**SOFTWARE ENGINEERING G6046**

**APPENDIX A: SPRINT DOCUMENTATION TEMPLATE**

|  |  |
| --- | --- |
| 1. **Summary data** | |
| Team number | 11 |
| Sprint technical lead(s) | Anson Wong |
| Sprint start date | 05/03/2021 |
| Sprint end date | 19/03/2021 |

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

|  |  |
| --- | --- |
| 1. **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.*

|  |
| --- |
| 1. **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.*   * *Each of the weapons in placed in a different room.* * *By tradition, Miss Scarlett goes first. The player shakes the dice and moves that number of squares along the passage to any room they like. Players take turns in a clockwise direction.* * Secret passages enable players to move from certain rooms to those indicated on the board. This can be done on a player’s turn without throwing the dice, and constitutes the move * Players may enter rooms by their doors only, but cannot enter and leave a room in the same move Entering a room ends the move count. If a player throws a 6 and only needs 4 to enter a room, they may enter the room and ignore the 2 moves left over. * No two pieces may occupy the same square. A room however may be occupied by any number of pieces and weapons. A player may choose to remain stationary during any number of turns in play.   Task Cards:   1. FOCUS: Full movement cycle 2. UI Handler (Tomasz) 3. User Controller (Tomasz) 4. Turn Controller (Abdullah) 5. Round Manager (Abdullah) 6. Board Manager (Danny) 7. Player Controller (Anson) 8. Tile (Anson) 9. Room (Danny) 10. Dice (Danny) |
|  |

|  |
| --- |
| 1. **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 Handler (Tomasz):
  + Shall:
    - Button for Rolling the dice
      * Calling the User Controller to roll dice
    - Display the rolled results
  + Should:
    - Button for next turn
* User Controller: (Abdullah)
  + Shall:
    - Call Round Manager to roll dice
  + Should:
    - Call Turn Manager to next turn
* Turn Controller (Abdullah)
  + Shall:
    - Storing the player turns
    - Passing the turn to the next player
* Round Manager: (Abdullah)
  + Shall:
    - Set its mode
    - Call Dice to roll and return the results
    - Call Board Manager to display tiles that the player can move on
    - Call Player move
* Board Manager: (Danny)
  + Shall:
    - Link the Player Token to player
    - Display and stores tile the player can move
    - Find all tiles the player can move with *bfs*
    - Highlight the movable tiles
* Player Controller: (Anson)
  + Shall:
    - Link itself to the respective token
    - Move the token to the selected tile
    - Move the token to the selected room and call Room to handle that
* Tile: (Danny)
  + Shall:
    - Know it’s grid placement on the board
    - Can check if the tile is occupied by a token
* Room: (Danny)
  + Shall:
    - Store positions of players and weapons
    - Store players and weapons in that room
    - Store connected shortcuts
    - Store all exit tiles
* Dice: (Danny)
  + Shall:
    - Roll the dice
    - Return the results when ready

|  |
| --- |
| 1. **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.*    Shortcut UML |

|  |
| --- |
| 1. **Test plan and evidence of testing** |
| *You should consider:*   * *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.* * *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:   * Dice * Camera * Moving the player * Moving weapons in and out * One for short cuts |

|  |
| --- |
| 1. **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?* |

Task Cards:

1. FOCUS: Full movement cycle
2. UI Handler (Tomasz)
   1. Buttons works
3. User Controller (Tomasz)
   1. It works
4. Turn Controller (Abdullah)
   1. Can swap to the next player
   2. Remove player
   3. Get next player
5. Round Manager (Abdullah)
   1. On hold
   2. Move works
6. Board Manager (Danny)
   1. It works
7. Player Controller (Anson)
   1. It works
8. Tile (Anson)
   1. It works
9. Room (Danny)
   1. It works
10. Dice (Danny)
    1. It works

Broken and fixed movement when merging for movement in and out of rooms.

Unfortunately, some of our teams mates are occupied with other works and wasn’t able to complete the assigned task card

We have showed a semi broken prototype to Mr. Raffles and he seems impressed