# CliffHanger

## Incremental and Regression Testing

## Classification of Components

1. **Front:** The front handles UI and WebGL portion of our web application it is responsible for what a user sees and interacts with. This section is composed of a JavaScript running on ThreeJS.

    a. **Animator:** The animators are a set of functions that the backend uses to interact with the UI. This can perform graphical actions such as player movement, falls, drops etc.

    b. **GameStage:** The game stage is used to initialize the UI and run the main thread. It controls parameter that are global to the game like lighting and graphical effects.

        i. **Global:** It houses variables that are used globally in the game stage.

        ii. **Builder:** It initializes all global game elements and build the game platform on screen based on the size the backend provides.

        iii. **Main:** It runs the main animation frame which is responsible to render all actions and it also take mouse input for orbital controls.

    c. **3DObjects:** This class acts as an interface for the back end to load STL objects and get references to said objects for manipulation.

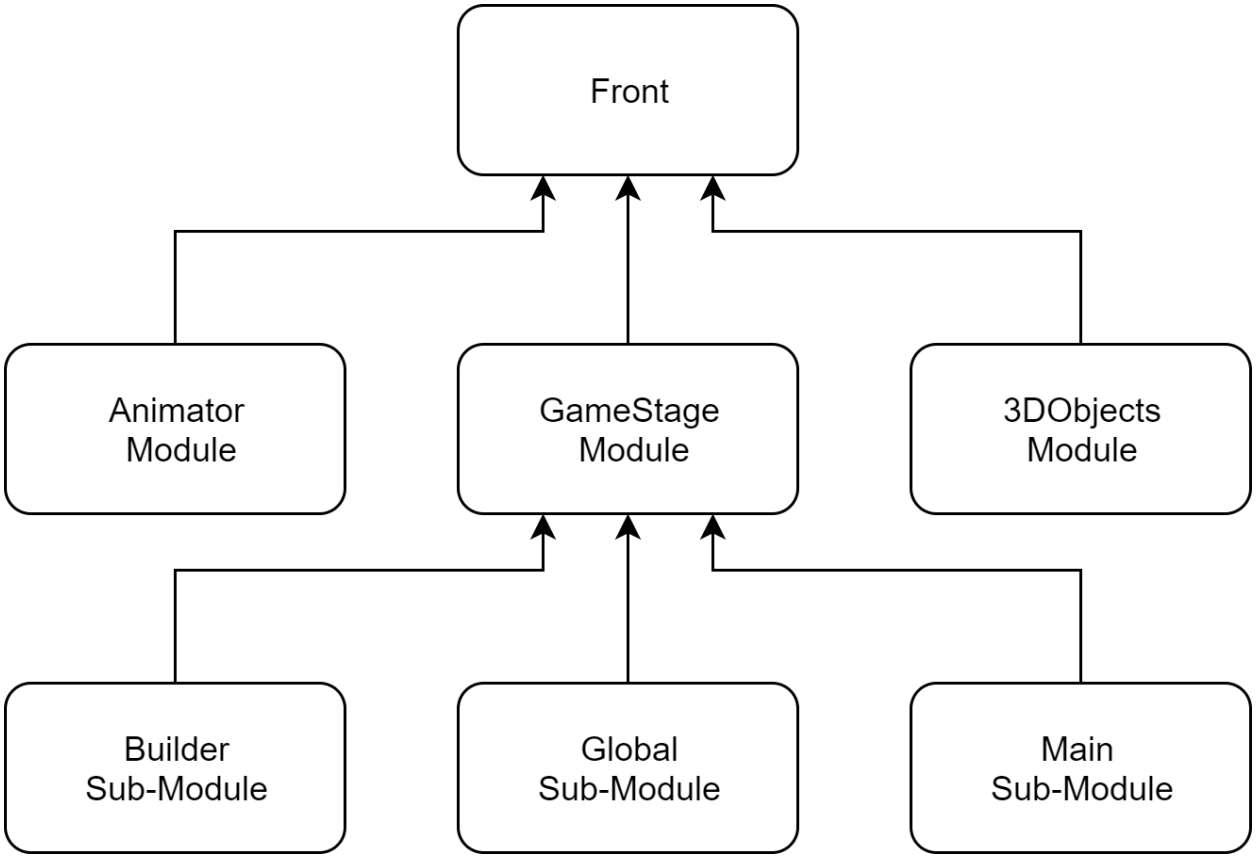


*Figure 1.* The Front-end

2. **Back:** The back handles the a mathematical representation of the game state. It uses arithmetic to compute game actions and command the UI to display changes.

    a. **GamePlay:** This section is specifically responsible for handling a single seat game.

        i. **Item:** It handles the items that a player has, uses and gets used on. It also handles the distribution of items on the stage.

        ii. **Player:** It handles player statistics, actions and items. It is used to account for user interaction in the backend.

        iii. **BoardSpace:** The board space is responsible for sizing the stage. Periodic reduction in board size, and positioning of everything on the board.

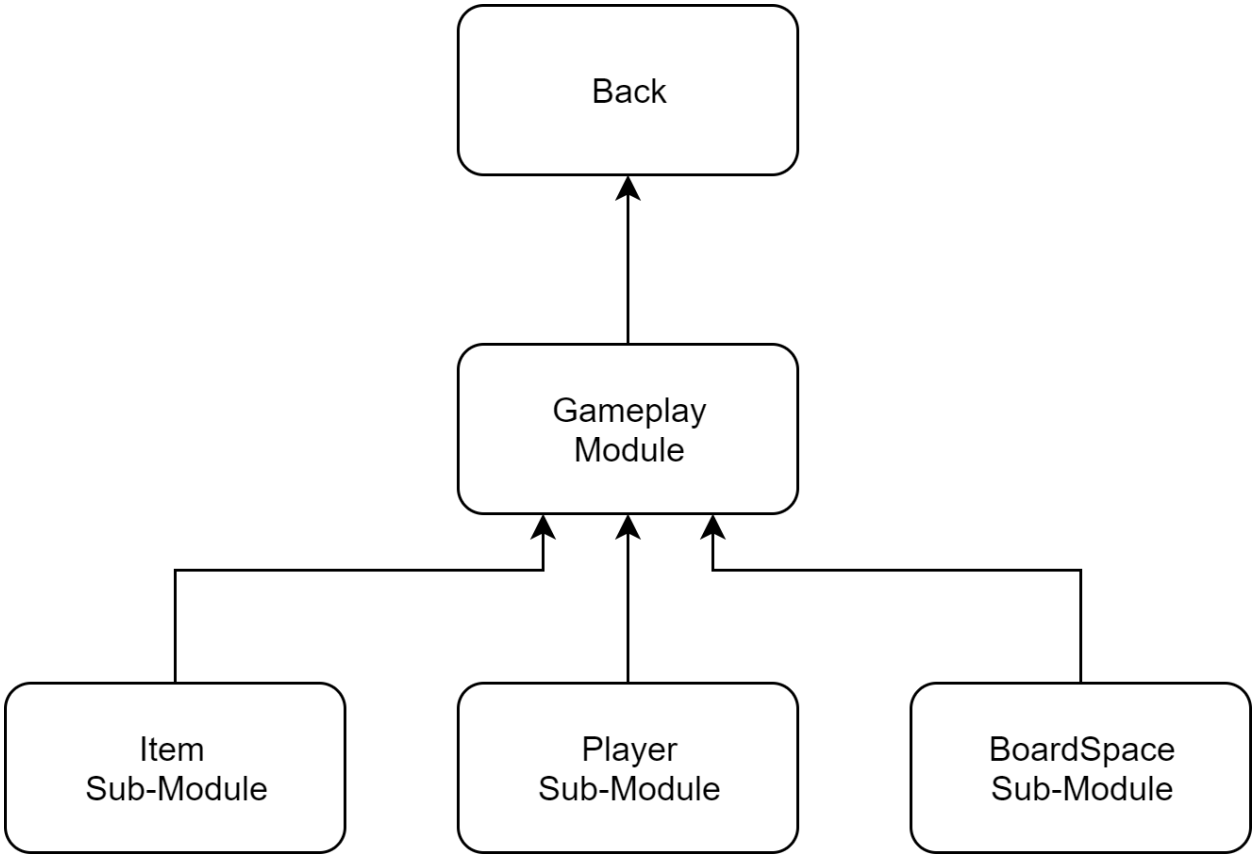


*Figure 2.* The Back-end

# Type of Incremental Testing Used:

We decided to go with a **bottom-up** approach. Our design allowed each individual to work on multiple separate and independent modules. This meant that we could test them independently and them put them together to test them. The integration of sub modules took some effort but since we had a design in mind, it worked out excellently. The bottom-up approach allowed us to do development more independently so we wouldn't have to wait for each other to test something. Our testing was also **automated** using **NodeJS**, **Jest** and **Mocha**. To run a test we went to the module directory and ran the command.

```
npm test
```

Here is an example of one such output.

```
C:\Users\mailb\Desktop\Anirudh Pal\PU\JUNIOR\SEM 6\CS
408\CliffHanger\Front\GameStageDev>npm test

> gamestagedev@1.0.0 test C:\Users\mailb\Desktop\Anirudh
Pal\PU\JUNIOR\SEM 6\CS 408\CliffHanger\Front\GameStageDev
> jest

 PASS  .\gameStageRegression.test.js
  √ Test if Screen Width and Height are correctly obtained. (3ms)
  √ Test if Camera has right Aspect Ratio and not 1.
  √ Test if Camera FOV is resonable (1ms)
  √ Test if Camera Far Plane is resonable
  √ Test if Camera Near Plane is resonable (1ms)
  √ Test if Ambient Light has Color that is Not Default
  √ Test if Point Light has Color that is Not Default
  √ Test if Material has Color that is Not Default
  √ Test if Geometry Size is resonable Large
  √ Test if Geometry Size is resonable Small

Test Suites: 1 passed, 1 total
Tests:       10 passed, 10 total
Snapshots:   0 total
Time:        1.441s
Ran all test suites.
```

## Incremental Testing

| **Module** | **GameStage** | | |
|---|---|---|---|
| **Sub-Module** | **Global** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |

| | | | |
|---|---|---|---|
| **1** | When choosing a random position to drop an item, the random number generated was multiplied by the wrong board size. As it did not account for changing the size after shrinking. | High | Added the proper formula to get the current board size. |
| **2** | The color assignment was done though decimal representation. | Low | All meshes were changed to type Lambert, Phong or Basic. Tests were added to test correct types of meshes. |
| **3** | Lights was assigned types that was not expected (eg, point light instead of ambient light and vise versa. | High | Check if the correct type of light is passed and add tests for the constructors. |
| **4** | Failure of other functions due to undefined scene to add objects to. | High | Verify that scene was created and has a matrix. |
| **5** | Camera was not showing the play board at a visible angle. | Medium | Define a new Frame of view and use it as the initial position of the camera when starting the game. |

| **Module** | **Animator** | | |
|---|---|---|---|
| **Sub-Module** | **Global** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |

| | | | |
|---|---|---|---|
| **1** | Passing meshes or models to color change method was not working properly for different functions in animations, as there are a number of different objects on screen. | High | Instead of passing objects to another js file function, we created a global array of objects, and simply passed indices of the models in the array, making it much simpler to use inside of different animation functions. |
| **2** | Not all meshes on screen were responding to change in properties such as color. | Low | All meshes were changed to type Lambert, Phong or Basic. Tests were added to test correct types of meshes. |
| **3** | The visibility of plates falling was not changed to false after certain z-position threshold. | High | Added tests to check the visibility of plates. Upon animation of falling, the visibility must be false and can be checked using the tests. This made sure that the plates were visible and invisible |
| **4** | Number of plates initialized on screen was incorrect as the for loops used for initialization were using single index. | Medium | Added test to check for initial number of components in the plate array. Corrected the for loop logic in code. |

| **Module** | **GamePlay** | | |
|---|---|---|---|
| **Sub-Module** | **Item** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |

| | | | |
|---|---|---|---|
| **1** | The aspect ratio of the camera was wrong because of using width by width instead of using width by height. | Medium | The change was made in the source file in addition to the creation of a test to check that the aspect ratio is width by height. |
| **2** | *possibleAttacksBy*(item) did not initially account for the board shrinking. | High | Added a conditional check using predefined variables which keep track of the boundaries of the boardspace. |
| **3** | *useItem*() accepted a string parameter which described which direction the user wanted to use an item. Not all items fit this description which caused some items to not work properly. | Low | *useItem*() now accepts a boardspace parameter. Calculations are done using the player's current position and the boardspace to find the direction the user intends to use an item. |

| **Module** | **Gameplay** | | |
|---|---|---|---|
| **Sub-Module** | **Player** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |
| **1** | *killPlayer*() only removed the player from the linked list but did not remove them from the boardspace they were on. | Medium | Set the corresponding boardspace's player attribute to NULL. |

| | | | |
|---|---|---|---|
| **2** | *healHealthBy*(amount) in *Player*.js would heal the player's health beyond the maximum limit (100) and *damageHealthBy*(amount) would decrement it beyond the *minimum limit* (0). | Medium | Added conditions to make sure it does not exceed the boundaries. |
| **3** | Lights was assigned types that was not expected (eg, point light instead of ambient light and vise versa. | High | Check if the correct type of light is passed and add tests for the constructors. |

# Regression Testing

| **Module** | **GameStage** | | |
|---|---|---|---|
| **Sub-Module** | **Global** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |
| **1** | Scalability of objects created such as the plates of the board being to small for the user. | Medium | Created a test to check that the values assigned is considered reasonably big to be viewed. |
| **2** | The cameras were assigned to close to the board that the board didn't look like it exists. | High | Define a range check that is not to close and not to far so that the camera would include the whole board. |
| **3** | The aspect ratio being assigned to 1 due to wrong height and width assignment. | Medium | Have a check to ensure that the aspect ratio is not 1 to prevent wrong height and width assignments. |

| Module | Animator | | |
|---|---|---|---|
| **Sub-Module** | **Global** | | |
| **Defect Number** | **Problem** | **Severity** | **Solution** |
| **1** | Adding color change code in move animation caused errors, upon keypress c. | Medium | Improper passing of index of object in array was the issue. The solution was to remove the *findIndex()* method for sending index. |
| **2** | Color change was not taking place due to use of incorrect method. | Medium | Change material type to Lambert Material because Normal material type doesn't let change of color. Then used setHex method to change colors on keypress c. |

| Module | GamePlay | | |
|---|---|---|---|
| **Sub-Module** | **Player** | | |
| **Defect No.** | **Description** | **Severity** | **Solution** |
| **1** | Making *useItem*()'s parameter a string did not allow the UI script to communicate the users' intentions well enough. | High | *useItem*() now has a parameter for the boardspace the user selects. Within *useItem*(), that boardspace is used to calculate which direction the user is attacking or moving. |
| **2** | Using variables for boundary checks in *shinkBoard*() led to issues | Low | Switched to constants. Issue with logic fixed. |

| | | | |
|---|---|---|---|
| | displaying the shrinking board in the UI. | | |
| **3** | Adding a status variable to Player objects broke all instances of Player objects. | High | Adjusted each Player object instantiation to include the status variable. |
| **4** | Adding color change code in move animation caused errors, upon keypress c. | Medium | Improper passing of index of object in array was the issue. The solution was to remove the *findIndex()* method for sending index. |
| **5** | Dropping an item does not show on the test board. | Low | Print the index of the item on the block where the item was dropped. |

# Update Product Backlog

| Backlog ID | Functional Requirement | Completed | Hours |
|---|---|---|---|
| 1 | As a user, I would like to play a good looking game. | | 30 |
| 2 | As a user, I would like to read the rules of the game. | | 2 |
| 3 | As a user, I would like to move on my turn. | DONE | 10 |
| 4 | As a user, I would like to attack another player on my turn. | DONE | 10 |
| 5 | As a user, I would like to push another player on my turn. | DONE | 10 |
| 6 | As a user, I would like to see what special items I have. | | 6 |
| 7 | As a user, I would like pick up special items. | | 6 |
| 8 | As a user, I would like to utilize the special items. | | 12 |
| 9 | As a user, I would like to see a description of special | | 2 |

| | | | |
|---|---|---|---|
| | items. | | |
| 10 | As a user, I would like to pick which direction I want to move. | DONE | 4 |
| 11 | As a user, I would like to pick which direction I want to attack. | DONE | 4 |
| 12 | As a user, I would like to pick which direction I want to push. | DONE | 4 |
| 13 | As a user, I would like to set up traps on the playing field. | | 8 |
| 14 | As a user, I would like to see which floor tile will fall next. | DONE | 10 |
| 15 | As a user, I would like to see my health bar. | DONE | 4 |
| 16 | As a user, I would like to see my stats (weapon,items,etc). | | 12 |
| 17 | As a user, I would like to consume a health potion. | | 2 |
| 18 | As a user,I would like to know which player turn it is. | DONE | 4 |
| 19 | As a user, I would like each turn to take a limited amount of time so the game will go faster. | | 4 |
| 20 | As a user, I would like to see how much time is left in each turn. | | 4 |
| 21 | As a user, I would like to end my turn. | DONE | 2 |
| 22 | As a user, I would like to play with other people. | DONE | 10 |
| 23 | As a user, I would like to use the mouse/keyboard to control the game. | DONE | 10 |
| 24 | As a user, I would like to be shown a win screen upon winning. | | 4 |
| 25 | If time allows, allow players to connect from different computers to play the game. | | 12 |
| 26 | If time allows, create a leaderboard and track user statistics. | | 12 |

| | | |
|---|---|---|
| **TOTAL** | **82 hrs Completed** | **198** |