**APPENDIX A: SPRINT CYCLE 2 - Player and player token**

|  |  |
| --- | --- |
| 1. **Summary data** | |
| Team number |  |
| Sprint technical lead(s) | Ethan, Sarah |
| Sprint start date | 7/3/21 |
| Sprint end date | 14/4/21 |

|  |  |
| --- | --- |
| 1. **Individual key contributions** | |
| **Team member** | **Key contribution(s)** |
| Ethan | Programmer - game logic and mechanics |
| Umar | Sprint documentation |
| Will | JavaDoc testing |
| Adam | JavaDoc testing |
| Sam | JavaDoc testing |
| Sarah | Programmer - GUI |

|  |
| --- |
| 1. **User stories / task cards** |
| * The 6 persons in the house are represented by playing pieces: Col Mustard, Prof Plum, Rev Green, Mrs Peacock, Miss Scarlett and Mrs White * The game shall be playable for a minimum of 2 players and a maximum for 6 players * The weapons are as follows: dagger, candlestick, revolver, rope, lead piping and spanner. * The weapons are represented by small tokens * Each of the weapons is placed in a different room. * When a player thinks they know the 3 murder cards, then they can make an “accusation” directly after making their suggestion. |

|  |
| --- |
| 1. **Requirements analysis** |
| F4 - The player shall be able to accuse another player of the murder only once. The accusation must include the murderer, murder weapon and room of the crime.  F33 - the player will be represented by a graphical token on the board GUI  NF1 - The software shall be written in Python to ensure maximum portability across platforms  D3 - The characters must be aesthetically pleasing and not be obviously offending to cultural groups. The character images must also uniquely identify a character  NF4 - The software design shall accommodate future updates or maintenance through well designed code  NF7 - The software will implement an appropriate GUI for the user that represents that of a board game  NF6 - Images can be implemented into the board to increase the visual aspect of the game. This is not necessary for the game to run but would assist the game's visuals.  D4 - The weapon models should not offend any underage participants  F7 - Each weapon is represented by a graphical token and shall be placed in a separate room in the board.  F19 - When a suggestion is made the player that made the suggestion will be shown one of the other players cards if the card is one from the suggestion. (If there are no cards to be shown then the player suggestion will be thecards that ae in the murder envelope.  F26 - A player may make a suggestion only when their playing piece is in the room mentioned in the suggestion. |

|  |
| --- |
| 1. **Design** |
| The player is a class. The following image shows a UML diagram of the Player class. As you can see, the player class includes the dice roll function, the accusation and the ability to get the deck. This sprint will only include the ability to accuse other players. The dice roll will be implemented in the player movement sprint.    This sprint also required the creation of the character images and tiles for the game. The images show the finished character images that will be displayed in the game. The tiles the characters will be displayed on are quite small, so the images had to be resized to fit this, resulting in slightly pixelated images as shown. This shouldn't be an issue as they are still distinguishable. For clarity, the full resolution images will be displayed on the home screen when the player gets to choose their character.      The player token shall appear on the board as the size of one of the tiles, the player will take up the entire tile and can therefore traverse the board using the grid.  The next image shws the use case diagram relevant to the accusation part of the sprint. This shows the actions that result in the use of an accusation.    This image shows the sequence diagram for the lead up to an accusation. |

|  |
| --- |
| 1. **Test plan and evidence of testing** |
| import unittest  from src.playertoken import PlayerToken  from src.card import Card  from src.board import Board  from src.player import Player  class MyTestCase(unittest.TestCase):  # Unfinished test  def test\_move\_by\_direction(self):  board = Board()  card = Card('', 0, '')  player = Player('', '', '')  playerToken = PlayerToken(10, 10, card, board, player)  playerToken.move\_by\_direction(1, 1)  if \_\_name\_\_ == '\_\_main\_\_':  unittest.main()  import unittest  from src.player import Player  from src.carddeck import CardDeck  from src.card import Card  class MyTestCase(unittest.TestCase):  def test\_something(self):  self.assertEqual(True, True)  def test\_init(self):  player = Player("name", 1, "symbol")  self.assertTrue(player)  def test\_add\_to\_hand(self):  player = Player("name", 1, "d")  card1 = Card("card1", 1, "a")  card2 = Card("card2", 2, "b")  card3 = Card("card3", 3, "c")  cards = {}  cards.append(card1)  deck = CardDeck()  dict1 = deck.convert\_dict\_and\_add\_to\_deck(cards)  player.add\_to\_hand(dict1)  self.assertEqual(first, second)  def test\_check\_hand(self):      if \_\_name\_\_ == '\_\_main\_\_':  unittest.main() |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **System Testing** | | | | | | | |
| **ID** | **Req** | **Description** | **Inputs** | **Expected** | **Actual** | **Pass/Fail** | **Action** |
| 1 | F33 | Check that the player model appears on the screen | Visual inspection | The player model appears on the screen | There is no graphical player image on the screen, the game is text based | Fail | Implement GUI and link to player token so image is visible on screen |
| 2 | F33 | Check that the player image appears in the correct size - as the size of a single tile | Visual inspection | The player image appears correctly sized | There is no graphical player image on the screen, the game is text based | Fail | Implement GUI and link to player token so image is visible on screen |
| 3 | NF1 | Check that the code for the player is written in python | Visual inspection | That the code for the player class is programmed in python | The code for the player class is programmed in python | Pass | n/a |
| 4 | D3 | Check that the character images are aesthetically pleasing | Visual inspection | The player images are aesthetically pleasing | The player images are aesthetically pleasing | Pass | n/a |
| 5 | D3 | Check that the character images don't obviously offend any cultural groups | Visual inspection | The player images don't obviously offend any cultural groups | The player images don't obviously offend any cultural groups | Pass | n/a |
| 6 | D3 | Check that the character images uniquely represent a character | Visual inspection and try to distinguish characters | The player images are appealing and uniquely identify a character | The player images are appealing and uniquely identify a character | Pass | n/a |
| 7 | NF7 | Test that the software implements an appropriate GUI for the user that represents that of a board game | Visual inspection and compare to board game version | the software implements an appropriate GUI for the user that represents that of a board game | The GUI for the player is not implemented however the GUI for the weapon has been implemented | Fail | Implement GUI for the players and add to the board GUI |
| 8 | NF6 | Test that images are used in the game. | Visual inspection | Images are used and display correctly in the game GUI | There are images for the weapons however not the player | Pass | n/a |
| 9 | D4 | Test that the weapon models do not offend any underage participants in the game | Visual inspection | The weapon models are appropriate and do not offend any participants | The weapon models are appropriate and do not offend any participants | Pass | n/a |
| 10 | F7 | Test that the weapons are represented by graphical tokens | visual inspection | The graphical models of the weapons appear on the board | The graphical models of the weapons appear on the board | Pass | n/a |
| 11 | F7 | Test that the graphical token of the weapon represents the correct weapon type | Compare the weapon image on the board to the corresponding weapon on the text file | The graphical models of the weapons represent the correct weapon | The graphical models of the weapons represent the correct weapon | Pass | n/a |
| 12 | F7 | Test that all weapons start in separate rooms | Visual inspection | All weapons start in separate rooms | All weapons start in separate rooms | Pass | n/a |
| 13 | F33 | Test that the graphical player token corresponds to the correct player token | Compare the player image on the board to the corresponding player on the text file | The player token corresponds to the correct graphical token | The player GUI is not implemented so the player is represented by a symbol | Fail | n/a |
| 14 | NF4 | Test that the code is built well enough to support maintenance and updates in the future if needed | n/a | The code is structured well enough to support future updates and maintenance | The code is structured well enough to support future updates and maintenance | Pass | n/a |
| 15 | F33 | Test that the graphical player image appears on the coordinates as the player token | Visual inspection and compare image coordinates to text file coords | The player token and image are on the same tile at all times | The player GUI isn't not implemented therefore the player is not represented by an image | Fail | Implement player GUI and link to board |
| 16 | F7 | Test that the graphical weapon image appears on the coordinates as the weapon token | Visual inspection and compare image coordinates to text file coords | The weapon token and image are on the same tile at all times | The weapon token and image are on the same tile at all times | Pass | n/a |
| 17 | F7 | Test that the weapon token are the correct size - the size of a tile | visual inspection | The weapon images are the size of a single tile | The weapon images are the size of a single tile | Pass | n/a |
| 18 | F4 | Test that the player has the ability to accuse another player of the murder | Click the accuse button on the UI | The player can press the accuse button to make an accusation | The player can press the accuse button to make an accusation | Pass | n/a |
| 19 | F4 | Test that the player has the ability to suggest another player for the murder | Select the suggestion button on the UI | The player can press the suggest button to make an accusation | The player can press the suggest button to make an accusation | Pass | n/a |
| 20 | F4 | Test that the player can input the murder weapon when accusing | Select the weapon after pressing the accuse button | The player can input any weapon after pressing the accuse button | The player can input any weapon after pressing the accuse button | Pass | n/a |
| 21 | F4 | Test that the player can input the murderer when accusing | Select a player after pressing the accuse button | The player can input any player after pressing the accuse button | The player can input any player after pressing the accuse button | Pass | n/a |
| 22 | F4 | Test that the player can input the murder room when accusing | Select a player after pressing the accuse button | The player can input any room after pressing the accuse button | The player can input any room after pressing the accuse button | Pass | n/a |
| 23 | F19 | Test that when a suggestion is made the player that made the suggestion will be shown one of the other players cards if the card is one from the suggestion. | Press the suggestion button on the UI | The player that made the suggestion is shown one of the other players cards if the card is one from the suggestion. | The player that made the suggestion is shown one of the other players cards if the card is one from the suggestion. | Pass | n/a |
| 24 | F19 | Testing that upon accusing, if there are no cards to be shown then the players suggestion will be the cards that are in the murder envelope | Pressing the suggestion button on the UI when all cards are visible | The players suggestion will be the cards that are in the murder envelope | The players suggestion will be the cards that are in the murder envelope | Pass | n/a |
| 25 | F26 | Test that A player may make a suggestion only when their playing piece is in the room mentioned in the suggestion. | Pressing the suggestion button when their player piece is in the mentioned room | The player can successfully make a suggestion | The player can successfully make a suggestion | pass | n/a |
| 26 | F6 | Test that A player cannot make a suggestion when their playing piece is not in the room mentioned in the suggestion. | Pressing the suggestion button when their player piece is not in the mentioned room | The player cannot make a suggestion, ie. the button does not do anything, or an error appears | The AI has not been implemented | Fail | n/a |

|  |
| --- |
| 1. **Summary of sprint** |
| *You should consider and discuss:*   * *Did you achieve your objectives for this sprint?*   + *Yes* * *Is there a working prototype?*   + *Yes* * *What went well, and what did not go well? If things did not go well, what have you learned and what will you do differently for the next sprint?*   + *The implementation was fairly linear and due to lots of the movement being offloaded to the player token it was easy to interface with it, however the lack of time gave stress and made the productivity go down and it took longer than what we wanted, even if it was a small sprint* * *Is there any feedback from the customer?*   + *no* * *The following image shows the prototype for the current sprint. This shows that the weapon graphics have been implemented and that they all start in separate rooms. THe buttons for suggestions and accusations are present but not functional, this will be added in a future sprint* * *The GUI* * The Cli |