## Final Architectural Plan

Prepared for Michael Landreth

Project Mentality

Prepared by Carson Perry

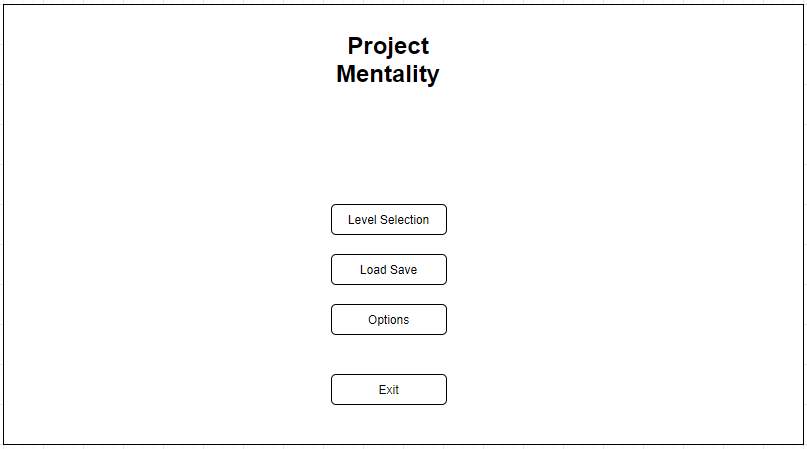
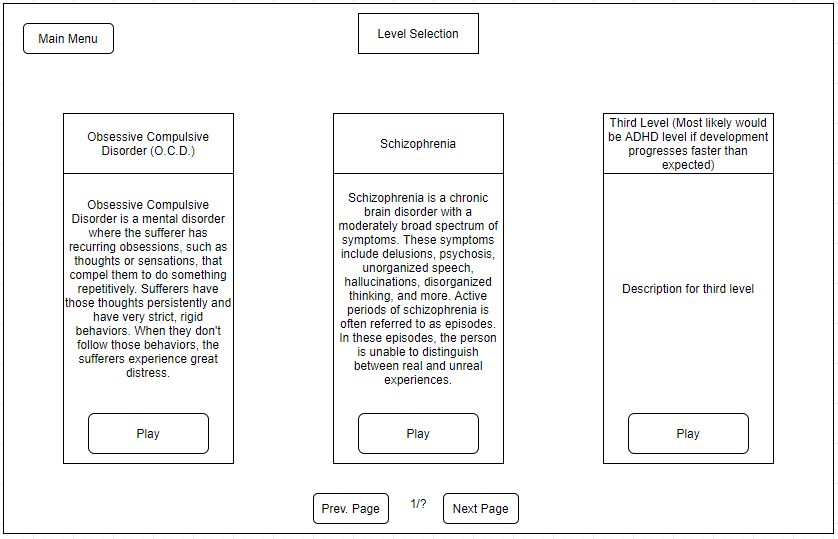
|  |
| --- |
| Design Planning Summary |

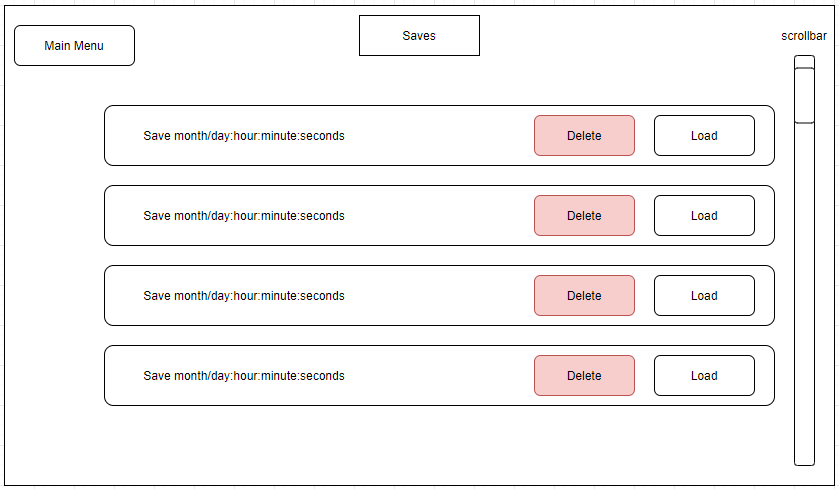
1. This development project is the result of the lack of mental illness awareness within kids and the harm that it may cause in interpersonal relationships with those that suffer from mental illnesses. This project will help alleviate those issues by providing a game to help people understand the perspectives of those with the included mental illnesses. This game will be comprised of multiple levels that depict a different mental illness. The player will go through these levels as a character with that mental illness. I believe that video games are the perfect medium for relaying perspective. Someone can talk about their experiences, but it’s still difficult to see their perspective and be in their shoes, especially when their brain works in a fundamentally different way. However, with a video game, the player is able to act as someone with that illness and see things that translate well into the symptoms of those mental illnesses. This is a much easier way to change perspective.

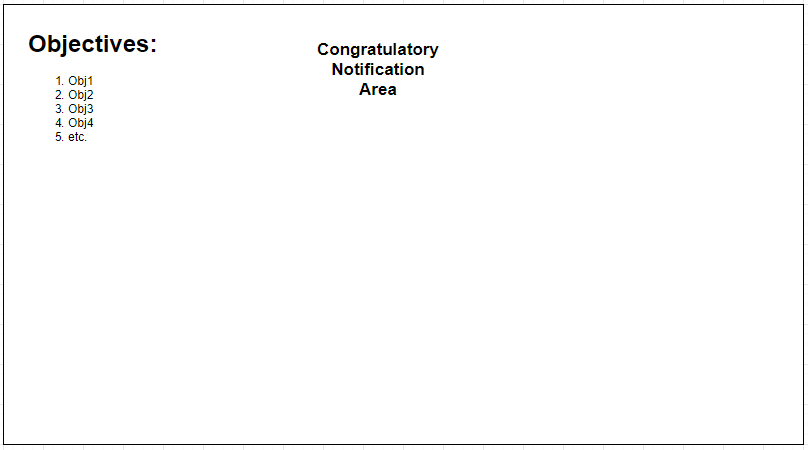
|  |
| --- |
| Overview of Design Concepts |

1. The User will select a level in the main menu, then be loaded into the level. Not all levels will have a UI. In fact, the OCD level is the only level with planned UI outside of text appearing at the bottom of the screen for dialogue. Minimal UI is to help with immersion, however with the OCD design, the UI serves the purpose of placing a sense of importance of the objectives of a typical morning, though there’s no real reason to do the tasks in that order, and it’s purely in the players head that they need to do those tasks in that order, just as someone with OCD thinks. The UI for that level serves the purpose of furthering perspective in the players head.

UI Screenshots

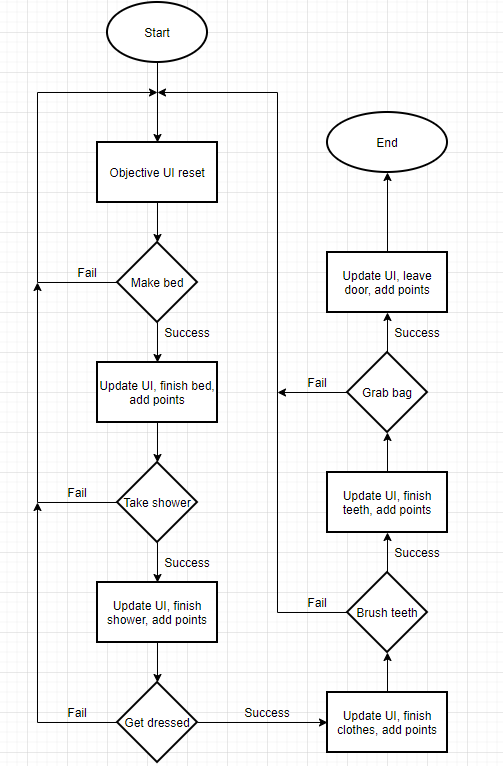
* 1. Main Menu:
  2. Level Selection Menu
  3. Load Save Menu

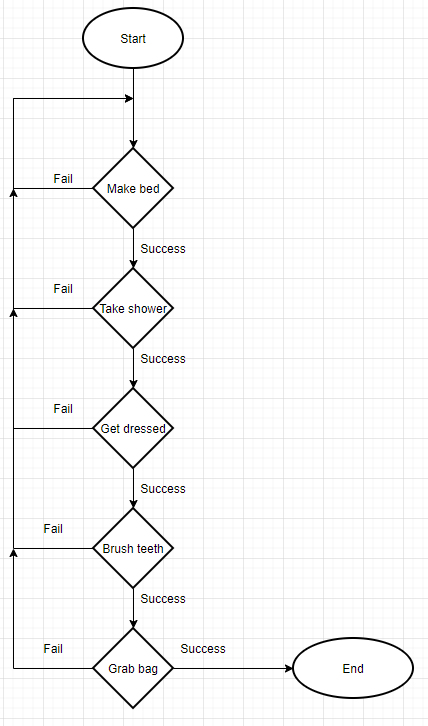


* 1. OCD Level:

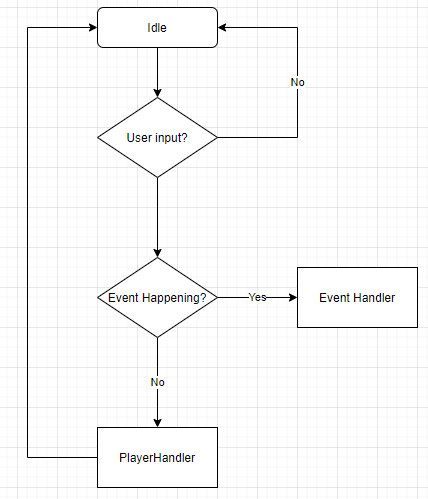
OCD Level UI and Objectives Control Flow Diagram

The UI and the actual hidden objects are on two separate systems. The UI’s purpose is to get in the head of the player by use of the on-screen objectives and point system. There’s no real reason to complete the objectives in this specific order, however the player will feel as though they have to because all their progress is reset and they have to redo everything if they get something wrong or complete something out of order. However, as more objectives are added on and become increasingly ridiculous and have no point, the player will eventually learn that they could have completed all of the objectives regardless of order and left without any repercussions. But it feels bad to lose all your points and progress. This helps to emphasize the distress that sufferers of OCD have when breaking their behaviors. By showing the players that the only reason to do any of those objectives in order, or redo them after failing, was all in their head, it helps empathize the player with the symptoms of OCD.

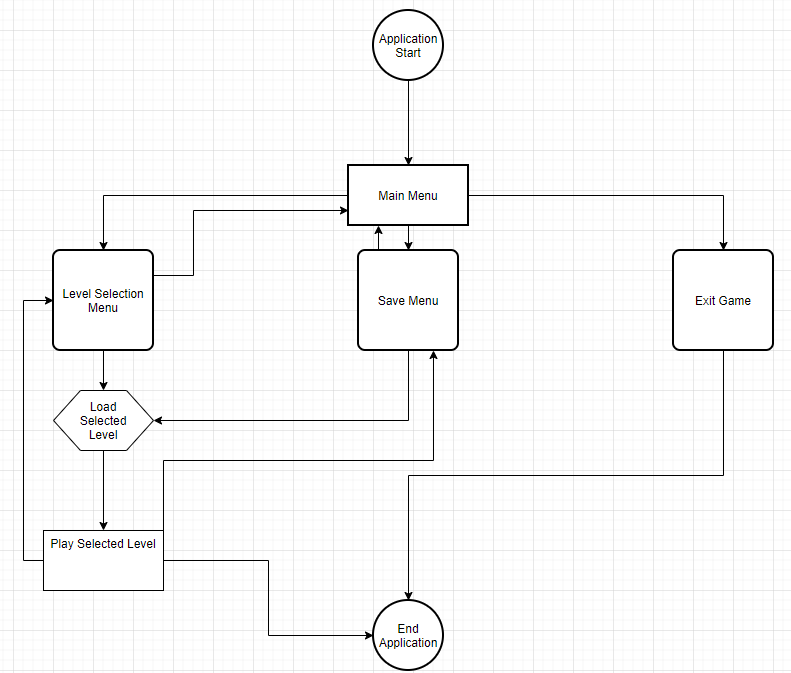




User Input Flowchart



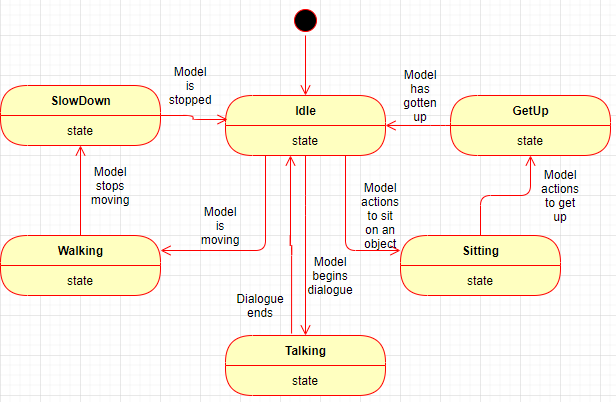
User Experience Flowchart



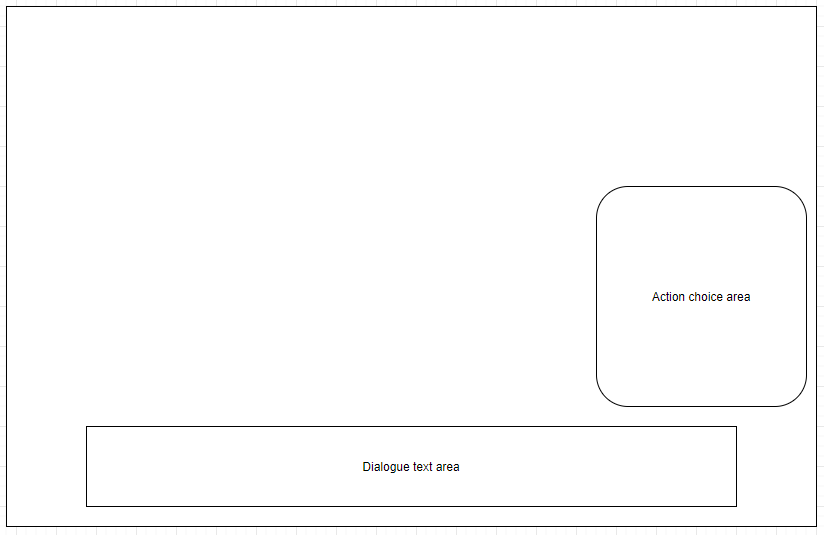
Schizophrenia Level:

The schizophrenia level won’t have much of a UI except for a text box at the bottom, and options on the right side of the screen. This will be the first level to include other characters outside of the player. In the previous level about OCD, the only character is the player, and there’s no need to render an entire player object into the scene, so there will only be bits and pieces that the player will see as animations that trigger from actions. The animation that plays for the model depends on the object’s current state. The state diagram shows how those states interact with each other, with each state having its own animation. In this level, the player will go through events in a fantasy world. The purpose of the setting being a fantasy world is to disconnect the player’s knowledge of reality and the world, from the game’s world. This way, it’s much easier to confuse the player on what is and isn’t real in the world. If you had a realistic level and had a train fall out of the sky, that would immediately break immersion and the player would immediately understand that the character’s schizophrenia is active. But, if you’re in a world where trains fly and can break down and fall from the sky, that is a much more reasonable thing to happen. The player will go through multiple events that contain real and unreal elements. Choosing bad options during an episode would bring bad consequences. Such as fighting off a perceived burglar, who is actually your friend, or breaking a mirror because it’s presented as a portal to allow an invading army into your room. Even an event as simple as believing you are having a conversation with a friend, but when you chat with them later, find out that conversation never actually took place.

Character Model State Diagram:

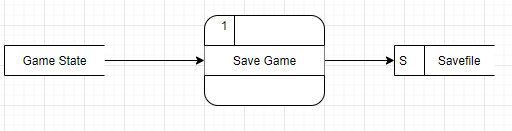


Schizophrenia Level UI:



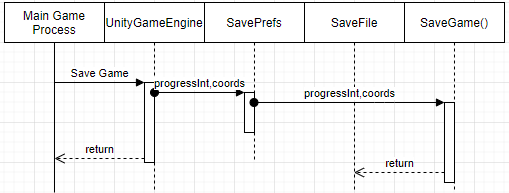
Saving:

The Dataflow for saving is very simple, the game state provides information for the save game process, which then creates a file with the save information



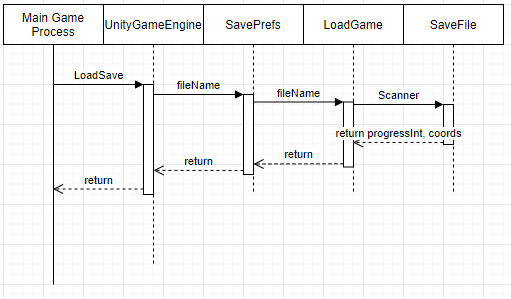
As the player progresses through each level, regardless of whether the player is aware, they are passing checkpoints. These checkpoints will increment a progress integer that allows the program to know where they player has progressed to. This int is what will be saved, along with coordinates of where the player is on the map. This information will be serialized and verified within the SavePrefs C# script to ensure that the information actually works with the game, and isn’t a custom file to break the application, or corrupted data.

Save Game Sequence Diagram:



For loading a game, when clicking on the available save files, it will send the name of the chosen file. This file is what will supply the game with the information to render the game to what it was during when the game was saved.

Load Game Sequence Diagram:

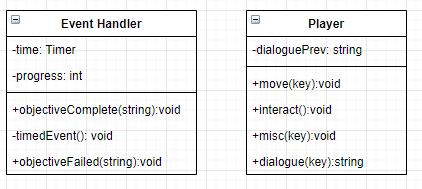


1. Use the template to list the project deliverables.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Deliverable Acceptance Log | | | | | |
| ID | Deliverable Description | Comments | Evaluator (internal or external as applicable) | Status | Date of Decision |
| 1 | UI Wireframes | Wireframes for the planned UI of the Main Menu and the OCD level | Carson Perry | Complete | 02/18 |
| 2 | User Input Flowchart | Flowchart for how user input is interpreted and sent | Carson Perry | Complete | 04/04 |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |

|  |
| --- |
| Detail Solution Architecture |

1. The User will be able to choose which level to load from the main menu. This will begin to load assets that the level requires and set the scene. The only model classes used in the scenes will be an Event handler (an object that will be used to detect alarms, and collisions with event causing areas) and the Player class model as well. There is no need for other models, as the Event Handler can control and manipulate multiple objects in the scene, and is the only model we need for scene objects. There won’t be any need for software setup, however hardware wise, there are some specs required for running this game. Unity games require an Operating system of Windows 7 SP1+, macOS 10.12+, or Ubuntu 16.04+. The CPU must be able to handle SSE2 instruction sets, and the graphics card (GPU) should be capable of using DX10.



1. With this being a completely offline, single-player video game, there’s little to do with security. However, on the main site where this game will be distributed, I will provide an md5 checksum. This is often used to ensure that a program hasn’t been tampered with after downloading. This will help people make sure they have the correct version in case they get it from somewhere else.
2. Use the template to list the hardware and software technologies.

|  |
| --- |
| Hardware and Software Technologies |
| 1 - Windows 7 SP1+, macOS 10.12+, or Ubuntu 16.04+ |
| 2 – CPU that can handle SSE2 instruction sets |
| 3 - GPU capable of using DX10 |
| 4 – Unity Game Engine |
| 5 – FL Studio |
| 6 - Blender |

|  |
| --- |
| Revision and Signoff Sheet |

**Change Record**

|  |  |  |
| --- | --- | --- |
| **Date** | **Editor** | **Revision Notes** |
| 04/04/21 | Carson Perry | Initial draft for review/discussion |
| 05/22/21 | Carson Perry | Adding more diagrams and explanations |
| 05/23/21 | Carson Perry | Added more OCD diagrams and description, Schizophrenia description and Model state diagram |

|  |
| --- |
| **Overall Instructor Feedback/Comments** |