Rogue Chess

Travis Head
Jose Herrera

CONCEPT OF OPERATIONS

REVISION – 1
1 February 2022

CONCEPT OF OPERATIONS
FOR
Rogue Chess

TEAM <15>

APPROVED BY:

_____
Project Leader Date

_____
Prof. S. Kalafatis Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
0.0	02/01/2022	Jose Herrera		Draft Release
1.0	2/23/2022	Travis Head		Revision 1

Table of Contents

List of Figures

1. Executive Summary

The sponsor, Prunav Dhulipala, has requested development of a Rogue Chess phone app. This app will behave like traditional chess except that there is a chance the user's pieces will not obey commands and will choose their own movements. The randomization is determined by the user selecting one of three difficulty settings at the start of the match. There will be an easy, medium, and hard mode that corresponds to a disobedience chance of 10%, 30%, and 50% respectively. The opponent's artificial intelligence will not be affected by the difficulty setting. It will play in a traditional style using the neural net algorithm which, learns the players moves and predicts the next move the player will make. The artificial intelligence will make its decisions relatively quickly to keep the user engaged; this time will be less than 2 seconds. The app will assist the player in their valid move choices by highlighting available spaces on the chess board. The app will be available to android smartphones that have connection to the internet. This program is only intended for a single player to use at a time for proof of concept. The storage space will be kept to a minimum of 20 MB. A game timer will show users how fast they beat the opponent which will provide a reference for their skill.

2. Introduction

The purpose of this document is to present the development of a Rogue Chess App, an app of a new chess variation for Android products. This chess variance will occasionally ignore the instructions of the user and choose a random valid movement choice for the chess piece chosen against a traditional artificial intelligence for chess. This will challenge the user's typical strategies in chess and provide a unique experience that is unavailable elsewhere online or in person.

2.1. Background

The game of chess has existed for around 1500 years and has since spread all across the world in popularity. Both tournaments and casual games have been a large factor in propagating its success. In modern times, people have frequently looked towards apps and websites to host the game for practice and personal enjoyment. As such, there are several websites and apps that provide the player with customizable traditional versions or unconventional games modes.

Chess has long-standing popularity amongst veterans of the game and consistently has new players eager to learn. While there are millions of possible outcomes during a single game, there are still players seeking new experiences. Many variations in board design, number of pieces, and types of pieces have all been adapted to alter the player's experience. Nevertheless, there is not a widely available variation of chess where the pieces "go rogue" and do not listen to the user consistently.

2.2. Overview

Our chess app will fill the void amongst existing variations of chess. The board and chess pieces of the app will be standard for play and will be accessed via an android smartphone. The user will be able to select the difficulty of the game; harder modes raise the chance of the user's pieces disobeying commands. If a piece is determined to disobey, it will choose a random valid position instead. If the King is in check, all pieces will listen to the user for that round. The King will always listen to the user. The opponent's artificial intelligence (AI) will use an alpha-beta algorithm and cost functions to determine the best possible moves. The AI will be uploaded into the Cloud for faster decision making.

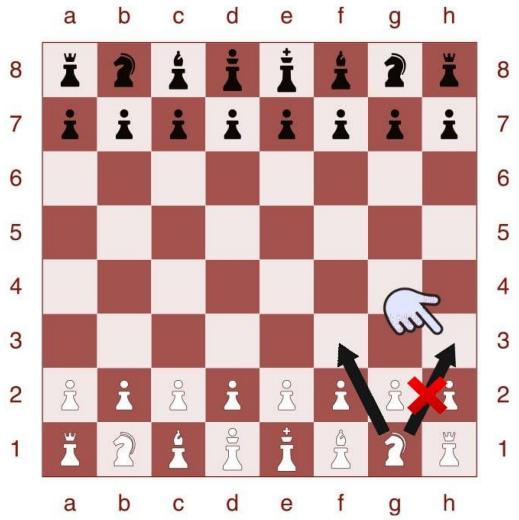

Figure 1: Disobedient Piece Movement

2.3. Referenced Documents and Standards

https://www.javatpoint.com/ai-alpha-beta-pruning
https://developer.android.com/studio
https://www.chess.com/terms/chess-pieces

3. Operating Concept

3.1. Scope

Rogue Chess is a unique way of playing the game of chess due to the unexpected change of piece movements. An algorithm will be implemented to the chess pieces which will create a randomization percentage depending on the game mode the user is on, which will determine whether the chess piece will listen to the user or not. The implementation will keep the user thinking ahead in order to adapt to the different scenarios the chess pieces may have. An AI (Artificial Intelligence) will be implemented to the single player game in order to play against the player. The AI response time will be two seconds or less in order to keep the player engaged. A function will be implemented to highlight the available chess piece moves so the player plays according to the official rules of chess. The game will be uploaded to the Cloud to be able to be made into an app every android user could use.

3.2. Operational Description and Constraints

The Rogue Chess will be a phone application that may be used by any android user. The game will follow the rules of chess except for the user's piece's constraints. The pieces have the option whether to follow the user's instructions or make a random move while following the valid move set of the chosen piece. The legal moves the pieces may make will be highlighted in yellow. The game will be a single player mode with three difficulties. The three difficulties will be based on the percentage of obedience of the chess pieces. The game will be a single player mode which means an AI (Artificial Intelligence) will be implemented to make moves for the opposing sides in order to beat the user.

The following constraints include:

- Making a compatible app for android users which is within 20Mb of storage.
- The app is not able to crash when the user exits the app or turns off the phone.
- The game may not crash due to high usage.
- The graphics should be clear and visible to the user.
- The AI response time should be two seconds or less.
- The AI should be intellectual enough to beat the user.
- The randomized piece movements (if any) need to be legal moves.
- If the randomization happens, the piece needs to move to a different space other than the chosen space.
- The randomization factor needs to follow the percentage of the set difficulty.
- We will be using $75 out of our $200 budget to buy an android phone
- Time restraints include: having to research how to play the chess game and applying working functions to create a chess game app.

3.3. System Description

- Android Phone Application: The subsystem will be built in Android Studio which uses the Java coding language. The app will include the graphics of the chess game which includes the board and the chess pieces.
- Application: The application will consist of multiple games modes (easy, medium, and hard) with each consisting of different probabilities.
- Artificial Intelligence: This subsystem is the implementation of an AI which will be playing against the player. The subsystem will consist of a Recurrent Neural Network that will be trained to take in the current board state and output a move that has the highest probability of winning. It will be stored in the Cloud for optimal computation time.

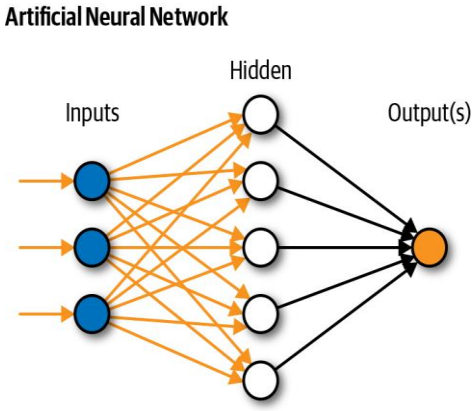

Figure 2: Neural Network

3.4. Modes of Operations

The Rogue chess will be composed of an AI system which will provide movements for the single player mode. The AI will have a min/max algorithm imbedded in order to make a move that will benefit the AI the most to beat the player. The player will have a click and move system with the assistance of highlighted boxes to make a legal chess move. However, depending on the difficulty the player is playing in, the chess piece may make a legal random move the player did not choose to do.

3.5. Users

The user base will be provided by chess-playing fanatics who want to try a new way of playing chess. The level of training needed for the user to use the application is basic chess knowledge, since the game will provide highlighted boxes of the legal possible moves. The basic instructions will be provided in the beginning of the game. The user will need knowledge on how to download an application via cellular phone. The marketing may be targeted to people who want to challenge themselves and exercise their brain functionality. The application may be monetized by having advertisements or having a payment fee to remove the advertisements.

3.6. Support

A set of instructions with rules and expectations will be provided on the main menu page. There will be an algorithm which will highlight the legal moves the user may make to provide a support system.

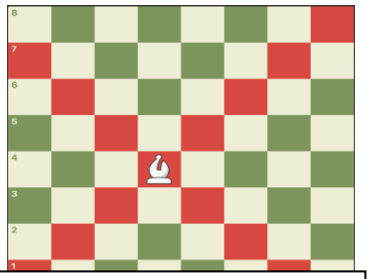

Figure 3: Highlighted Boxes of Available Moves

4. Scenario(s)

4.1. Single Player Game

Rogue chess is a single player chess game which follows the same conventions as a regular chess game. However, the chess piece may or may not want to follow the move the player wants to make depending on the game mode. The king piece will always listen to the player, and when the king piece is in check, all the pieces will listen.

4.2. Easy Mode

When the player selects easy mode, the player has a 90% chance that the chess pieces will listen.

4.3. Medium Mode

When the player selects medium mode, the player has a 70% chance that the chess pieces will listen.

4.4. Hard Mode

When the player selects hard mode, the player has a 50% chance that the chess pieces will listen.

4.5. Brain Teaser

The application may be used as a brain teaser to enhance brain functions such as thinking ahead, or a reflex enhancement in order to react to different unexpected movements. The application also allows the player to enhance patience and adaptation skills.

5. Analysis

5.1. Summary of Proposed Improvements

Improvements that the proposed system will provide:

- The game has a randomizer algorithm for the chess pieces.
- The AI will be programmed to analyze the best possible steps to beat the player.
- The click and move algorithm would make it easier for the player to move the pieces.
- The highlighted support system will allow someone to learn how to play chess legally.
- If the King is in danger, the King will be highlighted red to indicate it is in danger.

5.2. Disadvantages and Limitations

Expected disadvantages and limitations that the proposed system will have:

- Making a compatible app for android users which is within 20Mb of storage.
- The app is not able to crash when the user exits the app or turns off the phone.
- Finding a compatible phone under the budget may be difficult.
- The algorithm will be developed in Java format, which will be a limitation for someone who does not know how to code in Java.
- Researching how to play the game of chess in time to be able to code a game of chess.

5.3. Alternatives

The alternatives of Rogue Chess include:

- A regular conventional chess game which allows the chess pieces to listen 100% of the time.
- A chess game with different board shapes.
- An algorithm which will make the player's move for the player in case they get stuck.
- Java allows a parallel running capability that other languages do not allow.
- Neural Network AI will work faster than the alpha beta pruning with less problems.

5.4. Impact

Concerns may include:

- Collection of personal data without the consent of the user.
- Using software from other works without giving credit to the creator is a violation of proprietary rights.
- Only available to android user limits the audience's accessibility.

Positive impacts include:

- A fun way of playing chess
- New chess players will be able to learn how to play chess with the highlighted tile algorithm.
- Will help players win due to knowing when the king is danger.

Rogue Chess

Travis Head
Jose Herrera

FUNCTIONAL SYSTEM REQUIREMENTS

REVISION – Draft
21 February 2022

FUNCTIONAL SYSTEM REQUIREMENTS
FOR
Rogue Chess App

PREPARED BY:

_____
Author Date

APPROVED BY:

_____
Project Leader Date

_____
John Lusher, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	02/16/2022	Travis Head		Draft Release

Table of Contents

List of Tables

List of Figures

6. Introduction

6.1. Purpose and Scope

The purpose of this project is to improve upon a traditional chess app by making the game more challenging for the user. This app will allow the user to choose between three difficulty settings which correspond to how likely the user's chosen pieces will disobey the intended movement. The user interface menu will allow for a single or 2-player game mode. Also, rather than using common methods for the opponent's artificial intelligence, this app will use neural networking to make its decisions in 2 seconds or less. In game, the user's selected piece's valid moves should be highlighted to assist the user's experience.

This specification defines the technical requirements for the development items and support subsystems delivered to the client for the project. Figure 3 shows a representative integration of the project in the proposed CONOPS. The verification requirements for the project are contained in a separate Verification and Validation Plan.

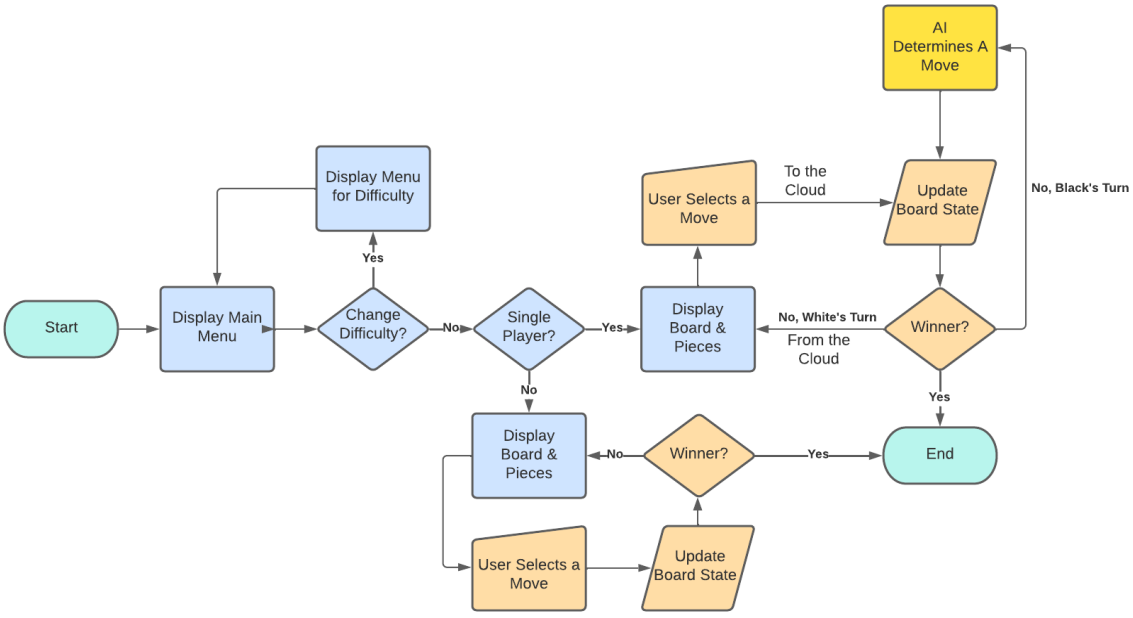

Figure 3. Rogue Chess Flowchart

6.2. Responsibility and Change Authority

Changes to performance requirements will only be made by Prunav Dhulipala, or by all team members with the consent of Prunav Dhulipala. Consensus of all team members is required for any changes to implementation. The team leader, Travis Head, is responsible for the fulfilment of all project requirements. Subsystem owners are responsible for their respective subsystem requirements as shown below:

- Travis Head: Artificial Intelligence
- Jose Herrera: User Interface and Game Functionality
- Applicable and Reference Documents

6.3. Reference Documents

The following documents are reference documents utilized in the development of this specification. These documents do not form a part of this specification and are not controlled by their reference herein.

Document Number	Revision/Release Date	Document Title
1	3.10.2	Python Standard Library
2	2.8.0	TensorFlow
3	2015	Official Chess Rules

6.4. Order of Precedence

In the event of a conflict between the text of this specification and an applicable document cited herein, the text of this specification takes precedence without any exceptions.

All specifications, standards, exhibits, drawings or other documents that are invoked as “applicable” in this specification are incorporated as cited. All documents that are referred to within an applicable report are considered to be for guidance and information only, except ICDs that have their relevant documents considered to be incorporated as cited.

7. Requirements

7.1. System Definition

The Rogue Chess App is a traditional chess app except for that the user’s selected pieces may randomly move to valid spaces not chosen by the user. There will be three difficulty settings that determine the randomness of disobedience. The user will have the option play against an Artificial Intelligence (AI) or against another user in a 2-player mode; both users will have the same difficulty setting applied to their pieces. The project is divided into 3 main subsystems: User Interface, Game Functionality, and Artificial Intelligence.

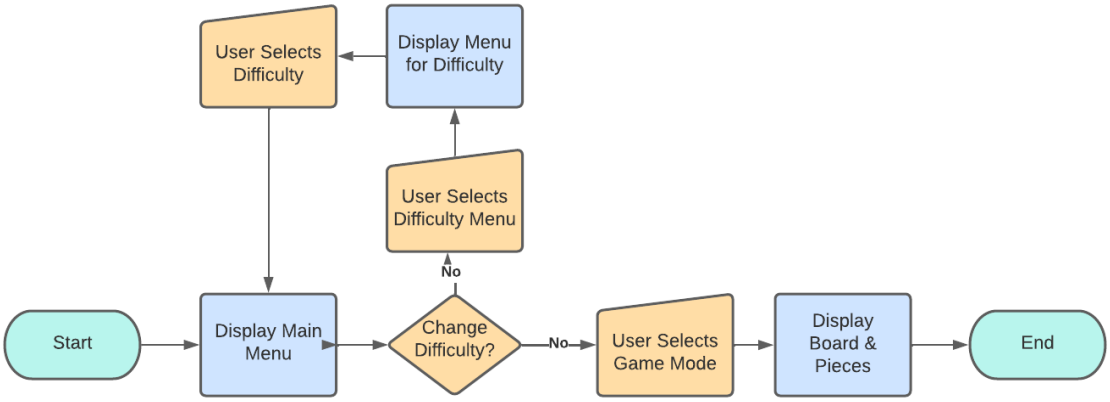

Figure 2: User Interface and Graphics

The user interface will consist of a tappable menu and graphics for the game. The Main Menu will have options for Difficulty, Single Player and 2-Player. When the Difficulty menu is selected, a new menu will replace the original with three tappable options for Easy, Medium, and Hard. The default difficulty will be easy. Once selected, the user will be returned to the Main Menu with the difficulty saved until the app is closed. When the Single Player or 2-Player options are selected, the menu will close, and the game will begin with the user as White and the opponent as Black.

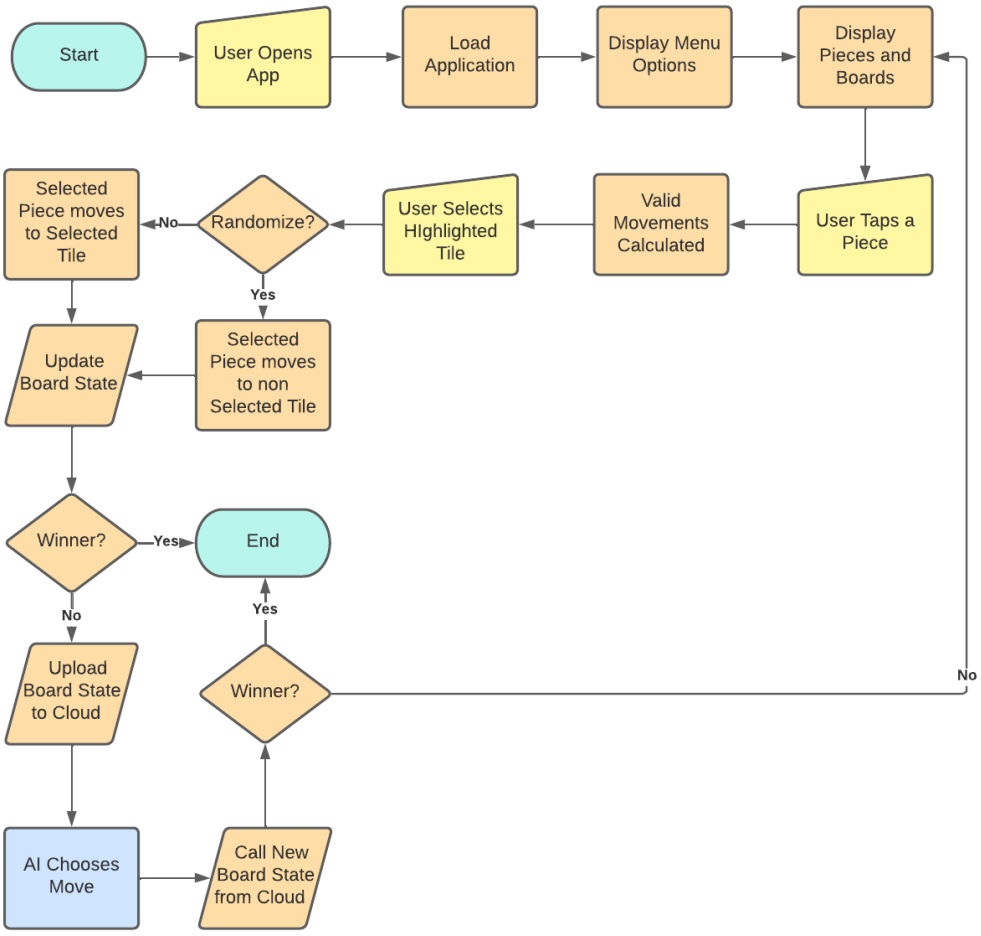

Figure 3: Game Functionality

The game functionality consists of the creation of the app and the in-game rule set. When the user selects a piece by tapping, valid move options will be highlighted on the board. The user should then tap the highlighted space which will move the piece to a desired location or re-tap the selected piece to unselect it. Then, after the user selects a highlighted position and depending on the difficulty setting, the chosen piece will move to the indicated position or to another valid position if possible. The king will always listen to the user and all pieces will listen to the user if the King is in check.

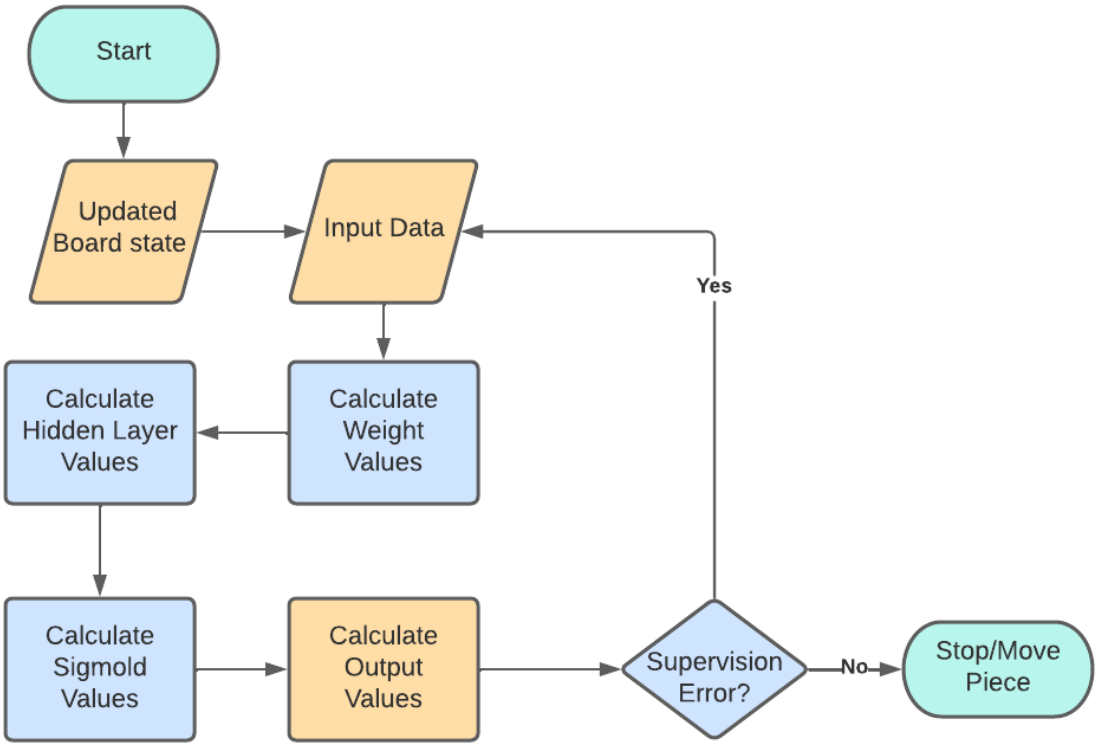

Figure 4: Neural Network for Artificial Intelligence

The moves of the artificial intelligence of the opponent will be performed using a neural network. The software will be developed using Python with the TensorFlow library and trained using a relatively large sample of board instances as supervised learning. It will be uploaded to Microsoft Azure for faster processing. When the user makes a move in single player mode, the board instance will be sent to the program via wireless internet, a decision will be made, and a signal for the opponent's piece's movement will be sent back to the Rogue Chess app to be performed.

7.2. Characteristics

7.2.1. Functional / Performance Requirements

7.2.1.1. Data Size Requirements

The app data size shall not exceed 20Mb.

Rationale: These are system requirements based on sponsor specifications. Smaller data size is easier to download.

7.2.1.2. Randomization Difficulty

The probability of obedience for a selected piece in game will correspond to the chosen difficulty; this will be 10% in easy mode, 30% in medium mode and 50% in hard mode.

Rationale: These are system requirements based on sponsor specifications. This provides variation to the game.

7.2.1.3. Chess Piece Movement

Each respective chess piece shall move according to the standard rules of chess.

Rationale: Chess piece movement must be functional and understandable for a chess game.

7.2.1.4. App Menu Functionality

a) On opening the app, a menu with Single Player, 2-Player, and Difficulty will be displayed in a different menu.

b) Tapping Single Player mode will begin replacement menu with options for Easy, Medium, Hard, and clicking the difficulty will play a game against the artificial intelligence.

c) Tapping 2-Player mode will begin a game where both sets of pieces can be selected.
Rationale: A menu for selecting game modes and difficulty provides the user customization.

7.2.1.5. App Graphics

a) The visual chess pieces and board will be recognizable as a chess game.

b) Tiles will be highlighted to indicate valid moves for selected pieces.

Rationale: These are system requirements based on sponsor specifications.

7.2.1.6. Artificial Intelligence Rating

The Artificial Intelligence should challenge a chess player of an Elo rating of 1300.

Rationale: The Artificial Intelligence should represent a user that could beat novice opponents on regular occasion.

7.2.1.7. Artificial Intelligence Movement

The AI will be able to input a valid piece movement on its turn.

Rationale: The Artificial Intelligence needs to be able to function as a player.

7.2.2. Physical Characteristics

The Rogue Chess does not have any physical requirement.

Rationale: This project is entirely software and has no physical components.

7.2.3. Electrical Characteristics

7.2.3.1. Inputs

a. The presence or absence of any combination of the input signals in accordance with ICD specifications applied in any sequence shall not damage the Rogue Chess App, reduce its life expectancy, or cause any malfunction, either when the unit is powered or when it is not.

b. No sequence of command shall damage the user's phone, reduce its life expectancy, or cause any malfunction.

Rationale: By design, should limit the chance of damage or malfunction by user/technician error.

7.2.3.1.1 External Commands

The Rogue Chess App shall document all external commands in the appropriate ICD.

Rationale: The ICD will capture all moves in a given match.

8.2.3.2. Outputs

8.2.3.2.1. Data Output

The Rogue Chess App does not store data.

Rationale: The Rogue Chess App does not make use of profiles.

8.2.3.2.2. Diagnostic Output

The Rogue Chess App may include a diagnostic interface for error logging.

Rationale: Provides the ability to control things for debugging manually and a way to view/download the piece's movements and obedience.

8.2.3.2.3. Raw Video Output

The Rogue Chess App does not use any video recording devices.

8.2.4. Environmental Requirements

The Rogue Chess App does not have any specific environmental requirements.

Rationale: As the app is a software project, there are not any physical elements that it must adhere to.

8.2.5. Failure Propagation

The Rogue Chess App shall not allow propagation of faults beyond the Rogue Chess App interface.

8.2.5.2. Failure Detection, Isolation, and Recovery (FDIR)

8.2.5.2.1. Built In Test (BIT)

The Rogue Chess App shall have an internal subsystem that will generate test signals and evaluate the Rogue Chess App responses and determine if there is a failure.

8.2.5.2.1.1. BIT Critical Fault Detection

The BIT may be able to detect a critical fault in the Rogue Chess App 95 percent of the time.

Rationale: This is an internal prevention to not damage the user's smartphone.

8.2.5.2.1.2. BIT False Alarms

The BIT may have a false alarm rate of less than 5 percent.

Rationale: This requirement will limit the number of errors to the user and ensure stability.

8.2.5.2.1.3. BIT Log

The BIT may save the results of each test to a log that shall be stored in the application.

Rationale: This assists the developer to monitor failure rates and trends.

8.2.5.2.2. Isolation and Recovery

The Rogue Chess App should provide for fault isolation and recovery by enabling subsystems to be reset or disabled based upon the result of the BIT.

Rationale: This will enable a full reset or shutdown of the application in extreme circumstances.

9. Support Requirements

The Rogue Chess App requires a functioning smartphone running an Android Operating System that has connection to wireless internet. It also requires connection to Microsoft Azure to interact with the artificial intelligence. User's must provide a wireless internet connection and power to the smartphone to interact with the program.

Appendix A: Acronyms and Abbreviations

Below is a list of common acronyms and abbreviations:

BIT	Built-In Test
Hz	Hertz
ICD	Interface Control Document
AI	Artificial Intelligence

Appendix B: Definition of Terms

Elo Rating	The calculated relative skill of players in a zero-sum game of chess.

Rogue Chess

Travis Head
Jose Herrera

INTERFACE CONTROL DOCUMENT

REVISION – 1
FEBRUARY 21, 2022

INTERFACE CONTROL DOCUMENT
FOR
Rogue Chess

PREPARED BY:

_____
Author Date

APPROVED BY:

_____
Project Leader Date

_____
John Lusher II, P.E. Date

_____
T/A Date

Change Record

Rev.	Date	Originator	Approvals	Description
-	02/19/2022	Jose Herrera		Draft Release
1	02/20/2022	Jose Herrera		Revision 1

Table of Contents

List of Tables

List of Figures

10. Overview

The following document will provide a detailed description of the subsystems which will be used for the implementation of the Rogue Chess. The first subsystem is the use of the implementation of the Artificial Intelligence with a neural network. The second, will be a description of how the application will be built. Lastly, a description of the device that will be used to test the Rouge Chess application.

11. References and Definitions

11.1 References

Build your first app : Android Basics : android developers. Android Developers. (n.d.). Retrieved February 21, 2022, from https://developer.android.com/training/basics/firstapp

Carter, D. S. and S. (n.d.). *Tensorflow - Neural Network Playground*. A Neural Network Playground. Retrieved February 21, 2022, from TensorFlow Example

Tutorialspoint . (n.d.). *JAVA Tutorial* . Java - Basic syntax. Retrieved February 21, 2022, from https://www.tutorialspoint.com/java/java_basic_syntax.htm

11.2. Definitions

AI	Artificial Intelligence
App	Application
CPU	Control Processing Unit: portion which retrieves and executes instructions.
	Neural Network: A technique for building program that learns from data.
	Via: by means of.

12. Physical Interface

The physical interface is the android phone which will be used to test the application.

12.1 Weight

152g. There are no weight requirements since this is a programming project.

12.2. Dimensions (inches)

2.74 x 5.61 x 0.31 (width x height x thickness) These are the dimensions needed for the app due to the backgrounds being set to specific dimensions.

13. Thermal Interface

The subsystems written in detail do not require a thermal interface.

14. Electrical Interface

The electrical interface will be provided by the phone used in the testing process.

14.1 Signal Interfaces

The signal interface is a touch bar which will be controlled by a sensor.
The Sensor types include: Accelerator, Barometer, Fingerprint sensor, Gyro Sensor, Geomagnetic Sensor, Hall Sensor, HR Sensor, Proximity Sensor, RGB, and Light Sensor.

14.2. Video Interfaces

The Phone uses a proximity sensor to detect every touch the user does. The proximity sensor detects exactly where the user touched.

Video Playing Format	Video Playing Resolution
MP4, M4A, 3GP, 3G2, WMV, ASF, AVI, FLV, MKV, WEBM	UHD 4K (3840 x 2160) @ 60 fps

Table 1: Video Format/Resolution

14.3. CPU Interface

Exynos: Octa-core (4x2.3 GHz Mongoose & 4x1.6 GHz Cortex-A53).

15. Communications / Device Interface Protocols

The following sections will discuss protocols for communication.

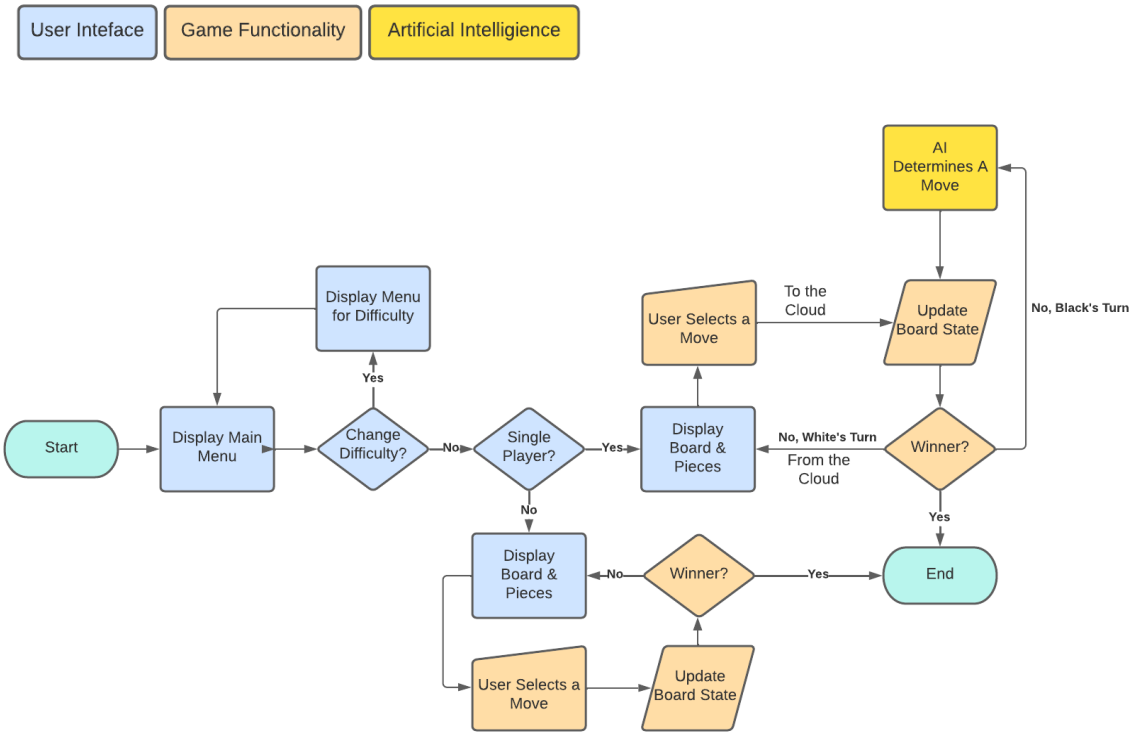

Figure 1: Application Diagram

15.1 Wireless Communications (WiFi) (Samsung Galaxy s7)

802.11a/b/g/n/ac 2.4G+5GHz, VHT80 MU-MIMO

15.2. Video Interface

The video interface will be the built in phone display from the user. The graphics will be created via software in Android Studio.

15.3. User Control Interface

The user control interface will be provided by a web design which will be uploaded to the cloud. The cloud will then allow adjustments to the game.

15.4. Command Line from User Interface

The user will control the commands the app/ game would like to do. The following commands will be provided by a series of functions implemented with Android Studio.

Function	**Command**
Start Game	User selects game mode
Click on piece	User selects piece
Move piece	User selects new position to move
Restart game	Restart button
Back to main menu	Main Menu function

Table 2: User Function/Command

Rogue Chess
Travis Head
Jose Herrera

VALIDATION AND EXECUTION PLAN

REVISION – Draft
21 February 202

	2/14	2/21	2/28	3/7	3/14	3/21	3/28	4/4	4/11	4/18	4/25	Date
System Update 1												2/15/22
Research											Completed	2/22/22
Game Menu Appears											Behind Schedule	2/23/22
Midterm Presentation											Not Started	
Implement a Chess Board											In Progress	
Functional AI												
Functionality of Menus												
System Update 2												
Chess Piece Movement												
AI Training												
AI Rating												
Final Design Presentation												
Final System Demo												
Final Report												

Paragraph #	Test Name	Success Criteria	Methodology	Status	Responsible Engineer
8.2.2.1	Data Size Requirements	Application is under 20Mb.	View file size and validate.	Untested	Jose
8.2.3.1	Randomization Difficulty	90%, 70%, or 50% obedience rate corresponding to Easy, Medium, Hard.	Record obedience successes on 100 moves from each setting.	**TESTED**	Jose
8.2.3.2	Chess Piece Movement	Each type of Piece moves only to respective ruleset.	Place each piece in the center/edge of an empty board and input a move to expected and unexpected valid tiles.	**TESTED**	Jose
8.2.3.4	App Menu Functionality	Main Menu is visible, and each option is selectable. Selecting a game mode starts a match. Difficulty option opens a new menu that saves selected setting.	Open the application and validate main menu is visible. Select each option on the menu.	**TESTED**	Jose
8.2.1.5	App Graphics	Menus, board, pieces, and highlighted tiles are all visible when needed.	Open the application and validate visibility. Start a match in each mode and select/deselect pieces.	**TESTED**	Jose
8.2.1.6	Artificial Intelligence Rating	Elo Rating of at least 1300.	Run the AI against another chess program for 10 matches at ratings of 900, 1100, 1200, 1300, 1400, and 1500.	Untested	Travis
8.2.1.7	Artificial Intelligence Movement	Artificial Intelligence can input a valid move on its turn.	Input a board state and run the AI with a written output.	**TESTED**	Travis

Rogue Chess
Jose Herrera

GRAPHICS AND USER INTERFACE SUBSYSTEM REPORTS

REVISION – Original
30 April 2022

GRAPHICS AND USER INTERFACE SUBSYSTEM REPORTS FOR Rogue Chess

PREPARED BY:

_____
Author Date

APPROVED BY:

_____
Project Leader Date

_____
John Lusher II, P.E. Date

_____
T/A Date

Rev.	Date	Originator	Approvals	Description
1	04/30/2022	Jose Herrera		Original

Table of Contents

List of Videos

List of Figures

1 Introduction

The purpose of this document is to present the development of a Rogue Chess App, an app of a new chess variation for Android products. This chess variance will occasionally ignore the instructions of the user and choose a random valid movement choice for the chess piece chosen. The system is broken down into two sub categories the game interface and the user interface, each of which have been designed and tested. With all of the subsystems working properly, the implementation shall be quick and easy to handle into the system specified in the previous reports.

2. Graphics Subsystem Report

2.1 Subsystem Introduction

The graphics subsystem is designed for the app's cosmetics in order to have an appealing application. The graphics are made in Android Studios with the intention of having a soothing feeling while playing the game of chess. The button interface should easily guide the user into selecting the proper modes to play the game. There are also highlighting features to help the user select the proper movements and win the game.

2.2 Buttons

Figure 1: Main Menu

Figure 2: Single Player Menu

Figure 3: Pawn Promotion Menu

The buttons are designed to show what the player is selecting by having the selection displayed on the button. In Figure 1, the main menu is displayed with the game mode selection. In Figure 2, the difficulty menu is displayed with three different difficulties, easy, medium, and hard. Lastly, in Figure 3, the pawn promotion appears when a piece reaches to the other side for a pawn promotion of their choosing, queen, bishop, rook, or knight. As well, as the undo button appears throughout the game for whenever, a player would like to undo the move they made prior to the next player making their move.

2.3 Design

Figure 4: Game Over

Figure 5: Game Design

The design of the game is based on a regular chess game with the black pieces on one side and the white pieces on the other. There is a nature background in order to have a soothing appeal to the player as they play the game. The chess board is still the ordinary color as a regular chess game for less confusion. When one of the teams beats the king, a “Game Over” text appears in the middle of the screen to indicated that the game is over. The game setup is still the same setup as a regular chess game.

Figure 6: Main Menu Error

Figure 7: Chess Board Error

Although the graphics work perfectly when the phone is vertical. I did run across a few issues when the phone is horizontal. The main menu seems to partially disappear, causing the buttons to go away. This may be due to the constraint settings I have implemented in Android Studios. Changing the layout view to a linear layout may fix the issue due to the linear layout allowing the app to freely move every time the phone dimensions are changed. However, for the purposes of this app, we will be using the dimension 2.74 x 5.61 x 0.31 as mentioned in the ICD Dimensions portion. In Figure 7, I ran across the issue of an unplayable black piece board due to the design of the “Rogue Chess” banner hanging down. As a result, a function that I am missing may be able to make the banner disappear in order for the black pieces to be playable again in horizontal mode. Both of these can be avoided by turning off the auto rotate feature on the phone. The gid view allows the chess board to appear.

2.4 Highlights

Figure 8: Kill Highlight

Figure 9: Movement/Selected

Figure 10: King in Check

Highlights are designed to help navigate the user into selecting the next move legally. In Figure 7, the pawn piece (teal) is in position to “kill” the bishop piece. Therefore, for any piece that has a red highlight, that means the piece is able to be taken over by the player. In Figure 8, the teal piece means the piece is selected for movements which are indicated by the darker blue pieces for the legality of the game since each piece has a certain move, they are able to make. Lastly, when the King is in check (aka. In danger), the King piece is highlighted in a dark red color to indicate that the piece is in danger and may want to prepare for a movement if allowed.

2.5 Subsystem Conclusion

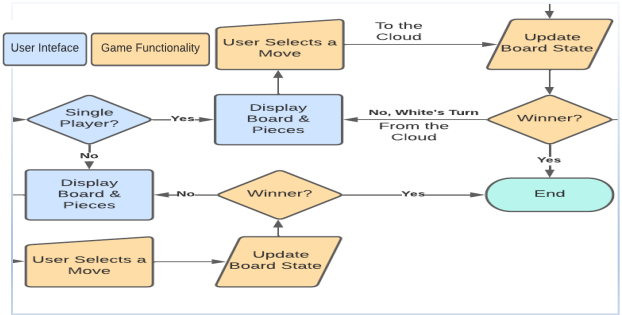

Figure 11: Game Functionality Flow Chart

The game interface is successful despite the minor detailed horizontal flaw. The backgrounds are displayed perfectly with the “fit XY” size functionality. The buttons do not move from their placement due to the constraint layout. All the buttons are placed and colored for easy access to the user. Lastly, the highlights appear bright for easy viewing and are colored coded to the specificity of a particular functionality as mentioned above. The settings are visible and are direct without any confusion.

3. User Interface Subsystem Report

3.1 Subsystem Introduction

The user interface subsystem is designed for the app's functionality in order for the user to have a playable game. The functions are made in Android Studios with the intention of creating an application to play the game of chess with a few misdirection. The button interface should be able to switch in between screens and be able to start the game. The chess pieces should only move to a highlighted area only when it is the user's turn and should not move to any illegal (non-highlighted) areas. The chess pieces should also be able to "kill" a piece is the piece is highlighted red and if the piece is "killed" then the piece should disappear. As well as, if a piece reaches to the end, then the pawn promotion menu should pop up and replace the pawn to the user's choosing. Lastly, the last placement of the user should be saved for an undo usage button to work.

3.2 Buttons

Video 1: Button Demo https://youtube.com/shorts/cEms019yF9w?feature=share

As shown in Video 1 the buttons work as intended. The single player button switches to the difficulty menu where the different modes are displayed. The mode buttons start the designated difficulty game for the user to play. When the difficulty mode menu is displayed the user has the option to go the main menu by tapping the home button. When the user selects the player vs player button then the game automatically starts a game for the face-to-face game. During the game the user is allowed to undo a move before the next player makes a move by pressing the undo button.

3.3 Board

Video 2: Clicking Function https://youtube.com/shorts/-ISQGY6DNTU?feature=share

Presented in Video 2 is the demo of the user trying to select a random area that is not a legal move however, the piece does not move due the functions created in android studio. The piece only moves when the box is highlighted is a dark blue color indicating the legal spaces. The piece "kills" another piece that is highlighted in red and the piece that is "killed" disappears as it should.

3.4 Chess Pieces

Video 3: Chess Demo https://youtu.be/WVrvMzoAAN4

Figure 12: Pawn Promotion

Figure 13: Chess Piece Movement

In Video 3, the piece is promoted due to the piece reaching the side of the opponent without being “killed” however, prior to being promoted a pawn promotion menu appears with the four pawn options that the piece may be promoted to. As shown in Figure 11, the pawn changed to the selected option and now has the movements the new pawn acquires. In Figure 12, the movement of the pieces is shown after the fact showing the functionality of the pieces. The video also appears to show when the King is highlighted in red meaning the King is in check. The piece which has the King in check is allowed to “kill” the King and end the game, which is when the “Game Over!” sign appears.

3.5 Subsystem Conclusion

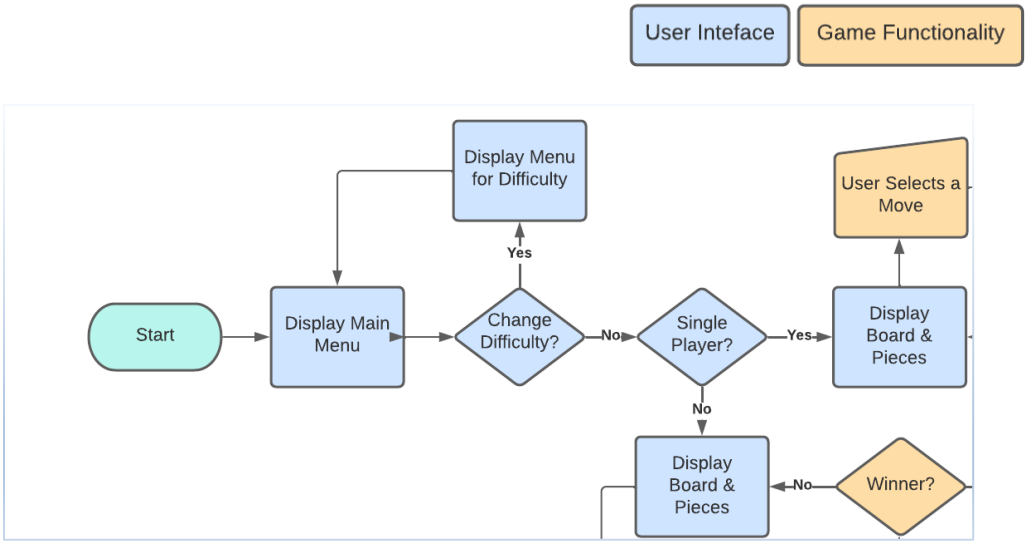

Figure 14: User Interface Flow Chart

The user interface subsystem works perfectly without any errors. The built-in invalid move prevention works as intended and does not allow any pieces to move outside of their valid movements. The screens change based on the button that is pressed without any glitches or slow movement time. The app is very responsive to every move the user makes. The pieces move as intended for the purposes of this app. All the menus appear and are very visible to the user.