Software Requirements Specification

for

ChessCloud - Chess Archive and Analytics

Release 1.0

Prepared by:
Isabella Robert Llorens (idr7)
Wiley Bozada (pwb26)

Revision History - Detail description in Inspection Report

Name	Date	Reason For Changes	Version
Isabella Robert	9/16/21	Initial Draft	1.0 draft
Isabella Robert	9/19/21	Added more detail to draft, expanded and developed feature list	1.0 draft
Isabella Robert	9/20/21	Edits to SRS, Diagrams Added	1.0 draft
Isabella Robert	9/21/20	Final revisions	1.0 draft

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) describes the software functional and nonfunctional requirements for the release 1.0 of the ChessCloud web application. The purpose of this SRS document is to provide a detailed overview of our software product, its parameters and goals. This document describes the project's target audience, user interface, and software requirements. The development team will use this SRS to implement and verify the correct functioning of the system.

1.2 Project Scope and Product Features

ChessCloud will allow for local and CPU play of chess games. These games are stored as user data to provide chess game archiving with a variety of analytics and comparisons. A detailed project description is available in the Vision and Scope section [2]. The section in that document titled “Scope of Initial and Subsequent Releases” lists the features that are scheduled for full or partial implementation in this release.

2. Vision and Scope

2.1 Business Requirements

2.1.1 Background, Business Opportunity, and Customer Needs

Chess players grow their skill and awareness by careful comparison of their previous games and strategies. Avid chess players would benefit from the ability to store, load and organize historic data to compare their chess strategies (a lengthy process when done manually). Furthermore chess players could benefit from the insight of chess engines whether as an opponent or an assistant. To provide the most high-level use of these engines, chess players would want to be able to load their new or historic game to any specific position and configure the CPU opponent both off and on. Chess players would also benefit from purely local play where they can play hypothetical moves for their opponent and analyze them.

Many chess players have voiced the desire for a easily usable web application which delivers these features. Chess.com, a major chess web application, has attempted to deliver some of the features above, but falls short of the customizability and accessibility desired. Chess.com’s archives are not built in a way that allows user organization into folders. Many chess players will improve through their use of chess analytics and want to archive the games they played as a novice. Chess.com does not permit the external importing of chess games, which is a key feature to user accessibility. Features like chess engine selection and the toggle off and on of the CPU opponent are important in game features to derive deeper analytics and are not currently supported on Chess.com. Finally the developers of ChessCloud believe chess analytics should be free to use, unlike similar features in Chess.com.

2.1.2 Business Objectives and Success Criteria

BO-1 Have 30 users sign up for accounts within the first three months with 30% of the users being active

SC-1 Have 60% of registered users store at least one personal chess game within a month of signing up

SC-2 Have 40% of registered users compare their strategy using the chess engines

2.1.3 Business Risks

RI-1 Not enough users might sign up for the system. Since the project replaces an activity often done manually by chess players, there may be a resistance towards using new tools that differ from practiced techniques.

RI-2 The application becomes more valuable and provides more insight when more chess games are stored in the user's archives. New users may not import data and fail to see the value in the long term use of the application.

2.2 Vision of the Solution

2.2.1 Vision Statement

ChessCloud will deliver data driven analytics tools to chess players via a web application. This piece of software will help organize chess players' historic data and improve their skill through insightful analysis. ChessCloud will collect user data either by reading saved PGN files, or allowing a user to play the game on the built-in chess board either locally or against a CPU. Users will be able to save their games to their associated account and receive comparisons between their live and historic games. Stored games will be easily organized in a folder system-based archive. Users will also be able to load historic matches to any board point to be played differently or played against a CPU. The CPU and chess engine assistant will allow the user to decide which engine they play with based on their preferences and needs. Major Chess web applications like Chess.com do provide analytics for users, but there are no chess aides that provide the accessibility and customization of ChessCloud.

2.2.2 Major Features

FE-1 Play local chess games
FE-2 Toggle play against a CPU in chess games
FE-3 Chess engine selection
FE-4 Chess Assistant: Similar games lookup during user gameplay
FE-5 Chess Assistant: Engine recommendations of optimal moves
FE-6 Import game to registered user file system
FE-7 Filesystem: Organize games in an easily navigated file system
FE-8 Filesystem: Load games
FE-9 Log in to existing account
FE-10 Sign-up for a new account

2.2.3 Assumptions and Dependencies

AS-1 Users will have access to a browser and the Internet

AS-2 Users will understand the basics of PGN notation
AS-3 Users will understand the rules and basic mechanics of chess
DE-1 Users will register for accounts for long term data tracking

2.3 Scope and Limitations

2.3.1 Scope of Initial and Subsequent Releases

Features	Release 1	Release 2	Release 3
FE-1	Fully implemented		
FE-2	Not Implemented	Not Implemented	Fully implemented
FE-3	Begin to write queries (if time permits)	Fully implemented	
FE-4	Not Implemented	Not Implemented	Fully implemented
FE-5	Not Implemented	Not Implemented	Fully implemented
FE-6	Fully implemented		
FE-7	Fully implemented		
FE-8	Fully implemented		
FE-9	Fully implemented		
FE-10	Fully implemented		

2.3.2 Limitations and Exclusions

LI-1 Without current ownership of a server or a contract with a cloud computing provider we are unable to specify how much web traffic ChessCloud can currently handle.
LI-2 This application is unable to host online chess matches.
LI-3 We will not confirm emails upon sign up, therefore accounts with an invalid email will not be able to have their password recovered.

2.4 *Business Context*

2.4.1 Stakeholder Profiles

Stakeholder	**Major Value**	**Attitudes**	**Major Interests**	**Constraints**
Avid Chess	Database for current	strong	Computed	Internet usage

Players	PGN, highly customizable	commitment through release 3	analytics are much faster than by hand, easy to use archives	
Novice Chess Players	Free to use (new players are less likely to pay for analytics)	strong enthusiasm, but might not find immediate use	Detailed user documentation	Internet usage, chess knowledge

2.4.2 Project Priorities

Dimension	**Driver**	**Constraint**	**Degree of Freedom**
Schedule			Release 1 is planned to be available by 10/8/21, release 2 is planned to be available by 10/13/21, release 3 is planned to be available by 10/20/21
Features		All features scheduled for the first release of 1.0 must be completely operational	
Quality		Tests will be written for each of the features of which all of them must pass in order to verify functionality.	
Staff	The team is composed of two developers who will also be testers for the project. One of the developers is also a team leader.		
Cost			The expenses will come from the web hosting services and for the database.

3. Overall Description

3.1 Product Perspective

ChessCloud is a web application that archives user matches and provides analytics for them. The system is created to aid avid chess players in their study, as well as provide free chess analytics for new users. The system is expected to expand throughout multiple releases with the initial release providing basic functionality and each successive release having additional features.

3.2 User Classes and Characteristics

Guest User

A Guest User is any person who accesses the web application without signing on. Ideally, the guest user will have a strong understanding of chess and its gameplay. Guest Users would use the application to play local games, CPU games, and use the engines for analysis. The frequency which the Guest User will use the application depends on each individual and their desire to study chess. Guest User interaction with the app will be limited because they will not have the file system available until an account is made. The Guest User must be able to sign up for a full account while utilizing the app. The Guest User will be able to select from a variety of chess engines and toggle between CPU and local play. The guest user will be prompted to sign up in order to receive the features associated with an account.

Registered User

A Registered User is any person who has completed signing up for an account and subsequently signed into their account. Ideally Registered Users will have a populated archive of PGNs, a strong understanding of chess and its gameplay, and experience comparing chess matches. These traits will maximize the value Registered Users can gain from the application. Registered Users will interact with and traffic ChessCloud based on personal conditions and interests in chess analysis. Unlike Guest Users, Registered Users will have their own file system which stores and organizes PGN chess games. These file systems will link to the chess assistant to provide queries of similar games. Registered Users will gain more insight as they play and upload more games; therefore it can be assumed Registered users are more likely to utilize the app more than once and over a longer time period. Similar to Guest Users, Registered Users would use the application to play local games, CPU games, and use the engines for analysis. All games played locally and with the CPU will be saved into the file system. Registered Users will be able to load previous games to whichever point in the match where they can select from a variety of chess engines and toggle between CPU and local play.

3.3 Operating Environment

OE-1 The website should be able to operate on the latest versions of most web browsers.

OE-2 ChessCloud should allow user access from an internet connection without any restrictions.

3.4 Design and Implementation Constraints

CO-1 The system will use ASP.NET in order to create an interactive web front end application for the user.

CO-2 The system will use SQL as the query language for the API and will use MySQL as the database.

CO-3 The system will use C# (MVC) for the backend.

CO-4 The system will use the ASP.NET WebClient class in order to run the application.

CO-5 The system will use IntelliTest in order to create tests for the C# and the .NET framework.

3.5 User Documentation

UD-1 Help documentation will be available in the help section of the web application. The help documentation will cover all features and how to use them.

UD-2 Tooltip popups will be displayed for all core features that are not obvious to the game of chess. These can be disabled.

3.6 Assumptions and Dependencies

AS-1 Users will be able to receive all support through documentation and tooltips.

DE-1 The database will be able to query similar games more quickly than it takes the user to move a piece.

DE-2 Engines will be able to integrate in a streamlined and efficient manner.

4. System Features

The Feature Tree is shown in Appendix B: Figure 1.

4.1 Local Play

4.1.1 Description and Priority

Local play is the primary way users will be able to save their games, moves, and overall strategies. The user plays as both white and black pieces, able to go back and forth on moves throughout the whole game at any time. The game autosaves at predetermined timestamps to the recent games folder. At any point, the user can choose to switch from Local Play to CPU Play. At the end of the game, the game is automatically saved to recent games and the user will be able to choose to save it in the cloud filesystem. Priority: High

4.1.2 Stimulus/Response Sequences

Stimulus: user presses the Local Play button

Response: board is set up so that the user is able to move the corresponding color's pieces,

Stimulus: user presses the Move Undo button

Response: board is updated to reflect the state before the current piece was moved

Stimulus: user presses the Move Redo button

Response: board is updated to reflect the state after the current piece was moved

Stimulus: player finishes a game
Response: game is automatically saved to the cloud, player can also choose where it is saved in the filesystem

4.1.3 Functional Requirements

Play.SetBoard	Sets up board state, if the user loaded a game, it would set the board accordingly, otherwise it sets the initial board state
Play.Move.Input	User's move input
Play.Move.Input.Validate	Verifies that the input move is valid
Play.Move.White	Changes the state of the board to reflect the move
Play.Move.Black	Changes the state of the board to reflect the move
Play.Move.MoveUndo	Board is updated to reflect the state before the current piece was moved
Play.Move.MoveRedo	Board is updated to reflect the state after the current piece was moved
Play.Autosave	Game progress is autosaved at predetermined timestamps to the Recent Games folder
FileSys.Game.Create	Creates a new game file in Recents for the game to be autosaved and saved
FileSys.Game.Save.Recent	As a game ends, it is automatically saved to the Recent Games folder
FileSys.Game.Save	Prompted to choose where to save the game (PGN)
Play.Switch.CPU	Switches from Local Play to CPU Play from the current board state

4.2 CPU Play

4.2.1 Description and Priority

CPU play is another way users will be able to save their games, moves, and overall strategies. The user can choose whether to play black or white pieces, with the chess engine of their choice controlling the leftover color. The player is able to go back and forth on moves throughout the whole game at any time. The game autosaves at predetermined timestamps to the recent games folder. At any point, the user can choose to switch from CPU Play to Local Play. At the end of the game, the game is automatically saved to recent games and the user will be able to choose to save it in the cloud filesystem. Priority: Low

4.2.2 Stimulus/Response Sequences

Stimulus: user presses the CPU Play button
Response: user is prompted to pick a piece color, and a chess engine
Stimulus: user chooses a piece color, and a chess engine
Response: board is set up so that the user is able to move their chosen color's pieces
Stimulus: user presses the Move Undo button

Response: board is updated to reflect the state before the current piece was moved
Stimulus: user presses the Move Redo button
Response: board is updated to reflect the state after the current piece was moved
Stimulus: player finishes a game
Response: game is automatically saved to the cloud, player can also choose where it is saved in the filesystem

4.2.3 Functional Requirements

Play.SetBoard	Sets up board state, if the user loaded a game, it would set the board accordingly, otherwise it sets the initial board state
Play.Move.Input	User's move input
Play.Move.CPU	CPU's move input
Play.Move.Input.Validate	Verifies that the input move is valid
Play.Move.White	Changes the state of the board to reflect the move
Play.Move.Black	Changes the state of the board to reflect the move
Play.Move.MoveUndo	Board is updated to reflect the state before the current piece was moved
Play.Move.MoveRedo	Board is updated to reflect the state after the current piece was moved
Play.Autosave	Game progress is autosaved at predetermined timestamps to the Recent Games folder
FileSys.Game.Create	Creates a new game file in Recents for the game to be autosaved and saved
FileSys.Game.Save.Recent	As a game ends, it is automatically saved to the Recent Games folder
FileSys.Game.Save	Prompted to choose where to save the game (PGN)
Play.Switch.Local	Switches from CPU Play to Local Play from the current board state

4.3 Chess engine selection

4.3.1 Description and Priority

Before playing a CPU Play game, the user is prompted to choose from 3 different chess engines to play with. Priority: Medium

4.3.2 Stimulus/Response Sequences

Stimulus: user presses the CPU Play button
Response: User is prompted to pick from 3 different chess engines
Stimulus: user selects chess engine 1
Response: the CPU Play chess engine used is chess engine 1
Stimulus: user selects chess engine 2
Response: the CPU Play chess engine used is chess engine 2
Stimulus: user selects chess engine 3

Response: the CPU Play chess engine used is chess engine 3

4.3.3 Functional Requirements

Play.CPU.Engine1	Play CPU with Chess Engine 1
Play.CPU.Engine2	Play CPU with Chess Engine 2
Play.CPU.Engine3	Play CPU with Chess Engine 3

4.4 Chess Assistant: Similar games lookup during user gameplay

4.4.1 Description and Priority

During Local Play or CPU Play, the Chess Assistant Similar Games collapsible tab will show a list of similar games played in the user's saved games that change as the player continues playing moves. Priority: Low.

4.4.2 Stimulus/Response Sequences

Stimulus: user inputs moves in a game

Response: the Chess Assistant Similar Games collapsible tab will show a list of similar played games in the user's saved games

4.4.3 Functional Requirements

FileSys.SearchSimilar	Searches the saved games database for similar moves and game progressions
Assistant.Similar.isCollapsed	Determines if the Similar Games tab is collapsed
Assistant.Similar.Games	The similar games list

4.5 Chess Assistant: Engine recommendations of optimal moves

4.5.1 Description and Priority

During Local Play or CPU Play, the Chess Assistant Engine Recommendations suggest next moves and strategies in a collapsible tab. Priority: Low

4.5.2 Stimulus/Response Sequences

Stimulus: user inputs moves in a game

Response: the Chess Assistant Engine Recommendations collapsable tab will show the recommended moves and strategies

4.5.3 Functional Requirements

Engine.SearchSuggested	Searches suggested moves and strategies according to the state of the board

Assistant.Engine.isCollapsed	Determines if the Similar Games tab is collapsed
Assistant.Engine.Moves	The recommended moves and strategies

4.6 Import Game

4.6.1 Description and Priority

Users can upload a PGN, either through a text file or a test input box; the game is saved in the filesystem. The PGN will be verified as valid when it is uploaded. The imported game can then be loaded in via the filesystem. Priority: High

4.6.2 Stimulus/Response Sequences

Stimulus: user clicks the Import Game button

Response: a pop up appears prompting the user to upload a PGN text file or to input text in PGN format in a text box

Stimulus: user uploads a PGN text file

Response: PGN format is validated, if valid, the user is prompted to choose between saving the game to the filesystem. if invalid, an error message will appear

Stimulus: user enters a PGN through the input box

Response: PGN format is validated, if valid the game is saved to the filesystem. if invalid, an error message will appear

4.6.3 Functional Requirements

Import.PopUp	Popup that prompt the user to either upload a PGN text file or to input text in PGN format in a text box
Import.FileUpload Validate.PGN.File Import.TextInput Validate.PGN.Text Import.Store.PGN.File	File Upload of a plain text file Validates the contents of the plain text file as a valid PGN format Text input of the PGN format Validates the contents of the input text as a valid PGN format Process PGN to generate FEN and store new FEN for queries
FileSys.Game.Save	Saves the game in the filesystem

4.7 Organizing Filesystem - Loading Games

4.7.1 Description and Priority

Users can modify and personalize the filesystem by creating folders and games, and moving folders and saved games. Priority: High.

4.7.2 Stimulus/Response Sequences

Stimulus: User clicks create folder button
Response: User is prompted to select where the folder is created and input a unique name
Stimulus: User clicks create game button
Response: The load game pop-up appears, via which the Game is created and that the input name is unique
Stimulus: User clicks move folder button
Response: User is prompted to select where the folder will be moved to
Stimulus: User clicks move Game button
Response: User is prompted to select where the Game will be moved to
Stimulus: User clicks delete folder button
Response: Selected folder and all its contents are deleted from the filesystem
Stimulus: User clicks move Game button
Response: Selected Game is deleted from the filesystem

4.7.3 Functional Requirements

FileSys.Folder.Create	Creates a folder in the filesystem
FileSys.Game.Create	Calls the import pop-up
Validate.FolderName	When creating a folder, verifies the name is unique
Validate.GameName	When creating a game, verifies the name is unique
FileSys.Folder.Move	Moves the folder to another specified location in the filesystem
FileSys.Game.Move	Moves the game to another specified location in the filesystem
FileSys.Folder.Delete	Deletes a folder and all its contents from the filesystem
FileSys.Game.Delete	Deletes a game from the filesystem
FileSys.Game.Export	Creates plain text file with PGN for export
Redirect.Home	When the game is chosen for play, the page is redirected to the Home Page

4.8 Loading Games

4.8.1 Description and Priority

Users can load any game saved in the filesystem, where it is playable in Local Play. Priority: High.

4.8.2 Stimulus/Response Sequences

Stimulus: User loads a saved game from the filesystem
Response: Page redirects to the Home Page where a Local Play match will begin with the state of the board reflecting the saved game

4.8.3 Functional Requirements

Play.Game.Load FileSys.Game.Load	Loads in the game onto the board Retrieves the game from the database
Redirect.Home	When the game is chosen for play, the page is redirected to the Home Page

4.9 Log in to existing account

4.9.*1* *Description and Priority*

The user is prompted to input an email and password corresponding to their account. By submitting, the email and password will be verified to correspond with a user. If the input is valid, the user is logged in and redirected to the Home Page. If the user forgets their password, an email is sent with the link to recover the account and change the password. Priority: High.

4.9.2 *Stimulus/Response Sequences*

Stimulus: User clicks the Login button

Response: Email and Password are validated to a corresponding user, if valid, the user is logged in and the page is redirected to the Home Page, otherwise an error message appears

Stimulus: User clicks the Forgot Password button

Response: An email is sent with the link to recover the account and change the password

4.9.3 *Functional Requirements*

Validate.User.Email	Validates the email to make sure that it corresponds to an existing user
Validate.User.Password	Verifies that the password corresponds to the account the email corresponds to
Login.ForgotPassword	Sends the user an email to recover their account and change their password
Message.Error.Login	If either the email or password do not correspond to an existing user, shows an error message
User.isLoggedIn	States that the user is logged in for the session
Redirect.Home	Once the user is created, it redirects to the Home Page

4.10 Sign-up for a new account

4.10.1 *Description and Priority*

The user is prompted to input a name, valid and unique email address, an initial password, and a confirmation password. By submitting, the email and passwords will be verified. Passwords will need to have at least 8 characters, two uppercase letters, and a

number. If the input is valid, the account is made, and the user will be redirected to the home page. Priority: High.

4.10.2 Stimulus/Response Sequences

Stimulus: User clicks the Sign-up button

Response: Email and Password are validated, if valid, new user is made, logged in and the page is redirected to the Home Page, otherwise an error message appears

4.10.3 Functional Requirements

Validate.Email	Validates the email to make sure it is in line with the email format and that it is not already in use by another user
Validate.Password	Determines that both passwords are equal and comply with the security requirements
Message.Error.SignUp	If either the email or password are invalid, shows an error message
User.CreateNew	Creates a new user with the validated information
User.isLoggedIn	States that the user is logged in for the session
Redirect.Home	Once the user is created, it redirects to the Home Page

5. External Interface Requirements

5.1 User Interface

UI-1 Compatible with all internet browsers

UI-2 All pages will be centered around the home page; various pop ups will be displayed when needed. A diagram of the webpage relations is available in Appendix B: Figure 2

UI-3 A mockup for the home page can be found in Appendix B: Figure 3

5.2 Hardware Interface

HI-1 No hardware specification required

5.3 Software Interface

SI-1 Database API

SI-1.1 The live game board will query similar FEN positions and load the corresponding PGN files

SI-1.2 Import feature will process a PGN and correctly store it to the database

SI-1.3 Live games played will populate a data file which will be stored in the database

SI-2 Chess Engines

SI-2.1 When chess pieces are moved in local play the selected engines provide their suggested move

SI-2.2 When the user makes a move against a cpu the chess engine will provide the response move

SI-2.3 The specific chess engines for SI-2.1 and SI-2.2 will be configurable through a comprehensive list

5.4 Communications Interface

CI-1 Emails will be sent for account recovery in case of a forgotten password

CI-2 Users will be able to download saved games as plain text file containing the PGN of the game

6. Nonfunctional Requirements

6.1 Performance Requirements

PE-1 The application will accommodate at least 500user profiles

PE-2 Queries should be resolved within 15 seconds

PE-3 Similar games will be queried and shown in less than 10 seconds after a move is made

PE-4 All pages should load within 40 seconds

6.2 Safety Requirements

There are no safety requirements for this software.

6.3 Security Requirements

SE-1 All user account data will be encrypted

SE-2 Users must log in to save and access data

SE-3 Users can only access their own data

SE-4 Defense against SQL Injection in all text input boxes

SE-5 URL cloaking

6.4 Software Quality Attributes

Availability-1 The application should be available to users for 99% of the time

Reliability-1 The server should be up 99% of the time apart from scheduled maintenance

Robustness-1 If a Registered User’s session drops their current game will be saved

Usability-1 The application should be easy to navigate and well explained

Appendix A: Data Dictionary

.NET Framework	=	A coding framework from Microsoft to develop a variety of applications
ASP.NET	=	Extension of .NET to provide dynamic webpage development
Avid Chess Player	=	A chess player who has a level of chess mastery and deep knowledge of the game
Black	=	One of the two players in a chess game; players are named according to their piece color
Board	=	The setting of a chess match. In ChessCloud this board is used to save, play and analyze
C#	=	.NET, object oriented programming language
Chess Assistant	=	The panel which holds suggested moves and similar games queries
Chess Engine	=	A software that analyzes a board position and determines optimal moves
Chess.com	=	A popular chess web application
CPU	=	Stand for computer and is used to represent a computer controlled opponent
CPU Play	=	The games played against a CPU
FEN	=	Chess notation used to describe the current state of the board
File system	=	The organizer of saved user game data
Guest User	=	Any user who accesses the web application without signing on
Home	=	The default screen for the web application which contains the chess board, chess assistant and prompts to other screens
Import	=	The process of loading a PGN file into the database
IntelliTest	=	An automatic tester for .NET applications
Load games	=	The process of loading a game onto the live board

Local Play	=	A game with no CPU users that is played on one machine
Move	=	The action which defines a turn in chess; one piece is placed on a different space on the board
MVC	=	(Model, View, Controller) A software design pattern where the view displays the model and the controller modifies it
MySQL	=	Open source relational database manager
Novice Chess Player	=	A chess player who knows only the basics of the game
Optimal moves	=	The move determined by software to have the highest probability of long term success
PGN	=	Chess notation which stores each move and the order they occur in
Registered User	=	Any person who has completed signing up for an account and subsequently logged in
Similar Games	=	Games which share the FEN notation with the live game
SQL	=	Query language for relational databases
SQL Injection	=	A malicious action taken by users to produce a query which harms the database
Stored games	=	A user's saved games in the database; accessible through the file system
Tooltip Popups	=	Small helpful hints that appear when the mouse is over an action loaded part of the user interface
URL cloaking	=	Modifying the original URL to hide origins and provide better usability
White	=	The player other than black in a chess match

Appendix B: Diagrams

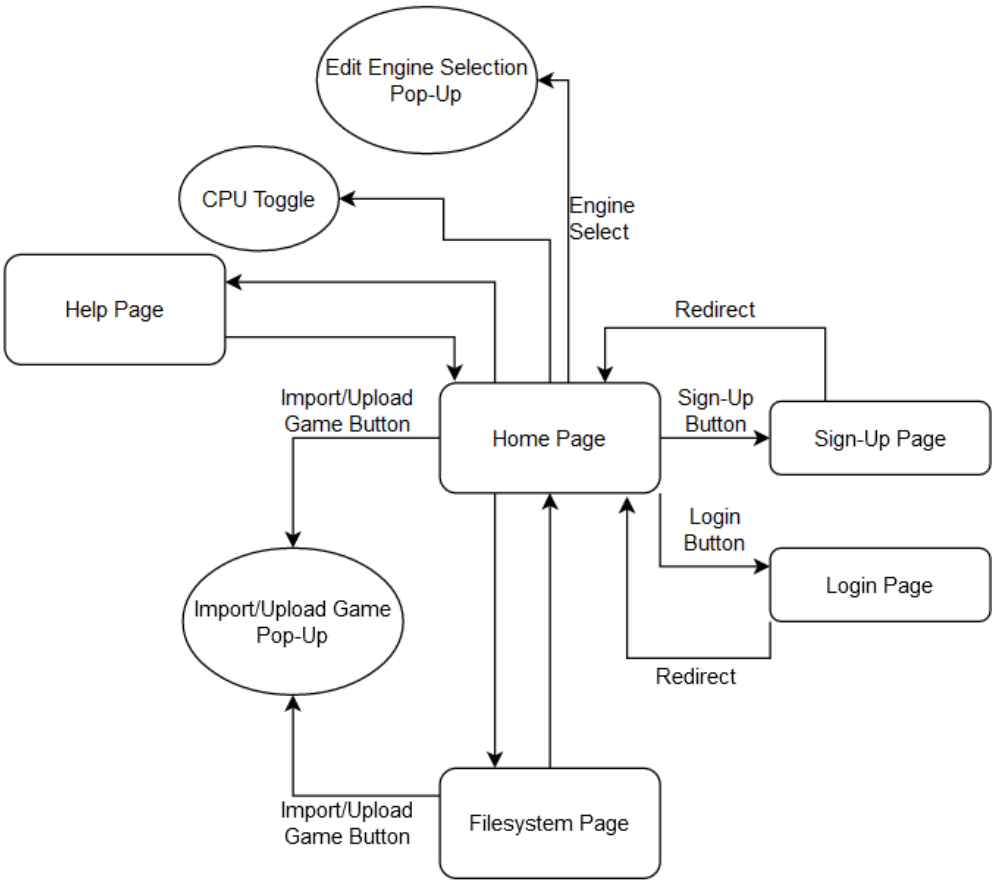

Figure 1.
Interface Diagram

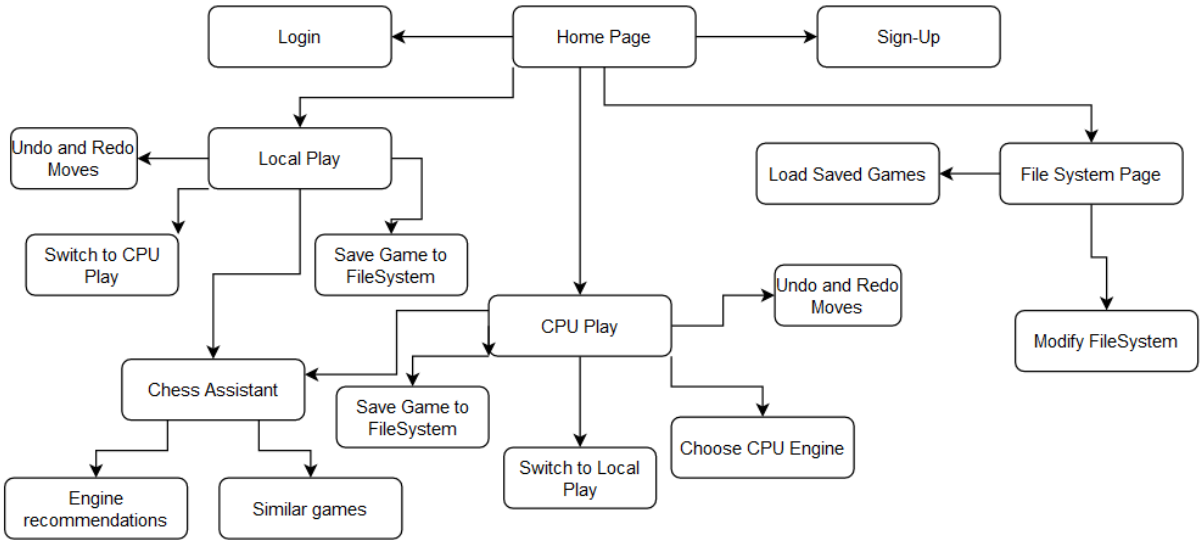

Figure 2.
Feature Tree

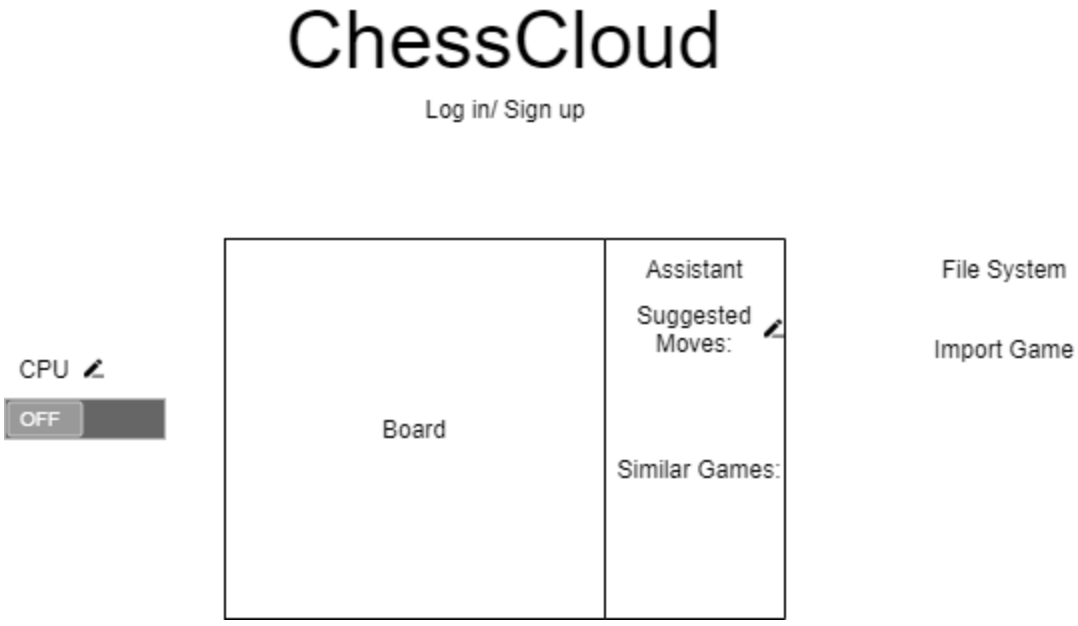

Figure 3.
Sample UI Design (All pens and unmarked texts are linked to pop-ups or auxiliary pages)

Inspection Log

9/16/20 **Initial Draft**

Initial Draft was written and discussed in a group meeting. Important details in the functioning of the project were agreed upon. The draft consisted of an outline of the whole document with minor descriptions and initial ideas.

9/19/20 **Expand on Draft**

Added more detail to the draft and elaborated more on paragraphs and ideas. Expanded and developed feature list completely**.**

9/20/20 **Edits to SRS, Diagrams Added**

Designed and added the Interface Diagram and the Feature Tree. Made overall changes and improvements on the SRS.

9/21/20 **Final Revision**

Added the Table of Contents and Appendix A. Inspect the whole document for any mistakes or any areas to improve upon.