# **System and Software Architecture Description (SSAD)**

**Newlette Coins**

**Team 06**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Members** | **Name** | **Email** | **Primary Role** | **Secondary Role** |
|  | Akshaya Ravichandran | ravichaa@usc.edu | Requirements Engineer | UML Modeler |
|  | John Leibowitz | jleibowi@usc.edu | IIV&V | Quality Focal Point |
|  | Nitin Surana | nsurana@usc.edu | Life Cycle Planner | Software Architect |
|  | Remya Ramachandran | remyaram@usc.edu | Feasibility Analyst | Implementer |
|  | Santhoshi Priyanka Gooty Agraharam | gootyagr@usc.edu | Project Manager | Tester |
|  | Theerapat Chawannakul | tchawann@usc.edu | Builder | Implementer |
|  | Vujjini Anuraag | vujjini@usc.edu | Implementer | Prototyper |

# **Version History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date** | **Author** | **Version** | **Changes made** | **Rationale** |
| 10/8/16 | TC | 1.0 | * Initial draft | * Initial draft for use with Instructional ICM-Sw v1.0 |
| 10/9/16 | NS | 1.1 | * Updated artifacts and use-cases | * For better requirements coverage |
| 11/28/16 | TC | 2.0 | * Updated topics * Tehnical-Dependent Model * Architectural Styles | * Describe in-depth details of the system |
| 12/3/16 | TC | 2.1 | * Adjust layout * Update table of contents, figures, tables | * For better reading |

# 

# 

# **Table of Contents**

**SYSTEM AND SOFTWARE ARCHITECTURE DESCRIPTION (SSAD)**

**VERSION HISTORY**

**TABLE OF CONTENTS**

**TABLE OF TABLES**

**TABLE OF FIGURES**

1. Introduction

[1.1. Purpose of the SSAD](#_35nkun2)

1.2. Status of the SSAD

2. System Analysis

2.1. System Analysis Overview

2.2. System Context

2.2.1. Artifacts & Information

2.2.2. Behavior

2.2.2.1. User Management

2.2.2.2. Gameplay

2.2.3. Modes of Operation

2.3. System Analysis Rationale

3. Technology-Dependent Model

[3.1. Design Overview](#_2fk6b3p)

3.1.1. System Structure

3.1.2. Design Classes

[3.1.2.1. Backend Project](#_2y3w247)

3.1.2.2. Frontend Project

3.1.3. Process Realization

3.1.4. State Diagram

3.2. Design Rationale

4. Architectural Styles, Patterns and Frameworks

4.1. Frameworks

# **Table of Tables**

[Table 1: Actors Summary](#_4i7ojhp)

[Table 2: Artifacts and Information Summary](#_49x2ik5)

Table 3: Register: Process Description

Table 4: Register: Successful

Table 5: Register: Failure

Table 6: Login: Process Description

Table 7: Login: Successful

Table 8: Login: Failure with invalid username or password

Table 9: Logout: Process Description

Table 10: Logout: Successful

Table 11: Edit Profile: Process Description

Table 12: Edit Profile: Successful

Table 13: Edit Profile: Failure with invalid profile information

Table 14: Change Password: Process Description

Table 15: Change Password: Successful

Table 16: Change Password: Failure with different new password and confirm password

Table 17: Change Password: Failure with invalid current password

Table 18: View Profile: Process Description

Table 19: View Profile: Successful

Table 20: Select Multiplier: Process Description

Table 21: Select multiplier: Success

Table 22: Select multiplier: Failure with not enough points

Table 23: Add Bomb: Process Description

Table 24: Add Bomb: Successful

Table 25: Add Bomb: Try to place more than the game’s bomb limit

Table 26: Remove a bomb: Process Description

Table 27: Remove a bomb: Success

Table 28: Remove a bomb: Click on a grid with no bomb

Table 29: Detonate: Process Description

Table 30: Detonate: Successful

Table 31: Detonate: Not enough points scenario

Table 32: Detonate: Network issues scenario

Table 33: Change Sound Level: Process Description

Table 34: Change Sound Level: Mute sound

Table 35: Change Sound Level: Unmute sound

Table 36: Adjust volume: Process Description

Table 37: Adjust volume: Increase volume

Table 38: Adjust volume: Decrease volume

Table 39: View History: Process Description

Table 40 : View History: Successful

Table 41: View Score: Process Description

[Table 42: Hardware Component Description](#_1ljsd9k)

[Table 43: Software Component Description](#_2koq656)

[Table 44: Supporting Software Component Description](#_3jtnz0s)

[Table 45: Architectural Styles, Patterns, and Frameworks](#_338fx5o)

# **Table of Figures**

Figure 1 System Context Diagram

Figure 2: Artifact Diagram

Figure 3: Use Case Diagram

Figure 4: Hardware Component Class Diagram

Figure 5: Software Component Class Diagram

Figure 6: Deployment Diagram

Figure 7: Supporting Software Component Class Diagram

Figure 8: Backend Project Class Diagram

Figure 9: Frontend Project Class Diagram

Figure 10: Sequence Diagram - User places and detonates bombs

Figure 11: In-game State Diagram

1. **Introduction**
   1. **Purpose of the SSAD**

This System and Software Architecture Description is created to describe the in-depth essential detail of the project that includes

* Software architecture
* Technology, tools and framework
* Software Lifecycle
* Security

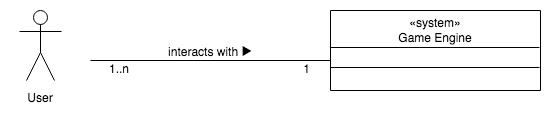
* 1. **Status of the SSAD**

This is the first version of SSAD. It contains the System Analysis - context, artifact, behavior and Architectural styles, patterns & frameworks.

1. **System Analysis**
   1. **System Analysis Overview**

The primary purpose of the Newlette Coins project is to provide the users with an easy to understand board game that can be played on both touch-based and pointer-based devices including phone, laptop, tablet & desktop. The system keeps track of all the games played by a user along with points win/lose. The system also maintains a leaderboard listing the top scorers in the game.

* 1. **System Context**

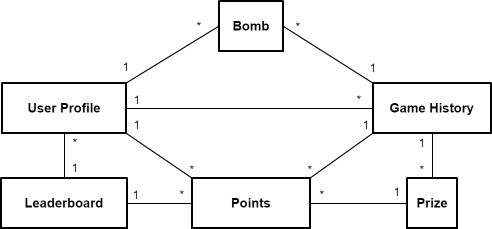


**Figure 1 System Context Diagram**

**Table 1: Actors Summary**

|  |  |  |
| --- | --- | --- |
| **Actor** | **Description** | **Responsibilities** |
| *User* | *General public including people of all ages.* | * *Register and login into the game* * *Plays the game by selecting a multiplier bid and placing bombs* * *Views his/her game history* * *View leaderboard* |

* + 1. **Artifacts & Information**

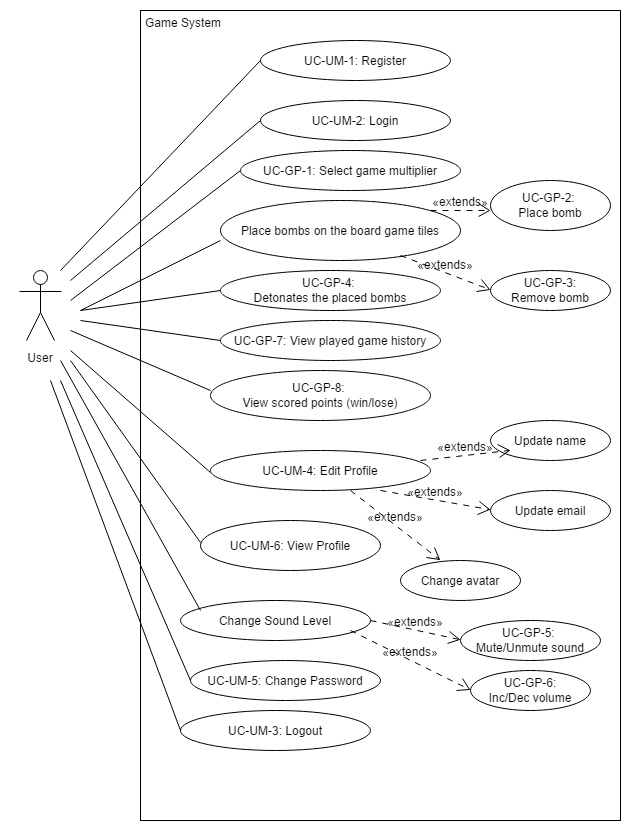


**Figure 2: Artifact Diagram**

**Table 2: Artifacts and Information Summary**

|  |  |
| --- | --- |
| **Artifact** | **Purpose** |
| ATF-1: User Profile | Contains all profile information about the user including name, avatar(profile image), email, login credentials and earned points from playing games |
| ATF-2: Game History | Contains all historical data of played game by user including location of bombs, earned prizes and total points |
| ATF-3: Leaderboard | Contains the ranking information of users based on their points |
| ATF-4: Points | Contains all points of the user |
| ATF-5: Prize | Contains all prizes created by the system when a user clicks detonate button |

* + 1. **Behavior**



**Figure 3: Use Case Diagram**

* + - 1. **User Management**
         1. **Register a new account**

**Table 3: Register: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-1: Register |
| **Purpose** | To create a user login credentials to login to the game |
| **Requirements** | The user must use these information to create an account   * Firstname * Lastname * Email * Password |
| **Development Risks** | Validate all the input to avoid query injections. |
| **Pre-conditions** | The email address must be unique and valid |
| **Post-conditions** | A new user account is created |

**Typical Course of Action**

**Table 4: Register: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Open game website in the web browser | The web browser shows Newlette Coins login page. |
| **2** | Click on ‘New Account’ | The web browser redirect to register page |
| **3** | Input Firstname, Lastname, Email, Password |  |
| **4** | Click on ‘Register’ | System validates the email address and password combination, creates a new user account and login user into the game |
|  |  | System redirects the user to game screen. |

**Alternate Course of Action**

**Table 5: Register: Failure**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Open game website in the web browser | The web browser shows Newlette Coins login page. |
| **2** | Click on ‘Register’ | The web browser redirect to register page |
| **3** | Input Firstname, Lastname, Duplicated Email, Password |  |
| **4** | Click on ‘Register’ | System pop ups an error message indicates that the email is already existed |

* + - * 1. **Login**

**Table 6: Login: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-2: Login |
| **Purpose** | To allow user to use their login credentials to login into the game |
| **Requirements** | The user must use these information to login   * Username * Password |
| **Development Risks** | None |
| **Pre-conditions** | The user has already an account |
| **Post-conditions** | The game screen is displayed |

**Typical Course of Action**

**Table 7: Login: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Open game website in the web browser | The web browser shows Newlette Coins login page. |
| **2** | Input Username and Password |  |
| **3** | Click on ‘Login’ | System validates the username and password and |
|  |  | System redirects the user to the game screen. |

**Alternate Course of Action**

**Table 8: Login: Failure with invalid username or password**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Open game website in the web browser | The web browser shows Newlette Coins login page. |
| **2** | Input Username and invalid Password |  |
| **3** | Click on ‘Login’ | System validates the username and password and pop ups error message indicates that his credentials is invalid |

* + - * 1. **Logout**

**Table 9: Logout: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-3: Logout |
| **Purpose** | To allow user to logout from the game |
| **Requirements** | The user must click on Logout button |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and he is in a game screen or board screen |
| **Post-conditions** | The login screen is displayed |

**Typical Course of Action**

**Table 10: Logout: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with Logout link will be visible |
| **2** | Click on ‘Logout’ | System invalidates this user’s session |
|  |  | System redirects the user to the login screen |

* + - * 1. **Edit Profile**

**Table 11: Edit Profile: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-4: Edit Profile |
| **Purpose** | To allow user to change/update his information (Avatar, Name, Email) |
| **Requirements** | The user must click on Edit Profile button |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and he is in a game screen or board screen |
| **Post-conditions** | The user’s profile is updated |

**Typical Course of Action**

**Table 12: Edit Profile: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with Edit Profile link will be visible |
| **2** | Click on ‘Edit Profile’ | System redirects the user to edit profile page |
| **3** | Enter information required |  |
| **4** | Click on ‘Save’ | Save updated user profile and display pop up message indicates that his user profile is saved |

**Alternate Course of Action**

**Table 13: Edit Profile: Failure with invalid profile information**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with Edit Profile link will be visible |
| **2** | Click on ‘Edit Profile’ | System redirects the user to edit profile page |
| **3** | Enter information required with some invalid informations |  |
| **4** | Click on ‘Save’ | Edited user profile is not saved and display pop up message indicates which field is invalid |

* + - * 1. **Change Password**

**Table 14: Change Password: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-5: Change Password |
| **Purpose** | To allow user to change his/her login credentials |
| **Requirements** | The user must click on Change Password button |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and he is in a game screen or board screen |
| **Post-conditions** | The user’s password is updated |

**Typical Course of Action**

**Table 15: Change Password: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with ‘Change Password’ link will be visible |
| **2** | Click on ‘Change Password’ | System redirects the user to change password page |
| **3** | Input these information   * New Password * Confirm Password * Current Password |  |
| **4** | Click on ‘Change’ | Save new user password and display pop up message indicates that his user profile is updated |

**Alternate Course of Action**

**Table 16: Change Password: Failure with different new password and confirm password**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with ‘Change Password’ link will be visible |
| **2** | Click on ‘Change Password’ | System redirects the user to change password page |
| **3** | Input these information   * New Password * Confirm Password * Current Password |  |
| **4** | Click on ‘Change’ | User password doesn’t change and |
|  |  | System display pop up message indicates that the new password and confirm password must match |

**Table 17: Change Password: Failure with invalid current password**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with ‘Change Password’ link will be visible |
| **2** | Click on ‘Change Password’ | System redirects the user to change password page |
| **3** | Input these information   * New Password * Confirm Password * Current Password |  |
| **4** | Click on ‘Change’ | User password doesn’t change and |
|  |  | System display pop up message indicates that the user inputted wrong password |

* + - * 1. **View Profile**

**Table 18: View Profile: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-UM-6: View Profile |
| **Purpose** | To allow user to view his personal information |
| **Requirements** | The user must click on My Profile link |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and he is in a game screen or board screen |
| **Post-conditions** | The profile page is displayed |

**Typical Course of Action**

**Table 19: View Profile: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with My Profile link will be visible |
| **2** | Click on ‘My Profile’ | System redirects the user to the his profile page |
|  |  |  |

* + - 1. **Gameplay**
         1. **Select multiplier**

**Table 20: Select Multiplier: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-1: Select multiplier |
| **Purpose** | Allow a user to choose multiplier factor which cause a user to spend more points in order to earn more prizes. (High risk high return) |
| **Requirements** | The user must click on predefined board images shown in the game screen |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in board screen  The user have enough points for that multiplier factor |
| **Post-conditions** | The multiplier factor is selected |

**Typical Course of Action**

**Table 21: Select multiplier: Success**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of the multiplier factor button | System show some effects on that button |

**Alternate Course of Action**

**Table 22: Select multiplier: Failure with not enough points**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of the multiplier factor button which requires more point than the user’s current point | The button is disabled and user is unable to click it |

* + - * 1. **Place bombs on a board game tiles: Add a bomb**

**Table 23: Add Bomb: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-2: Add a bomb |
| **Purpose** | Allow a user to freely place a bomb on given grids |
| **Requirements** | The user must click on given grids |
| **Development Risks** | Finding attractive images and animations of a bomb |
| **Pre-conditions** | The user is logged in and currently in board screen  The user cannot place more bombs than the current board limitation. |
| **Post-conditions** | A bomb is placed on a selected grid |

**Typical Course of Action**

**Table 24: Add Bomb: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of given grids | A bomb appears in the canvas of the web browser on a selected grid |
|  |  | Number of user bombs is reduced |

**Alternate Course of Action**

**Table 25: Add Bomb: Try to place more than the game’s bomb limit**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of given grids while the number of placed bombs are at limit | Nothing happens |

* + - * 1. **Place bombs on a board game tiles: Remove a bomb**

**Table 26: Remove a bomb: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-3: Remove a bomb |
| **Purpose** | Allow a user to remove a placed bomb on given grids |
| **Requirements** | The user must click on given grids with a bomb |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in board screen  There is a bomb on a selected grid |
| **Post-conditions** | A bomb is removed on a selected grid |

**Typical Course of Action**

**Table 27: Remove a bomb: Success**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of given grids with a bomb on | A bomb disappears in the canvas of the web browser on a selected grid |
|  |  | Number of user bombs is increased |

**Alternate Course of Action**

**Table 28: Remove a bomb: Click on a grid with no bomb**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of given grids with no bomb | Nothing happen |

* + - * 1. **Detonate all bombs**

**Table 29: Detonate: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-4: Detonate bombs |
| **Purpose** | Allow a user to detonate placed bombs to earned points and prizes |
| **Requirements** | The user must click on a detonate button |
| **Development Risks** | The sequence of animations to be shown  The security issues in a connection between the game and backend server such as CORS |
| **Pre-conditions** | The user is logged in and currently in board screen  There are bombs on a selected grid  The user must select his desired multiplier factor  The user must have enough points for selected multiplier factor |
| **Post-conditions** | The user gains/loses points (based on points they earned and points they spent)  The earned prizes and points are recorded in the system |

**Typical Course of Action**

**Table 30: Detonate: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on one of multiplier factors button | The selected button shows an effect indicated that it has been selected |
| **2** | Click on a detonate button | Bombs’ explosion animation are animated |
|  |  | Prizes are shown on each grid. |
|  |  | Number of point earned and number of point spent are shown on the screen |
|  |  | Reset the board for new round   * Remove all bombs * Fill user’s bombs * Reset number of point earned and point spent |

**Alternate Course of Action**

**Table 31: Detonate: Not enough points scenario**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on a detonate button while user doesn’t has enough points | System pop ups an error message indicate that user has not enough points to play with selected multiplier |

**Exceptional Course of Action**

**Table 32: Detonate: Network issues scenario**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on a detonate button while there is no internet connection or some network issues | System pop ups an error message indicate that user has no internet connection |
|  |  | No points will deducted from the user |
|  |  | The user will be redirected to login page |

* + - * 1. **Change Sound Level: Mute or Unmute sound**

**Table 33: Change Sound Level: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-5: Mute/Unmute sound |
| **Purpose** | Allow a user to mute or unmute sound |
| **Requirements** | The user must click on Mute/Unmute button |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in game screen or board screen |
| **Post-conditions** | The game sound is mute or unmute based on previous status |

**Typical Course of Action**

**Table 34: Change Sound Level: Mute sound**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on Mute button | The game sound is mute |
|  |  | Mute button is changed to Unmute button |

**Table 35: Change Sound Level: Unmute sound**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on Unmute button | The game sound is unmute |
|  |  | Unmute button is changed to Mute button |

* + - * 1. **Change Sound Level: Adjust sound volume**

**Table 36: Adjust volume: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-6: Inc/Dec Volume |
| **Purpose** | Allow a user to adjust sound volume |
| **Requirements** | The user must click on Volume level slider |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in game screen or board screen  The game sound is not muted |
| **Post-conditions** | The game sound volume is adjusted |

**Typical Course of Action**

**Table 37: Adjust volume: Increase volume**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Adjust volume level slider to the right | The game sound volume is increase |

**Table 38: Adjust volume: Decrease volume**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Adjust volume level slider to the left | The game sound volume is decreased |

* + - * 1. **View played game history of the user**

**Table 39: View History: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-7: View played game history |
| **Purpose** | Allow a user to view his historical gameplay data such as earned points and locations of placed bombs |
| **Requirements** | The user must click on My History link |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in game screen or board screen |
| **Post-conditions** | The gameplay history page is shown |

**Typical Course of Action**

**Table 40 : View History: Successful**

|  |  |  |
| --- | --- | --- |
| **Seq#** | **Actor’s Action** | **System’s Response** |
| **1** | Click on his avatar picture on right top of the window | A dropdown list with ‘My History’ link will be visible |
| **2** | Click on ‘My History’ | System redirects the user to gameplay history page |

* + - * 1. **View scored points (win/lose)**

**Table 41: View Score: Process Description**

|  |  |
| --- | --- |
| **Identifier** | UC-GP-8: View scored points |
| **Purpose** | Allow a user to view his total points and his win/lose records |
| **Requirements** | None |
| **Development Risks** | None |
| **Pre-conditions** | The user is logged in and currently in game screen or board screen |
| **Post-conditions** | None |

**No Action is required**

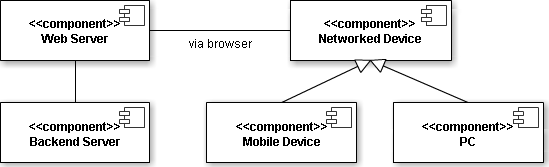
* + 1. **Modes of Operation**

The system Newlette Coins as we envision it has only one mode of operation.

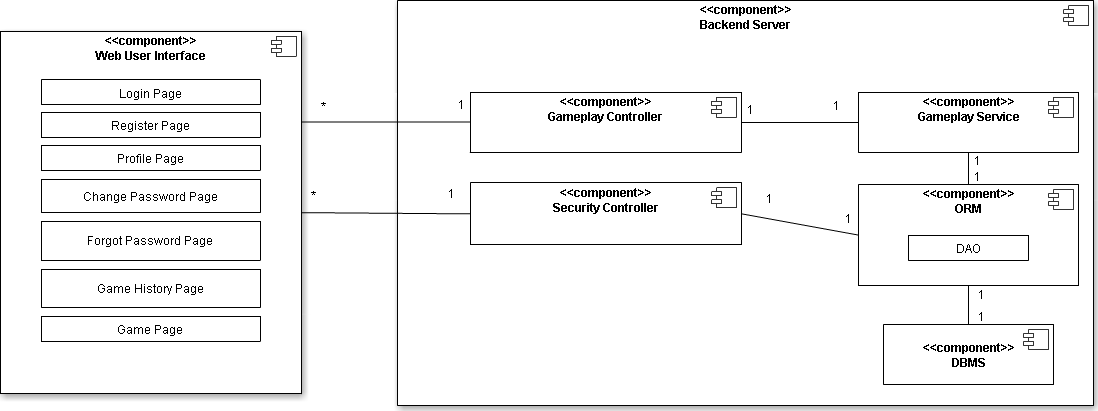
* 1. **System Analysis Rationale**

The primary purpose of Newlette Coins project is to provide a web-based game which will work on multiple devices including phones, tablets, laptops and desktops. The board game allows the user to place bombs in order to detonate tiles and find the treasure beneath. Points are awarded for different prize items. Game history for a player is maintained and the game includes a leaderboard showing users with maximum points.

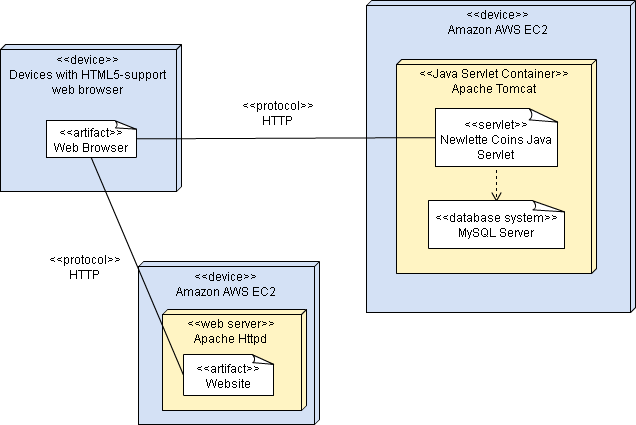
1. **Technology-Dependent Model**
   1. **Design Overview**
      1. **System Structure**



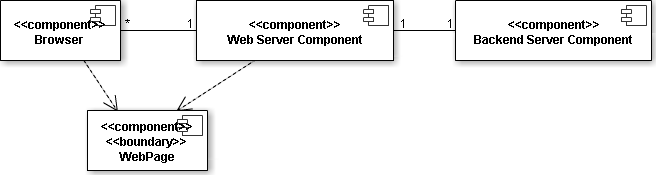
**Figure 4: Hardware Component Class Diagram**



**Figure 5: Software Component Class Diagram**



**Figure 6: Deployment Diagram**



**Figure 7: Supporting Software Component Class Diagram**

**Table 42: Hardware Component Description**

|  |  |
| --- | --- |
| **Hardware Component** | **Description** |
| Networked Device | Any device that is connected to internet. User can open a browser with that device and play Newlette Coins. |
| Mobile Device | One of the networked devices that user can play the game on. |
| PC | One of the networked devices that user can play the game on. |
| Web Server | The server that our frontend application will be running on it. |
| Backend Server | The server that our backend application and database of our system will be running on it. |

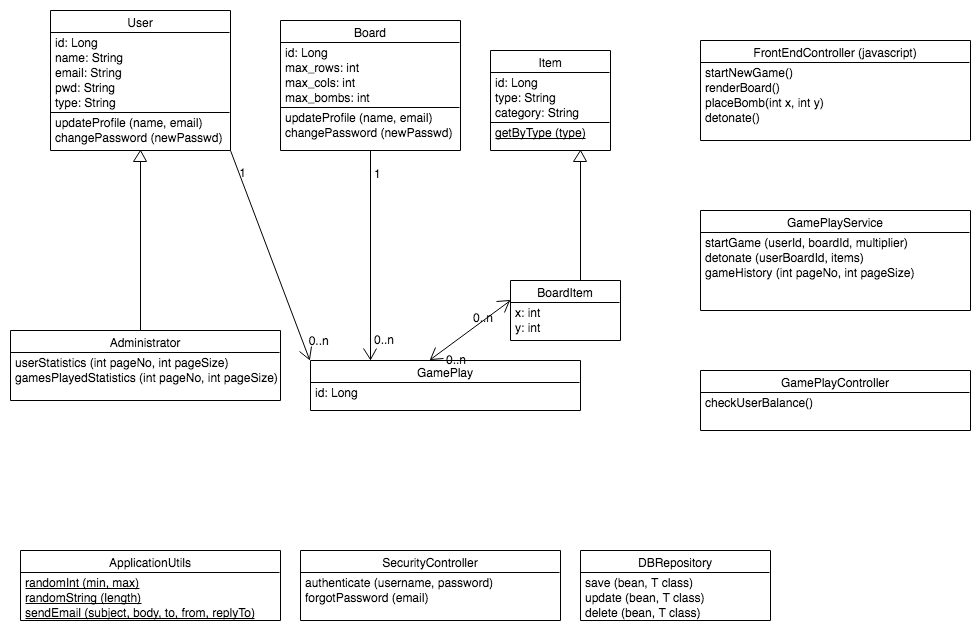
**Table 43: Software Component Description**

|  |  |
| --- | --- |
| **Software Component** | **Description** |
| Backend Server | This component contains all endpoints for requests about gameplay. It provides a response for each incoming request from the web server created by user’s events. |
| Gameplay Controller | A component that is responsible for calculate all the game’s constraints, user’s points, user’s items. And, act as an endpoint to receive incoming requests about gameplay from users. |
| Security Controller | This component contains an endpoint for requests about user credentials and receive incoming requests for register user, login, change password, reset password. |
| Gameplay Service | A component to calculate all business logics about gameplay such as calculate user’s points, items’ location, points used and items earned based on designed ratio. |
| ORM | Data Access Objects classes that are used for connecting to database. |
| DBMS | Represents the database of our system. |
| Login Page | Page for user login |
| Register Page | Page for user register |
| Forgot Password Page | Page for user who forgot his password |
| Change Password Page | Page for user to change his password |
| Profile Page | Page for user profile |
| Game History Page | Page for user’s played game history |
| Game Page | Page for user to play Newlette Coins |

**Table 44: Supporting Software Component Description**

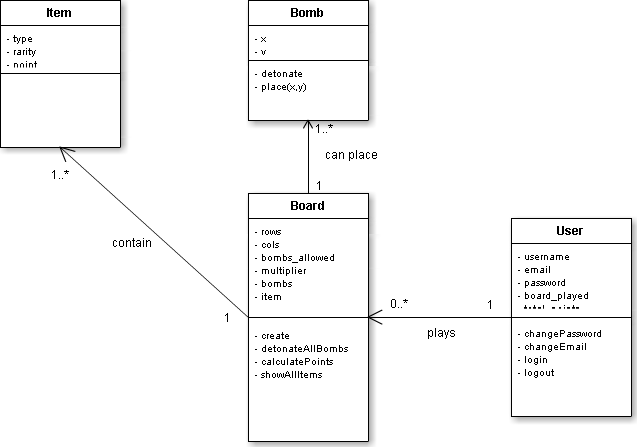
|  |  |
| --- | --- |
| **Support Software Component** | **Description** |
| Browser | An internet browser that opens Newlette Coins web application. It is responsible for render all game pages and user interfaces. |
| Web Server Component | The server component that presents the web browser with all static files such as HTML and Javascript files. |
| Backend Server Component | The server component where all Newlette Coins’ business logics are calculated. |

* + 1. **Design Classes**
       1. **Backend Project**



**Figure 8: Backend Project Class Diagram**

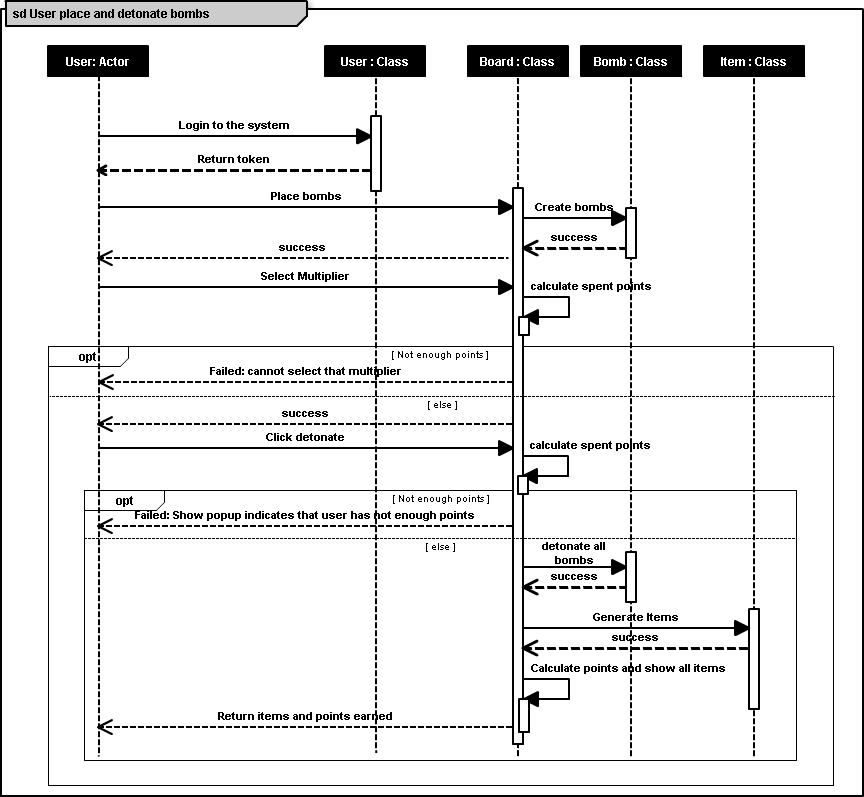
* + - 1. **Frontend Project**



**Figure 9: Frontend Project Class Diagram**

* + 1. **Process Realization**

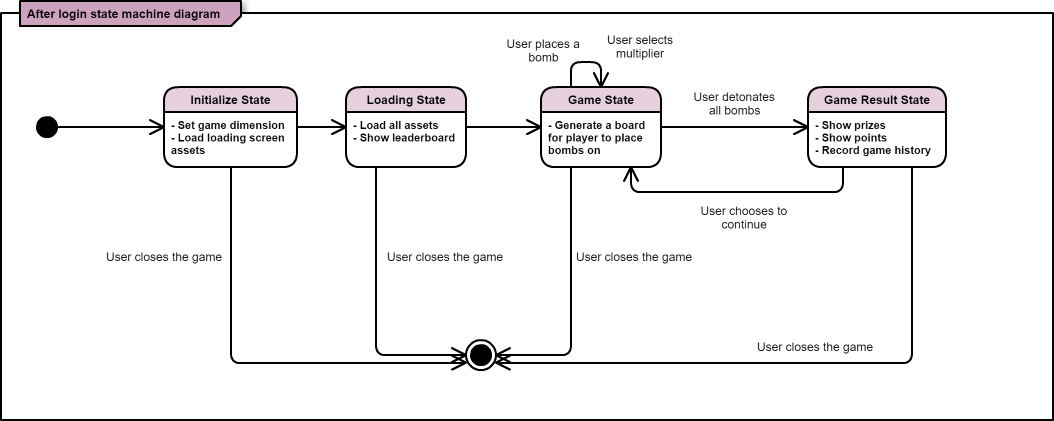
For the main function of the game(User places and detonates bombs), we decided to create a sequence diagram to demonstrate its feature.



**Figure 10: Sequence Diagram - User places and detonates bombs**

* + 1. **State Diagram**

The game process flows have been controlled using states. In order to make sure that we can understand the flow of our game’s main function, we decided to create a state diagram.



**Figure 11: In-game State Diagram**

* 1. **Design Rationale**

By decided to develop a mobile game, Newlette Coins, we have chosen HTML5 language, a programming language that is fully supported across multiple devices, as a primary language to develop the game. Thus, to develop a complexed game using only HTML5 language is not an appropriate way to do. So, we use Phaser.JS (Desktop and mobile HTML5 game framework) to develop Newlette Coins because it provides many functionalities to develop a HTML5 game. Moreover, the client, Crazy Cool Apps Co. Ltd, has many experience using this framework.

On the backend side, we decided to use Spring Framework for an implementation. This framework is an application framework that facilitates a developer to develop Java application with ease. Furthermore, the source code can be easily maintained by the maintainers because of its architecture that enforces separation of concerns by separate the business logic via many artefacts such as controller, service and repository.

We separate the frontend part and the backend part into different servers to improve security measures and scalability potential. This provide the maintainer ability to track which server will consume more resources and then they can increase the resource of that device. Moreover, they can create a load balancer on both frontend and backend server because both of them are stateless application.

1. **Architectural Styles, Patterns and Frameworks**
   1. **Frameworks**

In the table below, you can see list of architectural styles, patterns and frameworks that we used in the implementation of Newlette Coins.

**Table 45: Architectural Styles, Patterns, and Frameworks**

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Benefits, Costs, and Limitations** |
| 3-Tier Architecture | Modular code design with services, controllers, repositories and models. | Easy maintenance and loosely coupled code |
| Spring | Java library to provide dependency injection support | Provides ability to inject objects without worrying about the hassle of initialization. Ensures fast development Free & open source |
| Spring Data | Java framework | Provides an extensible and pre-defined framework for common database operations. The framework is time-tested and stable, thereby no bugs and large community support Faster development Free & open source |
| Hibernate | Java object relational mapping framework | Highly popular ORM framework provides ability to treat db objects as core java objects, thereby saving time to marshall/un-marshall between db tables and java entities Free & open source |
| TestNG | Java testing framework | Easy test setup and configuration through annotations with hooks at class & method levels Free & open source |
| Mockito | Java mock framework | Provides ability to mock un-necessary objects during unit-testing with pre-defined invocation results Free & open source |
| PhaserJS | Javascript framework | Desktop and mobile HTML5 game framework.  Free & open source |
| MySQL | Relation Database Management System | An open-source relational database management system (RDBMS). |
| State Design Pattern | Design Pattern | Easy to manage the scene, the data and the status of the game. |