ChessEDU
Software Requirements Specifications

Version <2.1>

Revision History

Date	Version	Description	Author
25/10/2022	1.0	First Draft	Adair Torres
26/10/2022	2.0	Second Draft	Grant Jones
30/10/2022	2.1	Edited for consistent formatting and spacing, updated table of contents.	Adair Torres

Table of Contents

Software Requirements Specifications

1. Introduction

1.1 Purpose

The purpose of this document is to define and describe the software requirements for a web application that will teach the users the basics of chess.

The intended use of this document is for software developers of the web application.

1.2 Scope

The chess app, ChessEDU, using interactive lessons will teach users how to move each piece, various attacking patterns, checks, defending, pins and skewers, two main line openings for white and black, and en passant.

ChessEDU could be used to teach anyone to play chess with the use of the lesson modules. Users of ChessEDU will be able to showcase their knowledge of the game by playing against a friend with the app in a pass and play format. The objective of ChessEDU is to make it easier to learn the intimidating game of chess. Beginner players of chess use ChessEDU to learn the basics of chess like individual piece movement, attacking patterns, and checks. Intermediate players of chess use ChessEDU to learn more about advanced aspects of chess like defending, pins and skewers, openings, and en passant. Advanced players of chess use ChessEDU as a review of the basic and advanced concepts of chess.

1.3 Definitions, Acronyms, and Abbreviations

Chapter: Groups of lessons

GM: Gameplay Module

LM: Lesson Module

SM: Server Module

SRS: Software Requirements Specifications

1.4 References

No references.

1.5 Overview

The rest of this document contains a succinct description of the ChessEDU software system (Section 2), and the software requirements specifications for the system (Section 3).

| ChessEDU	Version: <2.1>
Software Requirements Specifications	Date: <30/10/2022>
chessedu_srs	

2. Overall Description

2.1 Product perspective

Chess can be a very intimidating game to learn, and thec hess population is growing after the release of the Queen's Gambit on Netflix. Chess is a fun game and has alternative benefits like the development of higher-level thinking. Without an easy way to learn the game, many people may feel discouraged to learn chess because of its complicated nature. A chess tutorial app can solve this problem.

2.1.1 System Interfaces

The ChessEDU system is to be developed is a stand-alone tool that can be accessed through the Internet. It consists of in four major components: a Server Module, a Database, a Lesson Module, a Gameplay Module.
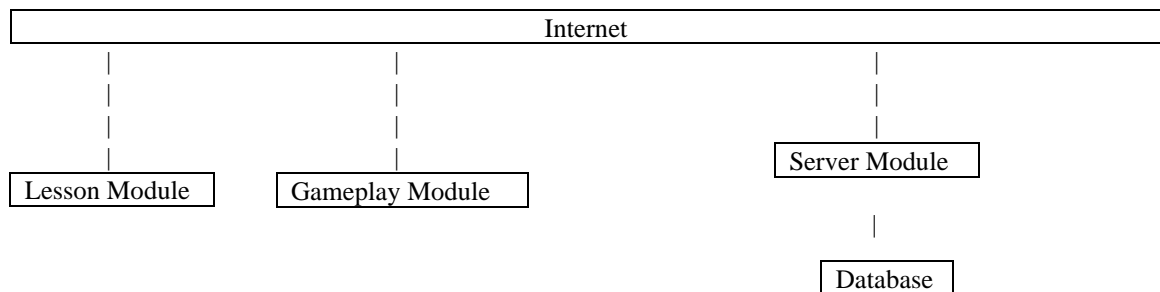
The LM allows users to log onto the ChessEDU system, so users are able to track their progress of the lessons they've completed. The SM allows the LM and GM to connect to it and is as an interface between the modules and the database. The GM allows a user to play a game of chess in a pass and play format. The database can be any type of database and doesn't have to be developed within the ChessEDU system, provided that the SM can interact with the available database system. All components must execute on Windows.

2.1.2 User Interfaces

The LM and GM must provide a user interface that is utilized through Flask. The SM must be able to launch on command but doesn't require a user interface. The database will not have a user interface.

2.1.3 Hardware Interfaces

All components must be able to perform on a personal computer.

2.1.4 Software Interfaces

The LM and GM must be Python scripts running within Flask. The SM must run within a web server available for Windows.

2.1.5 Communication Interfaces

The LM and GM will communicate with the server over a TCP/IP connection. The SM and database will be located on the same host.

2.1.6 Memory Constraints

The LM and GM must be able to operate within 64MB. The SM and database must be able to operate within 128MB.

2.1.7 Operations

The operation of the LM and GM must be easy and intuitive for users. No specific technology knowledge or experience should be required to use the app. The SM will be installed and maintained with no interaction with existing software and not require any technical skills from the network administrator.

Backup operations will be defined.

Recovery operations will be defined in case of network failure, user machine failure, and database failure.

2.2 Product functions

The two main functions of ChessEDU are to teach users basic and advanced concepts of chess and to allow users to practice their new skills in a pass and play game format.

A lesson will teach the user a concept of chess and then ask the user to apply the newly acquired knowledge in a puzzle. Once the user has completed all the lessons for the given chapter, the user will have completed the chapter and be able to move onto the next chapter.

The user will also be able to showcase their knowledge with their friends in a pass and play format where the users will take turns moving their respective pieces on the same device.

The database stores user configurations, which consists of settings, email, username, and password.

2.3 User characteristics

Users are primarily chess beginners and intermediates. Users want to learn how to play the game of chess through interactive modules.

2.4 Constraints

The system will have user authentication security.

2.5 Assumptions and dependencies

No specific assumptions or dependencies.

2.6 Requirements subsets

None.

3. Specific Requirements

3.1 Functionality

3.1.1 Lessons

3.1.1.1 Each lesson will be interactive, meaning each lesson will have at least have one way for the user to apply or demonstrate the acquired knowledge.

3.1.1.2 Each chapter ends with a cumulative review.

3.1.2 Settings

3.1.2.1 User can toggle the following settings: highlighting possible moves after selecting a piece, highlighting of hanging pieces, highlighting of pinned pieces, highlighting of pieces in or entering capture spaces, notification when a player is in check, confirm each move, Auto-Queen when pawn promotes, highlighting last move, sounds, and change theme.

3.1.3 Playing the Game

3.1.3.1 A pair of users can play a game on a single device.

3.1.3.2 During the game, the user has the option to resign, offer draw, and see previous moves.

3.1.3.3 If a player offers a draw, the other player will have the option to accept or decline the draw.

3.1.3.4 If the player accepts the draw, then a popup will state "Draw" with the option to play again or go back to the home screen.

3.1.3.5 If the player declines the draw, then the other player will be notified that the player declined the draw, and the game will resume.

3.1.3.6 After checkmate, resignation, or stalemate, a popup will state "Black wins", "White wins", or "Draw" with the option to play again or go back to home screen.

3.1.4 Home Screen

3.1.4.1 Home screen will have the option to play a game or go to lessons.

3.1.4.2 If the user chooses to play a game, then the user will be provided a board to play with a friend locally.

3.1.4.3 If the user chooses to go to lessons, then the next recommended lesson will pop up with the remaining lessons underneath.

ChessEDU	Version: <2.1>
Software Requirements Specifications	Date: <30/10/2022>
chessedu_srs	

3.2 Use-Case Specifications

3.2.1 Account Creation

3.2.1.1 Actor is an unregistered user.

3.2.1.2 Server needs to request email, username, and password.

3.2.1.3 A user account with this information is created in the database afterwards.

3.2.1.4 If the login is already taken, then the process will loop to prevent the actor from progressing until unique account information is entered.

3.2.1.5 If the actor is already signed in, then this specific actor should not be allowed to enter the system and prompt them if they want to sign out instead.

3.2.2 Signing In

3.2.2.1 Actor is a registered user that is not logged in.

3.2.2.2 The actor needs to be prompted with a login page.

3.2.2.3 Login page needs to contact the database to validate the user.

3.2.2.4 Actor should be redirected to a menu with their account specific features.

3.2.2.5 If the actor is signed in, then the specific actor should not be allowed to enter this system and the actor will be prompted if they want to sign out instead.

3.2.2.6 If the actor enters invalid information, the interface will prevent the actor from progressing and needs to prompt them about what is invalid (ex: wrong password).

3.2.3 Signing Out

3.2.3.1 Actor is a registered user that is logged in.

3.2.3.2 Any other user should not be able to see the option to sign out.

3.2.3.3 Server will be contacted that the user wants to sign out.

3.2.3.4 Actor is redirected to the view of the page a signed-out user can see.

3.2.4 Selecting a Lesson

3.2.4.1 Actor is any user (unregistered/registered)

3.2.4.2 Actor needs access to a catalog of available lessons

3.2.4.3 Server needs to gather lessons from the database that contains them, handling any connection issues in the process

3.2.4.4 Actor can choose to enter a lesson, which hands control to a new display

3.2.4.5 As a lesson progresses, the interface includes different options displayed to the user: a user can flip from one page to the next/previous page, interface accesses pages from the server database, pages

need to display text to the user in an easily readable manner, interactive elements (such as small chess boards).

3.2.4.6 The interactive elements includes: the user can move pieces in real time, element provides feedback (ex: "good move!") after an action, element accurately updates the position of pieces, a piece can be "locked" so the user cannot move it

3.2.4.7 Actor can leave the lesson at any point and save progress if logged in

3.2.5 Playing on Your Own

3.2.5.1 Actor is any user (unregistered/registered)

3.2.5.2 Actor can leave this page at any time

3.2.5.3 Control alternates between a "white" and a "black" player (white goes first).

3.2.5.4 The other player's pieces are locked during one player's turn.

3.2.5.5 Player is prompted that it is their move.

3.2.5.6 Player can move a piece to a position.

3.2.5.7 If the movement is invalid for that piece: do not update the board, prompt the player that the move is invalid, and give the player the ability to move again.

3.2.5.8 If the movement is valid for that piece: update the board.

3.2.5.9 If the movement captures a piece: the piece is removed from the board.

3.2.5.10 Control shifts to the opposing player when the current player makes a valid move

3.2.5.11 If the player is in "check": a different set of movement rules should be applied so that the user has to address the threat.

3.2.5.12 The interface should check if the board state after a move produces either a checkmate or stalemate.

3.2.5.13 If there's a checkmate, the player who last moved is the "winner" of the game

3.2.5.14 If there's a stalemate, the interface should prompt the players that there is a draw

3.2.6 Editing User Settings

3.2.6.1 Actor is a registered user.

3.2.6.2 All other actors should not have access to this menu.

3.2.6.3 The actor should be prompted with a list of settings.

3.2.6.4 When the actor updates a setting, the server should receive the request and save the change to the database.

3.2.6.5 If a setting is sensitive ("change password", "update email"): the actor should be prompted to

validate their information.

3.2.6.6   After three attempts of validating their information, they should be signed out

3.2.6.7   Email confirmation for specific setting changes like password will be sent.

3.3   Supplementary Requirements

3.3.1   Client / Server

3.3.1.1   The user interacts with the system through a mobile or a web browser

3.3.1.2   Either will execute HTML and Java queries to the web engine using JSP

3.3.1.3   The engine communicates with the database using SQL

3.3.1.4   The database returns the data to the engine which, in turn, returns the resulting package to the browser or application, displaying the results afterwards.

3.3.2   Security

3.3.2.1   Individual users have the option to create accounts within the system for tracking personal progress and data.

3.3.2.2   Account credentials will need to be stored within the database, made publicly inaccessible, and encrypted to guarantee security.

3.3.2.3   No information beyond bare minimum identification and module progress will be stored to minimize the impact of a data leakage should security measures fail.

3.3.3   Accessibility

3.3.3.1   System should include accessibility the certain groups of users depend on in order to utilize any software, including but not limited to: text narration, screen readers, variable font size, audio captions, and color correction.

4. Classification of Functional Requirements

Functionality	Type
Each lesson will be interactive, meaning each lesson will have at least have one way for the user to apply or demonstrate the acquired knowledge.	Desirable
Each chapter ends with a cumulative review.	Optional
User can toggle the following settings: highlighting possible moves after selecting a piece, highlighting of hanging pieces, highlighting of pinned pieces, highlighting of pieces in or entering capture spaces, notification when a player is in check, confirm each move, Auto-Queen when pawn promotes, highlighting last move, sounds, and change theme.	Optional
A pair of users can play a game on a single device.	Desirable
During the game, the user has the option to resign, offer draw, and see previous moves.	Essential
If a player offers a draw, the other player will have the option to accept or decline the draw.	Essential
If the player accepts the draw, then a popup will state "Draw" with the option to play again or go back to the home screen.	Essential
If the player declines the draw, then the other player will be notified that the player declined the draw, and the game will resume.	Essential
After checkmate, resignation, or stalemate, a popup will state "Black wins", "White wins", or "Draw" with the option to play again or go back to home screen.	Optional
Home screen will have the option to play a game or go to lessons	Essential
If the user chooses to play a game, then the user will be provided a board to play with a friend locally.	Essential
If the user chooses to go to lessons, then the next recommended lesson will pop up with the remaining lessons underneath.	Desirable
Actor is an unregistered user for account creation.	Essential
Server needs to request email, username, and password.	Essential
A user account with this information is created in the database afterwards.	Essential

If the login is already taken, then the process will loop to prevent the actor from progressing until unique account information is entered.	Essential
If the actor is already signed in, then this specific actor should not be allowed to enter the system and prompt them if they want to sign out instead.	Essential
Actor is a registered user that is not logged in for signing in.	Essential
The actor needs to be prompted with a login page.	Essential
Login page needs to contact the database to validate the user.	Essential
Actor should be redirected to a menu with their account specific features.	Essential
If the actor is signed in, then the specific actor should not be allowed to enter this system and the actor will be prompted if they want to sign out instead.	Essential
If the actor enters invalid information, the interface will prevent the actor from progressing and needs to prompt them about what is invalid (ex: wrong password).	Essential
Actor is a registered user that is logged in for signing out.	Essential
Any other user should not be able to see the option to sign out.	Essential
Server will be contacted that the user wants to sign out.	Essential
Actor is redirected to the view of the page a signed-out user can see.	Essential
Actor is any user (unregistered/registered) for selecting a lesson.	Essential
Actor needs access to a catalog of available lessons.	Essential
Server needs to gather lessons from the database that contains them, handling any connection issues in the process	Essential
Actor can choose to enter a lesson, which hands control to a new display.	Essential
As a lesson progresses, the interface includes different options displayed to the user: a user can flip from one page to the next/previous page, interface accesses pages from the server database, pages need to display text to the user in an easily readable manner, interactive elements (such as small chess boards).	Essential
The interactive elements includes: the user can move pieces in real time, element provides feedback (ex: "good move!") after an action, element accurately updates the position of pieces, a piece can be "locked" so the user cannot move it.	Desirable
Actor can leave the lesson at any point and save progress if logged in.	Optional

Actor is any user (unregistered/registered) for playing on their own.	Essential
Actor can leave this page at any time.	Desirable
Control alternates between a "white" and a "black" player (white goes first).	Essential
The other player's pieces are locked during one player's turn.	Essential
Player is prompted that it is their move.	Desirable
Player can move a piece to a position.	Essential
If the movement is invalid for that piece: do not update the board, prompt the player that the move is invalid, and give the player the ability to move again.	Essential
If the movement is valid for that piece: update the board.	Essential
If the movement captures a piece: the piece is removed from the board.	Essential
Control shifts to the opposing player when the current player makes a valid move.	Essential
If the player is in "check": a different set of movement rules should be applied so that the user has to address the threat.	Essential
The interface should check if the board state after a move produces either a checkmate or stalemate.	Essential
If there's a checkmate, the player who last moved is the "winner" of the game.	Essential
If there's a stalemate, the interface should prompt the players that there is a draw.	Essential
Actor is a registered user for editing user settings.	Essential
All other actors should not have access to the settings menu.	Essential
The actor should be prompted with a list of settings.	Desirable
When the actor updates a setting, the server should receive the request and save the change to the database.	Essential
If a setting is sensitive ("change password", "update email"): the actor should be prompted to validate their information.	Essential
After three attempts of validating their information, they should be signed out.	Optional
Email confirmation for specific setting changes like password will be sent.	Optional
The user interacts with the system through a mobile or a web browser.	Desirable

Either will execute HTML and Java queries to the web engine using JSP.	Essential
The engine communicates with the database using SQL.	Essential
The database returns the data to the engine which, in turn, returns the resulting package to the browser or application, displaying the results afterwards.	Essential
Individual users have the option to create accounts within the system for tracking personal progress and data.	Essential
Account credentials will need to be stored within the database, made publicly inaccessible, and encrypted to guarantee security.	Essential
No information beyond bare minimum identification and module progress will be stored to minimize the impact of a data leakage should security measures fail.	Essential
System should include accessibility the certain groups of users depend on in order to utilize any software, including but not limited to: text narration, screen readers, variable font size, audio captions, and color correction	Essential

5. Appendixes

None