**Sprint 1 – Group 39**

|  |  |
| --- | --- |
| 1. **Summary data** | |
| Team number | 39 |
| Sprint technical lead(s) | Arya Diznabi, Sam Banks |
| Sprint start date | 11/3 |
| Sprint end date | 18/3 |

|  |  |
| --- | --- |
| 1. **Individual key contributions** | |
| **Team member** | **Key contribution(s)** |
| Sam Banks | Writing code for the card class, sprint documentation |
| Geonwoo Lim | Writing code for player and dice class |
| Arya Diznabi | Writing the code for the room and board class, Market research |
| Dohyun Lee | Writing the code for the player class |
| Humza Satti | UML design, writing code for board class |
| Saif Zuqaili | UML design, Use Case diagram, multimedia |
| Subsin Sriprasert | UML design, multimedia |
| Nishan Deivendranbose | Requirements analysis, sprint documentation |

|  |
| --- |
| 1. **User stories / task cards** |
| * Produce a small prototype front end design that opens and displays the board, so that we can show the client and see if they like our idea for the UI and colours. * Draw up UML and Use Case diagrams. These will help the programmers to structure the code in a way that everyone understands. They will also be used to show the client so that they also have an idea as to how their game is laid out. * Create a few of the core classes in java code. |

|  |
| --- |
| 1. **Requirements analysis** |
| **Key functional requirements**   * The game should allow for 2 to 6 players to participate * Option for one or more of the users to be controlled by the computer, and should be able to play and challenge human players * A clear GUI * Only one ‘accusation’ can be made * Testing must be carried out to make sure the game is working properly and to free it of any potential bugs   **Non-functional requirements**   * Should be playable on PC’s + Mac * File size should not be large so is easily shared * Game should not be slow to load or slow to react to commands from players   **Domain requirements**   * Should be colourful and intuitive, a style that reflects the spirit and character of the original board game      * Software should be easy to use and understand |

|  |
| --- |
| 1. **Design** |
| During this first sprint we are focusing our design team on creating a UML diagram for the Cluedo game. This will help programmers to visualise how the classes will need to be laid out in the java package. This is useful as it helps to plan before the coding process so that everyone understands how the program is structured and the names of variables that will be used.  We are also producing a use case diagram; this should help for our front-end development to go more smoothly by knowing what the player interactions will be with the game.  Our main design objective this sprint is to produce a small prototype program that will show what we think the colours, GUI and multimedia will look like on the final product. |

|  |
| --- |
| 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.* |

|  |
| --- |
| 1. **Summary of sprint** |
| During this sprint, we managed to create a basic board class to give us a base as well as a few of the smaller classes such as Card, Room and dice. We don’t have a working prototype yet as it took slightly longer than expected to get everyone organised. Ideally, we would have a program that could run but we had a few group issues at the beginning getting started which set us back slightly with the coding. Our group didn’t have many confident programmers which resulted in some confusion. Eventually we got everything together and it started moving slightly smoother than before.  For next sprint we will try to plan and delegate the tasks better, and to also promote working together over discord or in labs rather than separately. This is because when we were working on code separately, it makes it harder in the future when the code eventually needs to be linked up. When programming the classes with each other, it helps when keeping up with how and when changes are being made to the code.  Overall, the first sprint due to the late start of the group was a bit rushed and we didn’t reach our main objective of having some sort of design prototype that we could show to the client. This should be finished very soon, however. We also didn’t manage to get much of the testing completed in time. This is because we may have been focusing our time into the wrong things. |