Software Requirements Specification

for

Chess and Checkers

Version 1.0 approved

Prepared by Group 5

Farjad Khan 19711

Muhammad Aahad -18637

Muhammad Abdullah Akhter -18632

Mohammad Anas Mohsen -18634

Roshaan Tariq khan -19781

Sahar Mirza -19736

Sumaiyah Chinoy -18650

Syed Muhammad Bilal Fahim -19677

Section No. 5215

October 14, 2021

Table of Contents

Table of Contents ii

Revision History iii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 2

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Functions 3

2.3 User Classes and Characteristics 3

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 5

3. External Interface Requirements 5

3.1 User Interfaces 5

3.2 Hardware Interfaces 6

3.3 Software Interfaces 6

3.4 Communications Interfaces 6

4. System Features 6

4.1 Creating a User Account 6

4.2 Signing a User 7

4.3 Play a Multiplayer Game 8

4.4 Play Single Player Game 8

4.5 Difficulty Level 9

4.6 Viewing Replay 9

4.7 Show Possible Moves 10

4.8 Leaderboard Management 10

4.9 Playing Chess Game 11

4.10 Playing Checkers Game 12

4.11 Change Theme 13

4.12 Admin System Features 14

5. Use Case Modelling 15

5.1 Use Case 1: Create Profile 16

5.2 Use Case 2: Check Duplicate Information 17

5.3 Use Case 3: Sign Into Profile 18

5.4 Use Case 4: Check Login Credentials 19

5.5 Use Case 5: Edit Profile 20

5.6 Use Case 6: Sign Out 21

5.7 Use Case 7: Play Game 22

5.8 Use Case 8: Single Player Game 23

5.9 Use Case 9: Multi Player Game 24

5.10 Use Case 10: View Leaderboard 25

5.11 Use Case 11: Chess Leaderboard 26

5.12 Use Case 12: Checkers Leaderboard 27

5.13 Use Case 13: View Replay 28

5.14 Use Case 14: Change Theme 29

5.15 Use Case 15: Delete User 30

5.16 Use Case 16: Block User 31

5.17 Use Case 17: Unblock User 32

5.18 Use Case 18: Reset Leaderboard 33

5.19 Use Case 19: Define Leaderboard Sorting Criteria 34

5.20 Use Case 20: Forgot Password 35

5.21 Use Case 21: Check Email Exists 36

6. Other Nonfunctional Requirements 36

6.1 Performance Requirements 36

6.2 Safety Requirements 37

6.3 Security Requirements 37

6.4 Software Quality Attributes 37

6.5 Business Rules 38

7. Other Requirements 38

Appendix A: Glossary 39

Appendix B: Analysis Model 41

Appendix C: To Be Determined List 42

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of this document is to give a detailed description of our chess and checkers game. It includes description of every individual part included in our games and also gives us answers on how we are going to implement the game itself. It lists down all the functional and non-functional requirements the user can expect. The main purpose of the project is to provide the user an intuitive and responsive experience for Chess & Checkers.

## Document Conventions

We have used the standard font size of 12, Times New Roman which is the standard for professional reports. In order to emphasize a topic importance we have used a bold font so that it is easier for the reader of the SRS to be prepared on what to expect in the body of the SRS. When it comes to priority, the more important the requirement the more in depth look we will have on it that’s not to undermine other requirements, but there definitely is some precedence at play here.

## Intended Audience and Reading Suggestions

The Intended Audience is:

-End Users

-Game Testers

-Developers

The document must be read from end to beginning for a complete understanding of

the project, However, it is written in sections and hence can be read as such.

For an overview of the document and the project itself, see Overall Description (Section 2).

For a detailed description of the game features, functional requirements see System Features (Section 4)

For a detailed description of the use cases see Use Case Modelling (Section 5).

Technical standards to which the team will hold the project are laid out in Other Nonfunctional Requirements (Section 6),

## Product Scope

The purpose of our chess & checkers game is to enable the user to experience the thrill and excitement of those games but in portable format. As the title of the game suggests, the user will be able to play either chess or checkers whatever he wishes to play. It will be available on desktop as well as mobile devices. Our game will allow the user to escape his day-to-day difficulties and embark on an intimidating journey. The goal of our project is to provide just that.

Our project provides a user-friendly GUI so it is easier for users to navigate through the menus. Players will be able to make their own accounts and identify themselves using a wide selection of avatars. Users will have the privilege to play single player and multi-player where the former will come with 3 difficulty levels (Rookie, Intermediate, Grandmaster) for different kind of chess and checkers player. The player can view the leaderboard and view his current standing as well as rewatch his previous games.

Our game will help keep covid restrictions in check as users will be making moves using their own devices not physical pieces which makes it easier for covid to spread.

From a business perspective, our game will include ads. Keeping security of our users of paramount importance, players will not have to encounter malicious ads as our sponsors will be thoroughly checked and verified before making it to our game.

## References

* Arxiv.org. 2021. [online] Available at: <https://arxiv.org/ftp/arxiv/papers/1307/1307.7096.pdf>.
* Bellevuecollege.edu. 2021. [online] Available at: <https://www.bellevuecollege.edu/wp-content/uploads/sites/135/2019/04/SRS_WebGLGame.pdf>.
* Dgp.toronto.edu. 2021. [online] Available at: <http://www.dgp.toronto.edu/~ppacheco/course/444/spec>.
* Jeff.fivefourteen.net. 2021. [online] Available at: <http://jeff.fivefourteen.net/warroom/wrs-srs.pdf>.
* Pages.drexel.edu. 2021. [online] Available at: <http://www.pages.drexel.edu/~pv42/thebiz/SRS-1.pdf>.
* Scribd.com. 2021. [online] Available at: <https://www.scribd.com/doc/305902506/Online-Game-chess-SRS-doc>.
* Se.rit.edu. 2021. [online] Available at: <http://www.se.rit.edu/~sis-io/documents/UseCase.pdf>.
* Unf.edu. 2021. [online] Available at: <https://www.unf.edu/~broggio/cen6016/SamplesLinksAndExercises/SampleStudentUseCases.doc>.
* GitHub. 2021. ChessGame/SRS.md at master · brandonalfred/ChessGame. [online] Available at: <https://github.com/brandonalfred/ChessGame/blob/master/SRS.md>.
* Sites.google.com. 2021. Software Requirements Specification - TankWars. [online] Available at: <https://sites.google.com/site/cop4331tankwars/software-requirements-specification>.
* Slideshare.net. 2021. The complete srs documentation of our developed game.. [online] Available at: <https://www.slideshare.net/Axphey/the-complete-srs-documentation-of-our-developed-game>.
* Casecomplete.com. 2021. Write Use Case Extensions. [online] Available at: <https://casecomplete.com/learn/write-use-case-extensions>.

# Overall Description

## Product Perspective

The Chess and Checkers is both an online and offline game platform. It facilitates in providing user an interactive and responsive platform to play chess and checkers either with the computer player or another player.

Both the games are usually played in the physical setting where a physical boards and game pieces

are required but we have implemented it in a virtual setting. It doesn’t require the player to carry all the pieces and board physically and find another player but instead an application where the player can play the game by being matched to another player according to their ease.

## Product Functions

The Perspective is to create a fully functional chess & checkers game. The user will be the player of the game.

* User can play a multi-player game or a single player game(i.e. Supported by the AI).
* The player can choose the difficulty level in the single player game.
* Decide whether to play chess or checkers.
* Player can change the theme of the board and pieces.
* The entire game is recorded so that the replay of the game is available for the player to be replayed again.
* The user can view the leaderboard containing records of the all the players playing the game. Leaderboard is updated after every game.
* The admin can reset and sort leaderboard, block and delete users.

## User Classes and Characteristics

The system consists of two types of User Classes Admin and Player.

### Features available to Admin:

* Admin can reset the leaderboard.
* Admin decides the leaderboard sorting criteria
* Admin can block and delete players.

### Features available to Player:

* A new player can create its own account with help of sign-up feature.
* A new player chooses an avatar.
* An existing player must have its unique credentials for login.
* An existing player can update his profile.
* An existing player can only play game upon logging in.
* Can choose whether to play a single player game or multi player game.
* Can choose whether to play chess or checkers.
* Can go back and select another game.
* Can restart a game when playing a single player game.
* Can choose the theme of the board and pieces.
* Can choose the difficulty level (Rookie, Intermediate, Grand Master) of the game in a single player game.
* Can view the leaderboard for Chess.
* Can view the leaderboard for Checkers.
* Can view the replays of the previous game it has played.

## Operating Environment

**Desktop Application:**

Our game will operate on browsers such as Microsoft Internet Explorer, Google Chrome, Mozilla Firefox, etc. In terms of hardware, a regular desktop would do as our game will not be too graphically demanding.

**Mobile Application:**

**Android:** Since the app will be created using react native. So, for android we will convert our code into the .sdk using android studio and deploy it to the google play store.

## Design and Implementation Constraints

**Minimum Hardware Requirements:**

* Microsoft Windows 7 or higher
* X86 Processor
* Android OS for mobile
* 2gb ram for Windows and Mobile

**Software Requirements:**

* The information of all users, game records must be stored in the database that is accessible by both the mobile and desktop app.
* Every user related information will remain confidential.
* The desktop application and mobile app user interface must be responsive
* React native will be used for frontend.
* The server will remain functional all the time.
* Users must login with their correct username and password to perform any action.

## User Documentation

We will provide two manuals one for chess and another for checkers. Chess Manual will contain all rules the player needs to know about chess before starting a game. Checkers Manual will have all rules the player needs to know about checkers. We also have a manual for how or application will work and how the player can switch from one game to another, view the leaderboard and replays.

## Assumptions and Dependencies

* **Assumptions:**
  + User PC will have Windows 7 or any Microsoft Windows of version greater than or equal to Windows 7 installed.
  + The user will have a decent and consistent internet connection
  + The mobile operating system will be Android.
  + Both the mobile and PC will be of decent specification
  + Both the mobile and PC will have free space to install the app.
  + There will be no power outage.
  + The user will have basic knowledge on how to operate a Windows or Mobile device.
  + The server will not crash or malfunction.
* **Dependencies:(**Further dependency will be noted down once we move into the later stages)
  + Bootstrap
    - Grid system based on HTML and CSS to make our application responsive
  + Font-awesome
    - Contain large collection of icons and symbol to use whenever necessary
  + Material UI
    - react user interface library
  + React-router-Dom
    - react router library to access everything related to routes link, history, router params.
  + Axios
    - react http request library.
  + React-dnd
    - react drag and drop library, will be useful for dragging chess and checkers pieces
  + React use-state, use-effect:
    - To handle and manage state across the website, states are your variables which are subject to change, since react works on the principle of one page applications, this allows us to update our variable (states) without reloading or affecting the dom element.

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

The Functional Requirements mentioned below in this section are given priority number according to the below mentioned criteria.

**Priority 1:** All requirements of this priority level must be implemented and verified. The system can’t work without these requirements.

**Priority 2:** Requirements of this priority are expected to be implemented but the system can work without these requirements. Unless and until their omission does not affect higher priority components.

## Creating a User Account

**4.1.1 Description and Priority**

In our chess and checkers game we need a way to identify which users are playing our game. We will need profiles to be maintained that represent each user. Having these profiles will also help in providing functionality for multiplayer games in our application and maintain leaderboards.

**4.1.2 Stimulus/Response Sequences**

**Step 1:** User requests to create a new account

**Step 2:** The user enters the details

**Step 3:** A new account is created

**4.1.3 Functional Requirements**

**REQ-1. A:** When the app is launched for the first time, the user must be prompted to create a new profile. The user will be asked to enter a nickname and select an avatar from a list of predefined avatars. **Priority 1**

**REQ-1. B:** The user shall be allowed to create a profile or switch to an existing profile. **Priority 1**

**REQ-1. C:** A user must create a profile if they want to play a game. **Priority 1**

**REQ-1. D:** A user must enter a unique username, valid email and password to create an account. **Priority 1**

**REQ-1. E:** A user must have unique credentials. The data is then validated to make sure there is not an existing user with those credentials. **Priority 1**

## Signing a User

**4.2.1 Description and Priority**

The user must be signed in to get hands-on functionality of the application.

            This feature allows the user to proceed to the next step of playing the

game. This feature will allow the game to uniquely identify each player and use his credentials later on in other features of the game.

**4.2.2 Stimulus/Response Sequences**

**Step 1:** The sign in screen prompts.

**Step 2:** The user enters the credentials

**Step 3:** The user gets access to Play Games, View Leaderboard, View Recorded Games.

**4.2.3 Functional Requirements**

**REQ-2. A:** The user signs in through username and password. **Priority 1**

**REQ-2. B:** The user credentials are validated from the Database. If correct, the user is logged in. **Priority 1**

**REQ-2. C:** An error message will be prompted if the user enters wrong user credentials. **Priority 1**

**REQ-2. D:**  If the user has forgotten their password, the user clicks on the “Forget Password” and a dialog box is prompted which asks for the user's email through which the password is recovered. **Priority 1**

**REQ-2. E:** The user shall be allowed to edit their profile when signed into the chess and checkers game. **Priority 2REQ-2. F:** The user shall be allowed to change their password, username and avatar when editing their profile. **Priority 2**

**REQ-2. G:** A signed in user shall be allowed to sign out from the application. **Priority 1**

**REQ-2. H:** A blocked user shall not be allowed to sign-in to the application. **Priority 1**

## Play a Multiplayer Game

**4.3.1 Description and Priority**

Our multiplayer option features both online and couch co-op style multiplayer. While online, Our game will, make a secure connection via internet , then find an online player of the game and make a connection between the two and then the game can commence. While offline, players can enjoy a retro style game of chess and checkers where both players will be using one device to play the game.

**4.3.2 Stimulus/Response Sequences**

**Step 1:** User enters the multi-player game mode.

**Step 2:** The screen loads and appears, prompting the player with two buttons

 “Play Chess” or “Play Checkers”

**Step 3:** The player presses one of the buttons, triggering its respective function.

**4.3.3 Functional Requirements**

**REQ-3. A:** Players shall be asked to select either Chess or Checkers. **Priority 1**

**REQ-3. B:** Upon choosing to start a new multiplayer game, the user is placed into a lobby where they will wait. If another user is also present in the lobby and has not been matched up then the two users are selected to play against each other in a game. **Priority 1**

**REQ-3. C:** A connection will be established between the devices of the two users involved in the game. The connections will be made possible through the IP addresses. **Priority 1**

**REQ-3. D:** If the user leaves the game after it is matched with another player, a loss is registered for the leaving user and win for the opponent. **Priority 1**

**REQ-3. E:** A timer starts when a player gets their turn. If the timer ends and the player has not made a move, the player’s turn is skipped. **Priority 2**

**REQ-3. F:** If the player skips a certain number of consecutive moves, then the game is forfeit and registered as a win for the opponent. **Priority 2**

**REQ-3. G:** The user shall be allowed to leave the lobby until he isn’t matched with another player and select another game type after initially selecting the type of game, i.e., Chess and Checkers. **Priority 2**

## Play Single Player Game

**4.4.1 Description and Priority**

In this mode the player can play with the AI incorporated into our game. The AI behaves in such a way that it provides three difficulty levels.

**4.4.2 Stimulus/Response Sequences**

**Step 1:** User enters the single-player game mode.

**Step 2:** The screen loads and appears, prompting the player to press “Choose Difficulty Level” button.

**Step 3:** The player presses one of the buttons, triggering its respective function.

**4.4.3 Functional Requirements**

**REQ-4. A:** Upon choosing to start a new single-player game, the user will be asked to choose a difficulty level: Rookie, intermediate, grand master . **Priority 1**

**REQ-4. B:** While playing the game in single-player mode, the user shall be allowed to restart the game. **Priority 2**

## Difficulty Level

**4.5.1 Description and Priority**

               The user shall be able to choose a difficulty level of the AI in a single player

              game. There will be 3 levels: rookie, intermediate, grand master. the default level

              shall be rookie.

**4.5.2 Stimulus/Response Sequences**

**Step 1:** User enters the difficulty level selection mode.

**Step 2:** The screen loads and appears, prompting the player to choose either of the three buttons, “Rookie” , “Intermediate” and “Grand Master”.

**Step 3:** The player presses one of the buttons, triggering its respective function.

**4.5.3 Functional Requirements**

**REQ-5. A:** The user will be asked to choose a difficulty level, there will be 3 levels: **Priority 1**

1. Rookie
2. Intermediate
3. Grand Master

**REQ-5. B:** Players shall be asked to select either Chess or Checkers. **Priority 1**

## Viewing Replay

**4.6.1 Description and Priority**

  Players can View their Match replays which can help them find out what mistakes

  they were doing and to improve their match strategies for later matches.

**4.6.2 Stimulus/Response Sequences**

**Step 1:** Player requests for match replay

**Step 2:** The recording for that game that player wants is fetched from the

              database and displayed to the Player.

**4.6.3 Functional Requirements**

**REQ-6. A:** The user can view the replays of games that they have previously completed. **Priority 1**

**REQ-6. B:** The moves made during a game are recorded so that the replay of the game can be viewed by the players involved. **Priority 1**

**REQ-6. C:** For a single-player game the replay can be viewed by only the player while for a multi-player game both players can view the replay. Any other party not involved within a game will not be allowed to view the replay of the game.

**Priority 2**

## Show Possible Moves

**4.7.1 Description and Priority**

When the user selects a piece, all possible moves available for the piece will be

highlighted. if the piece that is selected is in a position to capture the opponent's

piece then, that move will be shown with a different color to help the user.

**4.7.2 Stimulus/Response Sequences**

**Step 1:** User selects a piece

**Step 2:** All possible moves for that piece will be highlighted.

**4.7.3 Functional Requirements**

**REQ-7. A:** Depending on the game when the user picks a piece, the squares on which a valid move is possible will be highlighted. **Priority 2**

**REQ-7. B:** If the user clicks anywhere other than the highlighted squares, then the selected piece will be deselected.  **Priority 2**

**REQ-7. C:** If the user clicks on a second piece while a piece is already selected, then the second piece is selected, and the possible valid moves are shown for it.

**Priority 1**

**REQ-7. D:** The user shall be allowed to beat an opponent piece where it is possible, depending upon the highlighted square shown and within the rules of the game. **Priority 1**

## Leaderboard Management

**4.8.1 Description and Priority**

Players can view their rank in chess and checkers separately. Players can see where they stand relative to their peers. Leaderboard can develop the desire to play better in players.

**4.8.2 Stimulus/Response Sequences**

**Step 1:** Player chooses if they want to view the chess leaderboard or the checkers leaderboard.

**Step 2:** The selected game leaderboard is shown to the player with the ranks of all the players who have played.

**4.8.3 Functional Requirements**

**REQ-8. A:** The user can view the leaderboard containing records of the previous players. The leaderboard contains for each player record:

* + The wins
  + The losses
  + The win ratio (**Priority 1)**

**REQ-8. B:** When a game ends, the leaderboard records will be updated. The winner’s total wins are incremented and so are the loser’s losses. **Priority 1**

**REQ-8. C:** If a multiplayer game is forfeit, the player at loss will be recorded for the player in the leaderboard and correspondingly a win for the opponent. **Priority 1**

**REQ-8. D:** The leaderboard is sorted by the win ratio of each player. The number of wins can act as a tiebreaker if needed. **Priority 2**

**REQ-8. E:** 1. For single-player games, the wins and losses are not registered in the leaderboard. The user can quit or forfeit a single-player game without registering a loss. Leaderboard is only updated after multiplayer games. **Priority 2**

## Playing Chess Game

**4.9.1 Description and Priority**

This feature is a key feature in our project. As the name of our game suggests, this feature will allow us to play a game of chess.

**4.9.2 Stimulus/Response Sequences**

**Step 1:** Game Starts. the player with the light pieces (white) makes the first move, players move alternatively, with the player with the dark-colored pieces, making the next move.

**Step 2:** Moves validity is checked.

**Step 3:** It results in a win/loss after successful completion, or it is forfeited.

**4.9.3 Functional Requirements**

**REQ-9. A:** There shall be 6 different types of pieces: Pawn, Rook, Knight, Bishop, Queen, King. **Priority 1**

**REQ-9. B:** There shall be 2 knights, bishops, rooks and 1 Queen and King.

**Priority 1**

**REQ-9. C:**

**Castling:** The king can move 2 blocks towards a rook and the rook then moves 2 blocks over which the king crossed. The following requirements must be met before castling can happen:

* + The king has never moved. **Priority 1**
  + The rook has never moved. **Priority 1**
  + The king is not in the check. **Priority 1**
  + The king does not cross or end on a block in which it would be in check. **Priority 1**
  + There should be no piece in between the king and rook. **Priority 1**

**Movements:**

**REQ-9. D:** User shall move Pawn 1 block forward and can move 2 blocks on every pawn’s opening move. **Priority 1**

**REQ-9. E:** User shall move Rooks vertically or horizontally any number of blocks unless occupied by another piece. **Priority 1**

**REQ-9. F**: User shall move knight two blocks either vertically or horizontally followed by one block perpendicularly. **Priority 1**

**REQ-9. G:** User shall move Bishop diagonally any number of blocks. **Priority 1**

**REQ-9. H:** User shall move Queen vertically, horizontally or diagonally any number of blocks. **Priority 1**

**REQ-9. I:**User shall move King one block in any direction except for a block one which there is check (A block where an opponent’s piece can capture or move). **Priority 1**

**REQ-9. J:**  When requirements are met for castling (see castling requirements above), the king may move 2 blocks towards a rook, with the rook moving onto the block crossed over by the king. **Priority 1**

**Capturing:**

**REQ-9. K:**  Any piece other than a pawn, moving in its normal requirements, may move into a block occupied by an opposing piece, the friendly piece may capture the opposing piece. **Priority 1**

**REQ-9. L:**  Pawns shall capture by moving forward one space diagonally into an opposing piece. **Priority 1**

**Pawn promotion:**

**REQ-9. M:**  If a pawn reaches the opponent’s end the user has the choice to promote the pawn to a queen, rook, bishop, or a knight. **Priority 1**

**REQ-9. N:** When the game starts, the lighter colored piece makes the first move and then the other player takes the turn. **Priority 1**

## Playing Checkers Game

**4.10.1 Description and Priority**

This feature relates to the other half of our game, the game of checkers. In this feature the user will be able to play a game of checkers.

**4.10.2 Stimulus/Response Sequences**

**Step 1:** Game Starts. the player with the dark pieces (black) makes the first move, players move alternatively, with the player with the light-colored pieces, making the next move.

**Step 2:** Moves validity is checked.

**Step 3:** It results in a win/loss after successful completion, or it is forfeited.

**4.10.3 Functional Requirements**

**REQ-10. A:**  The player with the darker colored checker (black checker in a black and white theme game) makes the first move and then the other player takes the turn. **Priority 1**

**REQ-10. B:**  Check the validity of the move. **Priority 1**

**REQ-10. C:** A player can move the checker one diagonal space forward (towards your opponent checkers. **Priority 1**

**REQ-10. D:** You can move your checkers only on the dark squares. **Priority 1**

**REQ-10. E:** If your checker is in one diagonal space to the opponent’s checker, then the player can jump and capture the opponent’s checker by moving two diagonal spaces in the direction of the checker that the player is attacking. **Priority 1**

**REQ-10. F:** The opponent’s other side square must be empty for the player to move during capturing the opponent's checker. **Priority 1**

**REQ-10. G:** It is mandatory for the player to jump over the opponent’s checker if the opportunity arises. **Priority 1**

**REQ-10. H:** In case of multiple capturing opportunities, the player must choose one. **Priority 1**

**REQ-10. I:** If after a single capture, an opportunity for another capture is present then the player keeps on jumping checkers in the single turn until the player can’t capture anymore checkers **Priority 1**

**REQ-10. J:** When a player’s checker reaches the end of the opponent's side, the player crowns the checker. A checker is crowned and made a king checker by placing one of the players' captured checkers on top of it. **Priority 1**

**REQ-10. K:** King Checker can move forward and backward diagonally and can move only one diagonal space during a regular turn **Priority 1**

**REQ-10. L** The player whose checkers are the last to be left on the board wins the game. **Priority 1**

## Change Theme

**4.11.1 Description and Priority**

Users can choose the color scheme of their choice. The default theme will be black and white for checkers. For chess the default theme will be Dark brown and light brown. This feature will allow the player to choose different themes according to his liking. We will have a preset of themes which would include common combinations such as a ‘blue & red’ combination in the chess game and many others. Same rule applies to the checkers game, it will also include predefined common combinations.

**4.11.2 Stimulus/Response Sequences**

**Step 1:** User chooses the theme of their choice

**Step 2:** The board theme changes to the selected theme by the user

**4.11.3 Functional Requirements**

**REQ-11. A:** During a match of any game mode or any game type, the user shall be allowed to change the theme of the board and pieces from a set of predefined themes. **Priority 2**

## Admin System Features

**4.12.1 Description and Priority**

The admin is responsible for deleting or blocking users in case there is a suspicion for malicious act. The admin can choose the sorting criteria for the leaderboard, as well as responsible for resetting it.

**4.12.2 Stimulus/Response Sequences**

**Deleting/Blocking User:**

**Step 1:** The admin wants to delete a user.

**Step 2:** The admin deletes or blocks the specific user from the database.

**Leaderboard Manipulation:**

**Step 1:** The admin sorts the leaderboard according to the criteria of the user.

**Step 2:** The admin resets it when there is a new update.

**4.12.3 Functional Requirements**

**REQ-12. A:** The admin should be able to reset the leaderboard. **Priority 2**

**REQ-12. B:** The admin sorts the leaderboard according to the wins or win/loss ratio.

**Priority 2**

**REQ-12. C:** The admin should be able to delete or block/unblock a user. **Priority 1**

**REQ-12. D:** When the user that is deleted, all his information that is stored in the database, should be deleted alongside. **Priority 2**

# Use Case Modelling

## Diagram Description automatically generated

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Create Profile** | |
| Related Requirements | REQ-1. A, REQ-1. B, REQ-1. C, REQ-1. D, REQ-1. E | |
| Goal in Context | A user of the chess and checkers game creates a new profile. | |
| Preconditions | 1. The user has opened the chess and checkers game on their device. 2. The user is not currently logged into a profile. | |
| Successful End Condition | New user profile successfully created. | |
| Failed End Condition | Request for creation of new profile rejected. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to create a profile. | |
| Included Cases | **Check Information Duplication** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user requests the app to create a profile. |
|  | 2 | The system responds by asking the user to enter their email, password, username and selecting avatar. |
|  | 4 | The user submits the form after filling in the required details. |
|  | 5  **include::Check Information Duplication** | The system checks the details provided by the user. |
|  | 5 | The system responds by confirming profile creation. |
|  | 6 | The use case ends. |

## Use Case 1: Create Profile

## Use Case 2: Check Duplicate Information

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check Information Duplication** | |
| Related Requirements | REQ-1. D, REQ-1. E | |
| Goal in Context | Verifying that the details entered by the user do not correspond to an already existing profile. | |
| Preconditions | 1. The details provided by the user do not exist in the database. | |
| Successful End Condition | The details are valid and can be used for a profile. | |
| Failed End Condition | The details are invalid and are already being used. | |
| Primary Actors | User Database | |
| Secondary Actors | **None** | |
| Trigger | The profile details are provided to the system for verification. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the user supplies the email, password, and username to the system. |
|  | 2 | The User Database checks the user credentials records and checks whether the email, password, and username provided do not conflict with an already existing profile. |
|  | 3 | The User Database verifies the profile details and sends a response to the client application confirming the usability of the details for a profile. |
|  | 4 | The use case ends. |
| Extensions | Step | Branching Actions |
|  | 2.1 | The User Database rejects the user details supplied. |
|  | 2.2 | The User Database sends a response to the client application informing of the rejection. |
|  | 2.3 | The user is asked to supply valid details. |
|  | 2.4 | The use case continues at step 1. |

## Use Case 3: Sign Into Profile

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Sign Into Profile** | |
| Related Requirements | REQ-2. A, REQ-2. B, REQ-2. C, REQ-2. D, REQ 2- H | |
| Goal in Context | A user of the chess and checkers wishes to sign into an existing profile. | |
| Preconditions | 1. The user has opened the chess and checkers game on their device. 2. The user is not currently logged into a profile. 3. The user has a profile stored in the database. | |
| Successful End Condition | 1. User is successfully logged into the profile corresponding to the specified credentials entered. 2. The user gains access to the app’s features like playing multiplayer game. | |
| Failed End Condition | User is not logged into the requested profile. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to sign them into a profile. | |
| Included Cases | **Check Login Credentials** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user wishes to sign in to the game. |
|  | 2 | The system responds by asking the user to enter their details. |
|  | 3  **include::Check Login Credentials** | The system checks the details provided by the user. |
|  | 5 | The system signs the user in. |
|  | 6 | The use case ends. |

## Use Case 4: Check Login Credentials

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check Login Credentials** | |
| Related Requirements | REQ-2. B, REQ-2. C | |
| Goal in Context | Verifying the details entered by the user when login credentials are submitted. | |
| Preconditions | There is a record in the database corresponding to the user’s credentials. | |
| Successful End Condition | The credentials sent are successfully verified. | |
| Failed End Condition | The credentials sent are not verified. | |
| Primary Actors | User Database | |
| Secondary Actors | **None** | |
| Trigger | The credentials are provided to the system for verification. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the user supplies the email and password to the system. |
|  | 2 | The User Database checks the user credentials records and checks whether the email and password provided are correct. |
|  | 3 | The User Database verifies the user credentials and sends a response to the client application confirming the user’s sign in. |
|  | 4 | The use case ends. |
| Extensions | Step | Branching Actions |
|  | 2.1 | The User Database rejects the user credentials supplied. |
|  | 2.2 | The User Database sends a response to the client application informing of the rejection. |
|  | 2.3 | The user is asked to supply valid credentials. |
|  | 2.4 | The use case continues at step 1. |

## Use Case 5: Edit Profile

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Edit Profile** | |
| Related Requirements | REQ-1. E, REQ-2. E, REQ-2. F | |
| Goal in Context | A user of the chess and checkers game wishes to edit their profile. | |
| Preconditions | 1. The user has a profile stored in the database. 2. The user is signed in. | |
| Successful End Condition | The user’s profile is successfully updated according to the wishes of the user. | |
| Failed End Condition | The user’s profile is not updated. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to edit their profile. | |
| Included Cases | **Check Information Duplication** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins with the user asking the app to edit their profile. |
|  | 2 | The system responds with the user’s current profile state which includes the user’s email, password, avatar, and username. |
|  | 3 | The system renders the user’s profile by displaying the avatar and displaying the email, password, and username in input fields. |
|  | 4 | The user edits any of password, username, and the avatar. |
|  | 5 | The user submits the changes. |
|  | 6  **include::Check Information Duplication** | The system checks the details provided by the user. |
|  | 7 | The system updates the user profile. |
|  | 8 | The use case ends. |

## Use Case 6: Sign Out

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Sign Out** | |
| Related Requirements | REQ-2. G | |
| Goal in Context | A user of the chess and checkers game wishes to sign out. | |
| Preconditions | The user is already signed in. | |
| Successful End Condition | The user is successfully signed out of their profile. | |
| Failed End Condition | The user is not signed in. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user asks the app to sign them out of their profile. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The user asks the app to sign them out of their profile by clicking the Sign Out button. |
|  | 2 | The system signs the user out and deletes the session or cookie. |
|  | 3 | The app shows the option to sign in again to the user. |
|  | 4 | The use case ends. |

## Use Case 7: Play Game

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Play Game** | |
| Related Requirements | REQ-7. A, REQ-7. B, REQ-7. C, REQ-7. D  REQ-9. A, REQ-9. B, REQ-9. C, REQ-9. D, REQ-9. E, REQ-9. F, REQ-9. G, REQ-9. H, REQ-9. I, REQ-9. J, REQ-9. K, REQ-9. L, REQ-9. M,  REQ-9.N  REQ-10. A, REQ-10. B, REQ-9. C, REQ-10. D, REQ-10. E, REQ-10. F, REQ-10. G, REQ-10. H, REQ-10. I, REQ-10. J, REQ-10. K, REQ-10. L | |
| Goal in Context | The user wishes to play a game. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. | |
| Successful End Condition | The match starts and the user can play the game. | |
| Failed End Condition | The match does not start, and the user is disallowed to play a game. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to play a game by clicking the Play Game button. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user asks the app to play a game by clicking the play game button. |
|  | 2 | The app responds by asking user to select the game mode. |
|  | 3 | The user then selects the game mode. |
|  | 4 | The system takes the user to the game screen and the match starts. |
|  | 5 | The use case ends. |

## Use Case 8: Single Player Game

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Single player Game** | |
| Related Requirements | REQ-4. A, REQ-4. B  REQ-5. A, REQ-5. B | |
| Goal in Context | The user wishes to play a single player game against the computer. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. | |
| Successful End Condition | The match starts and the user can play a single player game against the computer. | |
| Failed End Condition | The match does not start, and the user is disallowed to play a single player game against the computer. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user asks the app to play a single player game. | |
| Included Cases | **None** | |
| Base Use Cases | **Play Game** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user asks the app to play a game by clicking the play game button. |
|  | 2 | The app responds by asking user to select the game mode. |
|  | **3** | **The user then selects the Single Player game mode.** |
|  | **4** | **The app responds by asking the difficulty level of the AI/computer.** |
|  | 5 | The system takes the user to the game screen and the match starts. |
|  | 6 | The use case ends. |
| Extensions | Step | Branching Actions |
|  | **5.1** | **The user elects to restart the game.** |
|  | **5.2** | **The match is restarted.** |
|  | **5.3** | **The use case continues at step 5.** |

## Use Case 9: Multi Player Game

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Multiplayer Game** | |
| Related Requirements | REQ-3. A, REQ-3. B, REQ-3. C, REQ-3. D, REQ-3. E, REQ-3. F, REQ-3. G. | |
| Goal in Context | The user wishes to play a Multiplayer game against another user. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. 3. The opponent user is also signed in. | |
| Successful End Condition | The match starts and the user can play the Multiplayer game against an opponent user. | |
| Failed End Condition | The match does not start, and the user is disallowed to play a Multiplayer game against opponent user. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user asks the app to play a multiplayer game. | |
| Included Cases | **None** | |
| Base Use Cases | **Play Game** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user asks the app to play a game by clicking the play game button. |
|  | 2 | The app responds by asking user to select the game mode. |
|  | **3** | **The user then selects the Multiplayer game mode.** |
|  | **4** | **The system then responds by placing the user into a lobby for being matched up with another player.** |
|  | **5** | **The user is matched up with another user and the multiplayer match begins.** |
|  | 6 | The use case ends. |
| Extensions | Step | Branching Action |
|  | **4.1** | **The user exits the lobby.** |
|  | **4.2** | **The use case continues at step 2.** |
|  | **5.1** | **The user forfeits the game.** |
|  | **5.2** | **The use case ends.** |

## Use Case 10: View Leaderboard

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **View Leaderboard** | |
| Related Requirements | REQ-8. A, REQ-8. B, REQ-8. C, REQ-8. D, REQ-8. E | |
| Goal in Context | A user of the chess and checkers game wishes to view the leaderboard. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. | |
| Successful End Condition | The leaderboard is shown to the user. | |
| Failed End Condition | The leaderboard is not shown. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to display the leaderboard. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the user asks the app to show the leaderboard. |
|  | 2 | The user selects the game type to view the leaderboard for. |
|  | 3 | The leaderboard is shown. |
|  | 4 | The use case ends. |

## Use Case 11: Chess Leaderboard

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Chess Leaderboard** | |
| Related Requirements | REQ-8. A, REQ-8. B, REQ-8. C, REQ-8. D, REQ-8. E | |
| Goal in Context | A user wishes to view the leaderboard for chess. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. | |
| Successful End Condition | The leaderboard for chess is shown to the user. | |
| Failed End Condition | The leaderboard for chess is not shown. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to display the leaderboard for chess. | |
| Included Cases | **None** | |
| Base Use Cases | **View Leaderboard** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the user asks the app to show the leaderboard. |
|  | **2** | **The user elects to view the leaderboard for chess.** |
|  | **3** | **The leaderboard for chess is shown to the user.** |
|  | 4 | The use case ends. |

## Use Case 12: Checkers Leaderboard

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Checkers Leaderboard** | |
| Related Requirements | REQ-8. A, REQ-8. B, REQ-8. C, REQ-8. D, REQ-8. E | |
| Goal in Context | A user wishes to view the leaderboard for checkers. | |
| Preconditions | 1. The app is properly started. 2. The user is signed in. | |
| Successful End Condition | The leaderboard for checkers is shown to the user. | |
| Failed End Condition | The leaderboard for checkers is not shown. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to display the leaderboard for checkers. | |
| Included Cases | **None** | |
| Base Use Cases | **View Leaderboard** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the user asks the app to show the leaderboard. |
|  | **2** | **The user elects to view the leaderboard for checkers.** |
|  | **3** | **The leaderboard for checkers is shown to the user.** |
|  | 4 | The use case ends. |

## Use Case 13: View Replay

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **View Replay** | |
| Related Requirements | REQ-6. A, REQ-6. B, REQ-6.C | |
| Goal in Context | A user of the chess and checkers game wishes to view the replay of a game they played. | |
| Preconditions | 1. The user is signed in. 2. The user has played at least one game. | |
| Successful End Condition | The replay is shown to the user. | |
| Failed End Condition | The replay is not shown. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to show a replay of a game they played. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The user requests the app to show a replay of a game they played. |
|  | 2 | The user selects the replay to view. |
|  | 3 | The replay is shown. |
|  | 4 | The use case ends. |

## Use Case 14: Change Theme

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Change Theme** | |
| Related Requirements | REQ-11. A | |
| Goal in Context | The user wishes to change the appearance of the board and pieces. | |
| Preconditions | 1. The user has a game in progress. | |
| Successful End Condition | The appearance of the board and pieces is changed. | |
| Failed End Condition | The appearance of the board and pieces is not changed. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user requests the app to change the theme of the board and pieces. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The user requests the app to change the theme of the board and pieces. |
|  | 2 | The system responds by showing the theme options to the user. |
|  | 3 | The user selects a theme. |
|  | 4 | The theme of the board and pieces is changed. |
|  | 5 | The use case ends. |
| Extensions | Step | Branching Actions |
|  | 2.1 | The user does not select a theme. |
|  | 2.2 | The theme is not changed. |
|  | 2.3 | The use case ends. |

## Use Case 15: Delete User

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Delete User** | |
| Related Requirements | REQ-12. C, REQ-12. D | |
| Goal in Context | The admin wishes to delete a user. | |
| Preconditions | 1. The admin is signed in. 2. The user to be deleted exists in the database. | |
| Successful End Condition | The chosen user is deleted from the database. | |
| Failed End Condition | The chosen user is not deleted. | |
| Primary Actors | Admin | |
| Secondary Actors | **None** | |
| Trigger | The admin user requests the system to delete a user. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the admin user requests the system to delete a user. |
|  | 2 | The system responds by showing all users registered on the chess and checkers game. |
|  | 3 | The admin selects the user to be deleted. |
|  | 4 | The user selected is deleted. |
|  | 5 | The use case ends. |

## Use Case 16: Block User

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Block User** | |
| Related Requirements | REQ-12. C | |
| Goal in Context | The admin wishes to Block a user. | |
| Preconditions | 1. The admin is signed in. 2. The user to be blocked exists in the database. 3. The user to be blocked is not already blocked. | |
| Successful End Condition | The chosen user is blocked from the database. | |
| Failed End Condition | The chosen user is not blocked. | |
| Primary Actors | Admin | |
| Secondary Actors | **None** | |
| Trigger | The admin user requests the system to block a user. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the admin user requests the system to block a user. |
|  | 2 | The system responds by showing all users registered on the chess and checkers game. |
|  | 3 | The admin selects the user to be blocked. |
|  | 4 | The selected user is blocked. |
|  | 5 | The use case ends. |

## Use Case 17: Unblock User

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Unblock User** | |
| Related Requirements | REQ-12. C | |
| Goal in Context | The admin wishes to Unblock a blocked user. | |
| Preconditions | 1. The admin is signed in. 2. The user to be unblocked exists in the database. 3. The user to be unblocked is blocked. | |
| Successful End Condition | The chosen user is unblocked from the database. | |
| Failed End Condition | The chosen user remains blocked. | |
| Primary Actors | Admin | |
| Secondary Actors | **None** | |
| Trigger | The admin user requests the system to unblock a user. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case begins when the admin user requests the system to unblock a user. |
|  | 2 | The system responds by showing all blocked users on the chess and checkers game. |
|  | 3 | The admin selects the user to be unblocked. |
|  | 4 | The selected user is unblocked. |
|  | 5 | The use case ends. |

## Use Case 18: Reset Leaderboard

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Reset Leaderboard** | |
| Related Requirements | REQ-12. A | |
| Goal in Context | The admin wishes to clear all records from leaderboard. | |
| Preconditions | 1. The admin is signed in. 2. The user to be deleted exists in the database. | |
| Successful End Condition | All records are cleared from leaderboard. | |
| Failed End Condition | The leaderboard remains unchanged. | |
| Primary Actors | Admin | |
| Secondary Actors | **None** | |
| Trigger | The admin user requests the app to reset the leaderboard. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The admin user requests the app to reset the leaderboard. |
|  | 2 | The system responds by clearing all the records in the leaderboard. |
|  | 3 | The use case ends. |

## Use Case 19: Define Leaderboard Sorting Criteria

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Define Leaderboard Sorting Criteria** | |
| Related Requirements | REQ-12. B | |
| Goal in Context | The admin wishes to define criteria for sorting of the leaderboard. | |
| Preconditions | 1. The admin is signed in. 2. The user to be deleted exists in the database. | |
| Successful End Condition | All records are sorted in the leaderboard by the admin’s criteria. | |
| Failed End Condition | The leaderboard remains unchanged. | |
| Primary Actors | Admin | |
| Secondary Actors | **None** | |
| Trigger | The admin user asks the app to allow defining leaderboard sorting criteria. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The admin user requests the app to allow defining leader board sorting criteria. |
|  | 2 | The admin defines the criteria. |
|  | 3 | The leaderboard is sorted. |
|  | 4 | The use case ends. |

## Use Case 20: Forgot Password

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Forgot Password** | |
| Related Requirements | REQ-2. D | |
| Goal in Context | A user of the chess and checkers game wishes to reset their password because they have forgotten it. | |
| Preconditions | 1. The user has a profile registered and stored in the database. 2. The user is not currently logged into a profile. | |
| Successful End Condition | The password reset code is dispatched to the user’s registered email. | |
| Failed End Condition | The request for resetting the password is rejected. | |
| Primary Actors | User | |
| Secondary Actors | **None** | |
| Trigger | The user asks the app to reset their password by clicking on the Forgot Password link or button. | |
| Included Cases | **Check Email Exists** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user clicks on the Forgot Password link or button. |
|  | 2 | The system responds by asking the user to enter the email of the profile of which they wish to reset the password. |
|  | 4 | The user enters the email and submits it to the system. |
|  | 5  **include::Check Email Exists** | The system checks if the profile with the submitted email exists in the user database. |
|  | 6 | The system dispatches an email containing the code and instructions for password reset. |
|  | 7 | The use case ends. |

|  |  |  |
| --- | --- | --- |
| **Use Case Name** | **Check Email Exists** | |
| Related Requirements | REQ-2. D | |
| Goal in Context | Verifying that the email provided matches to a registered profile. | |
| Preconditions | There is a profile matching with the email provided by the user in the database. | |
| Successful End Condition | The email is returned as verified by the system. | |
| Failed End Condition | The email is not verified by the system. | |
| Primary Actors | User Database | |
| Secondary Actors | **None** | |
| Trigger | The user asks the app to reset their password by clicking on the Forgot Password link or button. | |
| Included Cases | **None** | |
| Main Flow | Step | Action |
|  | 1 | The use case starts when the user enters the email and submits it to the system. |
|  | 2 | The system checks if the profile with the submitted email exists in the user database. |
|  | 3 | The system verifies that the email corresponds to a profile. |
|  | 4 | The use case ends. |
| Extensions | Step | Branching Action |
|  | 2.1 | The system does not find a profile matching to the email provided. |
|  | 2.2 | The email is not verified by the system and is rejected. |
|  | 2.3 | The use case ends. |

## Use Case 21: Check Email Exists

# Other Nonfunctional Requirements

## Performance Requirements

The system must be interactive, and the delay must be as less as possible. To avoid overloading of the server and keep its performance at optimal level the server should be spam proof and should have the functionality to ban the IP Address for a duration of 5 minutes from which it receives spam requests.

The Database should use optimize sorting, evaluation queries to minimize the loading time to preferably less than 2 second. The code should use try catch blocks to return optimal error response whenever it receives a bad request and to stop it from crashing whenever some app breaking bug occurs.

## Safety Requirements

The following safety requirements need to be paid attention to:

1. Any ads displayed on the app do not redirect the user to a malicious link and are skippable.
2. The game should not be too hardware intensive and should be able to run comfortably on even low-end machines without causing significant damage to the parts of the machine.
3. The users’ phone’s temperature must not exceed too much while playing the game.
4. While the game is running, it should not cause disruption to system processes.
5. The game asks the player to take a short break after completing a match to prevent eye strain.

## Security Requirements

* **Database:**

The database will be hosted on a private server. The only way to gain access to the server will be to use SSH (Secure Shell).  the server will only allow SSH requests from specific IP’s that will be configured into it, for authentication of SSH we will use public key authentications. The database will be protected using username and password, password will be generated using a keyboard mash. The user profile password will be stored in encrypted format. The password will be encrypted using Salted SHA-2.

The Database itself will be protected against SQL injection attacks by using parameterized

queries.

The server itself will be protected against DDOS attack by limiting broadcast and validation against spoofing.

* **Authentication and Authorization:**

For authentication and authorization purposes we will be using JWT tokens. Each request will contain an authorization header with a proper token. If the app received any request without the token or if the token is invalid the server will return the error status code 401 or if the request contains the proper token but the user doesn’t have the permission to access that service, the server will return the error status code 403.

* **Front-End Security:**

To protect the app from spam and abuse, the app will be integrated with google reCAPTCHA service.

## Software Quality Attributes

Attributes are defined in order of their importance.

1. **Usability:**
   * The site should be easy to use.
   * The layout should be simple and self-explanatory.
   * The buttons should have appropriate data-toggle attributes to explain their purposes and the images should also have appropriate alt tags.
2. **Audit Log:**
   * The app will Log every relevant information in an external text file which will be stored for a specific amount of time in the server.
   * User created logs will be store in file call userlog.txt in a format of “User id” “function perform” “date time”
   * All exception occurs during app lifetime will be store in a file call appException.txt in a format of “Exception” “date time created”
3. **Testability:**
   * The unit test should cover at least 90+% of the app. For both backend and frontend. The app should be tested for every scenario and cover user stories.
   * Every third-party library should be tested for security flaws or vulnerabilities.
4. **Upgradability:**
   * The app should be written using up-to-date OOP concepts. So, the code is maintainable and easy to change.
   * Upgrade to all third-party libraries use in the app must be performed in 30 days of their release.

## Business Rules

1. The chess game rules must be according to the guideline approved by The International Chess Federation.
2. The checkers game rules must also be according to the guideline approved by The World Draught Federation.
3. User Id and email must be unique
4. Users must log in to play the game, to watch his recorded gameplay and to update his profile.
5. Users can only play one game at a time.
6. Records must be updated regularly
7. User must input his old password first to reset his password

# Other Requirements

**Format of replay:**

Every game move will be stored in a table call Game Moves which will have a one to many relationship with our primary Game table, The data in Game Moves table will look like this. where p=pawn, N=knight, Q=Queen. For replay we will replay this whole scenario on the board.

|  |  |  |
| --- | --- | --- |
| Game Id | Move1 | Move2 |
| 1 | pe4 | pe5 |
| 1 | Nh3 | Qh4 |
| 1 | pb4 | Nf6 |

Appendix A: Glossary

* Authentication: the process of verifying who a user is
* Authorization: process of verifying what they have access to
* Bishop: Chess piece
* Check: A situation in which the king can be captured by other pieces
* Checkmate: A situation in which the king can be captured by other pieces and cannot escape. Game over.
* Chess: A strategy game played between two players and won by the one who captures the opponent's king.
* Checkers: Checkers is a game played on a board checkered with squares of two colors. Two players compete in checkers to have the last piece on the board.
* DOS (Denial of service) is a malicious attempt to disrupt the normal traffic of a targeted server, service or network by overwhelming it with requests.
* Google reCAPTCHA [ref](https://www.google.com/recaptcha/about/)
* IP Spoofing is the creation of Internet Protocol (IP) packets which have a modified source address in order to either hide the identity of the sender.
* JWT Tokens industry standard RFC 7519 method for representing claims securely between two parties. Certified by oAUTH.
* Keyboard mash, to create a password by blindly mashing the keyboard three times eg: fsdklfhiowfkldsdvklfhiocn.
* King : Chess piece
* Knight: Chess piece
* Limit Broadcasting, Limiting the number of requests a server will accept over a certain time window from the same user.
* Public-key authentication is a cryptographic system that uses pairs of keys. Public key and Private key in such a system, any person can encrypt a message using the intended receiver's public key, but that encrypted message can only be decrypted with the receiver's private key.
* Pawn: Chess piece
* Queen: Chess piece
* Rook: Chess piece
* SSH (Secure Shell) is a cryptographic network protocol for operating network services securely over an unsecured network
* SQL injections a web security vulnerability that allows an attacker to interfere with the queries that an application makes to its database.
* Salted SHA to encrypt each password with a uniquely generated value called salt.
* stalemate: when there is a tie between the 2 players. no possible moves left.
* 401 unauthorized error status code, part of HTTP status code.
* 403 forbidden error status code, also part of HTTP status code.

Diagram

Description automatically generatedAppendix B: Analysis Model

Appendix C: To Be Determined List

* Dependencies
* Design and Implementation Constraints