**SOFTWARE ENGINEERING G6046**

**APPENDIX A: SPRINT DOCUMENTATION TEMPLATE**

|  |  |
| --- | --- |
| * **Summary data** | |
| Team number | 11 |
| Sprint technical lead(s) | Anson Wong |
| Sprint start date | 13/04/2021 |
| Sprint end date | 18/04/2021 |

*The technical lead may vary from one sprint to the next. This is down to how you collectively organise your team.*

|  |  |
| --- | --- |
| * **Individual key contributions** | |
| **Team member** | **Key contribution(s)** |
| Anson Wong | Programmer, Planner |
| Danny Newsom | Programmer |
| Tomasz Czarnecki | Programmer |
| Abdullah Al-Hiyarat | Programmer |
| Mohammad Jallad |  |
|  |  |

*This data should help you to agree your peer assessment at the end of the project. If there is a dispute over your peer assessment, the markers will refer to this section as evidence to support a final decision.*

|  |
| --- |
| * **User stories / task cards** |
| *Provide text descriptions of any user stories or task cards you have selected for this sprint. These should naturally emerge from the user requirements document and discussion on Canvas. If you produce task cards, they should show the relative priority of the task for this sprint.*   * *When a player reaches a room, they can call out a “suggestion” by calling into that room any other person and any weapon.* * *When the suggestion is made, starting from the current player’s left, if they have one of the cards mentioned in the suggestion, then they must show one only of those cards to the current player (unseen by other players)* * *If the player on the left is unable to show a card, then we move to the next player to the left until a player is able to show a card to the current player. Once that happens, or all players have been tried leading back to the current player, then the turn of the current player is ended.*   Task Cards:   1. Fix and Complete accusation 2. Working on documents 3. Focus: Suggestion 4. Ability to pick Room and Character, but not cards 5. Ability to check if the token is in a room and what room the token is in 6. Ability to check which player has what cards 7. Ability to display a random card to the current player |

|  |
| --- |
| * **Requirements analysis** |
| *For the user stories/task cards selected, set out what key functional, non-functional and domain requirements you have identified. Remember that functional and non-functional requirements can be further categorised as mandatory (“shall”) and desirable (“should”). You can use free text descriptions or tabular formats. Remember that domain requirements cannot be acted upon directly. They require domain expertise to refine them into meaningful functional and non-functional requirements. All requirements should be SMART (Specific, Measurable, Achievable, Realistic and Time-Bounded). The requirements analysis does not need to be exhaustive, but should focus on things that are important for this sprint. They should also form a basis for testing.* |

* UI Controller: (Tomasz)
  + Shall
    - Fix and Complete accusation
      * Have a UI screen to let the player to chooses the characters, weapons and rooms
      * Combine the selection and pass it to the round manager
    - Suggestion
      * Only allow the player to select accusation when the player is in a room
      * The UI screen will have the room selected
      * The player cannot change the room
      * Pass the selection to the round manager
      * Wait for round manger to return a card to be displayed
  + Should
    - Have the ability to change their choices
    - Have the ability for the other players to select which card to display
* Round Manager: (Abdullah)
  + Shall
    - Suggestion
      * A method to accept the suggested list of cards
      * Loop through all the players via the Turn Manager
      * Get the first instance of the player that has that card
      * return which player has that card and that card it is
      * Return that results to the UI controller most likely
  + Should:
    - The first card to be check is randomised
* Player Master Controller: (Anson)
  + Shall
    - Be able to check if the player has a certain card in it’s deck

|  |
| --- |
| * **Design** |
| *Remember that you only need to do enough design to support the objectives of the sprint. For teams working with OO implementation languages (likely most of you), this would include a class diagram. You may find it useful to develop simple Application Programming Interfaces (APIs) for key classes. This will focus your attention on what each class needs to make available for other classes to use. It also supports good documentation practice and helps coders work together.*      Suggestion UML |

|  |
| --- |
| * **Test plan and evidence of testing** |
| *You should consider:*   1. *Unit/component level testing – typically achieved using automated test procedures such as Junit in Java. This level of testing demonstrates that individual classes are working as you intend.* 2. *System level testing – typically a human lead and documented test process that shows the prototype working as a whole entity.*   *Testing should show that the requirements you set out are being delivered on. They provide a means of showing that we have delivered what the user stores and task cards set out. Remember to identify a useful set of boundary test conditions.*  *Evidence of testing should demonstrate that the prototype achieved has been tested according to the test plan. If there are deficiencies, then these should be documented, as they will need further work in a subsequent sprint.*  Test Scenes:  -Suggestion  -Main scene (for UI)  **\*Please Refer to Testing Document** |
|  |

|  |
| --- |
| * **Summary of sprint** |
| *You should consider and discuss:*   * *Did you achieve your objectives for this sprint?* * *Is there a working prototype?* * *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?* * *Is there any feedback from the customer?* |

* UI Controller: (Tomasz)
  + The basic logic and scripts were completed, however when merged and put together with the rest of the work it provided some issues.
* Round Manager: (Abdullah)
  + The basic logic and scripts were completed, however when merged and put together with the rest of the work it provided some issues.
* Player Master Controller: (Anson)
  + Completed

Accusation was fixed and implemented into the test scenes, as well as integrated into the UI. The suggestion section of the scripts was completed, however it created some bugs and needed reconsideration due to it being written not optimally. UML documents for all classes were produced and used to help construct the remaining test scenes needed for proper testing. A find card method was also completed, which allowed for suggestions to be called more optimally. Some of the work needed to be done was moving more slowly than anticipated with one of our group members, however it was completed by the end of the cycle.