Sprint 2 Incremental Testing and Regression Testing

Team 23: Akhil Agrawal, Jisoo Cha, Alex Geier, Shivan Desai, Youshin Kim

# Classification of Components

* Start
  + input: None
  + output: None
  + Parent Dependency: None
  + Child Dependency: Main page
* Main page
  + input: User’s login session
  + output: Leaderboard information (show when logged in, not show when logged out)
  + Parent Dependency: LeaderBoard, Logout Page, login Page, sign up page, break page
  + Child dependency: LeaderBoard, Logout Page, login Page, sign up page, break page
* Logout page
  + input: None
  + output: None
  + parent Dependency: main page
  + Child dependency: main page
* sign up Page
  + input: Email, password, Display Name
  + output: None
  + parent Dependency: login page, break page, main page
  + Child dependency: login page, break page, main page
* Login Page
  + input: email, password
  + output: credential validation
  + parent Dependency: main page, break page, sign up page
  + Child dependency: main page, break page, sign up page
* Break Page
  + input: None
  + output: None
  + parent Dependency: main page, sign up page, login page
  + Child dependency: main page, sign up page, login page
* LeaderBoard
  + input: User’s session
  + output: if user is logged in, show scores. if not, request to login
  + parent Dependency: Main page
  + Child dependency: Main page

Top-down incremental testing

In this project, we followed top-down incremental testing. We tested tasks from start to leaderboard following the control flow. This allowed us to discover the defects occurring in the top of the program. Using top-down testing ensured the major components could work well which helped us test the rest of the parts of the system more conveniently because if the leaderboard works, it means data from the game is stored well, and login in session is implemented well.

# Incremental and Regression Testing

|  |  |
| --- | --- |
| Module | LeaderBoard |

Incremental Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Defect No. | Description | Severity | How To Correct |
| 1 | When entering the leaderboard page, a leaderboard is not sorted in order of scores | 2 | sorted the list before it is added to leaderboard list |
| 2 | When going to the leaderboard page, some of the navbar items are not shown. | 2 | Make sure to include all accessible pages in the html for the navbar. |

Regression Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Defect No. | Description | Severity | How To Correct |
| 1 | The ordering scores did not work. | 2 | The problem was that we sorted temporary list variable. After sorting the right variable, it worked |
| 2 | When switching between game modes on the leaderboard page, the scores are refreshed instantly instead of making the user wait 60 seconds to view refreshed scores. | 2 | Keep track of individual timers for each game mode that prevent the leaderboards from being refreshed every time the game mode is switched. |
| 3 | Clicking on the current game mode on the leaderboard page refreshes the scores for the current mode instead of making the user wait 60 seconds. | 2 | Disable the button for the current game mode on the leaderboard screen and re-enable it when the game mode is changed. |

|  |  |
| --- | --- |
| Module | Break page |

Incremental Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Defect No. | Description | Severity | How To Correct |
| 1 | The game should end when bricks hit the bottom of the game screen. | 1 | Add code that verify if bricks hit the bottom of the game screen. |

Regression Testing

|  |  |  |  |
| --- | --- | --- | --- |
| Defect No. | Description | Severity | How To Correct |
| 1 | Classic mode does not stop when bricks hit the bottom of the game screen therefore, leaderboard cannot be updated with data in realtime. | 1 | added data in database manually to check the functionality of leaderboards or add code that ends the game after shooting 10 times. |