SE 3XA3: Software Requirements Specification
Supreme Chess

Team 2, The Triple Grobs
Jonathan Cels (celsj)
Pesara Amarasekera (amarasep)
Rupinder Nagra (nagrar5)

April 12, 2021

Contents

1 Project Drivers **2**

1.1 The Purpose of the Project 2

1.2 The Stakeholders 2

1.2.1 The Client 2

1.2.2 The Customers 2

1.2.3 Other Stakeholders 2

1.3 Mandated Constraints 2

1.4 Naming Conventions and Terminology 2

1.5 Relevant Facts and Assumptions 3

2 Functional Requirements **3**

2.1 The Scope of the Work and the Product 3

2.1.1 The Context of the Work 3

2.1.2 Work Partitioning 3

2.1.3 Individual Product Use Cases 5

2.2 Functional Requirements 10

3 Non-functional Requirements **14**

3.1 Look and Feel Requirements 14

3.1.1 Appearance Requirements 14

3.1.2 Style Requirements 15

3.2 Usability and Humanity Requirements 15

3.2.1 Ease of Use Requirements 15

3.2.2 Personalization and Internationalization Requirements 15

3.2.3 Learning Requirements 15

3.2.4 Understandability and Politeness Requirements 15

3.2.5 Accessibility Requirements 16

3.3 Performance Requirements 16

3.3.1 Speed and Latency Requirements 16

3.3.2 Safety-Critical Requirements 16

3.3.3 Precision or Accuracy Requirements 16

3.3.4 Reliability and Availability Requirements 16

3.3.5 Robustness or Fault-Tolerance Requirements 16

3.3.6 Capacity Requirements 16

3.4 Operational and Environmental Requirements 17

3.4.1 Installability Requirements 17

3.4.2 Requirements for Interfacing with Adjacent Systems 17

3.4.3 Productization Requirements 17
3.4.4 Release Requirements 17
3.5 Maintainability and Support Requirements 17
3.5.1 Maintenance Requirements 17
3.5.2 Supportability Requirements 17
3.5.3 Adaptability Requirements 17
3.6 Security Requirements 18
3.6.1 Access Requirements 18
3.6.2 Privacy Requirements 18
3.7 Legal Requirements 18
3.7.1 Standards Requirements 18

4 Project Issues **18**
4.1 Open Issues 18
4.2 Off-the-Shelf Solutions 18
4.3 Tasks 18
4.4 Risks 19
4.5 Costs 19
4.6 Waiting Room 19
4.7 Ideas for Solutions 20

List of Tables

1 **Revision History** 1
2 Work Partitioning 5

List of Figures

1 Context Diagram 4
2 UML Use Case Diagram 10

Table 1: **Revision History**

Date	**Version**	**Notes**
2021-02-11	1.0	Revision 0 of the SRS.
2021-04-11	2.0	Addressed TA's remarks by changing the document to better match the template given. This included the addition of many sections.
2021-04-11	2.1	Made changes to use cases and requirements to match the system, as requirements were altered during the course of the development process.

Important Information

Text that has been struck through, ~~like this~~, was in the previous revision of the document and has been removed.

Text that is colored red, like this, is new to revision 1, and reflects the most recent information.

Text that is colored green, like this, is a comment to the TA's/professor and should not be included as a part of the content in this document.

The template was not correctly used for revision 0 of this document. Revision 1 has updated the template to better match the given template. Section and subsection headers that are new to revision 1 are not in red, only the new content of the sections.

Many functional requirements and use cases were altered or removed. Deleted requirements are struck through rather than removed, so they maintain a requirement name and number even though they are not technically requirements. For example, requirement FR5 no longer exists but there is no new FR5 in order to reduce confusion in traceability.

1 Project Drivers

1.1 The Purpose of the Project

The purpose of this document is to specify the system requirements and how the system interacts with the end user.

1.2 The Stakeholders

Intended stakeholders for this document include the development team, any end-users, the professor of the course, and the TAs of the course.

1.2.1 The Client

The clients for this project are the professor and TA's of the course. This is because they have tasked the development team with the creation of the software, and their input is used as direction throughout the development life-cycle.

1.2.2 The Customers

The customers for this project are any users who wish to play computer chess and chat online.

1.2.3 Other Stakeholders

Other stakeholders for this project include the development team, who are invested in the success and development of the project.

1.3 Mandated Constraints

a) Time constraint: The final product must be fully implemented by the end of the semester.

b) Resource constraint: The developers will have limited access to additional resources such as more developers, money, and software management tools.

1.4 Naming Conventions and Terminology

a) The Software Requirements Specification (SRS) is a detailed document which describes a software system to be developed.

b) The Web Content Accessibility Guidelines (WCAG) 1 are guidelines on making web content more accessible to people with disabilities.

c) ~~Artificial Intelligence (AI) in this document refers to a computer-controlled player.~~

1.5 Relevant Facts and Assumptions

a) It is assumed that the user has access to the required hardware and internet connection required to access and play the game without the hindrance of lag or other issues.

b) It is assumed that the user has access to I/O devices, such as a keyboard/mouse/monitor for a computer, or a touchscreen for a mobile device.

c) Based on the browser used for the web application, the **HTML** and **CSS** might be affected. This project will use the assumption that a user will be using the latest version of Google Chrome.

d) It is assumed that the user has proficiency in the English language.

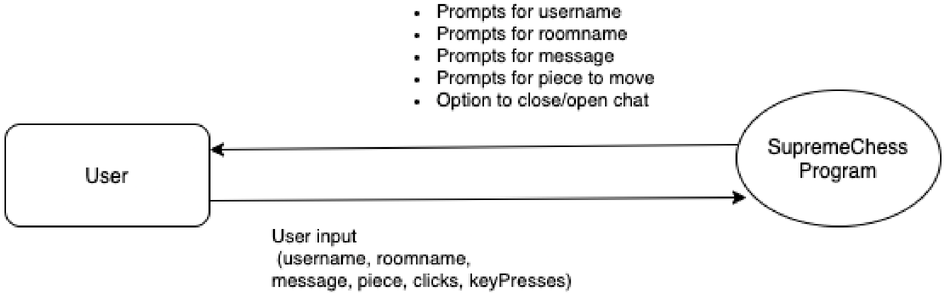
2 Functional Requirements

2.1 The Scope of the Work and the Product

The software product will provide the user with an implementation of a chess game. The product aims to provide an interactive and entertaining game-play experience utilizing web programming. The user will be able to play with other users online.

2.1.1 The Context of the Work

2.1.2 Work Partitioning



Prompts for username
Prompts for roomname
Prompts for message
Prompts for piece to move
Option to close/open chat
User
SupremeChess Program
User input (username, roomname, message, piece, clicks, keyPresses)

Figure 1: Context Diagram

Event Name	**Input/Output**	**Summary**
User enters username	Username(IN)	The user inputs a username to identify them and the program stores a record of this
User enters roomname	Room Name(IN)	The user inputs a roomname to have the chat and the program records this
User joins a room	• Username(IN) • Room Name(IN) • Join Button(IN) • Room(OUT)	The users input information (username and roomname) and an id (for their socket) is sent to the back-end to be recorded. The user is presented with their room
User sends message	• Message(IN) • Enter Key(IN) • Message(OUT)	The user enters a message to be sent and either presses the “Enter” key or clicks the Send button, this sends a record to the back-end with the message and user information pertaining to the message

Event Name	**Input/Output**	**Summary**
User decides to Close/Open Chat	- Close/Open Button(IN) - Chat(OUT)	The user clicks on the Close/Open Chat Button to either close (when the chat is open) or open (when the chat is closed) the chat. Upon clicking Close Chat the chat disappears, upon clicking Open Chat the chat appears, without displaying previous messages.
User moves a piece	- Drag and Drop Piece(IN) - Board(OUT)	The user drags and drops a piece on the board, if the move is valid the board is updated and redrawn on the screen.
User offers a draw	- Offer Draw Button(IN) - Accept Draw Button (IN/OUT) - Reject Draw Button (IN/OUT)	The user clicks on the Offer Draw button. This presents two buttons to their opponent (Accept and Reject Draw buttons), if the opponent clicks the Accept Draw button the game ends in a draw, if the opponent clicks the Reject Draw button the game continues.
User resigns	- Resign Button(IN)	The user clicks on the Resign button, which causes the current user to lose the game and the game ends.

Table 2: Work Partitioning

2.1.3 Individual Product Use Cases

UC1. **~~Select Opponent~~ Join Chat Room**

Initiating Actor: User

Actor's Goal: ~~To choose between playing against an AI or a human player.~~ To join a chat room to chat with other users in the same room.

Participating Actors: System

Pre-conditions ~~The system displays the menu of available options. The choices are "AI" and "Online Player".~~ The system prompts the user to input a username and room code.

Post-conditions The "Start Game" use case is initiated.

Flow of Events of Main Success Scenario:

i. ~~Host player selects one of two player options, "AI" or "Online Player".~~
ii. The application is initialized based on the choice made by the host. The game now starts.

UC2. **Start Game**

Initiating Actor: User

Actor's Goal: To start playing a game against the selected opponent.

Participating Actors: System

Pre-conditions ~~An "AI" or "Online Player" was selected as the opponent.~~ An opponent for the game has been chosen.

Post-conditions The board is setup and one or two players are connected and able to play the game.

Flow of Events of Main Success Scenario:

i. ~~Either the "AI" or "Online Player" opponent option is selected.~~
ii. The server connects the two players ~~together or connects one player to the AI~~ to a chat room.
iii. The board is initialized to the starting state and piece colors (black or white) are arbitrarily selected between the players.
iv. Control is passed to the user with the white pieces and the timers are initialized.

UC3. **Make Move**

Initiating Actor: User

Actor's Goal: To make a move in the game against the opponent.

Participating Actors: System

Pre-conditions The game has been started and the opponent has made their move.

Post-conditions The game may be terminated, or the opponent is now able to make their move.

Flow of Events of Main Success Scenario:

i. The user makes a legal move.
ii. The application registers that the user has made a move and then determines if the game must continue.
iii. If the game has not ended, the opponent is now able to make their move.

Flow of Events for Extensions (Alternate Scenarios):

i. The user may make a legal move.
ii. The application will register that the user has made a move, where it must determine if the game must continue.
iii. If the game has not ended, the opponent is now able to make their move.

UC4. **Check Legal Move**

Initiating Actor: User

Actor's Goal: To determine if the move made is valid.

Participating Actors: System

Pre-conditions

i. The game has started.
ii. The user has made a move.

Post-conditions The game may be terminated, or the opponent is now able to make their move.

Flow of Events of Main Success Scenario:

i. The user makes a move.
ii. The application will register that the user has made a move, where it must determine if the game must continue.
iii. The application will check the game logic to determine if the move is legal.
iv. If the game has not ended, the opponent is now able to make their move.

UC5. **Initiate Chat**

Initiating Actor: User

Actor's Goal: To initiate communication between players while concurrently playing a game.

Participating Actors: System

Pre-conditions:

i. The player has started a game against another player.

Post-conditions:

i. The player has communicated with the other player.
ii. The game-play is continued asynchronously.

Flow of Events of Main Success Scenario:

i. A game has started between two players.
ii. One of the players selects the chat function and types a message.
iii. The message is passed to the other player and made visible.
iv. The opponent is able to asynchronously send messages and continue game-play.

UC6. **End Chat**

Initiating Actor: User

Actor's Goal: To stop communicating with one's live opponent.

Participating Actors: User

Pre-conditions:

i. The player has selected an opponent.
ii. The player has started a game.

Post-conditions:

i. The player has made contact with the other player.
ii. The game-play is continued asynchronously.

Flow of Events of Main Success Scenario:

i. A player initiates chat with their opponent.
ii. One of the players can at anytime choose to end the chat session.
iii. One player ending the chat closes the chat for both players.

UC7. **Start Timers**

Initiating Actor: System

Actor's Goal: To start the timer for each player that represents how long each player has to make their moves in the game.

Participating Actors: User

Pre-conditions The game has been started and the player with the white pieces has made a move.

Post-conditions The timers have begun to countdown from their original values.

Flow of Events of Main Success Scenario:

i. The user with the white pieces makes a legal move.
ii. The timer for the user with the black pieces begins to count down from its initial value.
iii. The user with the black pieces makes a legal move, or that player's timer reaches 0.
iv. If a legal move is made, that timer is paused and the other timer begins to count down from its initial value.
v. If a timer reaches 0, the "Terminate Game" use case occurs and is overridden by "Timeout".

UC8. **Terminate Game**

Initiating Actor: Any of: User, or System

Actor's Goal: To determine if the game must be ended.

Participating Actors: None

Pre-conditions The game has been started.

Post-conditions The game has ended with a definite result.

Flow of Events of Main Success Scenario:

i. The system must start the game.
ii. The application determines if a user has offered a decision to end the game in a resignation or draw. The other user can choose to accept or reject this offer.
iii. The application determines if the user has won or lost the game through checkmate, stalemate or timeout. This signifies that one of the players can no longer make any moves.
iv. The game will end in a definite result if any of the previous events are true.

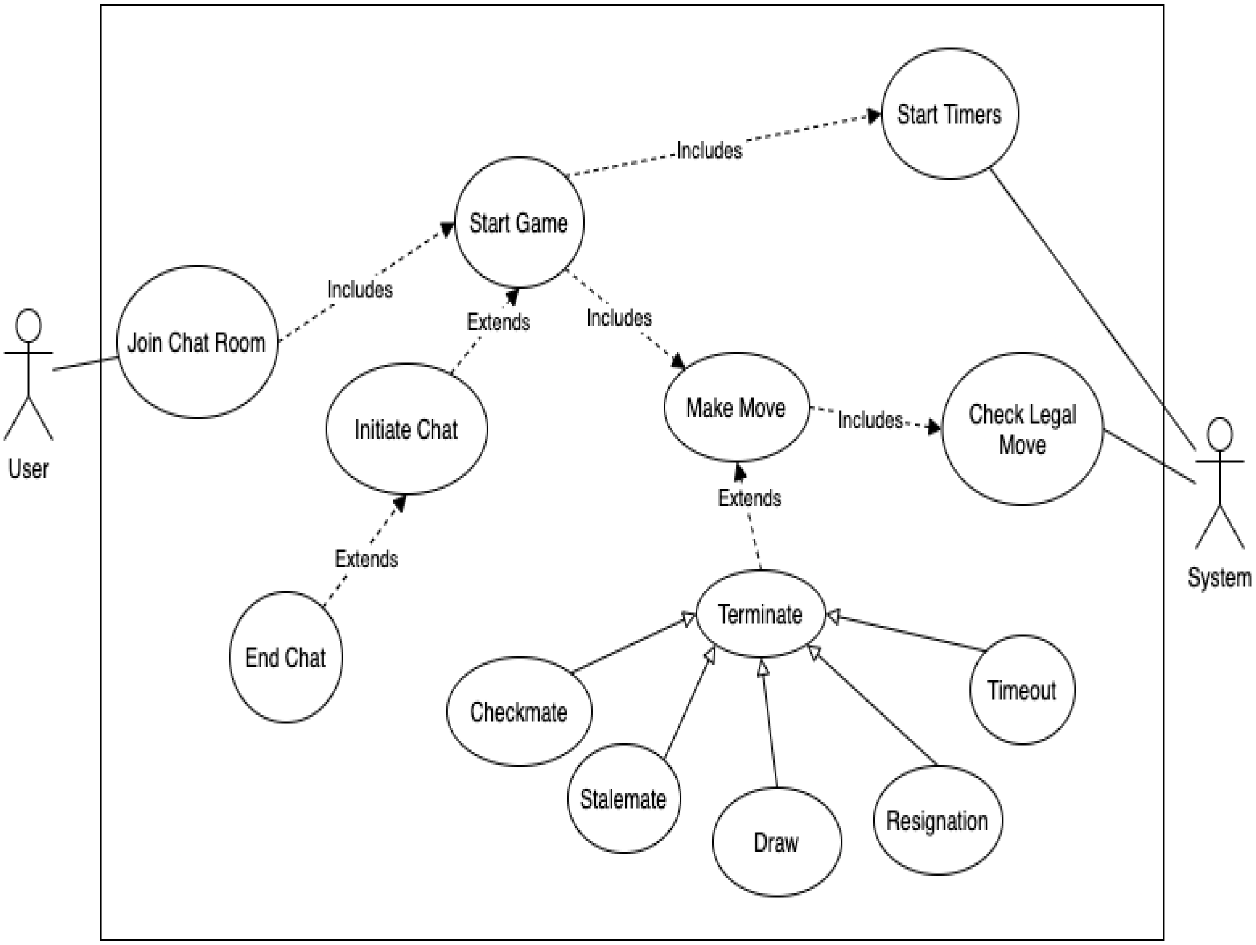


Figure 2: UML Use Case Diagram

2.2 Functional Requirements

This section of the SRS contains all of the functional software requirements. The functional requirements detailed here enable designers to design a system that satisfies these requirements, and enable testers to test that the system satisfies these requirements.

UC1. ~~**Select Opponent**~~ **Join Chat Room**

FR1. The user shall connect to the web application server.

FR2. ~~The user shall be immediately presented with the option of playing with an “AI” or “Online Player” in the middle of their screen. They shall use~~

~~the labeled buttons to choose between the two options in order to proceed.~~ The user shall be presented with a welcome screen.

FR3. The system shall present the user with options to input a username and room name as well as an option to join the specified chat room.

FR4. If the user attempts to join a chat room before inputting both a username and room name then the system shall prompt them to enter a username and room name. If the user attempts to join a chat room after inputting a username and room name, then the system shall send the user input data to the server and connect the user to the specified chat room. If a chat room with the specified name does not exist, it shall be created.

UC2. ~~**Play Against Live Opponent**~~

FR5. ~~If a user decides to "Play Against a Live Opponent" the system shall pair them against a user with the same invite link.~~

FR6. ~~While the system is awaiting another player, the system shall be present the user with the options to "Quit", or "Play Against AI".~~

FR7. ~~If a user decides to "Quit" or "Play Against AI" the current game shall be terminated.~~

FR8. ~~If a user selects the "Quit" option, the system shall return them to the "Select Opponent" use case.~~

FR9. ~~If a user selects the "Play Against AI" option, the system shall send them to the "Play Against AI" use case.~~

FR10. ~~The system shall present the players with a loading screen while the application is initializing the game. The loading screen shall last no longer than 5 seconds.~~

FR11. ~~Once initialized, the system shall present the players with a view of the board and transfer to the "Start Game" use case.~~

FR12. ~~If a user requests to start a game on any server with 15 concurrent games, the system shall reject the request and return to the "Select Opponent" use case.~~

UC3. ~~**Play Against AI**~~

FR13. ~~If a user selects "Play Against AI" the system shall immediately transfer to the "Start Game" use case.~~

FR14. ~~While playing against the AI the system shall have no option to initiate chat.~~

FR15. ~~If a user requests to start a game on any server with 15 concurrent games, the system shall reject the request and return the user to the "Select Opponent" use case.~~

UC4. **Start Game**

FR16. ~~If the user chooses "AI" from the "Select Opponent" use case, the server shall connect to an AI player and arbitrarily assign black or white to each player.~~

FR17. ~~If the user chooses "Online Player" from the "Select Opponent" use case, a shareable link shall be created with which a player can join the server. If a player joins, each player shall be arbitrarily assigned black or white.~~

FR18. The board and pieces shall be initialized, and the orientation of the board will be determined based on the colour of the users pieces.

FR19. The timer shall be initialized with a countdown of 10 minutes for each player, and the game control is dependent on the player with the white pieces.

FR20. The system shall give exclusive control to the player with the white pieces and start that player's timer.

UC5. **Make Move**

FR21. The system shall not allow the player to make a move until their opponent has played a move, with the exception of the first move of the game.

FR22. If a player clicks a piece, holds the mouse down, drags the piece to a square, and releases the mouse, the system shall register an attempted move.

FR23. The application shall confirm that the move is legal as defined by the rules of the game.

FR24. If the attempted move is legal, the move shall be sent to the server and the opposite player shall be given control.

FR25. If the attempted move is not legal, the move shall not be sent to the server and the player shall retain control.

FR26. The 'Terminate Game' use case shall be checked after every move is made.

FR27. If the game is not terminated, the timer for the other player shall resume.

UC6. **Check Legal Move**

FR28. The application shall register the move the user has made, and check the program logic to determine if the move can be made.

FR29. The move is then reflected in the board state and the opponent may now make their move.

FR30. Only one player will be allowed to make a move at once.

UC7. **Initiate Chat**

FR31. The system shall present the user with an option to “Start Chat” to start a chat.

FR32. If “Start Chat” is selected, the system shall open a chat window for both players.

FR33. If both players click on the “Start Chat” button simultaneously then the system shall only register one of the selections.

FR34. The “Start Chat” button shall be replaced by an “End Chat” button.

UC8. **End Chat**

FR35. The system shall present the user with an option to “End Chat” at any time once a chat has started.

FR36. If the chat is ended by one player the chat shall be disabled for both players. The “End Chat” button shall be replaced by the “Start Chat” button.

FR37. If both players click on the “End Chat” button simultaneously then the system shall only register one of the selections.

FR38. If the “End Chat” option is selected, the system shall clear all previous chat data.

UC9. **Start Timers**

FR39. Once a game has been confirmed a timer shall be started by the system.

FR40. The timer shall be set for a total of 10 minutes per player.

FR41. The timer shall begin for a player once it is their turn to move and will pause on their opponents turn.

UC10. **Terminate Game**

FR42. The system will have checks to establish if the game shall be terminated with the appropriate use case possibilities. These include the Checkmate, Stalemate, Draw, Resignation, and Timeout use cases.

UC11. **Checkmate**

FR43. The system shall determine if the opposing player is in ‘Check‘ 2 (state where the King is under attack) and has no legal moves for the King. This shall result in a victory with ‘Checkmate‘ for the current player.

UC12. **Stalemate**

FR44. The system shall determine if the opponent player is not in ‘Check‘ and has also no legal moves. This shall result in a draw with ‘Stalemate‘ for both players.

UC13. **Draw**

FR45. The system shall present the user with the ‘Offer Draw‘ button to offer a draw to the opponent. The draw must be accepted to end the game.

UC14. **Resignation**

FR46. The system shall present the user with a decision to end the game with the ‘Resign‘ button. This shall immediately end the game.

UC15. **Timeout**

FR47. The system shall determine if the user is out of time. This causes that player to lose the game.

3 Non-functional Requirements

3.1 Look and Feel Requirements

3.1.1 Appearance Requirements

LF1. The system shall use white, ~~black, and brown~~ light blue, light green, and dark grey as its primary colours.

LF2. The system shall use a minimalist design.

3.1.2 Style Requirements

LF3. The game shall have a calm and intuitive feel as users will be concentrated on making moves under a time constraint, making distractions undesirable.

LF4. ~~The game shall include sound-effects to inform that a move has been made.~~

LF5. The game shall not include background music.

3.2 Usability and Humanity Requirements

3.2.1 Ease of Use Requirements

UH1. The game shall not require any intricate keyboard presses or fast mouse movements.

UH2. The product shall be easy to use by 10 year old children. Ninety percent of a sample test panel of 10 year old children shall be able to successfully play a move within 5 minutes of starting a game.

3.2.2 Personalization and Internationalization Requirements

UH3. The game shall only display information in English.

3.2.3 Learning Requirements

UH4. The product shall be able to be used by members of the public over 10 years of age with no previous training. Ninety percent of a sample test panel of the public over 10 years of age shall be able to start a game and move a piece within 5 minutes of accessing the product.

UH5. ~~Instructions shall be in place to teach the user the rules of the game. Eighty percent of a sample test panel of the public over 12 years of age shall understand how to move each piece within 1 hour of accessing the product.~~

3.2.4 Understandability and Politeness Requirements

UH6. All symbols and words shall be naturally understandable by the user community and shall be similar to historically used Chess symbols 3.

3.2.5 Accessibility Requirements

UH7. System shall follow guidelines for correct colour contrast ratio for text to the background as stated in the WCAG 1.

3.3 Performance Requirements

3.3.1 Speed and Latency Requirements

PR1. The average time between player input and visual game response shall be less than 0.25 seconds.

PR2. The maximum time between player input and visual game response shall be less than 2 seconds.

PR3. The maximum time between a player input and that player's timer pausing shall be less than 0.1 seconds.

3.3.2 Safety-Critical Requirements

N/A. The system shall have no safety-critical components or hazards.

3.3.3 Precision or Accuracy Requirements

PR4. All timer values shall be accurate to ~~one decimal place~~ the nearest integer.

PR5. All material score values will be accurate to the nearest integer.

3.3.4 Reliability and Availability Requirements

PR6. The product shall be available for use 24 hours per day, 360 days per year on average.

3.3.5 Robustness or Fault-Tolerance Requirements

PR7. The product shall reconnect the player to the ~~game they were playing~~ chat room they were connected to if they lose and regain connection to the server.

3.3.6 Capacity Requirements

PR8. The product shall require a minimum amount of 2GB of RAM to function effectively.

3.4 Operational and Environmental Requirements

3.4.1 Installability Requirements

N/A. The system shall require no installation for access and use.

3.4.2 Requirements for Interfacing with Adjacent Systems

OE1. The application shall interface with an external server to make requests to the back-end of the application.

3.4.3 Productization Requirements

OE2. The product shall be deployed to a public website where users shall have access to it.

3.4.4 Release Requirements

OE3. The product shall be tested for possible bugs and issues, and will accordingly be fixed and redeployed to the product's website on a case-by-case basis.

3.5 Maintainability and Support Requirements

3.5.1 Maintenance Requirements

MS1. The product shall be maintained actively until the end of the semester.

3.5.2 Supportability Requirements

MS2. ~~The product shall come with an accessible set of instructions on the web-application and in the repository for rules of the game.~~

MS3. ~~The product shall provide users with a way of contacting the developers in order to report any bugs or issues they encounter.~~

3.5.3 Adaptability Requirements

MS4. The program shall be accessible from any internet browser on a computer or mobile device.

MS5. The product shall be deployed onto an external platform.

3.6 Security Requirements

3.6.1 Access Requirements

SR1. Any user shall be able to access and play the game.

SR2. The product repository shall only be accessible by the developers, TAs, and the professor.

SR3. Only the developers of the project shall be able to modify the files, domain, and the database of the product.

3.6.2 Privacy Requirements

SR4. There shall be no data collected on users outside of the moves they input.

3.7 Legal Requirements

3.7.1 Standards Requirements

LR1. The product shall be developed using the Airbnb JavaScript Style Guide 4.

4 Project Issues

4.1 Open Issues

The project uses a new version of React, which does not have full support from all testing dependencies, so testing React components is difficult.

4.2 Off-the-Shelf Solutions

There are other available products that provide similar and greater functionality to the application. Examples of this are websites like Chess.com and Lichess.

4.3 Tasks

The tasks for this project are set by the deliverable schedule in the course outline for Software Engineering 3XA3. These tasks include: The problem statement, development plan, requirements document, proof of concept demonstration, test plan, design documents, revision 0 demonstration, final demonstration, and test report.

4.4 Risks

There are two main risks that are likely to occur in the development of this project. The first and biggest risk is that intended features of the product may not be completed within the time-frame for the project. This could cause the design of the system to be worse than if the product was designed without those features in mind. This risk is likely to occur, specifically with the AI and multiplayer features of the product.

Another risk is that certain capacities will be measured inaccurately, such as server capacity. This will likely be due to a lack of resources and a lack of knowledge around best practices.

~~A lack of testing in the project might lead to more errors and unexpected results. It is important to discover as many bugs as possible before the final demonstration, which causes the software to be more reliable and easy to use. There is a risk that there may be uncaught errors and bugs in the program that could negatively affect the user experience.~~

4.5 Costs

The main cost for the system is the time of the development team and other stakeholders. However, there are other costs that could be incurred, such as having to purchase licensing for any software libraries used or purchasing a deployment server for the system.

4.6 Waiting Room

Multiplayer functionality that allows two users in the same chat room to start a game together or for a player to join a queue of online players.

An option to join a random chat room that has at least one other user already connected.

An option to play against an AI instead of another player.

4.7 Ideas for Solutions

Multiplayer functionality can be added similarly to the chat feature. The system would store individual board states for each player as well as either black or white, and each move would be passed between players. Whenever a move was made the board state would be updated accordingly.

An option to join a populated chat room could be added by storing a list of all chat rooms in use and then returning a random one when called, allowing the user to join that room.

An AI would have to be implemented, and options to play against the AI instead of another player would have to be added to the main screen.

References

1. Web content accessibility guide. https://www.w3.org/WAI/standards-guidelines/wcag/. Accessed: 2021-02-10.

2. Rules of chess. https://en.wikipedia.org/wiki/Rules_of_chess. Accessed: 2021-02-10.

3. History of chess sets and symbols. https://www.chesscentral.com/pages/chess-sets-pieces-boards/a-history-of-chess-pieces-and-chess-sets.html. Accessed: 2021-02-10.

4. Airbnb javascript style guide. https://airbnb.io/javascript/react/. Accessed: 2021-02-10.