and available value ranges:

- **startingMoney** - the amount of money allotted to the player at the start of the session. Can be any numeric above 0.
- **timeScale** – the multiplier used to modify the base time of the session. Can be any numeric.
- **moneyScale** – the multiplier used to modify money operations in the session. Can be any numeric.
- **numberOfRounds** – the number of rounds that took place in the session. This value should always be set to 3.
- **driverVisible** – allows a driver to be visible in the game if the Ride-Share or Drive yourself modes are selected. This value should always be true.

The [“SessionRound’N’”] section (where ‘N’ is an integer from 1 to 3) denotes variables that affect an individual round ‘N’. Below is a list of parameters and available value ranges:

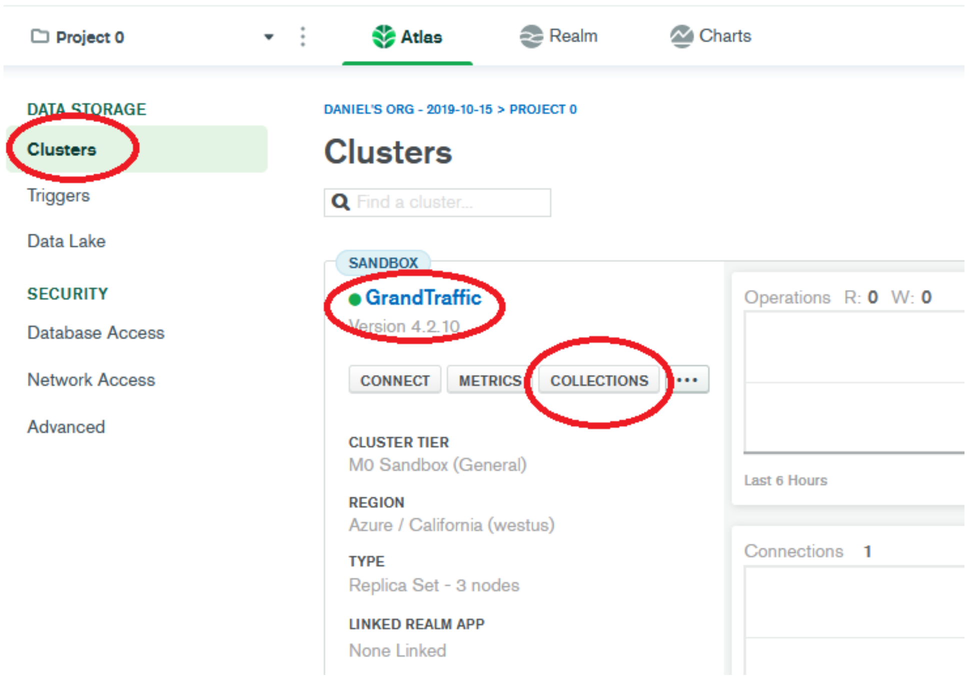
- **requestTime**- the time in seconds before the mobile phone popup appears for the player this round. Can be any integer above 0.
- **baseTime** – the internal counter used by the game to track the round duration. This value should always be set to 0.
- **additionalStopTim**e – the additional time in seconds the detour is claimed to take. This should be set proportional to additionalStopDistance and can be any integer above 0.
- **baseCost** – the base cost attached to selecting either the Ride-Share or Automated Vehicle modes of transport. Can be any numeric between 0 and startingMoney.
- **additionalCostSaved** – the amount by which the baseCost is reduced due to picking up a passenger. Can be any numeric between 0 and baseCost.
- **additionalStopDistance** - the minimum distance in meters that the additional stop must be from the next node, i.e. how far of a detour from the otherwise regular path is needed before the player can save money. Can be any integer above 0, but 50-500 has been found to work best.

2.1.3. Database

Once a player has completed the game, their gameplay data is automatically sent to the cloud-based database. This database is a MongoDB server hosted on the Atlas platform found at https://www.mongodb.com/cloud/atlas. To access this, the administrator user must have an account added to the Atlas project. Subsequent sections detail how to login to the database, as well as view and manipulate the data.

2.1.3.1 Accessing the Database

1. Go to https://account.mongodb.com/account/login
2. Login with the appropriate credentials
3. Navigate to *Clusters* on the left under *Data Storage* as seen in figure 7. Under *GrandTraffic*, click *Collections.*



CLUSTER TIER
M0 Sandbox (General)

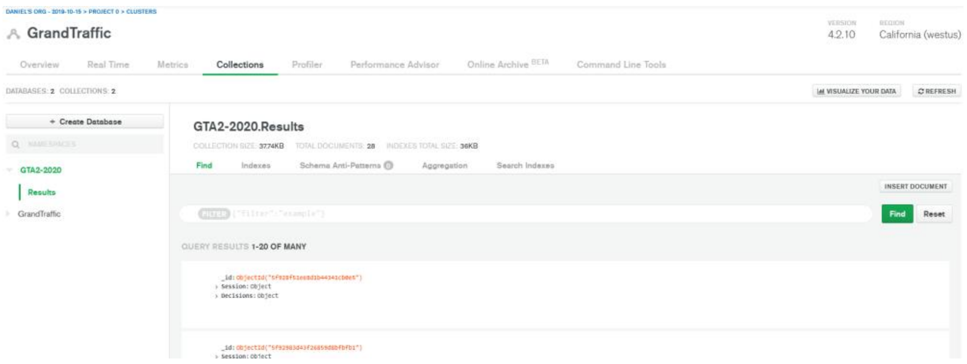
REGION
Azure / California (westus)

TYPE
Replica Set - 3 nodes

LINKED REALM APP
None Linked

Figure 7 – MongoDB Clusters

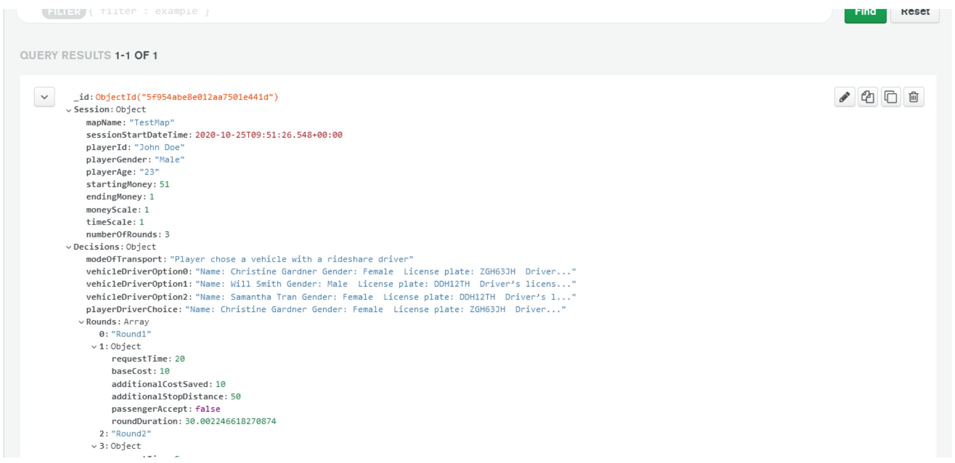
4. This view shows the databases available in the *GrandTraffic* environment. The gameplay data is stored in the *Results* collection of the *GTA2-2020* database (figure 8).



```
_id: ObjectId("5f928fb1ee8d1b44341cb0e5")
> Session: Object
> Decisions: Object

_id: ObjectId("5f92983d43f26859d8bfbfb1")
> Session: Object
```

Figure 8 – MongoDB results collection

5. Inspecting the Data
    a. In the Query Results view, the admin can view, update and delete the records within the collection (figure 9). A detailed summary of each element within a record can be found in Section *2.1.3.2 Database Field Information.*



```
_id: ObjectId("5f954abe8e012aa7501e441d")
v Session: Object
    mapName: "TestMap"
    sessionStartDateTime: 2020-10-25T09:51:26.548+00:00
    playerId: "John Doe"
    playerGender: "Male"
    playerAge: "23"
    startingMoney: 51
    endingMoney: 1
    moneyScale: 1
    timeScale: 1
    numberOfRounds: 3
v Decisions: Object
    modeOfTransport: "Player chose a vehicle with a rideshare driver"
    vehicleDriverOption0: "Name: Christine Gardner Gender: Female  License plate: ZGH63JH  Driver..."
    vehicleDriverOption1: "Name: Will Smith Gender: Male  License plate: DDH12TH  Driver’s licens..."
    vehicleDriverOption2: "Name: Samantha Tran Gender: Female  License plate: DDH12TH  Driver’s l..."
    playerDriverChoice: "Name: Christine Gardner Gender: Female  License plate: ZGH63JH  Driver..."
  v Rounds: Array
      0: "Round1"
    v 1: Object
        requestTime: 20
        baseCost: 10
        additionalCostSaved: 10
        additionalStopDistance: 50
        passengerAccept: false
        roundDuration: 30.002246618270874
      2: "Round2"
    v 3: Object
```

Figure 9 – MongoDB database records

   b. Click the dropdown arrow to expand the record. Individual objects can also be expanded via the dropdown arrow next to the object.
   c. Elements within the record can be edited by double clicking on the element.
   d. Records can be deleted using the bin icon on the right of the record.
   e. Records can be filtered via the *FILTER* found above the record results. This allows for querying data based on specific fields. For example, the query *{"Session.playerId" : "TestPlayerId"}* returns all records that contain *“TestPlayerId”* in the *playerId* element of the *Sessions* object. Further details on MongoDB querying can be found under the Help section on the dashboard or online via https://docs.mongodb.com/compass/master/query/filter.

2.1.3.2. Database Field Information

The database stores the following information:

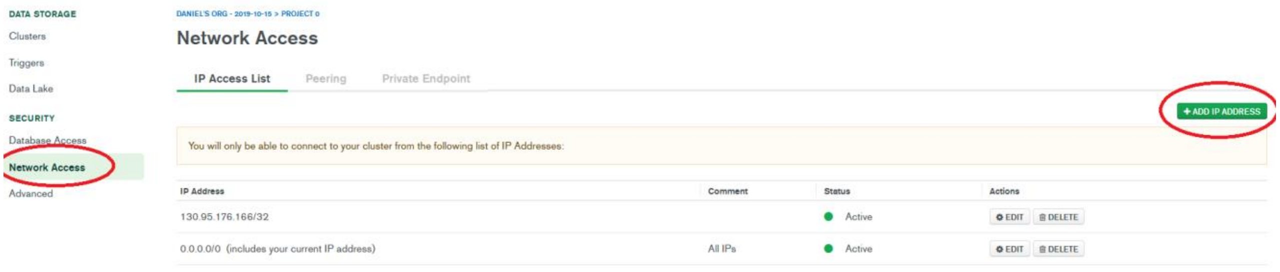
- *Session (Object)* - this object contains information about the game session itself.
    - **mapName –** the name of the map render loaded into the game session
    - **SessionStartDateTime** – the datetime that the session began in UTC.
    - **PlayerId** – the unique identifier of the player in the session.
    - **PlayerGender** – the gender of the player in the session.
    - **PlayerAge** – the age of the player in the session.
    - **startingMoney** – the amount of money allotted to the player at the start of the session.
    - **endingMoney** – the final amount of money the player ends the session with.
    - **moneyScale** – the multiplier used to modify money operations in the session.
    - **timeScale** – the multiplier used to modify the base time of the session.
    - **numberOfRounds** – the number of rounds that took place in the session
- *Decisions (Object)* - this object contains information about the choices the player makes throughout the game session
    - **modeOfTransport** – Which choice of transport the player made, whether it was Ride-Share, Drive yourself, or Automated Vehicle
    - **vehicleDriverOption1** – Information on the first driver the player is presented with in choosing their ride-share driver.
    - **vehicleDriverOption2** - Information on the second driver the player is presented with in choosing their ride-share driver.
    - **vehicleDriverOption3** - Information on the third driver the player is presented with in choosing their ride-share driver.
    - **playerDriverChoice** – The driver the player chooses from the 3 options presented.
    - *Rounds (Object)* - these objects contain information about the round parameters, as per the config file. There can be numerous rounds depending on the config file input.
        - **requestTime** – the time in seconds before the mobile phone popup appears for the player this round
        - **baseCost** – the base cost attached to selecting either the Ride-Share or Automated Vehicle modes of transport

- **additionalCostSaved** – the amount by which the baseCost is reduced due to picking up a passenger
- **additionalStopDistance -** the minimum distance that an intermediate node can be from the next node, i.e. how far of a detour from the otherwise regular path is needed before the player can save money
- **passengerAccept** - Whether the player accepts an additional passenger during this round.
- **roundDuration** – the total time taken for the round in seconds

2.1.3.3. Granting Permissions

MongoDB Atlas allows for whitelisting connections to restrict access to allowed parties. To manage the set of allowed connections:

1. Navigate to Network Access on the left side.
2. Adding a new IP address can be done via the ADD IP ADDRESS button
3. Editing or deleting addresses can be done via the buttons on the right of the interested IP address.



You will only be able to connect to your cluster from the following list of IP Addresses:

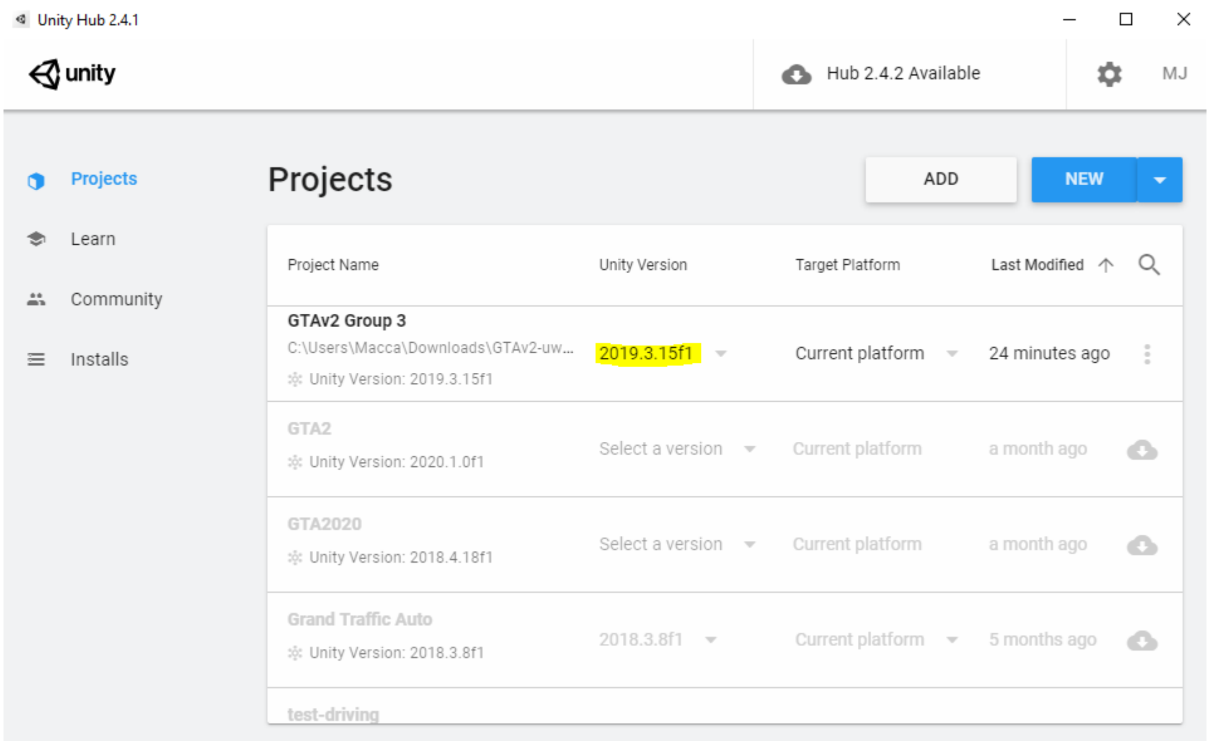
IP Address	Comment	Status	Actions
130.95.176.166/32		● Active	EDIT DELETE
0.0.0.0/0 (includes your current IP address)	All IPs	● Active	EDIT DELETE

Figure 10 – MongoDB Network Access

2.1.4. Unity Hub/Dashboard

This project is cloud hosted through Unity, and it was worked on using the Unity Collab cloud service. Since the code is stored here it is important to understand how to access the project from these services.

Firstly, is the Unity Hub, the location in which the project is provided to the user from the cloud. A user with access to the project in the respective organisation can access the project simply by opening the Unity Hub and selecting the correct Unity version to open the project. This can be seen below in figure 11.



Project Name	Unity Version	Target Platform	Last Modified
GTAv2 Group 3 C:\Users\Macca\Downloads\GTAv2-uw... Unity Version: 2019.3.15f1	2019.3.15f1	Current platform	24 minutes ago
GTA2 Unity Version: 2020.1.0f1	Select a version	Current platform	a month ago
GTA2020 Unity Version: 2018.4.18f1	Select a version	Current platform	a month ago
Grand Traffic Auto Unity Version: 2018.3.8f1	2018.3.8f1	Current platform	5 months ago
test-driving			

Figure 11 – Unity Hub

Once these are selected the project will open in the Unity editor, and all the project code can be found. Next is the cloud stored location for the project. A user can go the Unity Hub, click their initials at the top right and select 'Manage Organizations', which will open in a browser. Next, select the organisation the project is a part of and then select 'Projects' on the left. The project will be visible there, with all the cloud build statistics and analytics. From the organisation the user can also manage the seats for Unity Collab and add users to the organisation to access the project, which is helpful for future work.

2.2 Players

2.2.1 Log in Scene

The first scene presented to the players is the log in screen (figure 12). This screen prompts the user to input three fields:

- Player ID - unique identifier for the player in the session
- Gender – the player’s gender
- Age – the player’s age

These parameters are collected for research purposes as they provide the administrator (researcher) information on the choices made in the game by different demographics. Once filled in, the player can click the *'Play'* button to move to the next screen of the game (section 2.2.2).

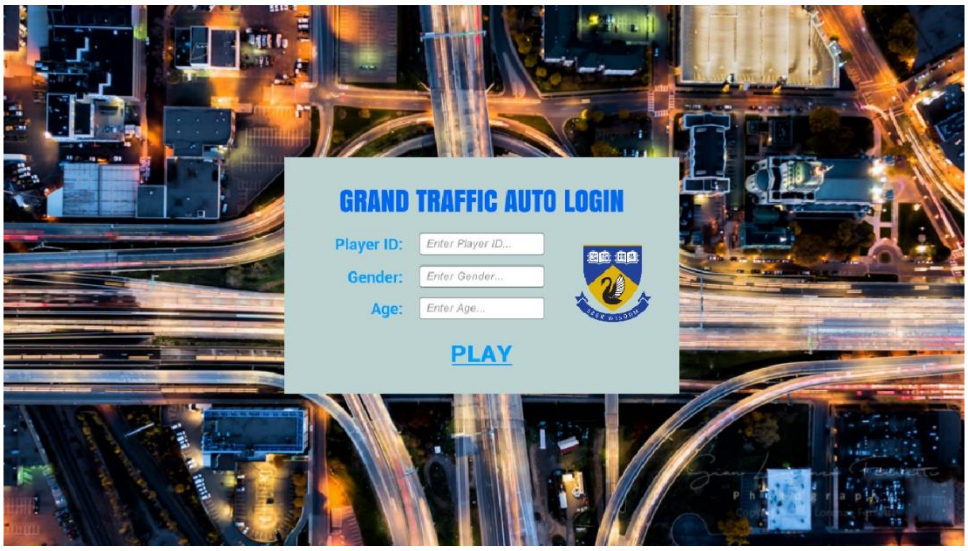


Figure 12 – Log in screen

2.2.2 Menu Scene

After logging in, the player will be taken to a game scene that displays menu 1 shown in figure 13 which presents the mode of transportation choices available. The player must press one of the confirm buttons to move on to the next scene.

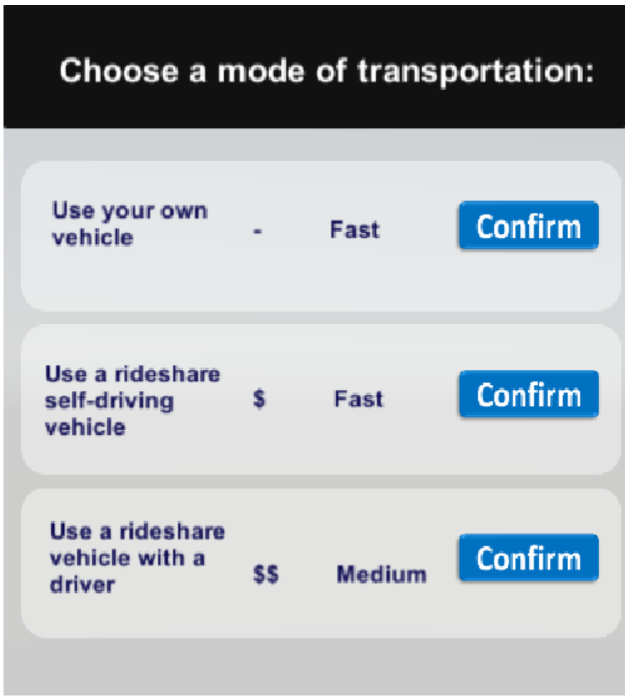

Choose a mode of transportation:
Use your own vehicle
-
Fast
Confirm
Use a rideshare self-driving vehicle
$
Fast
Confirm
Use a rideshare vehicle with a driver
$$
Medium
Confirm

Figure 13 – Menu 1

a. If the player decides to use their own vehicle then they will be taken directly to the main game scene (section *2.2.3*)
b. If the player decides to use a rideshare self-driving vehicle then they will be taken directly to the main game scene (section *2.2.3*)
c. If the player decides to use a rideshare vehicle with a driver then they will be taken to a scene that displays the menu visible in figure 14.

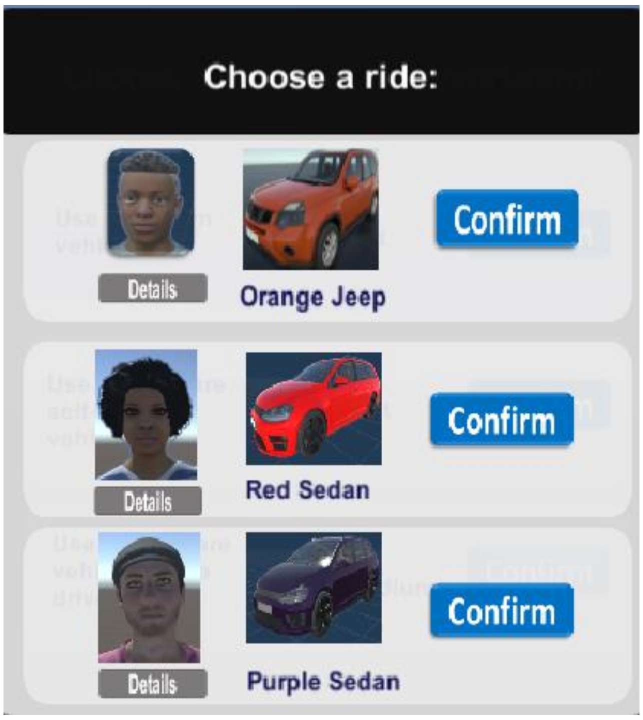
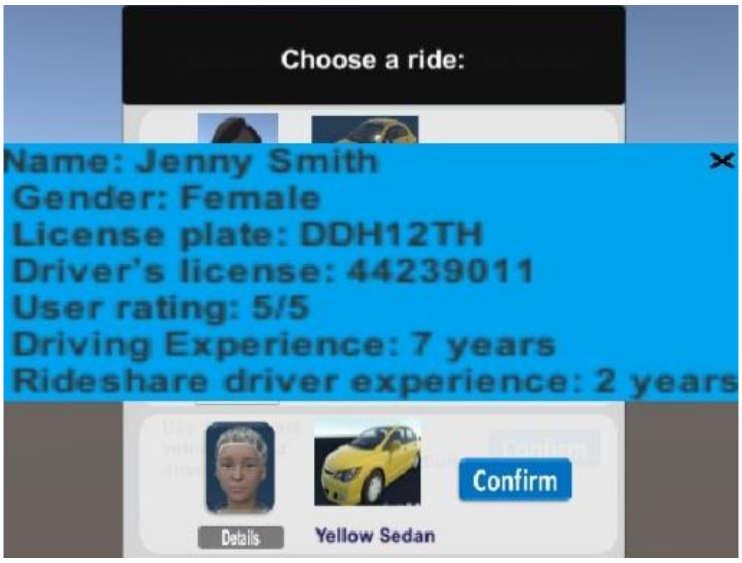


Figure 14 – Menu 2

The player must select a driver and is presented with three options. There are 20 driver options in total and every time the game is run, 3 options are randomly presented to the player. They will be able to see the driver's picture, vehicle, vehicle description and a button labelled details. The details button presents the player with information regarding the driver (figure 15). This information is presented to help the player make a choice on which driver to select. Once the player makes a choice, they must press the confirm button to move on to the main game scene (section 2.2.3).



Name: Jenny Smith
Gender: Female
License plate: DDH12TH
Driver's license: 44239011
User rating: 5/5
Driving Experience: 7 years
Rideshare driver experience: 2 years

Figure 15 – UI text box of a driver's information

2.2.3 Main Game Scene

After making a choice in section *2.2.2*, the player will be presented with the main game scene. The camera presents the player with a first-person point of view of the scene. A minimap is displayed in the bottom left corner of the scene and displays a top view of the game world. The minimap mirrors the vehicle's movement throughout the game world. A full screen map can also be brought up by the player to view a larger section of the game world, shown in figure 16. This full screen map can be toggled by either clicking on the minimap or by pressing the *'m'* key. The player is also presented with information regarding their elapsed time, cost of the trip and the total amount of money the player currently has. This information can be found in the top right corner of the screen.

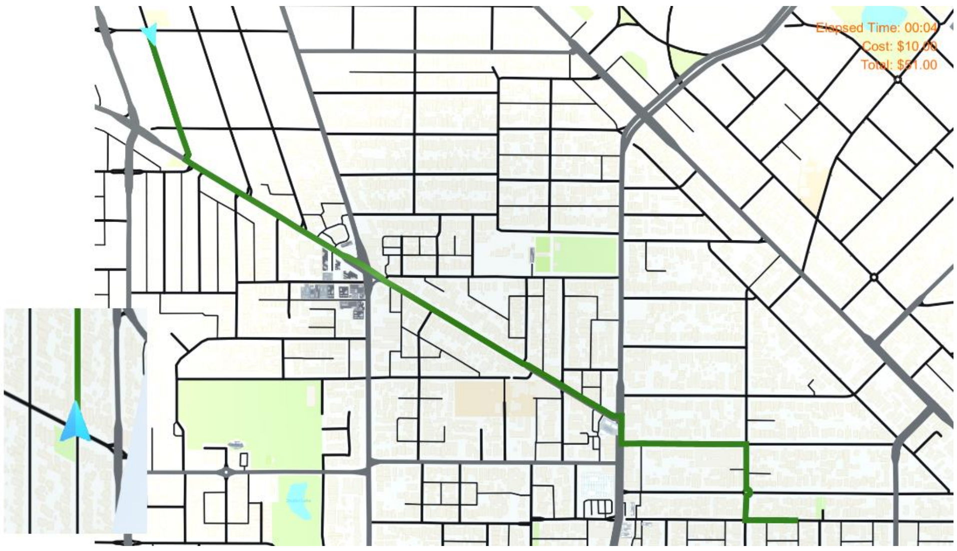


Figure 16 – Full screen map

The player has the capability to move the camera horizontally by moving their mouse cursor left and right. The vehicle will begin to move and travel along the predetermined path to the destination. The presence of a driver and the camera angle in the scene will differ based on the choices made by the players in section *2.2.2* and can be seen in the consecutive figures 17, 18 and 19:

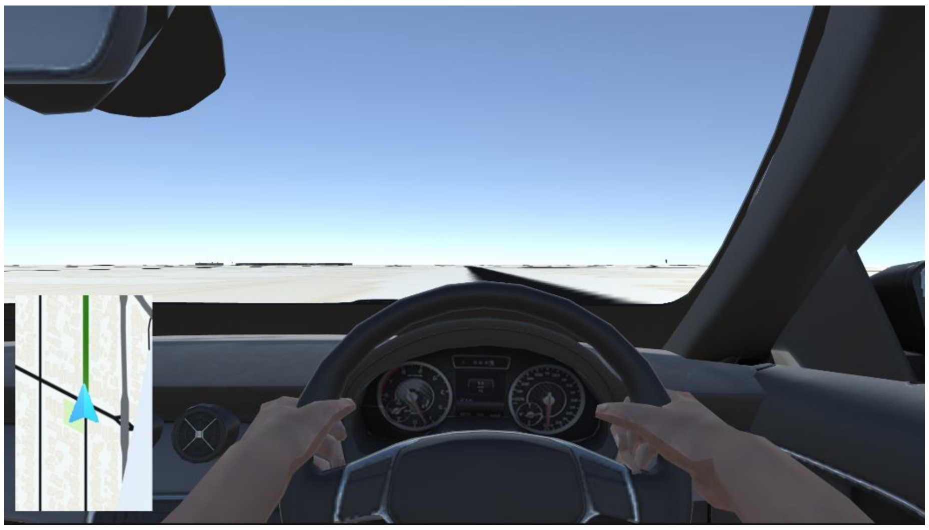


Figure 17 - Player decides to use their own vehicle

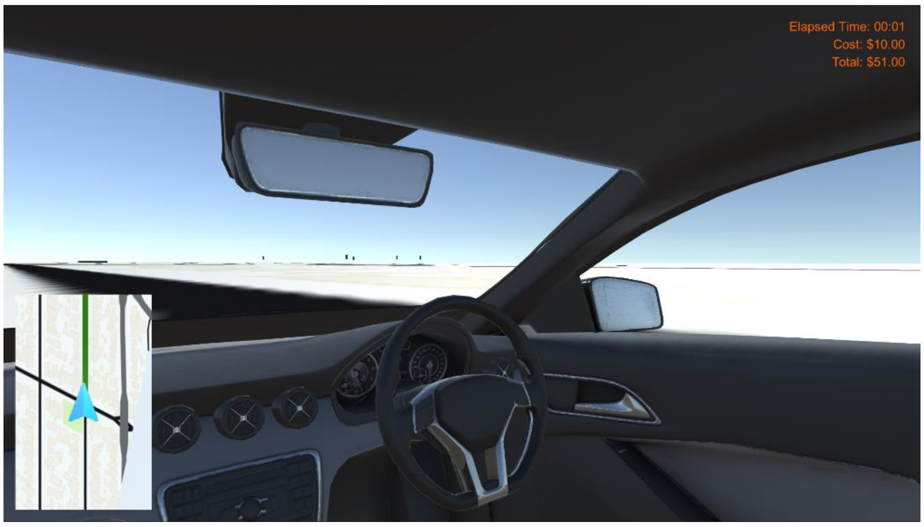


Figure 18 - Player decides to use a rideshare self-driving vehicle

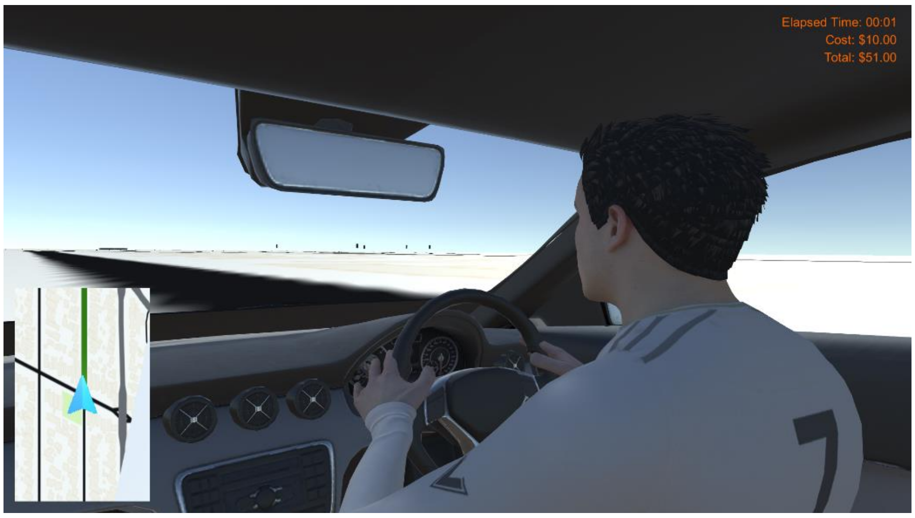
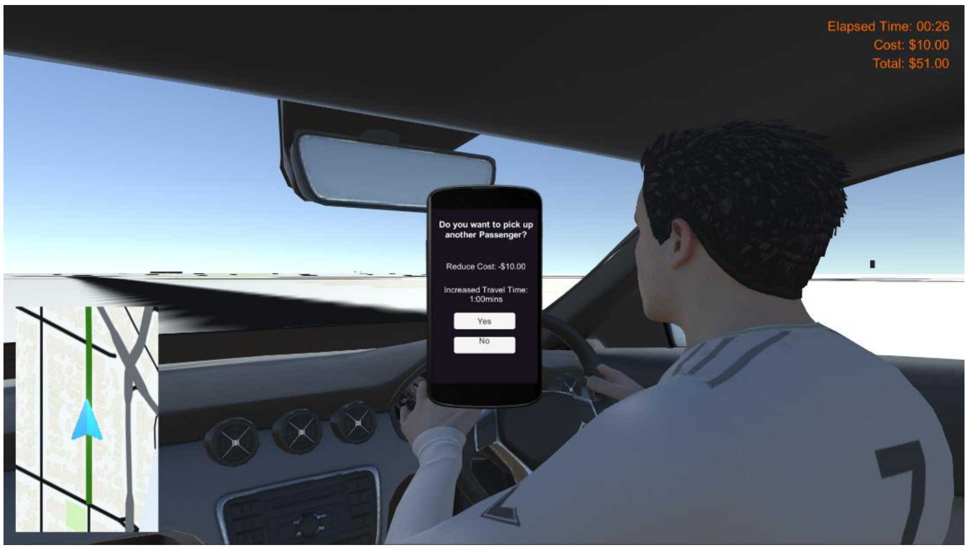


Figure 19 - Player decides to use a rideshare vehicle with a driver

If the player chooses either of the ridesharing options from section *2.2.2* (b or c) then throughout the main game they will be prompted to pick up other passengers during their trip. The prompt will appear as a mobile phone pop up as seen in figure 20. The mobile phone will display a notification asking the user whether they would like to pick up another passenger on their trip. The player makes a choice by pressing either the '*Yes'* or '*No'* button on the mobile. Agreeing to pick up another passenger increases the travel time of the trip by rerouting the player to pass through a 'pickup' location but decreases the cost.


Elapsed Time: 00:26
Cost: $10.00
Total: $51.00

Do you want to pick up another Passenger?

Reduce Cost: -$10.00

Increased Travel Time: 1:00mins

Figure 20 - Screenshot of mobile phone pop-up

In any of the game modes the player will face congestion in the form of AI controlled cars. Most of these cars follow pre-determined routes defined during the map creation, although a smaller percentage of them are smart traffic, which will update their routes during the game. The main impact these cars will have on the player is by causing the player to wait at intersections. There are two types of intersections that can occur; traffic light controlled and give-ways. Traffic light-controlled intersections, seen in figure 21, will allow one direction of traffic at a time, and will cycle through on a timer. All traffic lights are synchronised such that all lights in approximately the same direction will change together, e.g. all north facing lights would turn green at the same time. The give-way intersections are all of the other unmarked intersections, with cars giving way depending on a road priority and the direction they are turning. The road priority is based on which road a car is currently on, and is as follows;

Highest

1. A two-way road facing North-South
2. A two-way road facing East-West
3. A one-way road facing North-South
4. A one-way road facing East-West

Lowest

At a give-way intersection a car will give-way to cars on higher priority roads, and will obey usual road rules in regards to making turns, e.g. a car turning right will stop for cars in the oncoming lane. All cars, including the player car, will follow these intersection rules, resulting in the player's journey being impacted by other cars in a realistic manner.

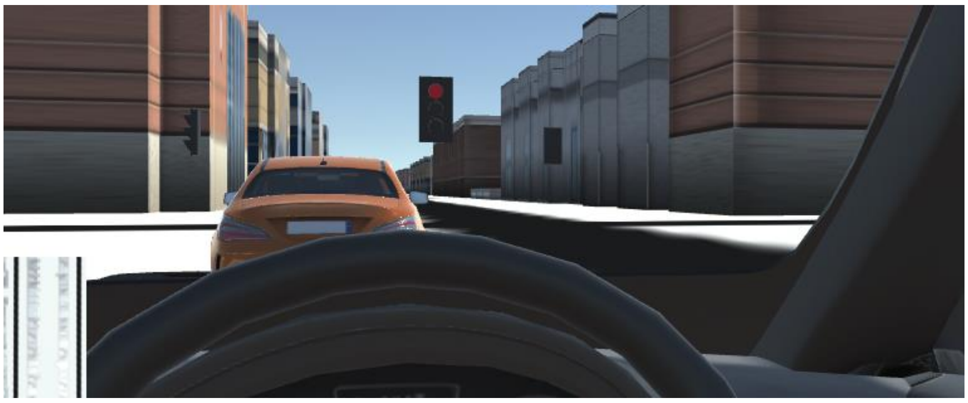


Figure 21 – A traffic light controlled intersection

3. Future Work

- Simulate passenger characters into the vehicles and calculate the time taken for a vehicle to travel along a path which can then be displayed and presented to the user in the menu screen.
- While the config file makes for a marked improvement over the game ID string used by the previous group, this method is not as accessible as having a user interface for the desired fields. Validation checking could be built into this interface; the current solution relies on astute usage by the admin.
- Dynamic Weather Conditions
    - Due to the time restrictive nature of the project and its reliance on the completion of the config file method, the weather implementation within the config file has been moved to the future work section. The Enviro Sky and Weather asset has been purchased and included within the Unity environment, so this should be used in any future works. The config file will need to include variables for each weather component as well as time, and these will need to be pulled into the game and applied with the Enviro Sky and Weather scripts. These will also need to be stored in the database as session information to allow for analysis later.
- Multiplayer
    - Currently, the game supports a single-player implementation; each session can only manage the one player. As the game was developed with this implementation as a basis, future work in producing other varied game modes would also be limited by this. The extension of the system to support

    multiplayer interactions would increase the range of development possibilities. From our initial research, we determined that this would be an arduous endeavour, given the time frame and the requirements for the current implementation. While it is possible to repurpose the game for multiplayer, we would recommend that this be a large focus for any undertaking groups, as it would require reworking much of the underlying codebase from the ground up (I.e. the game would need to be created with multiplayer in mind) as well as involving other known issues such as server hosting (as discussed below).

- Server Hosting
  - The current proposed server hosting solution was deemed to be insufficient for the deployment of the game. At best, the game would have been hosted on a small-scale, low resource virtual server which would have be partitioned for connecting users via Remote Desktop Protocol. The final game proved to be far too resource intensive, both in terms of computation and storage requirements, for the VM to support multiple concurrent connections. This restriction formed much of our decision making regarding final deployment, where we have opted for a hardware-based solution. While this was deemed satisfactory by the client, below we detail alternative solutions worth investigating by future groups:
    - Unity WebGL Deployment
      - Unity provides the WebGL framework that allows developers to build and publish content as JavaScript programs, using HTML5/JavaScript, WebAssembly, WebGL rendering and other web standards to run Unity content in a web browser WebGL. Unity WebGL provides many benefits. Resources can be hosted online, with accessibility chiefly determined by a user's download speed which is trivial for the UWA internet network. It also provides a platform-agnostic solution for accessing the game since it runs on a web browser. Extension of the current game to support WebGL involves developing and hosting a website; undertaking groups should be capable of performing this deployment and other requirements given adequate time.
    - VM Hosting
      - While VM hosting was deemed infeasible for the current game, it can still be explored as a potential solution, provided that the VM can support the game. This may involve upgrading the current VM or

sourcing a higher performance machine. From experience, negotiating with UWA IT for a larger-resource machine proved to be a tiresome, fruitless process; future groups should set aside time and expectation if pursuing this route. Based on the current implementation, an estimate of the game’s system requirements is detailed below in table 1. An adequate hosting solution should account for the multiple partitioning of a machine to service these requirements based on the number of desired concurrent users.

Table 1 - Game's system requirements

Component	**Minimum**	**Recommended**
CPU	Intel i3 or equivalent	Intel i5 or equivalent
RAM	4 GB	8 GB
Storage	10 GB	10 GB