SCHOOL OF COMPUTING (SOC)

Diploma in Applied AI and Analytics

ST1507 DATA STRUCTURES AND ALGORITHMS (AI)

2024/25 SEMESTER 1
ASSIGNMENT TWO (CA2)

~ Coffee Go Drone (shortest path finding algorithm) ~

Objective of Assignment

For this assignment you will have an opportunity to apply all that you have learnt with regards to data structures, algorithms, and object-oriented programming as to develop an application that can help a coffee delivery company study how their drones may be deployed to provide fast and efficient coffee delivery in the busy business district of Heatherthorn County.

Background

Coffee~Go~Drone is a startup company in Heatherthorn County that specializes in door-step coffee delivery for the busy office workers in the business district. Since last year they have started to deploy drones to deliver coffee at the office workers' doorsteps. Their drones are equipped with cameras and are currently remotely operated by Coffee~Go~Drone staff. Once an order for coffee comes in, a staff navigates the drone remotely from a pickup point and transports the coffee all the way to the client.

As the current way of working is labor intensive, and the staff may not always know the shortest route to the client, the company is keen to upgrade their drones with an autopilot feature. They hope that their drones can autonomously deliver the coffee by following a predefined path, and one that would also be the shortest path. That predefined path would then need to be calculated based on the city map and have to rely on an algorithm that calculates the shortest path possible between pickup and delivery location.

They are aware that it is risky to rely entirely on an autonomous drone that merely follows a pre-defined path, as there may be quite a few unforeseen circumstances that could cause troubles, such as unexpected road diversions, flight restrictions, traffic jams, and loss of GPS signals etc. Therefore, they want a hybrid system, whereby a staff can choose to navigate the drone either manually or letting it fly on autopilot where the circumstances would allow to do so.

Instructions and Guidelines:

1. This is a group assignment (you will work in pairs, only one group of 3 would be allowed if there is an odd number of total students).

2. This assignment accounts for **40%** of your final grade.

3. The assignment comprises a group component (70%) and an individual component (30%).

4. The submission date is **Wednesday 7 August 1:00 pm**.

5. The development will be carried out in Python using Anaconda.

6. The demonstrations/interviews will be conducted during the DSAA lessons in week 17/18. You are expected to explain your code and program logic. Take note that the interview is compulsory.

7. **50% of marks** will be deducted for submission of assignment within **ONE** calendar day after the deadline. **No marks shall** be awarded for assignments submitted **more than one day** after the deadline.

Warning: Plagiarism means passing off as one's own the ideas, works, writings, etc., which belong to another person. In accordance with this definition, you are committing plagiarism if you copy the work of another person and turning it in as your own, even if you would have the permission of that person.
Plagiarism is a serious offence, and if you are found to have committed, aided, and/or abetted the offence of plagiarism, disciplinary action will be taken against you. If you are guilty of plagiarism, you may fail all modules in the semester, or even be liable for expulsion.

Your job as team

As a team of AI programmers, you have been roped in to implement a simulator program for Coffee~Go~Drone. Your task is to develop a Python application that would allow the company to simulate the drones as they are going about delivering coffee in the business district of Heatherthorn County. The simulator program would allow the company to study their envisioned hybrid system whereby drones can be operated both manually and in autopilot mode.

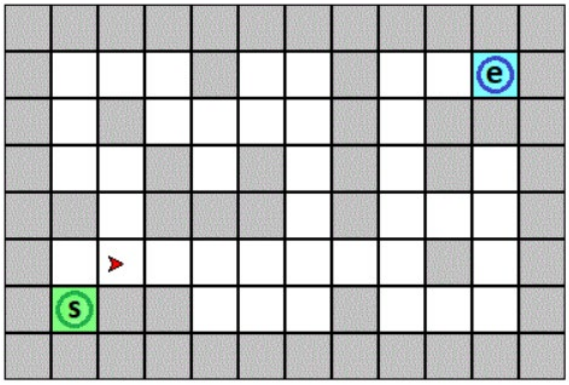
City map

Take note that the city is modelled as a grid with tiles. You are required to read the city map from a text file. You must follow the following conventions.

Character	Description of tile
X	*Building Tile (drone can't go here)*
.	*Road Tile (drone can fly here)*
s	*The start location of the drone*
e	*The end location to deliver the coffee*

Next is an example of a city map file (dimensions 8 rows by 12 columns), together with a screen shot of how the city map would then appear in the application.

```
XXXXXXXXXXXX
X...X..X..eX
X.X....X.XXX
X..X.X.X.X.X
XX.XXX.X...X
X........X.X
XsXX...X...X
XXXXXXXXXXXX
```



- Your application should allow for city maps up till at least dimensions of 16 rows by 24 columns.

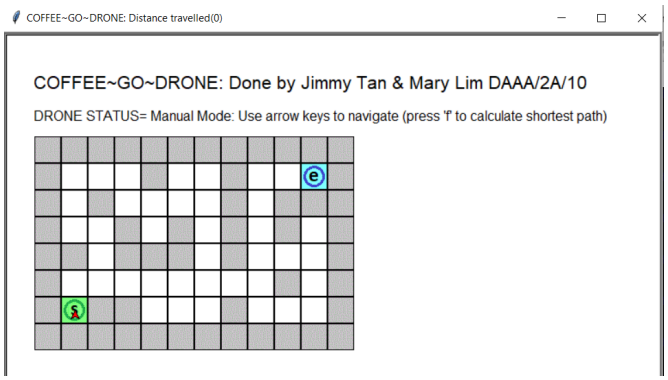
Drone Operations: autopilot mode

Below is a series of steps that illustrates a coffee delivery session that relies entirely on autopilot mode.

Step 1:

When we start the application, a window shows up displaying the city map that was specified when we started the application. Our drone will be located at the start position (the green tile at the left bottom) The drone is displayed as a red triangle (turtle icon). You must display a title on top of the window similarly as is shown below, but with the names and class of your team members. A status message displayed above the city map indicates that the drone is currently in manual mode, and as such it can be navigated manually with the arrow keys if the user wants so. In this case however, we will press 'f' as to start an autopilot session straight away.

Status message
Manual Mode: Use arrow keys to navigate (press 'f' to calculate shortest path)



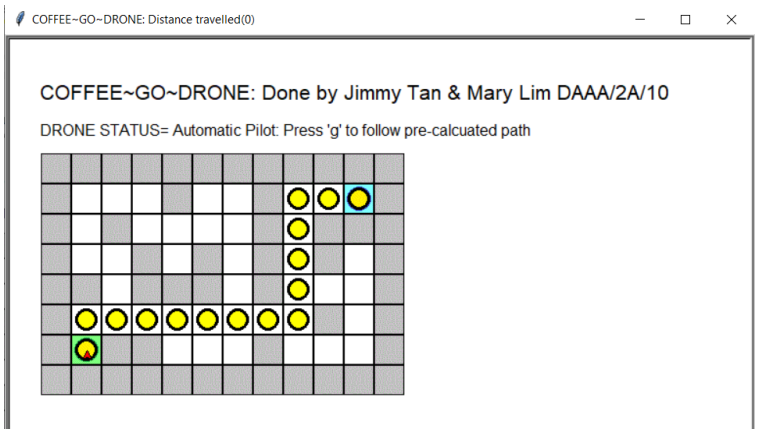
Take note, that the window title bar must display the company name, together with the current distance traveled as the drone moves along the path.

Step 2:

After we press 'f' the shortest path between the drone's current location and delivery point will be calculated. This pre-calculated path will be displayed with yellow circles that are plotted on the tiles that make up the path.

A status message displayed above the city map indicates that the drone is currently ready to take off in autopilot mode. Once we press 'g' the drone will take off.

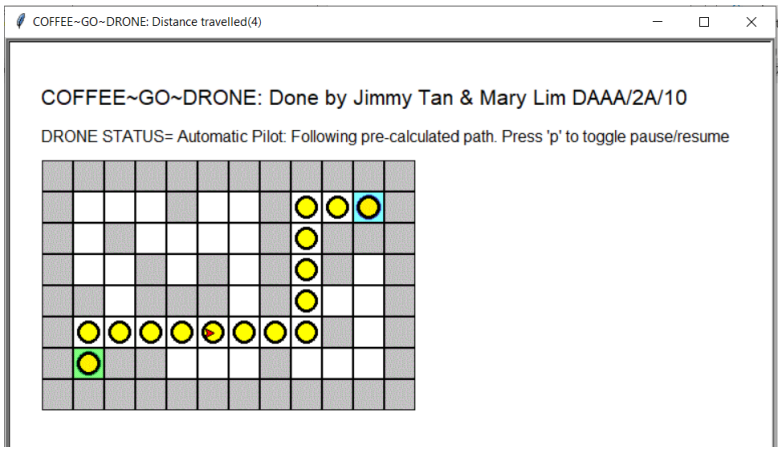
Status message
Automatic Pilot: Press 'g' to follow pre-calculated path.



Step 3:

After we press ‘g’ the drone will follow the pre-calculated path. A status message displayed above the city map indicates that the drone is currently following the path. The user may pause the flight anytime by pressing ‘p’. The drone will then be paused. By pressing ‘p’ again the flight resumes it’s flight.

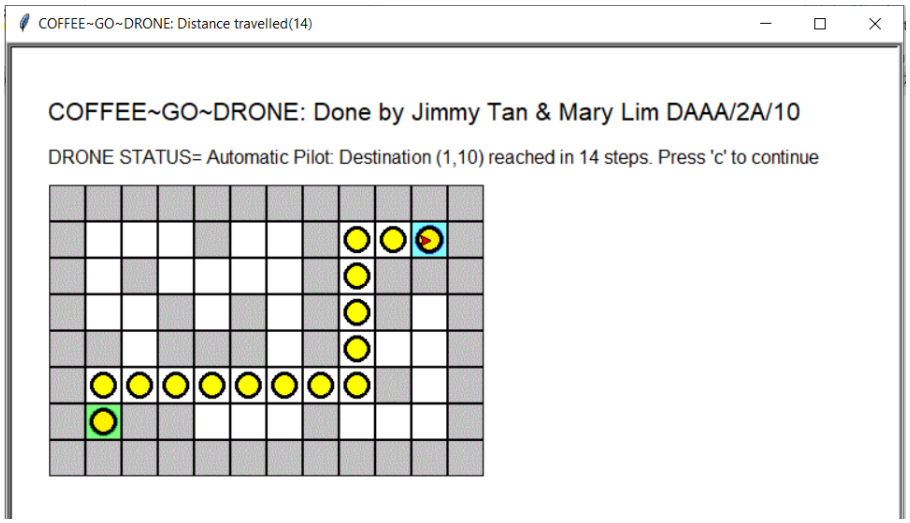
Status message
Automatic Pilot: Following pre-calculated path. Press ‘p’ to toggle pause/resume.



Step 4:

After the drone completes its flight, a status message will be displayed above the city map indicating the number of steps (tiles) that the Drone has traveled, as well as the location of destination tile. The user is reminded to press ‘c’ to continue operating the drone in manual mode.

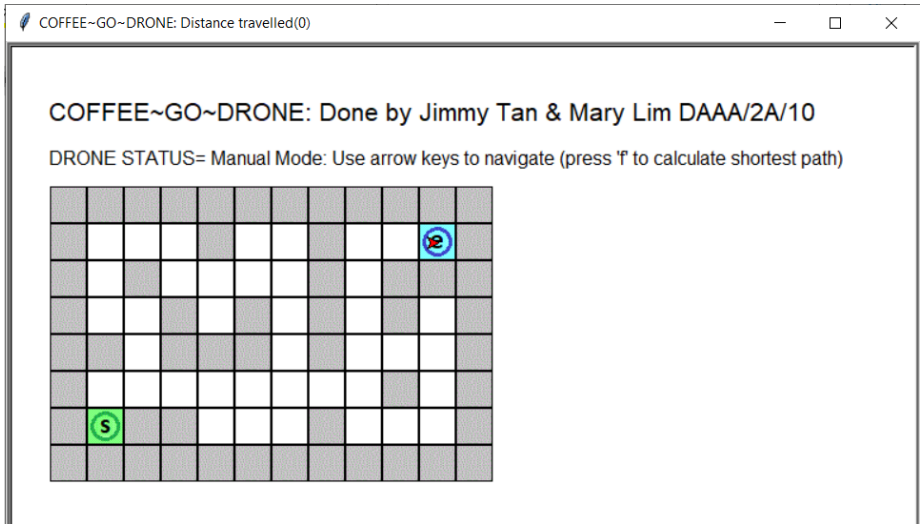
Status message
Automatic Pilot: Destination (1,10) reached in 14 steps. Press ‘c’ to continue.



Step 5:

After pressing ‘c’ the path will be cleared, and the drone will remain on the destination tile ready to be controlled manually.

Status message
Manual Mode: Use arrow keys to navigate (press ‘f’ to calculate shortest path)



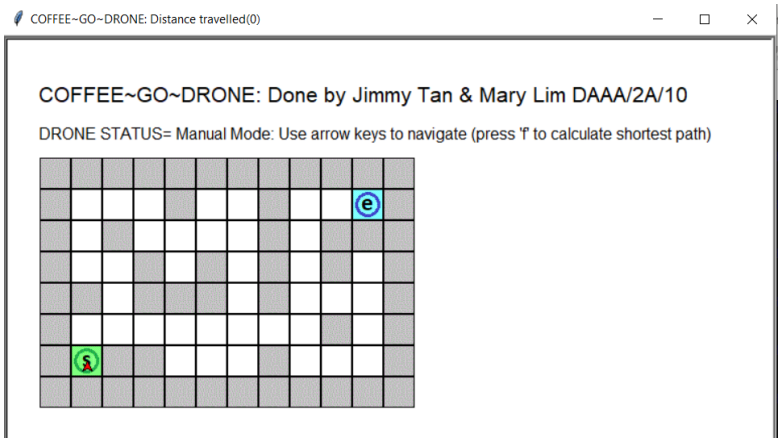
Drone operations: Manual mode combined with autopilot mode

Below is a series of steps that illustrates a coffee delivery session whereby the drone is initially controlled manually, and thereafter it continues in autopilot mode.

Step 1:

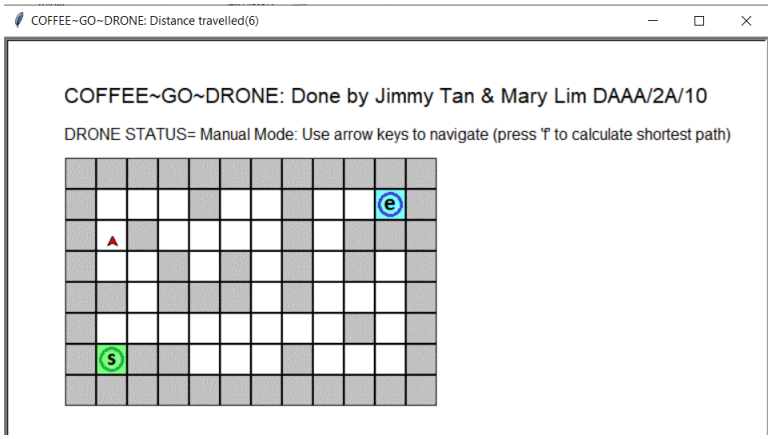
After we start the application, we may choose to operate the drone manually with the arrow keys.

Status message
Manual Mode: Use arrow keys to navigate (press 'f' to calculate shortest path)



Step 2:

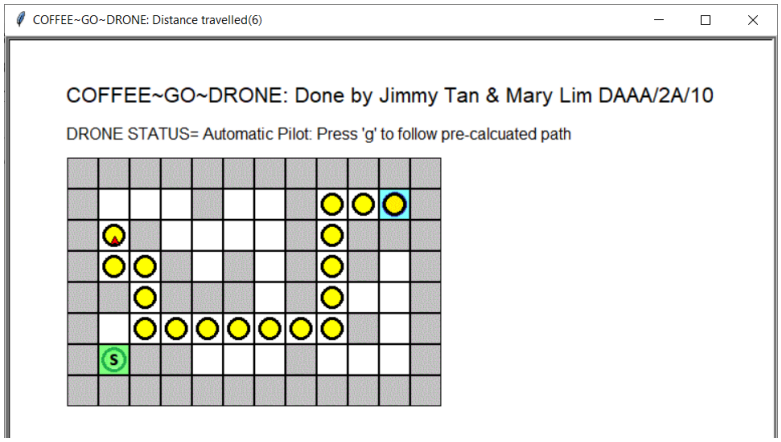
Using the arrow keys, we may navigate the drone to a specific location in the city from where we plan to start the autopilot. For instance, by following the next sequence of key presses, *Up, Right, Up, Up, Left, Up* it would bring the drone to the location as is shown below.



Step 3:

When we press ‘f’ the shortest path will be calculated between the drone’s current location and the coffee delivery location. Once we press ‘g’ the drone will take off.

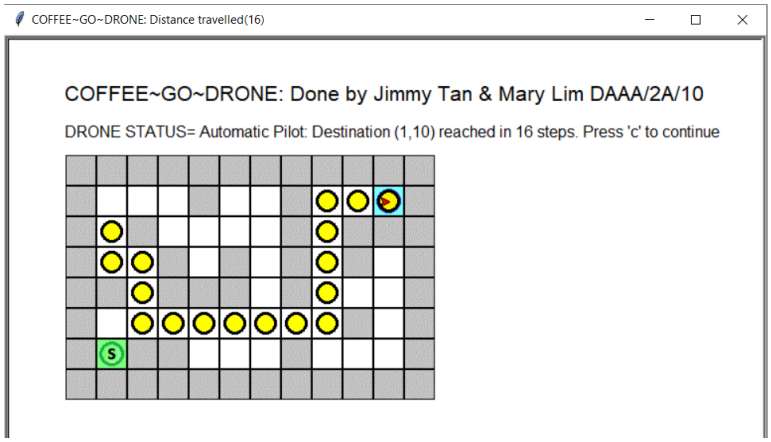
Status message
Automatic Pilot: Press ‘g’ to follow pre-calculated path.



Step 4:

After the drone completes its flight, the user may press ‘c’ to continue operating the drone manually.

Status message
Automatic Pilot: Destination (1,10) reached in 16 steps. Press ‘c’ to continue.



Requirements for Group Component (70%):

Starting the application.
A user needs to be able to start the application from the Anaconda Prompt as follows:

```
python main.py city001.txt
```

In this case the application would load the city map from the file 'city001.txt'

Drone operations.
A drone can either be navigated manually by the 4 arrow keys, or it can be instructed to follow a pre-defined path.

- As a drone goes from tile to tile, it may only move horizontally or vertically. A drone cannot move diagonally.
- Drones cannot fly over (or go through) tiles occupied with buildings.
- Drones are not allowed to venture outside the city.

Key Bindings.
Your application needs to support the following key bindings.

Key Press	Action	Description
'f'	find	Will calculate, and display, the shortest path from current drone location to coffee delivery location.
'g'	go	Will let the drone follow the current path.
'c'	continue	Will bring the drone back to manual operation after it has completed following a path.
'r'	reset	Resets the drone to the original start location, as when the application started. If there is any path set, it will be cleared.
'p'	pause/resume	Will toggle pause/resume a drone that is following a path.
'h'	show/hide	Will toggle show/hide of a path.
'left arrow'	move left	Will move the drone to the tile at left (if free) and only if drone is in manual operation.
'right arrow'	move right	Will move the drone to the tile at right (if free) and only if drone is in manual operation.
'up arrow'	move up	Will move the drone to the tile at top (if free) and only if drone is in manual operation.
'down arrow'	move down	Will move the drone to the tile at bottom (if free) and only if drone is in manual operation.
'q'	quit	Will quit the application

- You are required to design and write the Python application using an Object-Oriented approach (OOP). You should leverage on your knowledge that you have of encapsulation, inheritance, polymorphism etc.

- You may make use of Python's already built in data structures, such as list, tuple, dictionary and set. However, you should refrain from using the classes from the collection library. You are encouraged to write your own classes as to support the various data structures that you may need. Of course, you may refer to the lecture slides and lab tasks and expand further on those classes that we had previously developed in the tutorial and lab sessions.

- To run the application there should be no need to install additional libraries, other than those that ship already with Anaconda.

- Your application should not have to rely on any connection to the Internet.

- The OOP classes that you develop must all be placed in separate python files.

- The group will be requested to demonstrate the basic features of the application during the demonstration.

- Take note the group's demonstration should not exceed 15 minutes (including 5 minutes for Q&A).

Requirements for Individual Component (30%):

Each individual team member is required to implement two additional features to be added to the application. These two additional features will need to be demonstrated during the final presentation.

- The additional features will be graded on their technical sophistication and usability.
- Take note features within the same group must be different. So please check with your group member(s) first before embarking on implementing the extra features.
- Each group member must present/demonstrate his/her features (total presentation time not to exceed 10 minutes).
- Your PowerPoint slides must briefly describe what features you have implemented. You must include screen shots demonstrating the features in action. Please explain how the features work, and why they are useful. You are to include your Name, Class and Group Number in the first slide.
- You must submit the PowerPoint Slides (converted to pdf) together with a compulsory Peer Feedback (template will be provide on Brightspace) as well as a duly filled and signed Academic Integrity Form.

Final Deliverables

Your group's final deliverables must include:

(a) Group Report

A report (as pdf file) with a maximum **of 10** pages. This excludes cover page and the appendix with source listing and references. The report should contain:

a) Cover page with group number (your instructor will assign group numbers) names, ids, and class.
b) Description, and user guidelines, on how to operate your application (please include screen shots of your application in action).
c) Describe how you have made use of the Object-Oriented Programming (OOP) approach. You may elaborate of the classes that you have developed, and discuss on issues such as encapsulation, function/operator overloading, polymorphism, inheritance etc. Include a class diagram that displays the relation between the various classes you have developed.
d) Discussion on the data structures and algorithms you have developed (or used) for your application. You may discuss issues such as, the performance of the algorithms in terms of Big (O). Explain why you did develop certain data structures and explain why you deem these data structures suitable for the task(s) at hand. Include a table summarizing all the data structures that you have been using (those that you have developed and those already built in Python).
e) Include a summary of the challenges that the group has faced while developing the application (that should include both technical, as well as group-work challenges). Provide a summary of the key take aways and learning achievements that you have obtained from this project.
f) Include a clear description of the roles and contributions of each member in the team. Clearly state what each member has been responsible for, and what programming work has been carried out by each member.
g) **All** your python **source code listings** must be included as an appendix at the end of your report. You must clearly indicate in the source code listings who wrote what code. You may also include in the appendix those references from literature or internet that you may have consulted.

(b) Source Code

- You must submit all (*) the python files (.py files) that makes up your application. Ensure code is complete, and that it can run directly from the Anaconda Prompt.

  (*) Take note, that it includes the code for all the extra features that were coded by each team member as well.

Submission instructions

Group Submission:

- Group Leader to submit all the group deliverables (Source Code and Group Report) in the designated BrightSpace Drop Box.

- Important the source code must include all the extra features that were coded by the team members (so when we run the application, we can experience all the extra features that the team members have developed).

- You must submit it as one Zipped folder (RAR will not be accepted, only zip) whereby you label your submission as:

  CA2_GroupNumberClass.zip

  For example: *CA2_GR_10_DAAA_2A08.zip*

- Please ensure that you submit it by the stipulated deadline.

Individual Submission:

- Each individual Group Member is to submit his/her individual deliverables.

- Individual submission to include:

  - PowerPoint slides describing your two extra features (converted to pdf, maximum 8 slides)
  - Peer Feedback form
  - Academic Integrity Form (filled up & signed)

  Please ensure that you submit it in the designated Brightspace Drop Box for individual submissions.

- You must submit it as one Zipped folder (RAR will not be accepted, only zip) whereby you label your submission as:

  CA2_Final_ GroupNumberStudentNameClass.zip

  For example: *CA2_GR_10_JIMMY_TAN_DAAA_2B08.zip*

- Please ensure to submit it by the stipulated deadline.

Assessment Criteria

The group component of the assignment will be assessed based on the following criteria:

Assessment criteria (Group 70 %)	Marks awarded
GROUP COMPONENT (70 %)	
File IO & GUI: - Reads city map from file and displays it correctly. - Provides correct actions for various key-presses (f,g,c,r,p,h,q).	Max 10
Basic functionality of the application: Drone Manual Navigation: - Drone can be navigated manually with arrow keys. - Drone is being prevented to fly through (or over) buildings and will not be able to be steered out of the city. Drone Automatic Pilot: - Drone can find the shortest path (if there is one). - Drone follows the shortest path correctly.	Max 20
Programming techniques, robustness and readability of code: - Appropriate usage of classes and OOP technology. - Appropriate usage of data structures and algorithms. - Code is properly commented and neatly structured. - Application is free of crashes.	Max 20
Group Report: - The report follows the prescribed format. - The report is well written and comprehensive.	Max 10
Group's demonstration: - Group effectively demonstrates the basic features. - Group's ability to answer questions raised in Q&A.	Max 10
Group Total	**70**

The individual component of the assignment will be assessed based on the following criteria:

Assessment criteria (Individual 30 %)	Marks awarded
INDIVIDUAL COMPONENT (30 %)	
Extra Feature One - Technical sophistication. - Usability.	Max 10
Extra Feature Two - Technical sophistication. - Usability.	Max 10
Presentation & Demonstration: - PowerPoint slides. - Demonstration of features and Q&A.	Max 10
Individual Total	**30**

(*) Take note in case of group members with poor contribution to the group effort a multiplier may be applied (peer feedback may be taken in consideration).

~ End ~