Software Requirements Specification

Chess of Champions
Version 1.0

Created by:
Jeremy Bennett
Austin Herring
Akshay Sharma
Josh Weinstein

Revision History

Name	Date	Reason for Change	Version
Jeremy Bennett, Austin Herring, Akshay Sharma, Josh Weinstein	January 21, 2015	Initial Version	1.0

Table of Contents

Table of Contents

1. Introduction

This section of the document will serve as an introduction to the outlined program as well as to the document itself and its purpose.

1.1 Purpose

This document will collect and enumerate, in detail, the specific requirements for the chess playing program *Chess of Champions*, from a user perspective. That is, it will explain all of the requirements for user interaction and functionality for the final program. Its audience is divided; on one hand, the potential clients can use it to determine what the delivered program should look like and, therefore, if they would like to use it, and on the other hand, the developers and testers can use it as a reference to determine what it is their work should be actualizing and how well it succeeds in doing so.

1.2 Scope

The project to be completed under these specifications, to be named *Chess of Champions*, will be for playing player-vs-player matches of the classic game chess over the internet. Users will start the application and then be able to choose to play a game another user has created or create their own, with their own rule specifications. They will then make moves as in classical chess, until one of the players either wins or resigns, with the ability to chat in-app while the game progresses. Ultimately, the program will provide a quick and efficient, completely self-contained environment for two players to play fully complete games of chess with another, just as if they were sitting next to each other using a physical chess board.

1.3 Overview

The remainder of this document will go into much greater detail on the specific requirements of the described *Chess of Champions* program. First, it will give an overall description of how it is expected users will interface with the program, as well as describe what those users generally will look like. Then, it will enumerate, exactly, the specific requirements for the program, in a highly rigorous, highly detailed way and will also give mock-ups of actual user interfaces. After that, it will describe the use cases of the program and the general flows that the users will follow as they interact with it and play their games of chess,thus reaching the conclusion of what the currently discussed iteration of the product should accomplish. After detailing that, however, a section on system evolution will describe how the program may be altered in the future to enhance it and add features.

2. Description

2.1 Product Perspective

Chess of Champions will be a program which will allow two players to play games of chess with one another over the internet, providing, as much as possible, a self-contained way of doing so as if the two players were right next to one another, playing on a physical board.

2.1.1 Program User Interface

The program will present the user with everything he needs to play the game of chess over the internet. Upon opening the application, the user will be presented with the main screen, and he can, from there, decide to create a new game or load a previously saved game or input a previously created game's numeric ID to join that game. Then, he will be presented with a chess board, containing each player's pieces, and several smaller frames on the screen. Upon his turn, the player will be able to move his pieces around on the board to determine his move, and upon his opponent making a move, he will be able to watch it be carried out. In the peripheral frames, the user will be able to send and receive chat messages; view all of the previously made moves in the game; resign the game; and view the captured pieces and the remaining time for himself and his opponent to make their moves. The game board and all of these peripheral windows will allow the user to interact with the program to play a fully-fledged game of chess in a nearly isolated environment.

2.2 Product Functions

Chess for Champions will provide the following features:

- Creation or joining of chess game sessions to be played over the internet
- Ability to make all possible valid chess moves and to prevent all invalid ones
- Ability to resign from an ongoing game
- Game timers, as in professional chess, the length of which will be specified by the creator of the game
- Automatic determination of end-of-game situations such as checkmate and stalemate
- Text chat with an opponent
- Saving of in progress games and loading of previously saved games

2.3 User Description

The primary user of *Chess of Champions* is someone who desires to play Chess with a friend remotely. The ideal user knows all the rules of a standard game of chess as outlined by the *World Chess Organization* (WCO). The user should also have a way of contacting their friend beforehand in order to receive their unique ID to join their session of Chess. The program should, theoretically, support both novice and expert chess players.

2.4 Assumptions, Dependencies and Constraints

2.4.1 Assumptions

There will be free-to-use images of chess pieces available. If there turns out not to be, will will produce these on our own.

2.4.2 Dependencies

The user will need the latest version of the Java Runtime Environment (JRE7) or later in order to execute the application.

2.4.3 Constraints

The application must be completed within 6 weeks with a functioning final version presented at that time.

2.5 Requirements Apportioning

The priority levels of the following requirements are as follows:

Priority Level	**Description**
1	These requirements are essential to the application and must be included.
2	These requirements are highly desirable and most likely will be satisfied.
3	These requirements are desired but not required. These requirements should not be worked upon unless priority level 1 and 2 requirements are satisfied and verified.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 Client-Side Application

R1.1 Menubar Functionality

- R1.1.1 The application should allow the user to resign the game. If the user elects to resign, the game will display the opponent as victor. **Priority 1**
- R1.1.2 The application should allow the user to save the game. The game will be saved with the exact board state, current turn, and time remaining. **Priority 2**
- R1.1.3 The application should allow the user to load a game. Loading of a game can only happen before connecting to an opponent. The application will display the saved games to load and allow the user to pick one. **Priority 2**
- R1.1.4 The application should allow the user to host a game. When the user selects to host, he will be presented with the Host Game Screen. **Priority 1**
- R1.1.5 The application will allow the user to join a game. When the user selects to join a game, he will be presented with the Join Game Screen. **Priority 1**

R1.2 Host Game Modal

- R1.2.1 The application should give the user a unique id. **Priority 1**
- R1.2.2 The application should allow the host to set a time limit in minutes. **Priority 1**

R1.3 Join Game Modal

- R1.3.1 The application should allow the user to enter the unique id of the host. **Priority 1**
- R.1.3.2 If the unique id is incorrect, it should inform the user. **Priority 1**
- R.1.3.3 If the other player is available, it should start the match and set the host as the first player. **Priority 1**

R1.4 Boardgame Screen

- R1.4.1 The application should display to the user a standard chessboard with all pieces in the correct starting locations. **Priority 1**
- R1.4.2 The application should display the host's color as white and the opposing as black. The player's color should be on the bottom half of his chessboard. **Priority 1**

R1.4.3 The application should display the pieces of the opponent that the user has captured below the chessboard. The pieces that the opponent has captured will be displayed above the board. **Priority 3**

R1.4.4 The application should display how much time the user and the the opponent has remaining to make his move. The time is cumulative for the game, e.g., a 2 minute turn will bring your total time of 15 minutes to 13 after the turn. **Priority 1**

R1.4.5 The application should allow the user to drag a chess piece to a new position. The piece will stay in this position if it is a valid move, otherwise it will revert back to the original position. **Priority 1**

R1.4.6 If a user is in checkmate, the application should display a game over message indicating the player lost before he can make a move. **Priority 1**

R1.4.7 If the game is in stalemate, the application should display a game over message indicating the game ends in a draw. **Priority 1**

R1.4.8 The application should inform the user at the beginning of his turn if the user is in check and not allow moves that keep him in check or put his king in check. **Priority 1**

R1.4.8 The application will allow the user to enter a message in the chat box. This message will be saved in the window for both users to see. **Priority 3**

R1.4.10 The application will allow the user to save the game. This file can then be loaded at the hosting screen. **Priority 2**

R1.4.11 The application will allow the user to play again against the same opponent once a match has ended. **Priority 3**

R1.5 Gameplay Functionality

R1.5.1 The application will only allow the user to move his own pieces. **Priority 1**

R1.5.2 If the user moves a piece to a wrong position, it will revert the piece to it's original position. Thus, the application verifies all moves by the user. **Priority 1**

R1.5.3 If the user runs out of time, the application should deem this as a loss. **Priority 1**

R.1.5.4 Each piece may only according to its legal moves. The rules used are set by the World Chess Organization. **Priority 1**

R1.5.5 The application should allow the user to perform the Castle move by moving his King 2 rows to the left or right and the gameboard will perform the castle permitting it’s a legal move. **Priority 1**

R1.5.6 If a pawn achieves a pawn promotion, the user will be asked which piece he would like to replace the pawn with. **Priority 2**

R1.5.7 If a pawn is allowed to perform *En Passant*, the user must move the pawn to its allowed position and the pawn behind will be automatically captured. **Priority 1**

R1.5.8 If the user is in check and has a legal move to get out of check, then the application will not allow the user to select a move where he remains in check. **Priority 1**

R1.6 Chat Panel

R1.6.1 The application should contain a chatlog with a record of chat messages sent between players. Each message in the log will be contained in the window with Player1 or Player2 preceding the message. All messages will be displayed in sequential order of their reception times. **Priority 3**

R1.6.2 The application should have a textfield below the chatlog that allows the user to enter a message. **Priority 3**

R1.7 Move Log Panel

R1.7.1 The application should contain a log of the moves. Each line of the move log will contain the player (Player1 / Player2) and a line of the move. The move should be in the format of piece: row,col -> row,col (e.g. p: 1,2 -> 3,4). **Priority 1**

R1.7.2 The application should display two moves in one line for a Castle move. **Priority 2**

R1.7.3 The application should display a pawn promotion by the movement of the pawn and the name of the piece the pawn is replaced by. **Priority 2**

3.2 Non-functional Requirements

3.2.1 Deployment

R2.1 The application will be deployed as .jar file. Anyone with the .jar file will be able to host or join a game by connecting to the central server. **Priority 1**

3.2.2 Capacity

R3.1 The application server will be run on Tux. **Priority 2**

R3.2 The application will allow up to 10 players (5 pairs) to be playing chess at any time. **Priority 2**

3.2.3 Time and Speed

R4.1 A move, once played, should be performed with a reasonable time of 3-5 seconds, including time accounted for network latency. **Priority 2**

3.3 User Interface

3.3.1 Main Screen

The main screen is split into three separate parts: the menubar, side panel, and gameboard.

R5.1 Menubar

Figure 1 contains the menubar at the top of the screen. The menubar contains four buttons: connect, resign, save, and load. When clicked, the connect button opens a new modal window to allow a user to host or join a game; this is described in more detail below. The resign button will end the game and notify the other player of the resignation. Save will capture the current game state without user input and notify them once completed. Load will show the user a list of saved games and allow them to choose which to play using a unique game key. **Priority 1**

R5.2 Side Panel

Figure 1 contains the side panel on the left side of the screen. The side panel houses the interfaces of a move list, which displays what moves have been made during the game, a timer display, which shows the players how much time remaining to play the game, and a chat window where the players can talk and interact with each other while the game progresses. **Priority 1**

R5.3 Gameboard

Figure 1 contains the gameboard on the right side of the screen. The gameboard contains a border around a square of 8 x 8 alternating colored squares. On each of these squares, either a game piece or nothing will be placed. When a piece is taken, it will show up along the edge of the board to show the players what pieces have already been taken. **Priority 1**

3.3.2 Connect Screen

R6.1 Connect Screen

Figure 2 is the connect screen. The connect screen is seen when the connect button on the menubar is pressed by a player. Here, one can choose to host or join a game with the use of radio buttons. If one chooses to host a game, they will be assigned and shown a unique identification number to tell a friend so that their friend can join the same game. If one chooses to join a game, they must type a unique identification number into a text field. Once one decides to host or join a game, they must click the “Start” button to begin. **Priority 1**

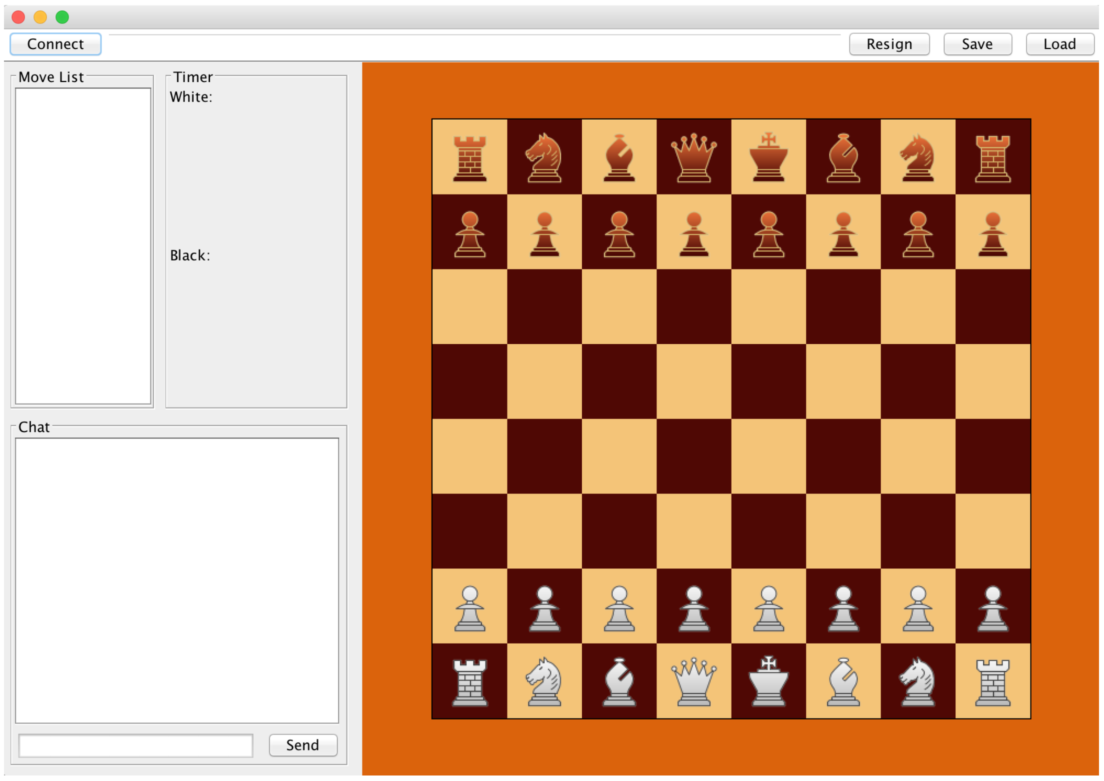
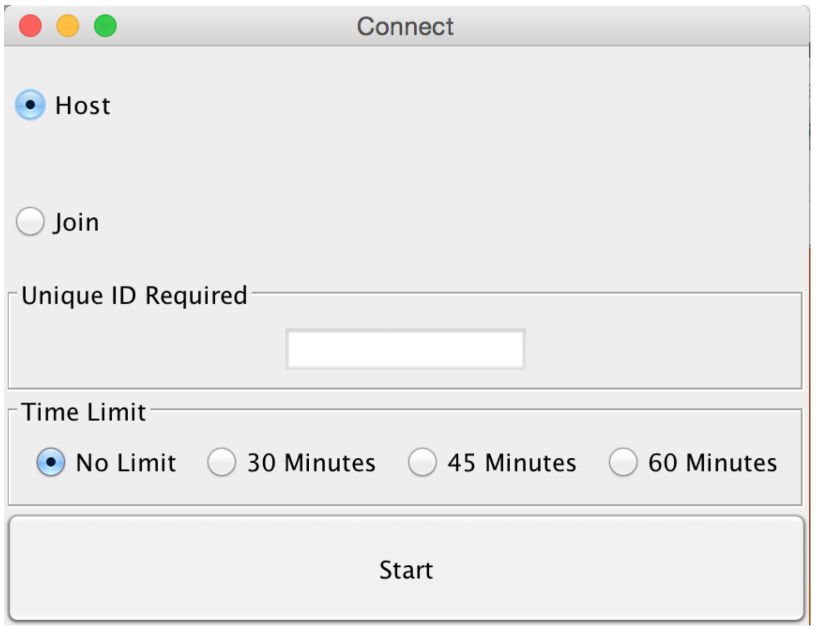


Figure 1: Main Screen


Connect

[✓ Host]

Join

Unique ID Required

Time Limit

[✓ No Limit] 30 Minutes 45 Minutes 60 Minutes

Start

Figure 2: Connect Screen

4. Use Cases

4.1 Use Case Flow

4.1.1 Host a game

Preconditions:
- The player desires to act as the host of the game
- The player has launched the chess application
- A game has not yet been started

Main Flow:
- The player observes the application generating a unique game identification code to be used by the second player for establishing a connection
- The guest player must join the game by using the identification code generated in the previous step

Postconditions:

- The host player waits for the chessboard to load and for game to begin

4.2.2 Join a game

Preconditions:
- There exists a host player who has already opted to host the game
- There exists a unique game identification code to be used for joining the game
- The player has launched the chess application
- A game has not yet been started

Main Flow:
- The player observes the application requesting the unique game identification code for establishing a connection
- The guest player enters the identification code for establishing the connection

Postconditions:
- A connection between host and guest players is established and the game begins

4.2.3 Play a move (game)

Preconditions:
- A connection between both players has been established
- The chessboard is displayed by the application along with all the pieces aligned in their original position, if it is a new game, otherwise the positions they were last placed in (if it is a loaded game)

Main Flow:
- The player plays a move by dragging chess piece from its existing position to its new destination position
- Move is validated by the application for being correct and consistent with game rules

Postconditions:
- Control is handed over to the other player to play a move

4.2.4 Save a game

Preconditions:
- A game has already been started or is in progress

- It is currently the turn of the player who wants to save

Main Flow:
- The player clicks on the dedicated button for saving the game
- Game configuration is saved locally on the host's machine

Postconditions:
- The players stop playing the game and exit the application

4.2.5 Load a game

Preconditions:
- The player has launched the chess application
- A game has not yet been started

Main Flow:
- The player clicks on the dedicated button for loading a game
- Game is loaded
- The player who loads the game now acts as the host player

Postconditions:
- The guest player now joins the host player using the game identification code associated with loaded game

4.2.6 Send a chat message

Preconditions:
- A game has been started or is in progress

Main Flow:
- The player sends an instant message to the another player using the application interface

Postconditions:
- The other player receives the chat message

4.2.7 View (received) chat messages

Preconditions:
- A game has been started or is in progress

- A chat message was sent by another player

Main Flow:
- The application window or panel displaying received chat message refreshes to display the latest received chat messages

Postconditions:
- -None-

4.2.8 Resign from game

Preconditions:
- A game has not yet been started, been started or is in progress

Main Flow:
- The player chooses to resign from the game
- The connection is terminated and session is closed

Postconditions:
- The unique game identification code associated with the game is destroyed
- The game associated with the current session can no longer be played or resumed
- The other player’s screen indicates the the game has now been terminated

Use Case Diagram

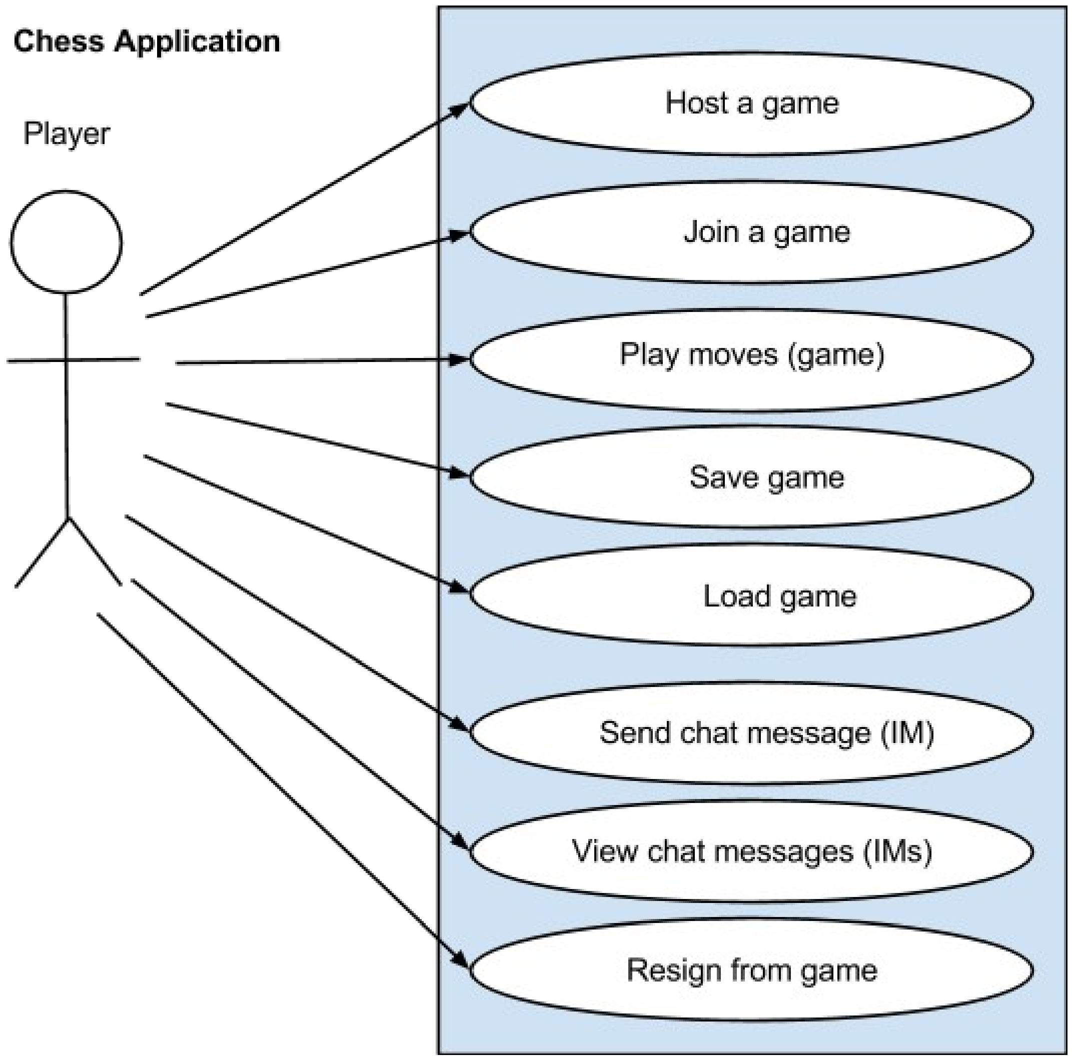


Figure 3: Use case diagram for a user (player) of the chess game application

Activity Diagram

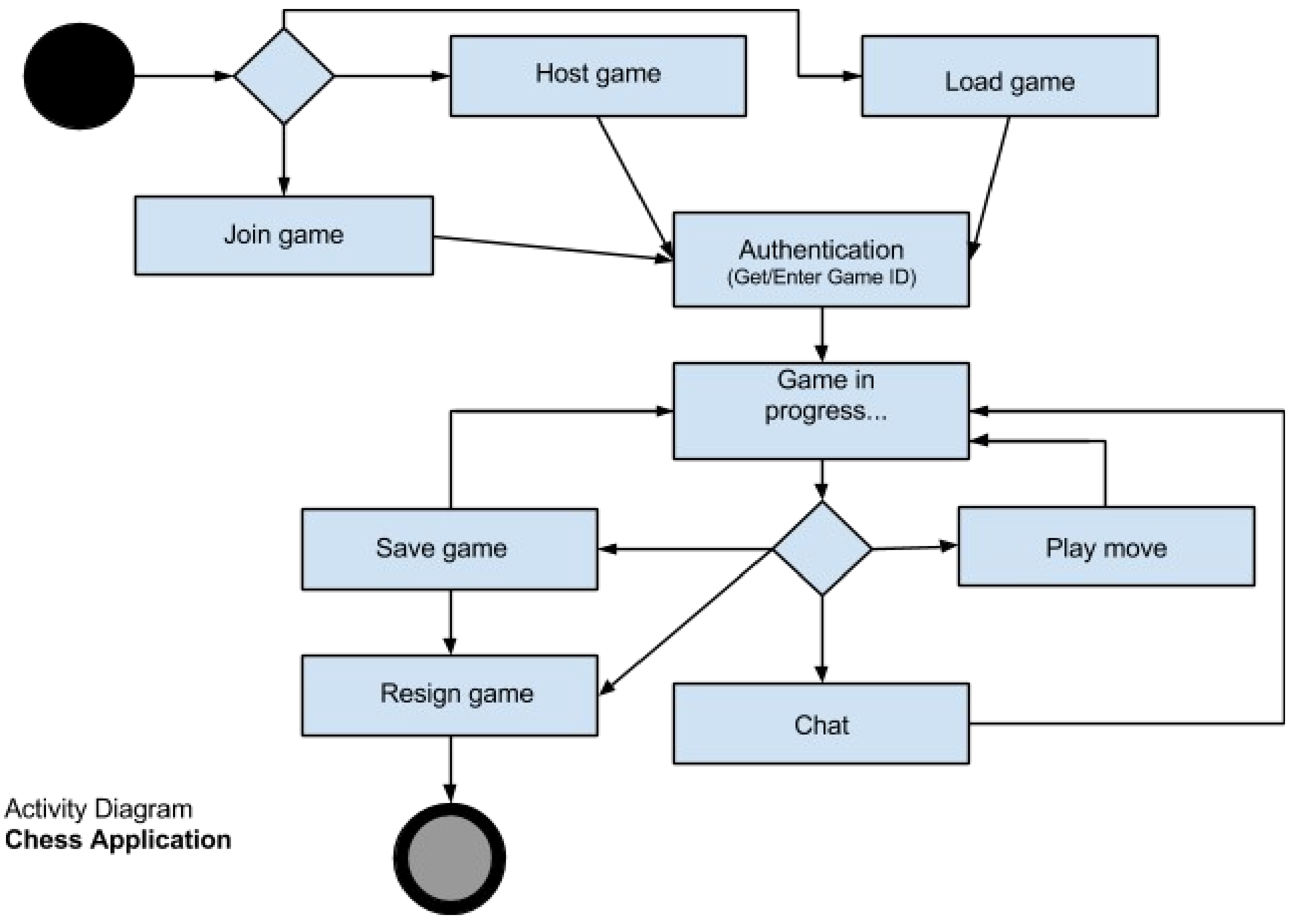


Figure 4: Activity diagram for a user (player) of the chess game application

5. System Evolution

Currently, the chess game will implement nearly all the functionality that may be required to play a full-fledged game of chess with another person, emulating the actual physical act as closely as possible. Because the game will support all regular chess moves and professional chess features such as timers, the extra functionality that can be added generally comes to “extraneous” features that enhance the playing environment.

For example, the communication channels could potentially be improved. The current iteration of the program is slated to support only text chat, but that does not emulate the actual game quite as closely as it could, so, in the future, voice, or perhaps even video, chat could be added.

Another possible realm of improvement is in the types of play offered. Currently, the program only supports two-player games of chess over the internet. In the future, it might be possible to add local two-player games, where both players sit at one computer and play one another; or it might be possible to add single-player games, where a player plays on his computer against an artificial intelligence opponent; or it might be possible to add games with random opponents, so that users do not have to have previously known the ID of the game they wish to join.

In any case, these are only a few examples of how this program, beyond simple improvements in usability, might change in the future. Though the specific improvements might not come to be, and others might appear in the future, these can act as a basis and an example of how this program might change in the future and how it is prepared to change.

6. Appendix

6.1 Glossary

Castle - A special move in Chess where a King moves two columns to the left or right and the rook of that side moves to the King's opposite side. See World Chess Organization's official chess rules for more details on the conditions to Castle.

En Passant - A special move in Chess where a pawn can capture an opposing pawn by attacking the previous position the opposing pawn was in. See World Chess Organization's official chess rules for more details on the conditions to En Passant.

Modal - A pop-up window that is not part of the main screen interface. The user must complete interaction with the modal before he can go back to interaction with the active screen.

Pawn Promotion - When a user's pawn reaches the opposite side of the board, the player selects a new piece of his own color to replace the pawn with.

Tux - Servers that are available to computer science students at Drexel that will be used for hosting.

World Chess Organization - Organization that sets the rules of chess for international play. See their website (http://www.fide.com/component/handbook/?id=124&view=article) for the complete set of rules which are the same rules this application should follow.