**Document 04 – Project 2 Wrap-up**

**This document is contained in your GitHub repository in a folder named *docs*.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Group | 6 | | Group Member Names |  |
|  | |  |  | | --- | --- | | 1. | Shaun Morrison | | 2. | Charles Samuel | | |  |  | | --- | --- | | 3. | Jeremy Craven | | 4. | John-Michael Kuczynski | |
|  |  |  |

1. **Video Demo**

|  |
| --- |
| **Deliverable**  Create a video for your project presentation. The agenda for your demo:   1. (1-3 minutes) Describe your project 2. (3-5 minutes) Discuss the design of your system using a class diagram.  * Explain at a high-level what each (or the most important) classes’ responsibilities are. Your goal is to give me a feel for your architecture and how the pieces fit together. * Don’t read off the list of methods! You can mention some key methods, or just describe what responsibilities each class has. If you need to go into more detail, it is fine, to show portions of the design and explain. * The class diagram should **not** show instance variables for any of the classes (in ObjectAid, select the class, right-click, and choose: *Show Attributes, None*) * The class diagram should **not** show methods for any Gui classes (in ObjectAid, select the class, right-click, and choose: *Show Operations, None*)  1. (3-5 minutes) Choose two tests (or sets of related tests) that you think are good tests. These can be unit or integration tests. Then, for each:  * Is this a unit or integration (or other) test? * What method(s) is being tested? * In what class(es)? * What does the method(s) do? * How did you test it? How did you decide on the inputs? * Did you use jUnit? * Any other comments?  1. (5-10 minutes) Pick a set of user stories to demo. Then, for each:  * Display a user story (text, in word or whatever) and read it, Expand on it if necessary. * Illustrate how the software fulfills it.   You may break this into 3 videos if you like. Just put the links here. |

|  |  |
| --- | --- |
| Link | https://drive.google.com/drive/folders/12OkPLugoSNUu3U0NmULOaDGdtR46irjU?usp=sharing |

1. **Development Status**

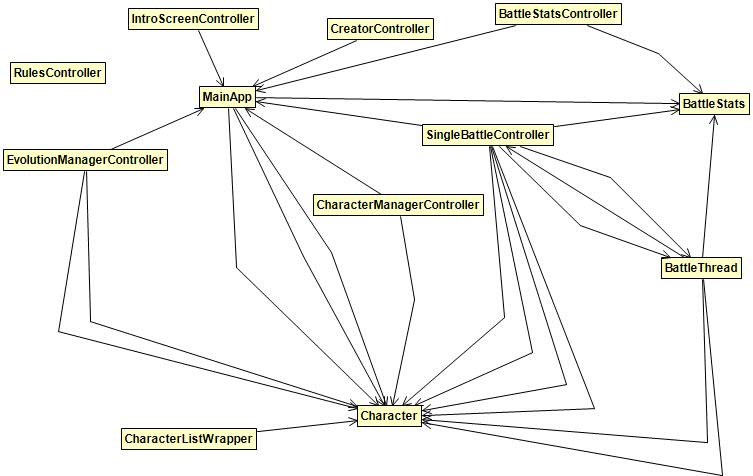
|  |
| --- |
| **Deliverable**  Provide a table of user stories that are completed, and another table of stories that are in progress. |

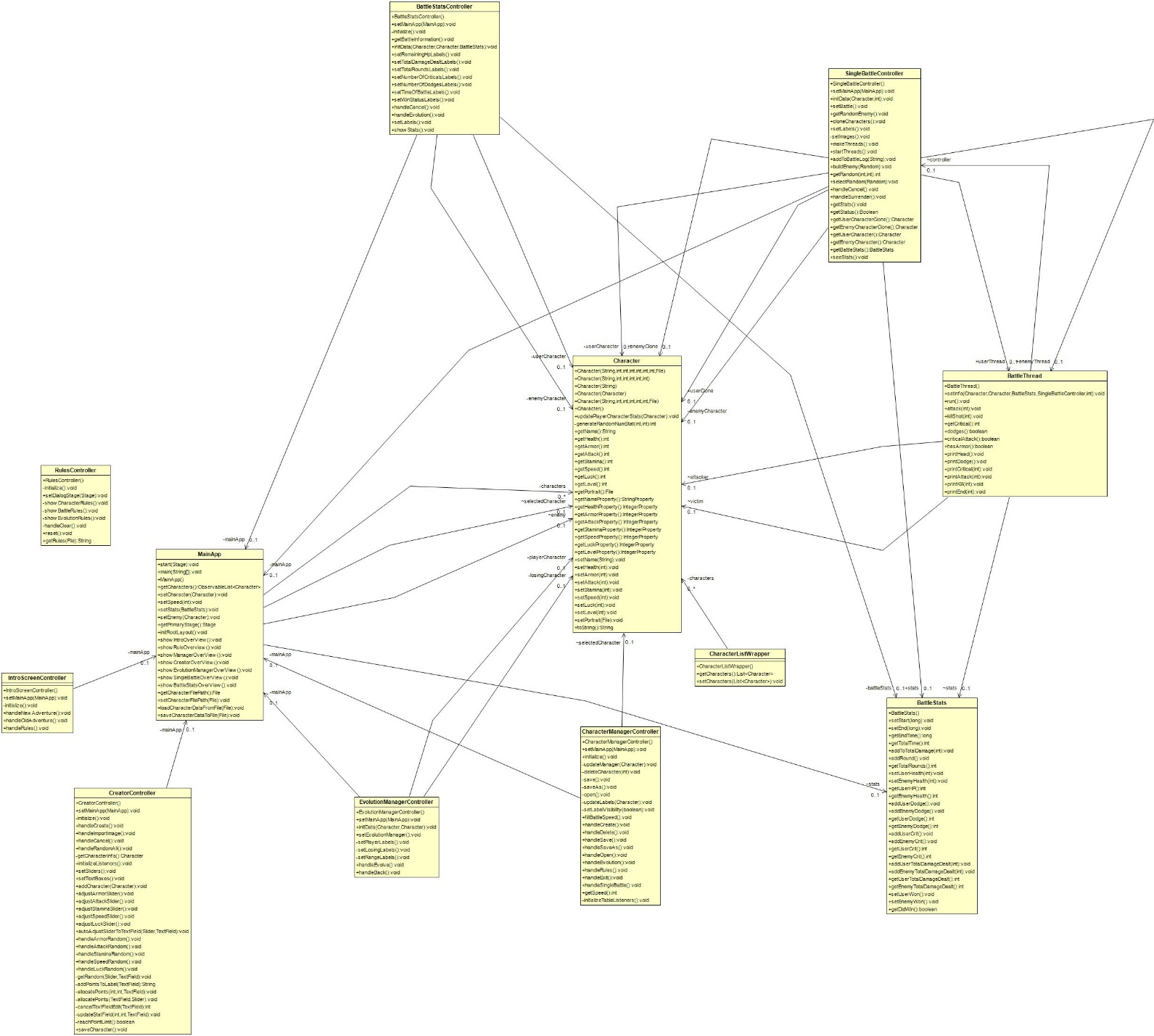
|  |  |  |
| --- | --- | --- |
| **Num** | **Completed User Story Title** | **Comments** |
| 1. | ***Display Rules*** | User can display rules in both intro and character manager |
| 2. | ***Character Deletion*** | User can delete a character |
| 3. | ***Character Saving*** | User can save characters |
| 4. | ***Character Creation*** | User can create a custom character |
| 5. | ***Creation Randomization*** | User can randomize character creation |
| 6. | ***Character Portrait*** | User can use default portrait or upload a custom |
| 7. | ***Character View*** | User can view each character in the character manager |
| 8. | ***Battle Speed*** | User can control the speed of the battle in the character manager |
| 9. | ***Battle Autonomous*** | User does not have to interact for battle to proceed |
| 10. | ***Battle Display*** | User can view what happens in each round of the battle |
| 11. | ***Surrender*** | User can choose to surrender early in the battle |
| 12. | ***Stats*** | User can view the stats after each battle |
| 13. | ***Evolution Visualization*** | User can view the evolution possibilities before evolving |
| 14. | ***Evolution*** | User can evolve their character with the enemy character |
| 15. | ***Character Level*** | User can view character level based on the total points of character |
| 16. | ***Battle Equality*** | User will be guaranteed equal battles based on user level |
| 17. | ***First Attack*** | User will know which character attacks first based on speed |
| 18. | ***Dodge*** | User’s character will have the possibility of dodging an attack |
| 19. | ***Random Enemy*** | User will be able to battle an enemy instead of creating an enemy |
| 20. | ***Multiple Attacks*** | Based on character stamina a character can achieve multiple attacks |

|  |  |  |
| --- | --- | --- |
| **Num** | **In-Progress User Story Title** | **Comments** |
| 1. | ***Evolution History*** | User will be able to see previous versions of the character |
| 2. | ***Team Creation*** | User will be able to create a team of characters |
| 3. | ***Team Battle*** | User will be able to battle the entire team as one |
| 4. | ***Portrait Drawing*** | User can draw their own portraits |
| 5. | ***Battle Pause*** | User will be able to pause a battle |

1. **Class Diagram**

|  |
| --- |
| **Deliverable**  Use Object Aid (or some other software) to develop a neat, legible, properly sized UML class diagram(s) showing your current design that exactly reflects your code. If your system is large, it is better to show a diagram with just the classes and their associations. Then, follow that with 2 or more diagrams showing the details of each class. You can use the class diagram from the demo. |





1. **Retrospective**
2. Meet as a group and discuss the following questions and provide a group written response below:
3. What worked well for us?

|  |
| --- |
| **Answer**  Weekly meetings on discord.  Creating a channel on discord that notifies use of any new pull requests.  Dividing up the work/user stories.  A member reviewing and testing another member’s code.  Leaving suggestions in coding reviews.  Pushing small pieces at a time |

1. What did not work well for us?

|  |
| --- |
| **Answer**  Don’t hear anything from teammates for several days.  Members not showing up to several meetings  Member’s working on the same component, which causes coding conflicts.  Some member’s putting little effort into the project.  More concreate due dates  Reviewing members work more frequently |

1. What actions can we take to improve our process going forward?

|  |
| --- |
| **Answer**  Use Junit to test our code before creating a pull request  Commit teammates to app features to prevent coding conflicts.  Increase response time from teammates.  Create deadlines for app features. These deadlines will be approved by the member working on said feature and finalize by the whole team. |