Ethan Powell

Professor Henderson

CSCI 325

September 27, 2019

Object-Oriented Project: Requirements Documentation

Abstract:

Our project will be a text-based RPG game that will allow one user to play through a story that the user can interact with and make decisions in. The program will use a text-based system for entering commands and helping to determine player location in relation to objects in the story or walls and boundaries. Our team members include Steven Yeung and Ethan Powell. The location of our repository is RPG-team-project.

Description of Program:

The program will start the user off in an area such as a house or a field where the user can move in cardinal direction to navigate the terrain. The user then will be able to interact with objects in the environment, picking up items and letters to direct the player where to go next. Finally, the goal will be to escape the area by navigating through a labyrinth and solving puzzles.

UI for the user:

A screenshot of a cell phone

Description automatically generatedThe program is projected to look similar to this picture if our group decides to purse the full GUI experience:

Otherwise the program will be completely text based looking similar to this picture:

A screenshot of a cell phone

Description automatically generated

Division of labor:

Currently the labor is divided 50/50 with Steven working on Puzzles and Items, and Ethan working on a text parser for the user commands as well as the layout of the labyrinth.

UML Diagram:

A close up of text on a whiteboard

Description automatically generated

|  |  |
| --- | --- |
| Use Case (Name): | Move around |
| ID: | Move around |
| Scope: | public |
| Priority: | first |
| Summary: | This is a general use case for navigating the environment. |
| Primary Actor: | User |
| Supporting Actors: | N/A |
| Stakeholders: | * N/A |
| Generalization: | Yes |
| Include: | * None |
| Extend: | None |
| Precondition: | N/A |
| Trigger: | When user types a move command |
| Normal Flow: | User enters “Move forward/backwards/left/right”  Child class is called for respective direction  Checks if user is trying to move into a wall  If not the user is moved. |
| Sub Flows: | N/A |
| Alternate Flow/ Exceptions: | Displays “You can’t go that way” if user tries to move into a wall |
| Post-Condition: | N/A |
| Open Issues: | N/A |

|  |  |
| --- | --- |
| Use Case (Name): | Move Forward |
| ID: | Move Forward |
| Scope: | private |
| Priority: | secondary |
| Summary: | Moves the player forward |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * none |
| Extend: | none |
| Precondition: | The user enters “Forward” |
| Trigger: | Same as above |
| Normal Flow: | User enters “Forward”  Check if there is a wall  If so, displays “You can’t go that way”  If not, updates the related coordinate |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Move back |
| ID: | Move back |
| Scope: | private |
| Priority: | secondary |
| Summary: | Moves the player backward |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * none |
| Extend: | none |
| Precondition: | The user enters “Back” |
| Trigger: | Same as above |
| Normal Flow: | User enters “Back”  Check if there is a wall  If so, displays “You can’t go that way”  If not, updates the related coordinate |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Move left |
| ID: | Move left |
| Scope: | private |
| Priority: | secondary |
| Summary: | Moves the player left |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * none |
| Extend: | none |
| Precondition: | The user enters “Left” |
| Trigger: | Same as above |
| Normal Flow: | User enters “Left”  Check if there is a wall  If so, displays “You can’t go that way”  If not, updates the related coordinate |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Move right |
| ID: | Move right |
| Scope: | private |
| Priority: | secondary |
| Summary: | Moves the player right |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * none |
| Extend: | none |
| Precondition: | The user enters “Right” |
| Trigger: | Same as above |
| Normal Flow: | User enters “Right”  Check if there is a wall  If so, displays “You can’t go that way”  If not, updates the related coordinate |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Pickup Item |
| ID: | Pickup Item |
| Scope: | public |
| Priority: | first |
| Summary: | Has the user pickup an item |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * n/a |
| Extend: | Add to Inventory, Fail to Pickup |
| Precondition: | User has to be next to an item |
| Trigger: | User enters “Pickup” |
| Normal Flow: | User is next to an item  User enters “Pickup”  If item can be picked up, add to player’s inventory  If not, fail |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Add to Inventory |
| ID: | Add to Inventory |
| Scope: | private |
| Priority: | second |
| Summary: | The user picks up an item, he/she will add it to his/her inventory |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | n/a |
| Include: | * n/a |
| Extend: | none |
| Precondition: | User has to have found an item to pickup |
| Trigger: | User enters “Pickup” |
| Normal Flow: | User is next to an item  User enters “Pickup”  Item is successfully added to his/her inventory |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Fail to Pickup |
| ID: | Fail to Pickup |
| Scope: | private |
| Priority: | second |
| Summary: | The user has found an item but is unable to acquire it. |
| Primary Actor: | user |
| Supporting Actors: | n/a |
| Stakeholders: | * n/a |
| Generalization: | n/a |
| Include: | * n/a |
| Extend: | n/a |
| Precondition: | User must be next to an item |
| Trigger: | User might not have reached a certain cap. |
| Normal Flow: | The user is next to an item  User enters “Pickup”  The user has not reached a certain level/stage in the game  Item is not added to his/her inventory. |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Solve Puzzle |
| ID: | Solve Puzzle |
| Scope: | public |
| Priority: | first |
| Summary: | This is a general class for solving puzzles |
| Primary Actor: | User |
| Supporting Actors: | N/a |
| Stakeholders: | * n/a |
| Generalization: | yes |
| Include: | * n/a |
| Extend: | Solve correctly, solve incorrectly |
| Precondition: | User is interacting with a puzzle |
| Trigger: | User submits a solution |
| Normal Flow: | User interacts with a puzzle  User submits an answer  Check to see if correct or not  Go to specific case regarding outcome |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Solve Puzzle correctly |
| ID: | Solve Puzzle correctly |
| Scope: | private |
| Priority: | secondary |
| Summary: | This is a class regarding the correct solution to a puzzle |
| Primary Actor: | User |
| Supporting Actors: | N/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * n/a |
| Extend: | n/a |
| Precondition: | User submits correct answer |
| Trigger: | Same as above |
| Normal Flow: | User interacts with a puzzle  User submits an answer  User’s answer is correct  Puzzle Boolean is set to true  Puzzle drops item and disappears |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

|  |  |
| --- | --- |
| Use Case (Name): | Solve Puzzle incorrectly |
| ID: | Solve Puzzle incorrectly |
| Scope: | private |
| Priority: | secondary |
| Summary: | This is a class regarding the incorrect solution to a puzzle |
| Primary Actor: | User |
| Supporting Actors: | N/a |
| Stakeholders: | * n/a |
| Generalization: | no |
| Include: | * n/a |
| Extend: | n/a |
| Precondition: | User submits incorrect answer |
| Trigger: | Same as above |
| Normal Flow: | User interacts with a puzzle  User submits an answer  User’s answer is incorrect  Puzzle Boolean remains false  Puzzle remains for correct answer and will not drop an item when solved |
| Sub Flows: | n/a |
| Alternate Flow/ Exceptions: | n/a |
| Post-Condition: | n/a |
| Open Issues: | n/a |

Communication:

For communication, our team plans to use Slack as our main form of conversing and discussing code.