spaceship rpg gAME

Text Based Game

<https://github.com/Evickerest/rpg/tree/main>

CIS\_350 Winter 2024

Professor: Dr Nandigam

Team members

Adrian F Perez-Lopez

Eric Moras

Phuc Le

Contents

[Project Information 3](#_Toc160970652)

[Features 3](#_Toc160970653)

[Screenshots 4](#_Toc160970654)

[Project Plan 9](#_Toc160970655)

[Requirements & Definition 9](#_Toc160970656)

[Development 9](#_Toc160970657)

[Verification 9](#_Toc160970658)

[Maintenance 9](#_Toc160970659)

[Umbrella Activities 9](#_Toc160970660)

[Requirements & Specification 10](#_Toc160970661)

[Use-Cases Diagram 10](#_Toc160970662)

[Use-Cases Diagram Description 11](#_Toc160970663)

[Natural Language Requirements 14](#_Toc160970664)

[Traceability Matrix 14](#_Toc160970665)

[Design 15](#_Toc160970666)

[Code Standards 15](#_Toc160970667)

[Static Analysis 15](#_Toc160970668)

[Code Documentation 22](#_Toc160970669)

[Configuration Management 26](#_Toc160970670)

[Verification 27](#_Toc160970671)

[Unit Tests 27](#_Toc160970672)

[Code Coverage 27](#_Toc160970673)

[Requirements Coverage 31](#_Toc160970674)

[Postmortem 32](#_Toc160970675)

[Earned Value 32](#_Toc160970676)

[Variances 32](#_Toc160970677)

[Lessons Learned 32](#_Toc160970678)

[Work Cited 33](#_Toc160970679)

# Project Information

The Role-Playing Game (RPG) project is a GUI game that will display actions using text. The main character, the space janitor, faces different challenges while cleaning shipwrecks; therefore, he acquires different fighting techniques to help him move up in rank, such as cadet and captain, and ultimately becomes a space king.

As the character moves through different maps, enemies will get stronger; thus, the player will be rewarded at the end of each challenge. These rewards will help him increase his stats by acquiring various combat tools.

## Features

* Leveling
  + As the player defeats enemies and bosses, they gain experience points.
  + Experience points (XP) can be used to level up, giving access to “Stat” points.
  + These points can be distributed to different facets of the character, including:
    - Strength
    - Dexterity
    - Vitality
    - Intelligence
* Stats
  + This feature will display how the points are going to be distributed while playing; in addition, the player will be awarded 5 free points to get the game going
* Username Input.
  + The user interface will get the player’s name; otherwise, Bob will be assigned if user does not want to provide a name.
* Game Screen.
  + A map will be displayed showing the player current position.
  + Stat information to keep track of the game.
  + Message screen showing the interaction of the player.
* Procedurally Generated Map
  + As the game starts, a completely random map layout is generated.
  + Each map is unique to the game session.
  + Each map layout can be determined based of the round’s hardness.
* Equipment
  + As the player explores the map, they can come across chests that allow them to acquire completely random equipment.
  + Equipment can be selected and deselected using the inventory GUI.
* Turn Based Combat
  + Once a player engages in combat, the player and enemies take turns fighting.
  + On the players turn, they can:
    - Attack
    - Defend
    - Use Item (Med Kit)
  + On the enemies turn, they can also:
    - Attack
    - Defend
* Randomized Enemies
  + Each enemy stat and name are randomized upon entering combat.
  + The number of enemies per room is randomized as well.
  + The actions an enemy takes during combat are random.
* Boss
  + There is one boss room per map.
  + The boss is randomized and is harder than other enemies.
  + Upon beating the boss, the round is won.
* Character Skills
  + A player can choose how best to use their equipment by investing in “Stat” points how they see fit.
* Shops
  + Players can spend the in-game currency to buy better equipment.
* Graphical User Interface
  + GUI to display:
    - Inventory
    - Combat
    - Main Screen
    - End Screen
    - Shops

## Screenshots

|  |
| --- |
| Name Input and GUI Screen upon new Game  A screenshot of a computer game  Description automatically generated |
|  |
| Initial Player Stat Creation  **A screenshot of a video game  Description automatically generated** |
|  |
| Main Game Screen  A screenshot of a video game  Description automatically generated |
|  |
| Combat Screen  A screenshot of a computer  Description automatically generated |
|  |
| Game Over Screen  A screenshot of a computer  Description automatically generated |

|  |
| --- |
|  |
| Game Won Screen |
|  |
| Shop Screen  A screenshot of a computer game  Description automatically generated |

|  |
| --- |
|  |
| Stats Screen  A screenshot of a cell phone  Description automatically generated |
|  |
| Inventory Screen  A screenshot of a video game  Description automatically generated |

|  |
| --- |
|  |
| Example Game |

# Requirements & Specification

## 

## Use-Cases Diagram

A diagram of a diagram

Description automatically generated

## Use-Cases Diagram Description

|  |  |
| --- | --- |
| Name: | Create User |
| ID | RPG\_01 |
| Brief Description. | The user can provide a name, which will appear on the game screen; however, if the user does not feel comfortable giving the name, the game software will default the player’s name to Bob. |
| Actors  Primary/Secondary | Player. |
| Triggers | New user who wants to play the game. There is no save game for now; therefore, returning users will have to start from square one. |
| Precondition | None |
| Primary Flow | 1. Player provides the name. 2. Start options will be displayed on the main screen to continue. 3. Exit button will be available if the player doesn’t want to continue. 4. The player refuses to provide name but want to continue. 5. Repeat option one workflow. 6. The player refuses to provide name and does not want to continue. 7. Exits game. |
| Alternative Flow | None. |
| Minimal Guarantee |  |

|  |  |
| --- | --- |
| Name: | Game Description |
| ID | RPG\_02 |
| Brief Description. | The player will be welcomed, and the game description is provided. |
| Actors  Primary/Secondary | Player. |
| Triggers | User must initiate Create User case for this action. |
| Precondition | None. |
| Primary Flow | The user reads instructions and clicks button to continue. |
| Alternative Flow | None. |
| Minimal Guarantee |  |

|  |  |
| --- | --- |
| Name: | Create Game |
| ID | RPG\_03 |
| Brief Description. | Players will check the initial stats and how they will be distributed and start the game button to create the game. |
| Actors  Primary/Secondary | Player |
| Triggers | Game Description. |
| Precondition | Players must follow the previous step to create the game. |
| Primary Flow | Because it is the initial phase, the player will receive five free points to play the game, and they don’t have to do anything else but start the game. |
| Alternative Flow | Player can manipulate or rearrange the stats in which the options are the following:   1. Strength 2. Dexterity. 3. Vitality. 4. Intelligence.   By changing the above settings, the player will lose one point for each change. |
| Minimal Guarantee | N/A |

|  |  |
| --- | --- |
| Name: | Choose Map/Start Game |
| ID | RPG\_04 |
| Brief Description. | After the game is created, the player will have three paths, each with a map. When a path is chosen, and the player starts selecting the given options, the player will be able to see their location on the map. |
| Actors  Primary/Secondary | Player |
| Triggers | N/A |
| Precondition | Start the Game. |
| Primary Flow | Players will read a message stating which part of the ship they want to explore/clean first. Also, before the player starts the game, they can double-check the stats and readjust them if needed. The inventory option will also be available on the main screen.   1. Entering a room: 2. Nothing happens and can rest after a battle. 3. Chest is found and key is needed to open. If lucky, good items can be found and taken. 4. Villains in the room and player goes into battle mode, or they can run. |
| Alternative Flow | N/A |
| Minimal Guarantee | N/A |

|  |  |
| --- | --- |
| Name: | Challenge Boss |
| ID | RPG\_06 |
| Brief Description. | To clear the room or current map, the player must face a stronger opponent(boss); defeating this character will make the player move to the next level. |
| Actors  Primary/Secondary | Player. |
| Triggers | Overcome all challenges that are assigned in each map. |
| Precondition | Chose Map. |
| Primary Flow | After clearing all the previous rooms and arriving at this stage, the player is worn out but collects different items to help them get back in shape(energetic). However, that last room, where the big boss resides, has a different feeling than the previous rooms in which the player realizes he is in trouble. What will he do?   1. **Fight:** The player defends himself and applies skills gained from previous battles. 2. **Run**: the main door is locked upon entrance.    1. Other doors are available but will suck him out of the battleship. |
| Alternative Flow | If the player did not collect boost items and does not have enough to face boos. Reset game for that specific map. |
| Minimal Guarantee | N/A |

|  |  |
| --- | --- |
| Name: | Reset Game |
| ID | RPG\_07 |
| Brief Description. | Players did not collect enough items to fight, boss can reset game, or if there is an issue with the software, administrator can rest the game without losing players collected items. |
| Actors  Primary/Secondary | Player  Administrator |
| Triggers | Unable to defeat boss or software issue. |
| Precondition | N/A |
| Primary Flow | 1. Because players must accumulate points and stats to play against the main villain character, they should have what it takes for the battle. However, if the player cannot win, he can reset the game and start again. 2. If the player cannot finish the challenge because of a software issue, the administrator can work on the problem and reset the game without losing the rewards collected if:    1. It is a software issue (bug)    2. A player testing the software. |
| Alternative Flow | N/A |
| Minimal Guarantee | N/A |

|  |  |
| --- | --- |
| Name: | End Game |
| ID | RPG\_08 |
| Brief Description. | When opting to end game, player’s window will be terminated and will have to start all over again (for now). |
| Actors  Primary/Secondary | Player  Administrator |
| Triggers | Finish all Leves |
| Precondition | N/A |
| Primary Flow | These options will be available for every player if they want to terminate the game anytime.  Players will be warned that they cannot recover any record if they want to log back in by closing or ending the game.  If it is determined that the player is cheating the system, the game will be terminated as well. |
| Alternative Flow | Reset Game |
| Minimal Guarantee | N/A |

# Design

A diagram of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

The class diagram below was designed using Lucidchart, and because some of the classes contain a lot of instances and methods, we decided to include the Class name only. An example of the MainGui class is shown below the class diagram displaying detailed information.

# Code Standards

Tried to adhere to Python’s PEP8 code style format. We ran pycodestyle <directory> from the terminal to find the problems on the next page. The majority of them (the E501 line too long errors) were left unresolved because there wasn’t a good break-point to continue it on a new line. Either the function calls themselves were too long or we were simply nesting a lot of calls in one line. There were also a few E731 do not assign a lambda expression, use a def errors that I left alone because I didn’t know how to rework them. I couldn’t find a way to get a numerical score on this but, considering that it’s just E501 errors, I think we got a 9/10 easily.

## rpg\Classes\text\_printer.py:38:13: E731 do not assign a lambda expression, use a def

## rpg\Classes\text\_printer.py:39:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\character\_gui.py:36:80: E501 line too long (94 > 79 characters)

## rpg\Classes\GUI\chest\_gui.py:37:80: E501 line too long (81 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:43:80: E501 line too long (97 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:44:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:66:80: E501 line too long (90 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:67:80: E501 line too long (85 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:78:80: E501 line too long (91 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:79:80: E501 line too long (86 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:83:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:86:80: E501 line too long (91 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:87:80: E501 line too long (86 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:93:80: E501 line too long (81 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:94:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:95:80: E501 line too long (90 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:102:80: E501 line too long (95 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:103:80: E501 line too long (90 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:265:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\fight\_gui.py:267:80: E501 line too long (91 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:37:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:50:80: E501 line too long (100 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:64:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:67:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:68:80: E501 line too long (88 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:73:80: E501 line too long (82 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:76:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:77:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:82:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:85:80: E501 line too long (87 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:86:80: E501 line too long (83 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:92:80: E501 line too long (95 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:93:80: E501 line too long (90 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:111:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:112:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:114:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:115:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:117:80: E501 line too long (97 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:118:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:120:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:121:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:123:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:124:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:126:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\inventory\_gui.py:127:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:62:80: E501 line too long (86 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:63:80: E501 line too long (88 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:85:80: E501 line too long (83 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:98:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:99:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:104:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:105:80: E501 line too long (81 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:107:80: E501 line too long (87 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:118:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:119:80: E501 line too long (86 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:182:80: E501 line too long (85 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:203:80: E501 line too long (88 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:207:80: E501 line too long (85 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:253:80: E501 line too long (91 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:254:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:285:80: E501 line too long (81 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:294:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:301:80: E501 line too long (83 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:304:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:306:80: E501 line too long (91 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:307:80: E501 line too long (82 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:310:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:311:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:317:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:319:80: E501 line too long (87 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:355:13: E731 do not assign a lambda expression, use a def

## rpg\Classes\GUI\main\_gui.py:458:80: E501 line too long (85 > 79 characters)

## rpg\Classes\GUI\main\_gui.py:559:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:38:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:49:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:51:80: E501 line too long (87 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:69:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:72:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:73:80: E501 line too long (88 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:78:80: E501 line too long (82 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:81:80: E501 line too long (89 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:82:80: E501 line too long (84 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:87:80: E501 line too long (83 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:90:80: E501 line too long (95 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:91:80: E501 line too long (85 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:96:80: E501 line too long (80 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:99:80: E501 line too long (87 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:100:80: E501 line too long (82 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:107:80: E501 line too long (81 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:108:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:109:80: E501 line too long (90 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:113:80: E501 line too long (86 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:115:80: E501 line too long (88 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:116:80: E501 line too long (83 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:134:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:135:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:137:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:138:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:140:80: E501 line too long (99 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:141:80: E501 line too long (93 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:143:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:144:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:146:80: E501 line too long (98 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:147:80: E501 line too long (92 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:149:80: E501 line too long (100 > 79 characters)

## rpg\Classes\GUI\shop\_gui.py:150:80: E501 line too long (93 > 79 characters)

## Static Analysis

We used the Pylint plugin to analyze and verify the project's compliance; the results of each class are listed below.

Pylint Static Code Analysis

|  |
| --- |
| \*\*\*\*\*\*\*\*\*\*\*\*\* Module GUI.main\_gui  Classes\GUI\main\_gui.py:12:0: R0904: Too many public methods (22/20) (too-many-public-methods)  \*\*\*\*\*\*\*\*\*\*\*\*\* Module Map.map  Classes\Map\map.py:56:8: C0206: Consider iterating with .items() (consider-using-dict-items)  \*\*\*\*\*\*\*\*\*\*\*\*\* Module Map.mapconstants  Classes\Map\mapconstants.py:9:0: R0903: Too few public methods (0/2) (too-few-public-methods)  \*\*\*\*\*\*\*\*\*\*\*\*\* Module Rooms.room  Classes\Rooms\room.py:99:20: E1120: No value for argument 'seq' in method call (no-value-for-parameter)  ------------------------------------------------------------------  Your code has been rated at 0.00/10 (previous run: 0.00/10, +0.00) |

# Justification

The original code analysis results from Pylint are not what is listed above. However, some warnings from Pylint can be ignored as we have justification for keeping the code the way it is:

* “Lambda may not be necessary.”
  + We get a lot of warnings for usage of lambda statements. However, lambdas are necessary for making Tkinter buttons working and other functions.
* “Too many instance attributes”
  + The max value for instance attributes that Pylint allows is 7. However, due to the way Tkinter works, 7 attributes are way too low for our classes. In release 2, we may be able to get the number of attributes down.
* “Similar lines in 2 files”
  + Pylint noticed that many of our GUI classes have the same code. However, the code that is mentioned is set up code that is necessary to be in the GUI classes.

The other results from above follow similar justifications for leaving them in. We don’t know why pylint returns a score of 0.

## Code Documentation

For code documentation, we used the pdoc API to document our code. Pdoc covers all classes, tests, and main files. Pdoc uses the definitions, arguments, and parameters that we have defined in multiline strings.

Link to class documentation html in GitHub: <https://github.com/Evickerest/rpg/blob/main/html/rpg/index.html>

index.html can also be found the project at RPG/html/rpg/index.main.

As there are too many classes to show, here are some image examples of the HTML Documentation:

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

## Configuration Management

Since we were using git/GitHub to track our project, each group member worked on developing the game software and pushed it into the remote repository when possible. Although we did not have a team lead, Eric would briefly describe what needed to be done so we could work on it when possible.

When committing the changes, we started with a brief description of each group member's changes. However, it took effort to follow sometimes, and we discussed that a detailed description would help us understand the changes better.

Although we did not write detailed information when committing, we used Discord constantly to describe what was done after pushing the changes to the GitHub repository; thus, it helped us better track the changes.

The URL address for the project is listed below.

<https://github.com/Evickerest/rpg.git>

## Code Verification

# Unit Tests

Unit Tests Done Using Python’s unittest module. Tests were separated by what classes they test and stored into the “Tests” folder. To run the tests, we made a “test.py” file which complied all the tests into a suite and ran had the runner fun the tests from there.

# Code Coverage

Automated unittest Coverage Report

A screenshot of a computer

Description automatically generated

The GUI files were tested both manually and through unit test.

FightGUI.py

* Manual testing was performed.
  + General Testing
    - Can’t leave until all enemies are killed.
    - Can’t open another window while a FightGUI window is still active.
    - Player and enemy information are correctly displayed and updated.
  + Enemy Targeting Entry Box
    - Retains previously attacked enemy
      * Clears itself after an enemy is killed
  + Attack Button
    - Does nothing if no enemy or invalid enemy is targeted (Invalid integer target, attempted to target enemy name, left targeting box empty, etc.).
  + Defend Button
    - Correctly increases player’s defense by 50% for 1 turn before reducing it back to normal.
  + Use Medkit Button
    - Correctly heals the player and uses a medkit.
    - Heals nothing if no medkits are left but still acts as a full turn.
  + Placeholder Button
    - Can be pressed but nothing will happen.
  + Enemy Killed
    - Player gains XP and Credits (currency) upon killing an enemy.
    - Enemy is correctly removed from the list of active combatants.
  + Player Killed
    - Destroys the window and prompts the end game method.
  + Exit Button
    - Available once all enemies are killed,
    - Deletes all buttons except for the use medkit button.
    - Cannot leave unless all enemies are killed.

ChestGUI.py

* Manual testing was performed.
  + General Testing
    - Can leave at any time.
    - Can’t open another window while a ChestGUI window is still active.
    - Item information is correctly displayed and updated.
      * What item can be looted.
      * How much it’s worth in Credits.
  + Loot Button
    - Correctly adds the item to your inventory.
    - Updates player inventory (backend).
    - Correctly closes window and runs exit\_room method.
  + Scrap Button
    - Correctly removes the item from the room and adds its Credit value to the player.
    - Correctly closes window and runs exit\_room method.

ShopGUI.py

* Manual testing was performed.
  + General Testing
    - Can leave at any time.
    - Can’t open another window while a ShopGUI window is still active.
    - Player and shop information are correctly displayed and updated.
  + Item Targeting Entry Box
    - Correctly allows you to select the various items.
    - Retains previously selected item.
  + Unequip Button
    - Correctly unequips the targeted item from your equipment and adds it to your inventory.
      * Replaces said equipment slot with a placeholder item.
    - Updates player information.
  + Equip Button
    - Correctly equips the targeted item from your inventory.
      * Swaps out equipment if necessary.
        + Adds previously worn equipment to inventory.
      * Updates player information.
  + Purchase Button
    - Buys item from shop.
      * Correctly adds it to the player’s inventory, removes it from the shop’s, and deducts player’s credits.
  + Sell Button
    - Sells item from inventory.
      * Correctly adds it to the shop’s inventory, removes it from the player’s, and adds player’s credits.
  + Buy Medkit Button
    - Buy medkits in exchange for Credits.
      * Confirmed that you can only buy if the player has enough credits.
    - Correctly updates player information.
  + Close Button
    - Available always.
    - Destroys the window.

InventoryGUI.py

* Manual testing was performed.
  + General Testing
    - Can leave at any time.
    - Can’t open another window while an InventroyGUI window is still active.
    - Player information is correctly displayed and updated.
      * Equipment
      * Inventory
      * Total stats
  + Item Targeting Entry Box
    - Correctly allows you to select the various items.
    - Retains previously selected item.
  + Unequip Button
    - Correctly unequips the targeted item from your equipment and adds it to your inventory.
      * Replaces said equipment slot with a placeholder item.
    - Updates player information.
  + Equip Button
    - Correctly equips the targeted item from your inventory.
      * Swaps out equipment if necessary.
        + Adds previously worn equipment to inventory.
      * Updates player information.
  + Drop Button
    - Correctly removes an item from the player’s inventory.
      * Fully deletes item, no way of regaining it.
  + Close Button
    - Available always.
    - Destroys the window.

CharacterGui.py

* Manual testing was performed.
  + General Testing
    - Can’t open window if another window is already present.
    - Can’t open another window until this window is destroyed.
  + LV UP Button
    - Checked that the lv-up button displays only if the player has enough XP to level up.
    - Also checked that the button disappears after a level up if the player no longer has enough XP to level up.
    - Lv-up also correctly adds 5 stat points.
  + Add Stat Points Buttons
    - Checked that the add stat points buttons work.
    - Also checked that the buttons only show if the play has unassigned stat points.
    - Also confirmed that leveling up with 0 stat points (and thus no + buttons) will create new + buttons.
  + Close Button
    - Works correctly and destroys the window.

MainGui.py and gamehandler.py

* Manual testing was performed via playing the game.
  + Main Menu
    - Exit Button works.
    - Start Button works.
    - Username entry box works.
      * If left empty, it sues the default player name instead.
  + Game Intro Screen
    - Displays the correct information.
      * Updates player name when necessary.
    - Click to Continue Button works correctly.
  + Character Creation Screen
    - Displays and updates player information.
    - Stat Increase and Decrease Buttons work correctly.
      * Prevents stats from going under 3 or over 12.
      * Player Max Health and current Health updates correctly when Vitality changes.
  + Game Over
    - Correctly displays information about your run (Time, Enemies Killed, Rooms Entered, Player Stats).
    - Correctly tells if you beat the boss (Victory)
      * Offers an exit button that works.
    - Correctly tells if you died (Loss)
      * Offers a retry with a new game button (Currently nonfunctional)
        + No buttons work in the new game.

## Roles/Responsibilities

Phuc Le

* Role – Product Dev. Team
* Specific Work
  + Unit Testing (With unittest module)
  + Source Code Documentation
  + Static Code Analysis (Pylint)
  + Code Standard Report (PEP8 Style)
  + GUI Implementation (Fight-, Chest-, Shop-, Character-, Inventory- GUIs, worked on player-system interactions except for graphics)
    - Character Creation as well
  + Implementation of the Character, Player, Enemy, and Item classes
* Self Reflection
  + Git was a lot more annoying than I expected. Merge conflicts were insanely difficult to resolve in PyCharm for some reason.
    - I wish we met up more often than ~2 weeks or so.
  + I was definitely very scattered in terms of what work I did and didn’t do so good at keeping the team up to date on what I was changing.
  + I also wish I helped out more on Tkinter specific code. I did not get the eventloop or how different layers of windows worked.

Adrian Perez:

* Gui dev. Team
* Documentation
* Design.

Reflection:

My lack of coding skills has put me in a very uncomfortable position because not contributing as much as my other team members is unfair. However, I try my very best to assist when I can, and of course, Git is not helping at all. I cannot say it all the time, but it has happened that I had to delete the entire project and clone it again because I cannot push and pull my work to or from GitHub.

The other two members have helped assist me when they can; thus, I will continue to work hard for the next phase.

Eric Moras:

* Role: Project Leader
* Specific Work:
  + Managed Internal Logic
    - Map and Edge classes.
    - Game handler functionality.
    - Is Game Won
  + Handled setting up meetings and discussing what needs to be done.
  + Oversaw how classes interacted with each other.
    - Subdivided the projects into necessary classes.
    - Managed the class injection into MainGUI and other GUIs.
  + Made some of the GUI.
    - Game loss and Win screens.
* Self-reflection:
  + Going in, I expected this to by like my last semester project where I just code and don’t really learn anything. But from this, I feel like I learned a lot. Going with GUI was a good idea because I was able to learn a lot about Tkinter and how to display something on the screen. Also being able to utilize what I have learned in class in an actual project has really helped cemented the concepts of software engineering for me. However, the biggest thing I feel like I have learned is working with people. I haven’t really worked on a project to completion before like this one. I do volunteer as a programming mentor at a robotics team, however, much of the code that I helped with is repeated code from previous years. Working with people taught be that programming isn’t much about getting done what you want to get done, its also thinking about what other people need to do and how that will all come together.

## 

## Work Cited

*Artworks by Artists from around the World | Artstation*, www.artstation.com/marketplace/game-dev/artworks. Accessed 7 Mar. 2024.

Creative, Make It. “Lights and Shadows: Sci-Fi Lift: After Effects.” *Behance*, 1999, www.behance.net/gallery/44429607/Lights-and-Shadows-Sci-fi-lift-After-Effects.

“Inside a Sci-Fi Spaceship Escape Pod That Is a Mind: Stable Diffusion.” *OpenArt*, openart.ai/discovery/sd-1006280114948030534. Accessed 9 Mar. 2024.

“Lifegoods.Com.” *MY*, my-lifegoods.com/. Accessed 9 Mar. 2024.

Santamarian, Ioana. *Retro Sci-Fi Kitchen, Ioana Santamarian*, 2019, www.artstation.com/artwork/2xZYWJ.

“Sci Fi Space Station Interior in Environments - UE Marketplace.” *Unreal Engine*, 1999, www.unrealengine.com/marketplace/en-US/product/sci-fi-space-station-interior.

*Sci-Fi Hangar Concept, Daniel Jakobi*, www.artstation.com/artwork/klgkbx. Accessed 7 Mar. 2024.

“The SHIPMAS Season Approaches: Countdown to 9 New Star Atlas Spaceships.” *Medium*, Star Atlas, 10 Dec. 2021, medium.com/star-atlas/shipmas-9-new-star-atlas-spaceships-f24afc7956b1.

“Spaceship Cabin.” *Cool Ideas in Photos*, en.idei.club/8121-spaceship-cabin.html. Accessed 1 Mar. 2024.

“Spaceship Hallway C4d 1920x1080 by Xcustomgraphix on DeviantArt.” *By xCustomGraphix on DeviantArt*, www.deviantart.com/xcustomgraphix/art/Spaceship-hallway-C4D-1920x1080-407417487. Accessed 7 Mar. 2024.

*Wallpaper Abyss - HD Wallpapers, Background Images*, wall.alphacoders.com/. Accessed 9 Mar. 2024.