**Contents**

1. [Requirement Analysis](#Ch1)
   1. [Functional Requirements](#Ch1_1)
   2. [Non-Functional Requirements](#Ch1_2)
2. [Use Cases](#Ch2)
   1. [Use Case 1 – Log in](#Ch2_1)
   2. [Use Case 2 – Registration](#Ch2_2)
   3. [Use Case 3 – Main Menu](#Ch2_3)
   4. [Use Case 4 – World creation](#Ch2_4)
   5. [Use Case 5 – Movement](#Ch2_5)
   6. [Use Case 6 – Coins](#Ch2_6)
   7. [Use Case 7 – Map visibility](#Ch2_7)
   8. [Use Case 8 – Exit](#Ch2_8)
   9. [Use Case 9 – Winning Condition](#Ch2_9)
3. **Requirement Analysis**

1.1 Functional Requirements

1. The user must log in to play the game.
   1. If the player is not registered to the game, he can register by giving a username, a password and an email.
2. The user chooses options from a menu (single, multiplayer, score or tutorial)
   1. If player selects single player game, he enters the level selection screen.
   2. If player selects multiplayer game, he enters the level selection screen.
   3. If player selects score, he sees the scoreboards.
   4. If player selects tutorial, the player sees the instructions of the game.
   5. If player selects exit or closes the tab, the connection with server terminates.
3. The world is loaded on the server’s memory.
4. The player can see a part of the dungeon (explored tiles), other players (human or bots), gold coins, passages and exit.
   1. A dungeon is a collection of rooms that are connected with passages.
      1. A room may have gold coins.
   2. A dungeon can be arbitrary size.
   3. Dungeon must contain the minimum gold coins so the player can win.
5. Player interacts with the dungeon:
   1. Indicating which way to move (UP, DOWN, LEFT, RIGHT)
   2. Picking up gold.
   3. Player can move around to reveal the room.
   4. Leaving the game.
6. Winning Condition:
   1. If the player collects all the gold coins, needed to win, and find the exit, player wins.
   2. Else if another player wins first or the player gives up, the player loses.
7. After winning condition, the player goes to result screen.

1.2 Non-Functional Requirements

1. System consists of 3 components: Presentation layer, Business Logic Layer and Database Layer
   1. Database Layer have a MySQL database that contains information about players and score table (server side).
   2. Business Logic Layer contains Web services and Models of game’s components. These components will be written in Java (server side).
   3. Presentation Layer contains GUI(view) for the user and controllers that interact with the server via Web services. These components will be written in HTML5 & CSS3 and JavaScript (client side).

**2.Use Cases**

2.1 Use Case 1 – Log in

UC1-1: Use Case: Log In

UC1-2: Author: QZ, XF

UC1-3: Date: 8-NOV-2016

UC1-4: Purpose: Log in to the play the game

UC1-5: Overview: Starts when the player opens the game page. System requests username and password for login. System validates username and password (if the username and password already exists in the player database) then player enters the Main Menu Screen, else alternative 1: Validation fails and log in failure message is given to the user.

UC1-6: Cross References: [R1](#R1)

UC1-7: Actors: Player

UC1-8: Pre-Condition:

UC1-Pre-1: The website must be loaded.

UC1-9: Post Condition:

UC1-Post-1: The player is prompted to Main Menu screen.

|  |  |  |
| --- | --- | --- |
| Actor Actions | Client System Actions | Server System Actions |
| 1. Open the game page. |  |  |
| 1. Input the username and password. |  |  |
| 1. Press “OK” button. |  |  |
|  | 1. Sends the username and password to server. |  |
|  |  | 1. Check if the username already exists in the membership database and validates the password. |
|  |  | 1. Sends response. |
|  | 1. Receives and parses the response. |  |
|  | 1. Jump to the main menu interface. |  |

UC1-10: Alternative flow of events:

* + Step 5: Username and password is not in the player database. Display an error message, and ask player to reenter their credentials.

UC1-11: Exceptional flow of events:

* Steps 4, ,6, 7: If the connection with the server is not established return an error message.

2.2 Use Case 2 – Registration

UC2-1: Registration

UC2-2: Author: AG

UC2-3: Data: 27-NOV-2016

UC2-4: Purpose: Player registers to the game

UC2-5: Overview: The user fills a form with his desired username, his email address and his desired password. If the username or email does not exist on the system’s database, then the system saves these details and returns to log in screen. Else, alternative 1: if username or email exists in database, then an appropriate message is returned to the user and the user must pick a different username or email. Alternative 2: if the email does not contain “@” and “.” characters the client must give an appropriate message.

UC2-6: Cross Reference: [R1.1](#R1_1), [R1](#R1)

UC2-7: Actors: Player

UC2-8: Pre-Conditions:

UC2-Pre-1: The registration web page must be loaded.

UC2-Pre-2: The player must not have an account.

UC2-9: Post-Conditions:

UC2-Post-1: The player returns to log in screen.

UC2-Post-2: The player’s information is stored into the server’s database.

|  |  |  |
| --- | --- | --- |
| Actor Actions | Client System Actions | Server System Actions |
| 1. Open the registration page. |  |  |
| 1. Input the username, the email and the password. |  |  |
| 1. Press “OK” button. |  |  |
|  | 1. Sends the username, the email and password to server. |  |
|  |  | 1. Stores the username, the email and the password to system’s database. |
|  |  | 1. Sends response. |
|  | 1. Receives and parses the response. |  |
|  | 1. Jump to the log in page. |  |

UC2-10: Alternative flow of events:

* Step 5: Username or email already exists on the system’s database. Appropriate message is returned to the player so the player can pick different username or email.

UC2-11: Exceptional flow of events:

* Step 4, 6, 7: If the connection with the server is not established return an error message.

2.3 Use Case 3 – Main Menu

UC3-1 Use Case: Player Choose Menu

UC3-2 Author: XF, QZ

UC3-3: Date: 8-NOV-16

UC3-4: Purpose: The user chooses options from the main menu.

UC3-5: Overview: The player is logged in and is prompted to main menu. There are four button: single player game, multiplayer game, score and tutorial. If the player selects single player game or multiplayer game, then he is prompted to level selection screen. The player must select a match level he wants to play in, the client sends a request to server and the server sends dungeon data to the client. The user is prompted into the dungeon map and starts playing the game. Else, alternative 1: if the player selects score, the game client sends a request to the server to retrieve the score table from the database. The server sends the data back to client. Else, alternative 2: if the player selects tutorial, them he is prompted to tutorial screen. Else, alternative 3: if the player selects the exit button, the player logs out of the system and returns to log in screen.

UC3-6: Cross References: [R2](#R2), [R2.1](#R2_1), [R2.2](#R2_2), [R2.3](#R2_3), [R2.4](#R2_4), [R2.5](#R2_5)

UC3-7: Actors: Player

UC3-8: Pre-Condition:

UC3-Pre-1: The player must be in menu (i.e. not the dungeon).

UC3-Pre-2: The player must already be logged in.

UC3-9: Post Condition:

UC3-Post-1: The player is sent to appropriate web page based on his/her action.

Single/Multiplayer button

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Begins when player click single player of multiplayer button |  |  |
|  | 1. Sends button request to the server |  |
|  |  | 1. The server initiates the player’s session. |
|  | 1. Jumps to level selection web page. |  |
| 1. The user selects a match to play in. |  |  |
|  | 1. The client sends a request to the server |  |
|  |  | 1. The server responds with dungeon data. |
|  | 1. Client receives the response and creates a graphical representation of the data. |  |

Score button

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Begins when player click score button |  |  |
|  | 1. Sends button request to server |  |
|  |  | 1. Retrieve the scoreboard from database. |
|  | 1. The user is prompted to score table web page and shows the data. | 1. Send response |

Tutorial

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Begins when player click Tutorial button |  |  |
|  | 1. The user is prompted to tutorial page. |  |

UC3-9: Alternative flow of events:

UC3-10: Exception flow of events:

* In steps where the client sends a request or the server sends a response, if the client does not receive any response in time, appropriate message should be displayed.

2.4 Use Case 4 – World Creation

UC4-1 Use Case: World Creation

UC4-2 Author: SK

UC4-3: Date: 13-NOV-16

UC4-4: Purpose: To create map, dungeon, coins, passages

UC4-5: Overview: After the match selection, dungeon is created. The client sends a request to server to retrieve graphical representation of the dungeon. Dungeon consists of rooms. Rooms are connected with each other with passages. Rooms must contain gold coins. Rooms can be of arbitrary size. Dungeon must contain minimum gold coins for win condition.

UC4-6: Cross Reference: [R3](#R3), [R4](#R4), [R4.1](#R4_1), [R4.2](#R4_2), [R4.3](#R4_3)

UC4-7: Actors: Player

UC4-8: Pre-Condition:

UC4-Pre-1: The player must select a match.

UC4-Pre-2: The player must already log on.

UC4-9: Post Condition:

UC4-Post-1: Player enters the game map and sees a graphical representation of the dungeon.

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Begins when player selects a match. |  |  |
|  | 1. Sends button request to server |  |
|  |  | 1. Dungeon is created with rooms, passages and coins. Server saves this information on its memory. |
|  |  | 1. Sends response with visible area by the player |
|  | 1. Receives and parses response |  |
|  | 1. The client draws the graphical representation of the dungeon. |  |

2.5 Use Case 5 - Movement

UC5-1 Use Case: Player Moves Character

UC5-2 Author: AG, MJ

UC5-3: Date: 8-NOV-16

UC5-4: Purpose: Move the player in one of the four directions

UC5-5: Overview: The player character’s location is a particular point on the map. The Player presses one of the keys W, A, S or D. These keys map to up, left, down, right respectively. The game client sends a request to the server with the details on the action. The server validates this input and decides whether or not the player character can move in that direction. If the character can be moved the server updates the character’s position in-memory. The server responds with the current location of the character. The client updates the dungeon graphical representation. Alternative 1: A wall is blocking the direction that the player wishes to move their character in. The server does not update the player’s position and responds with the current player location the same as it was.

UC5-6: Cross References: [R5](#R5), [R5.1](#R5_1)

UC5-7: Actors: Player

UC5-8: Pre-Condition:

UC5-Pre-1: The player must be in a dungeon (i.e. not the menu).

UC5-Pre-2: The map has been loaded.

UC5-9: Post Condition:

UC5-Post-1: The player’s position is the player’s previous position moved one unit in the chosen direction.

UC5-Post-2: The client’s graphical representation has updated.6

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Begins when player presses movement key |  |  |
|  | 1. Sends movement request to server |  |
|  |  | 1. Checks whether or not the player can move in that direction. |
|  |  | 1. Sends response |
|  | 1. Receives and parses response |  |
|  | 1. Updates dungeon graphical representation |  |

UC5-10: Alternative flow of events:

Step 3: The movement is illegal. Server doesn’t update player location, responds with player in the same location.

UC5-11: Exception flow of events:

Steps 2,4,5: The request or response network packets are dropped or corrupted. The sends a request for the current state of the system. If that request fails, the client displays an appropriate message regarding network connectivity problems to the player.

2.6 Use Case 6 - Coins

UC6-1: Use Case: Picking up Gold coins

UC6-2: Authors: AG, MJ

UC6-3: Date: 8-NOV-16

UC6-4: Purpose: Moves to gold coin to collect it.

UC6-5: Overview: The player moves to a tile with a gold coin on it. The server responds that the gold coin in no longer in its previous position, it increments player’s gold coin collection and moves the player to the location of the coin. The client updates the dungeon graphical representation.

UC6-6: Cross References: [R5](#R5), [R5.2](#R5_2)

UC6-7: Actors: Player

UC6-8: Pre-condition:

UC-6-Pre-1: The player must be in a dungeon (i.e. not in the menu)

UC-6-Pre-2: The player should be one unit away from the gold coin.

UC6-9: Post-condition:

UC-6-Post-1: The player’s gold coin collection is incremented.

UC-6-Post-2: The player moves to the updated location.

UC-6-Post-3: The client updates the dungeon graphical representation.

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. Player begins to move to a gold coin. |  |  |
|  | 1. The client sends a request to the server. |  |
|  |  | 1. The server receives the request. |
|  |  | 1. The server validates the action. |
|  |  | 1. The server increments player’s gold coin collection. 2. The server updates gold coin location. 3. Sends response |
|  | 1. Receives and parses the response 2. Updates the graphical representation. |  |
|  |  |  |

UC6-11: Exceptional flow of events:

* Same as [UC5-11](#UC6_11)

2.7 Use Case 7 – Map visibility

UC7-1 Use Case: Player movement reveals the map

UC7-2 Author: AG, MJ

UC7-3: Date: 8-NOV-16

UC7-4: Purpose: To discover the layout the dungeon

UC7-5: Overview: The player character is moving from one tile to another. The Server decides which tiles are visible to the player character. The server response includes the current state of the tiles now visible to the player character. The client adds these tile states to its memory. The client remains aware of previously discovered tiles but may not be aware of their current state, i.e. whether another player has moved their character to that location. The client updates the graphical representation including the newly visible tiles.

UC7-6: Cross References: [R5](#R5), [R5.3](#R5_3)

UC7-7: Actors: Player

UC7-8: Pre-Condition:

UC7-Pre-1: The player must be in a dungeon (i.e. not the menu)

UC7-Pre-2: The player is in a state of moving from one tile to the next

UC7-9: Post Condition:

UC7-Post-1: The client updates its memory with new tile states.

UC7-Post-2: The client now displays additional tiles that may not have been previously visible.

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
|  |  | 1. Begins when the server interprets a move command |
|  |  | 1. Decides which tiles are visible to the character |
|  |  | 1. Responds to move request, including the current state of tiles now visible to the player character |
|  | 1. Receives and parses the response |  |
|  | 1. Adds the updated and/or new tiles to its memory |  |
|  | 1. Updates the graphical representation |  |

UC7-11 Exceptional flow of events:

* Same as [UC5-11](#UC6_11)

2.8 Use Case 8 - Exit

UC8-1: Use Case: Leaving the Game

UC8-2: Authors: AG, MJ

UC8-3: Date: 8-NOV-16

UC8-4: Purpose: To exit the game.

UC8-5: Overview: The player presses a button to return to the main menu. The client sends a request to server to terminate the session. The server removes the player’s character.

UC8-6: Cross References: [R5](#R5), [R5.4](#R5_4)

UC8-7: Actors: Player

UC8-8: Pre-condition:

UC-8-Pre-1: The player must be in a dungeon (i.e. not in the menu)

UC-8-Pre-2: The player presses the exit button

UC8-9: Post-condition:

UC-8-Post-1: The player returns to main menu

UC-8-Post-2: The server removes the player from the game.

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. The player presses the exit button. |  |  |
|  | 1. Sends request to the server. |  |
|  |  | 1. Receives the request. |
|  |  | 1. Removes the player’s character. |
|  | 1. Returns to main menu. |  |

UC8-11: Exception flow of events:

* Step1: The player closes the browser to exit the game. The server notices that the client has not sent any request within a time limit and removes the player’s character.
* Step 2: Same as [UC5-11](#UC6_11)

2.9 Use Case 9 – Winning Condition

UC9-1: Use Case: Winning condition

UC9-2: Authors: AN

UC9-3: Date: 9-NOV-16

UC9-4: Purpose: To show the winning status and Main map with unlocked level.

UC9-5: Overview: The player won the game by passing through exit. The client sends the request to update the score. Server starts processing request. Server updates the results in scoreboard, unlock the new level and sends a response back to client to show the result screen with details which include time, coins or other entities player collected. Followed by client displays the main map to player, which shows the unlocked level with an option to return to main menu. Else alternative 1: another player exits the game first; the player returns to results screen with a “You lose” message.

UC9-6: Cross References: [R6](#R6), [R6.1](#R6_1), [R6.2](#R6_2), [R7](#R7)

UC9-7: Actors: Player

UC9-8: Pre-condition:

UC-9-Pre-1: The player must be in dungeon and pass through exit.

UC-9-Pre-2: The player must win the game (player must collect the minimum amount of coins.

UC9-9: Post-condition:

UC-9-Post-1: The player gets the result screen.

UC-9-Post-2: The server updates the scoreboard & sent the result screen.

|  |  |  |
| --- | --- | --- |
| **Actor Actions** | **Client System Actions** | **Server System Actions** |
| 1. The player won the game. |  |  |
|  | 1. Sends the updated result to the server. |  |
|  |  | 1. Receives the request. |
|  |  | 1. Update the scoreboard. 2. Unlock a new level 3. Sends the result screen. |
|  | 1. Shows the result screen 2. Shows the main map with unlocked level 3. Returns to main menu. |  |

UC9-11: Exception flow of events:

* Step1: If player loses the connection to the network after winning the game, client wait for a specified amount of time. If player comes back, client sends request to server and steps 2-8 will carry on. If the server hasn’t got any request within the time limit, then server will remove the player’s character.