Implementation

For Assessment 3 we adopted the project created by the Faceless Drones [1] and completed it, while fulfilling both our requirements from Assessment 2 [2] and the Faceless Drones' [3], with respect to the modifications (or lack of) presented in our requirements update section [4] and requirements table [5]. All additional code added to the project is commented with "`ADDITION BY WEDUNNIT`", and lines we changed have been commented "`UPDATED BY WEDUNNIT`" for reasons of clarity and simplicity. Additionally, in this report, we have abbreviated 'Non Player Character' to NPC.

3.1 Architecture

The architecture of our game is incredibly similar to the architecture of the Faceless Drones' project in Assessment 2 [4], with no significant changes to the structure or UML diagram [6]. There are several new methods and properties and three additional classes, described below.

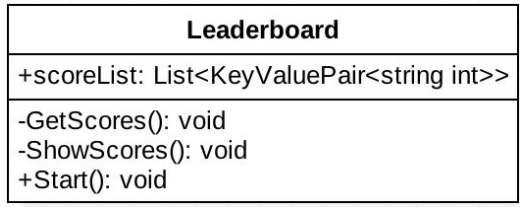
3.1.1 Additional methods and properties to existing classes

Method/ Property	**Purpose**
-Scenario.murdererName:String	A convenience property to store the name of the murderer.
-Scenario.GetRandomNonMurderingNPCName(string murdererName):String	Returns a random NPC's name. Used to generate more interesting verbal clues.
-Character.canBeQuestioned:bool	Property to allow a character to block the further questioning for our requirement 11a.
+Character.CanBeQuestioned():bool	Getter method for the above property. Needed to satisfy our requirement 11a.
+Character.AllowCharacterQuestioning():void	Sets Character.canBeQuestioned to true. Needed to satisfy our requirement 11a.
+Character.BlockCharacterQuestioning():void	Sets Character.canBeQuestioned to false. Needed to satisfy our requirement 11a.
-GameMaster.gameScore:float	Tracks the current game score for our requirement 13. A float type is used as small adjustments are made that would not be suitable for an integer variable.
+GameMaster.Update():void	Unity generated method for altering the score over time following our requirement 13a.
+GameMaster.Penalise(float penalty):void	Deducts 'penalty' from the current score following our requirement 13a.
+GameMaster.GainScore(float bonus):void	Adds 'bonus' to current score; called when clues are collected following our requirement 13.
+GameMaster.GetScore():int	Getter for the game's score; returns the integer cast of the float variable.
-LevelManager.greenTime:float	Property storing the remaining time for the score to be visually highlighted.
-LevelManager.isGreen:bool	Property denoting current status of score highlighting.
+LevelManager.OnScoreIncrease():void	When the player receives a bonus, this method sets the score to a bright green colour and resets its counter greenTime.

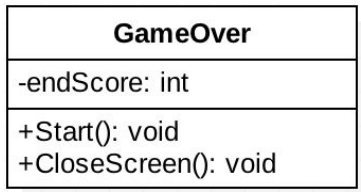
+LevelManager.Update():void	Unity generated method that decrements the greenTime counter on each frame. If the counter is 0 then the bright green highlighting is removed.
-InputManager.pauseIconPressed:bool	This property tracks the pause menu button status and is required as there are different input methods for accessing pause menu, improving upon our requirement 15.
+InputManager.ShowCluePanel(Item item):void	This method displays a panel containing information each time a clue item is collected.

3.1.2 New classes

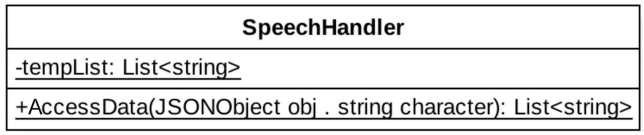
Two new standalone classes have been added to our implementation. One of these has been added to handle the leaderboard (requirement 14a) as shown. It does not interact with any other classes or scripts, although references to the Unity GameObjects representing text boxes are passed into Leaderboard through the Unity Editor.



Another of the additional classes is used to handle the end-of-game score reporting (requirement 13). This class interacts directly with the *GameMaster* class (using the `GameMaster.GetScore()` accessor), as well as interacting with numerous Unity *GameObjects* to both reset the game state, as well as display the final score and get the player's name.



Finally, the addition of *SpeechHandler* was made to allow the refactoring of the previous NPC speech lists into external JSON files to reduce the hardcoding within the system. It contains only a method `SpeechHandler.AccessData()` to return the relevant dialogue list when passed the NPCs name.



3.2 New Features

The following additional features have been developed wholly by Wedunnit, and are sorted in order of descending significance, based on our requirements.

3.2.1 Scoring

The most significant new feature implemented is the scoring system, completing our requirement 13. Initially the user has a score of 1,000. This reduces over time and whenever a character is accused incorrectly (satisfying our requirement 13a), and increases whenever a clue is acquired. This is achieved with variable `GameMaster.gameScore`, public accessor `GameMaster.getScore()` and modifiers `GameMaster.GainScore(float bonus)` and `GameMaster.Penalise(float penalty)`. This score is updated every frame in `GameMaster.Update()` where a small fraction is deducted as time progresses. It is also changed in `AccuseScript.AccusationResult(bool accusation)` which deducts a penalty of 200 if the player accuses incorrectly, and `ItemScript.OnMouseDown()` which adds 50 points when an item is collected.

3.2.2 Leaderboard

An additional feature related to scoring is the leaderboard, described in our requirement 14a. At the end of the game, the user is asked to input their name, which (with their score) is appended to `leaderboard.txt`, located in the root folder of the game (following our requirements 13b, 13c & 13d). When the user clicks on the Leaderboard button on the main menu, the leaderboard file is read as a List of KeyValuePairs `scoreList` such that `scoreList[index].Key == name` and `scoreList[index].Value == score`. This list is then sorted by value and displayed. The alternative to this approach would be to sort the list when data is appended to it; however, we have chosen to sort the data when viewing the leaderboard as it requires less overhead when running the game, and is safer as data is only being read or written, and not both simultaneously.

3.2.3 Blocked characters when player is wrong

Another feature that needed to be implemented in Assessment 3 was giving NPCs the option of refusing to answer questions if they have been incorrectly accused of the murder. This was denoted in our requirement 11a. We considered the possibility of the game blocking the player from completion (e.g. if every character is accused incorrectly or without sufficient evidence), but ultimately decided this would make the game too difficult. Instead, we added a condition that any character could only block the detective's questioning if there were still more clues to be discovered, as collecting a new clue allows the detective to resume questioning with all characters. This adds challenge to the game, without making it too frustrating.

3.2.4 Characters

Following our requirements 8a, we added four more characters to the program following the same method as the Faceless Drones for continuity, bringing the total to ten. We also added several new clue items in the same way the Faceless Drones did as *relevant* clues for the character. These have been drawn in a similar art style to maximise the continuity. Dialogue has been provided for all characters in a similar way which doesn't disrupt the detective's different styles as in our requirement 9c.

3.2.5 Clue Panel

A panel was added to appear on the screen whenever an item was collected by the detective. This is represented as a *GameObject* with *Panel* component within the Unity Editor, which parents other *GameObjects* representing Text, Images etc. When a clue item is clicked on, this Panel is populated with details and toggled within `ItemScript.OnMouseDown()`. We decided this would be a worthwhile addition to the game because while it does not fulfill any extra requirements, it certainly helps the user to understand the game more intuitively and should help make the game more enjoyable, therefore fulfilling requirement 21.

3.3 Updated Features

3.3.1 Replayability

In the initial project provided by Faceless Drones, the game was unable to be played multiple times in a row due to the GameMaster object created for each game not being destroyed at the end of each game. This similarly occured any time the main menu was returned to during the game, and contradicted our requirement 19 (multiple runs of the game wouldn't reset the murderer, the scenario would simply contain multiple murderers). This fix was accomplished by destroying the `GlobalScripts` and `NotebookCanvas` *GameObjects* when the game returns to the main menu. These objects contain the `Inventory` and `Scenario` script components which play an important role in keeping each playthrough different.

3.3.2 Pause menu

The pause menu was improved by giving it a darker background to make it very obvious that the game is paused (satisfying our requirement 16), and also by simplifying the available options. The project handed to us by the Faceless Drones contained a 'Settings' option within the pause menu that had no purpose. At time of transition, its purpose was as a placeholder. However, as it was not included in our requirements, we decided to remove it to maintain simplicity. A similar pane was removed from the main menu for the same reason.

3.3.3 Saving & Loading

While no previous code implemented it, we feel it necessary to mention here the removal of the saving and loading features from the game (the only physical artifact removed being the Load button from the main menu). The feature was removed from the implementation for several reasons. The main factor was the concern of whether the feature would see use; as the software is designed for demonstration in open days (requirements 20 and 22), it is unlikely that a user would come back to complete a game, and even more so when you consider the short timespan of the game. Thus, we have decided to remove this feature from our implementation.

3.3.4 Loading Speech

When we received the project from the Faceless Drones we found that the speech for the game was entirely hard-coded into the GameMaster script. We felt that this was very bad practice and difficult to maintain, so we created a new Class *SpeechHandler*. As shown above in our updated architecture, `SpeechHandler.AccessData()` is a static method that extracts the data from the specified JSON file-containing object and returns the list of speech lines for the given *NonPlayerCharacter*. An alternative method would've been to include a method within the *GameMaster*, class but we felt it better practice within Object Orientated Programming to set it as its own class; it could be adapted for other uses in the future. This addition has been tested via Unit testing (Unit test 31 [7]).

3.3.5 Player Movement

In the project produced by Faceless Drones, the player could click on an area in the room and the detective would move to the location clicked. However, beyond any aesthetics, this had no practical effect on gameplay, as transitions between rooms were only achievable via the map. As we felt that this had no practical impact on the gameplay, and could only serve to confuse and hinder the player's progress, we decided to remove the capability of the detective to move around in the scene. Movement around the RCH is still possible through the map, so requirement 5 (The user must be able to navigate to all rooms in the RCH) is still satisfied.

3.3.5 Minor changes

Below is a short list of other noteworthy changes we made to the software in order to improve its playability.

- Updated background images to be more detailed and interesting. This fulfils requirement 21.
- Character dialogue improved to be more realistic & relevant to each character. This fulfils requirement 21.
- Fixed the clue locations bug where all clue items would appear the same size in the same location in each room to be more appropriate for each room. This fulfils requirement 18.
- Audio clips were added to each room as well as sound effects to make the game more enjoyable and satisfying to play, as people are more likely to want to purchase or play a game if there are decent sound effects [8]. This fulfils requirement 21.
- The game UI has been scaled to work on any screen resolution, so it is more portable to users not playing on lab PCs. This fulfils requirement 23.
- The relevant NPC's name is displayed in their interrogation scene to give the user more clarity so they know who each character is. Previously, names were only mentioned in some of the clues. This fulfils requirement 21.
- Added a 'Credits' option on the menu that displays a panel containing attributions to Creative Commons material used in the game.

3.4 Requirements not completed

The only requirement we did not meet is the Faceless Drones' requirement F10; Saving and Loading. We didn't deem this feature necessary for such a small game as this, and the amount of time that we would've needed to spend on developing this feature was not worth the feature that would probably be under used as the game is very quick and can easily be completed within 10 minutes.

All other requirements were completed to specification, as demonstrated in our testing section [9] and table.

References

[1] Hitchin. Jack et al. “Cereal Killers Website” [Online] Available: http://www-users.york.ac.uk/~phj501/index.html [Accessed: 7- Feb- 2017].

[2] Cadogan. Henry et al. "Assessment 2 Requirements Table". N.p., 2017. [Online]. Available: http://wedunnit.me/webfiles/ass2/Updated-Requirements.pdf [Accessed: 18- Feb- 2017]

[3] Hitchin. Jack et al. “Faceless Drones Requirements” [Online] Available: http://wedunnit.me/webfiles/ass3/Req2.pdf [Accessed: 29- Jan- 2017].

[4] Cadogan. Henry et al. "Changes to Requirements Report". N.p., 2017. [Online]. Available: http://wedunnit.me/webfiles/ass3/Req3.pdf [Accessed: 18- Feb- 2017]

[5] Cadogan. Henry et al. "Updated Requirements Table". N.p., 2017. [Online]. Available: http://wedunnit.me/webfiles/ass3/Requirements_Table_Ass3.pdf [Accessed: 18- Feb- 2017]

[6] Hitchin. Jack et al. “Faceless Drones URL Diagram” [Online] Available: https://www.lucidchart.com/invitations/accept/b8c0909c-8f4d-4d20-9663-92bff042a46d [Accessed: 10- Feb- 2017].

[7] Cadogan. Henry et al. "Test Table". N.p., 2017. [Online]. Available: http://wedunnit.me/webfiles/ass3/TestTable3.pdf [Accessed: 18- Feb- 2017]

[8] S. Cunningham, V. Grout and R. Hebblewhite, "Computer Game Audio: The Unappreciated Scholar of the Half-Life Generation", in *Audio Mostly 2006 A CONFERENCE ON SOUND IN GAMES*, Piteå, Sweden, 2017, pp. 9-14. [Online] Available: http://s3.amazonaws.com/academia.edu.documents/5950083/10.1.1.112.2668.pdf?AWSAccessKeyId=AKIAIWOWYYGZ2Y53UL3A&Expires=1487425970&Signature=Awsgkmz3u%2BEgL2yvG%2Fuc6U9fvXk%3D&response-content-disposition=inline%3B%20filename%3DThe_Drum_Pants.pdf#page=7 [Accessed: 7- Feb- 2017]

[9] Cadogan. Henry et al. "Change Report". N.p., 2017. [Online]. Available: http://wedunnit.me/webfiles/ass3/Change3.pdf [Accessed: 19- Feb- 2017]