Mini Project Report (BCS-554)

On

"CHESS GAME UI"

Submitted in partial fulfillment for award of

BACHELOR OF TECHNOLOGY

Degree

In

COMPUTER SCIENCE & ENGINEERING

2024-25

Under the Guidance of:	Submitted By:
Mr. Vineet Srivastava	Dhairya Singh (2200330100088)
Assistant Professor	Naman Jain (2200330100141)
Ms. Chanchal Jayant	Lakshya Saxena (2200330100131)
Assistant Professor	

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

RAJ KUMAR GOEL INSTITUTE OF TECHNOLOGY

DELHI-MEERUT ROAD, GHAZIABAD

Affiliated to Dr. A.P.J. Abdul Kalam Technical University, Lucknow

SYNOPSIS

Future employment prospects for youngsters who learn **Python Programming** might be very diverse. With Python being used in many sophisticated disciplines, many new employment possibilities have emerged and will continue to emerge in the future.

It would strengthen their academic performance and their chances of securing aprosperous future profession. Even if your children decide not to embrace computer programming in the future, training them in Python today will enable them to develop life skills.

These days, creating video games may be quite lucrative. It can also be utilized as an educational and advertisement tool. Game creation may be really enjoyable and involves a lot of math, logic, physics, AI, and other subjects.

Python can also be used to make games, with features like collision detection, music, backgrounds, and sprites among many others.

PyGame is one of the greatest modules for programming games in Python and is used for game development. We can use pyGame module for creating a chess engine.[1]

Chess is a game played by two people on a single board. The goal is to capture the other person's king by strategically moving our pieces. We win the game by placing the other person's king in a position where it cannot move and will be captured. Chess has been played for many years and requires a lot of forethought and forward planning to win.

KEY FEATURES:

Interactive Chessboard:

A. Drag-and-drop functionality for smooth piece movement.

B. Real-time validation of legal chess moves to ensure rule compliance.

Game Modes:

A. Human vs. Human: Play against another person on the same device.

B. Human vs. AI: Challenge an AI opponent powered by the Stockfish chess engine.

AI Integration:

A. Uses the Stockfish engine for calculating optimal moves in Human vs. AI mode.

B. Configurable AI decision-making time for flexible gameplay difficulty.

Time Control:

A. Offers multiple preset time controls (e.g., 30+10, 10+2, 5+2).

B. Dynamic countdown for each player's remaining time during their turn.

User-Friendly Interface:

A. Intuitive buttons for starting a game, offering a draw, resigning, or settings.

B. Configurable settings menu for adjusting game mode and time control.

TECHNOLOGICAL STACK:

Python Programming Language: Python serves as the primary language for developing the Chess engine due to its simplicity, versatility,and extensive library support.

PyGame Library: Python PyGame library is used to create video games. This library includes several modules for playing sound, drawing graphics, handling mouse inputs, etc. It is also used to create client-side applications that can be wrapped in standalone executables.

Stockfish: Stockfish is the strongest chess engine available to the public and has been for a considerable amount of time. It is a free open-source engine that is currently developed by an entire community. Stockfish was based on a chess engine created by Tord Romstad in 2004 that was developed further by Marco Costalba in 2008. Joona Kiiski and Gary Linscott are also considered founders.[2]

Chess Library: The chess module is a pure Python chess library with move generation, move validation and support for common formats. We can play chess with it. It will help us to move the king queen, pawn, bishops and knights. We just have to import the chess library and with it, we can play chess. When we will import the chess library we have to call the function named board so that we can see the status of the chess board. [3]

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	SYNOPSIS	**ii**
	LIST OF FIGURES	**vii**
	LIST OF ABBREVIATIONS	**viii**
1.	**INTRODUCTION**	1
1.1	OBJECTIVE	1
1.2	BACKGROUND	2
1.3	MOTIVATION	4
1.4	PROBLEM STATEMENT	5
2.	**HARDWARE AND SOFTWARE REQUIREMENTS**	6
2.1	SOFTWARE REQUIREMENTS	6
	2.1.1 PYGAME LIBRARY	6
	2.1.2 STOCKFISH	6
	2.1.3 CHESS LIBRARY	6
	2.1.4 PYTHON IDE	7
2.2	HARDWARE REQUIREMENTS	7
3.	**SOFTWARE REQUIREMENTS SPECIFICATION**	8
3.1	PURPOSE	8
3.2	SCOPE	8
3.3	FUNCTIONAL REQUIREMENTS	8

3.4 NON-FUNCTIONAL REQUIREMENTS 10

4. APPLICATION ARCHITECTURE 11

4.1 INPUT PART 13

4.2 ALPHA-BETA ALGORITHM 13

4.3 METHODOLOGY FOR IMPLEMENTATION 15

4.3.1 CHESS GUI METHOD 15

- OBJECT ORIENTED PROGRAMMING 15
- PROCEDURAL PROGRAMMING DESCRIPTION 15
- EVENT DRIVEN PROGRAMMING 15
- AI INTEGRATION WITH NEURAL NETWORK 16
- WEB IMPLEMENTATION WITH FLASK/DJANGO 16

5. SOFTWARE FEATURES 17

5.1 FEATURES 17

5.1.1 GUI (GRAPHICAL USER INTERFACE) 17

5.1.2 SIDEBAR MENU 17

5.1.3 AI INTEGRATION 18

5.1.4 MISCELLANEOUS FEATURES 18

5.1.5 FUTURE POTENTIAL ENHANCEMENTS 18

6. PROJECT SNAPSHOTS 20

6.1 SAMPLE IMAGES 20

6.2 SAMPLE WORKING 21

6.3 AI VS HUMAN 21

6.4 RESIGN WINDOW 22

6.5 DRAW WINDOW 22

7. LIMITATIONS 23

7.1 USER EXPERIENCE 23

7.2 FUTURE CONSIDERATIONS 23

7.3 COMPLEXITY OF ALGORITHM 24

7.4 CHEATING IN CHESS 24

7.5 FUTURE DIRECTIONS 25

8. FUTURE SCOPE 26

9. CONCLUSION 27

10. REFERENCES 28

LIST OF FIGURES

CHAPTER NO.	TITLE	PAGE
1	Figure. 1.1 Persian Rule (14 th century AD)	2
1	Figure. 1.2 Lewis Chessmen	3
3	Figure. 3.1 Chess with traditional timer	9
4	Figure 4.1: Proposed System	12
4	Figure 4.2: Algorithm for engine evaluation	13
4	Figure 5.1: Stockfish	18
5	Figure 5.2: Chess Board Layout	19
5	Figure 5.3: AI vs Human	19
6	Figure 6.1: Sample Image	20
6	Figure 6.2: Sample Working	21
6	Figure 6.3: AI vs Human	21
6	Figure 6.4: Resign Window	22
6	Figure 6.5: Draw Window	22
7	Figure 7.1: Limitations	25
8	Figure 8.1: Future scope	26

LIST OF ABBREVIATIONS

1. GUI: Graphical User Interface
2. H VS M: Human vs. Machine mode
3. HCI: Human-Computer Interaction
4. AI: Artificial Intelligence
5. PGN: Portable Game Notation
6. FPS: Frames Per Second
7. CR: Chess Rules
8. SDLC: Software Development Life Cycle
9. MVC: Model-View-Controller
10. OOP: Object-Oriented Programming
11. IDE: Integrated Development Environment

CHAPTER 1

INTRODUCTION

Chess is an abstract strategy board game played on an 8x8 grid checkered board. The game is played by two players with each player beginning with 16 pieces (one king, one queen, two rooks, two knights, two bishops, and eight pawns) of contrasting colors. Player alternate turns to move their pieces with white moving first. The objective is to checkmate the opponent’s king by placing it under an inescapable threat of capture.

Played by millions around the world, chess is believed to have originated India around 7th century. It is one of the most popular board games and has attracted people from all walks of life such as scholars, warriors, strategists, men, women,and children. The rules and appearance of pieces reached today’s standard in the early 19th century.
With the development of computers and the internet, the way chess games played has also changed to a certain degree. Though World Championship are still played on a physical board, chess is now mostly played on display screens over internet. Likewise, our Chess game is a desktop application. It emulates experience of a real world scenario via GUI interface to which players can interact move their pieces. The game is engineered to follow the basic rules of chess i.e., the pieces only move according to valid moves for that piece.

1.1 OBJECTIVE

The main goal of this chess application project is to design a chess application that is fun, effective and simple to use. The project targets creating modes for Human vs Human and Human vs Machine supported by AI . The application offers clearness in its use through features like time limit settings, validating of a player’s turns at the point of making a move, and simple graphics. Other features, for example, resignation, draw, a new game, and a game timer improve the usability of the application. Moreover, the game message pop-ups have been designed to give an insight of certain phase/s in the game which are important to know. The application also offers a change to the mouse cursor to the ‘waiting for shot’ mode. Considering different program functions that are able to be added later, the application aims for enhancement.

1.2 BACKGROUND

The history of chess can be traced back nearly 1,500 years to its earliest known predecessor, called chaturanga, in India; its prehistory is the subject of speculation. From India it spread to Persia, where it was modified in terms of shapes and rules and developed into Shatranj. Following the Arab invasion and conquest of Persia, chess was taken up by the Muslim world and subsequently spread to Europe via Spain (Al Andalus) and Italy (Emirate of Sicily). The game evolved roughly into its current form by about 1500 CE.Romantic chess" was the predominant playing style from the late 18th century to the 1880s. Chess games of this period emphasized quick, tactical maneuvers rather than long-term strategic planning. The Romantic era of play was followed by the Scientific, Hypermodern, and New Dynamism eras. In the second half of the 19th century, modern chess tournament play began, and the first official World Chess Championship was held in 1886.

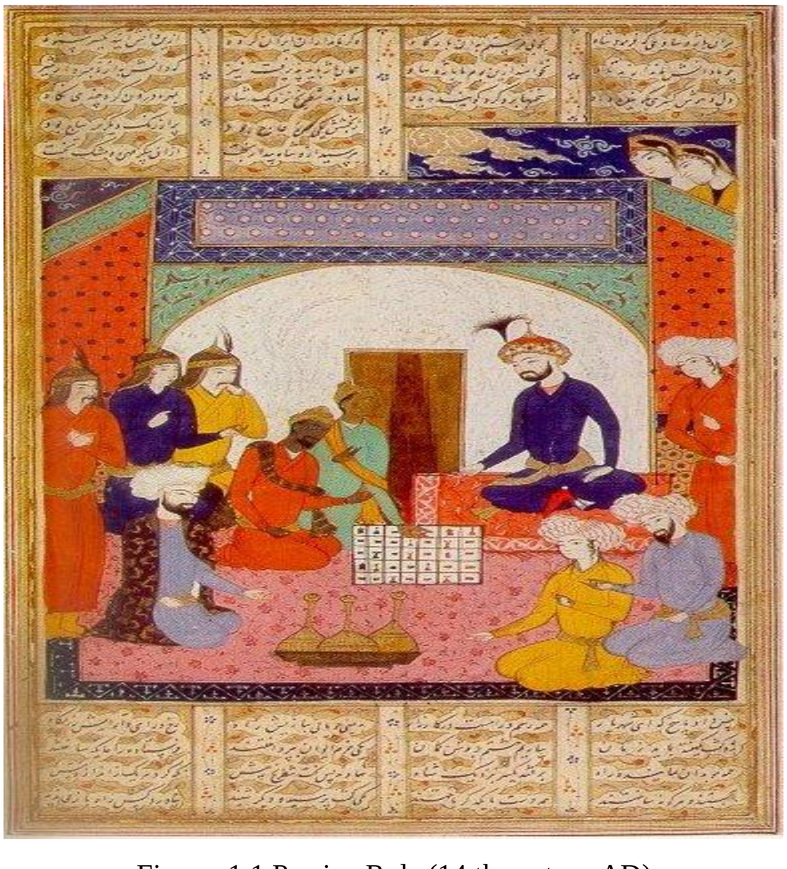


Figure. 1.1 Persian Rule (14 th century AD)

The 20th century saw great leaps forward in chess theory and the establishment of the World Chess Federation. In 1997, an IBM supercomputer beat Garry Kasparov, the then world chess champion, in the famous Deep Blue versus Garry Kasparov match, ushering the game into an era of computer domination. Since then, computer analysis – which originated in the 1970s with the first programmed chess games on the market – has contributed to much of the development in chess theory and has become an important part of preparation in professional human chess. Later developments in the 21st century made the use of computer analysis far surpassing the ability of any human player accessible to the public. Online chess, which first appeared in the mid-1990s, also became popular in the 21st century. In recent years, online chess through the use of internet chess servers have exploded in popularity. Internet chess servers have existed since 1992 with the creation of the subscription service Internet Chess Club,but today the majority of top-level players have moved to freemium websites like Chess.com (founded in 2007) and, to a lesser extent, free website Lichess (launched in 2010). These websites feature quick pairing systems and site-specific ratings for bullet, blitz, rapid, and/or classical time controls. [4]

They also include additional features such as puzzles, engine analysis, databases, user-created libraries, community blogs and forums, news and articles, video lessons, and more. They also host tournaments where top grandmasters compete for prizes, creating an around-the-clock chess media ecosystem. Chess engines have also radically transformed, incorporating new technologies into their move search and evaluation functions. Previously, dominant computers such as Rybka (released in 2005), Stockfish (2008), Houdini (2010), and Komodo (2010) used a hand-crafted evaluation using variables such as space and piece mobility. In 2017, chess engine AlphaZero won a controversial match against Stockfish using a relatively novel neural network system. Since then, all of the top engines, including Stockfish, Komodo Dragon, Leela Chess Zero (2018), and Torch (2023), have incorporated efficiently updatable neural networks (NNUEs) into their evaluation functions.

Figure. 1.2 Lewis Chessmen

1.3 MOTIVATION

The motivation behind building a Chess UI powered by Stockfish lies in the desire to create an engaging and intelligent platform that caters to chess enthusiasts of all skill levels. Chess, being a timeless and widely loved game, can become even more accessible through a user-friendly interface that allows players to enjoy, learn, and improve their skills. By integrating Stockfish, one of the most powerful open-source chess engines, the project leverages advanced AI to provide challenging gameplay and precise analysis, enhancing the overall experience. This UI also serves as an interactive learning platform, enabling players to explore move suggestions, understand strategies, and refine their skills with real-time feedback. Additionally, the project showcases technical expertise in designing seamless interfaces and integrating AI while fostering creativity in game design. It is a perfect blend of technology and creativity, aimed at offering both entertainment and education for chess enthusiasts.

1.4 PROBLEM STATEMENT

Chess is a popular game of strategy and skill, but players have a hard time finding what they want on existing platforms because some are inaccessible, others are not customizable, and still, others have uninteresting interfaces. Players report being constrained by a poor AI opponent, only a few available modes of play, or primitive timers. Notably, there is a gap that needs to be fulfilled by a suitable platform which is capable of addressing the needs of a casual chess player and a brutal chess enthusiast, perfecting the core mechanics of strategic gameplay, implementing real-time action, and creating the option for a timed tournament. Solving these problems requires an application that is both flexible and scalable with simple user interaction, a smart AI, and a well-designed rule engine. This project aims at filling that gap by developing an innovative chess application to meet the requirements of different classes of players. The problem that this chess application project addresses is systemic in nature; it pertains to the existing bottlenecks that are centered on the current digital chess applications.

It is true that chess is a game of wits played across the globe, but not many applications are able to provide the right 'experience' to the users. Barriers to engagement tend to cause much frustration with installation protocols, membership and internet access or the lack thereof, as casual users find it too cumbersome. Also, some consumable structures barely support adjusting to users' requests, for example, more sophisticated AIs, individual time limits, or even user-friendly designs. AI in chess, for instance, research by the Stockfish engine has received much focus on performance and strategy, although it often neglects the human factors such as engaging or feedback mechanisms.

Players, particularly novices, have difficulties in comprehending the movements or moves and strategies because there are no or few available educational aids, while advanced users are unhappy about general AI activity not being complex enough. The aim of the present project is to deal with such problems by integrating strong AI with a simpler interface and scalable design to accommodate the diversity in users and improve the chess-playing experience.

CHAPTER 2

HARDWARE AND SOFTWARE REQUIREMENTS

Computer chess includes both hardware (dedicated computers) and software capable of playing chess. Computer chess provides opportunities for players to practice even in the absence of human opponents, and also provides opportunities for analysis, entertainment and training. Computer chess applications that play at the level of a chess grandmaster or higher are available on hardware from supercomputers to smart phones. Standalone chess-playing machines are also available. Stockfish, Leela Chess Zero, GNU Chess, Fruit, and other free open source applications are available for various platforms. The first chess machines capable of playing chess or reduced chess-like games were software programs running on digital computers early in the vacuum-tube computer age (1950s). The early programs played so poorly that even a beginner could defeat them. Within 40 years, in 1997, chess engines running on super-computers or specialized hardware were capable of defeating even the best human players. By 2006, programs running on desktop PCs had attained the same capability. In 2006, Monty Newborn, Professor of Computer Science at McGill University, declared: "the science has been done". Nevertheless, solving chess is not currently possible for modern computers due to the game's extremely large number of possible variations.

2.1 SOFTWARE REQUIREMENTS

2.1.1 PyGame Library: Python PyGame library is used to create video games.

2.1.2 Stockfish: Stockfish is the strongest chess engine available to the public and has been for a considerable amount of time. It is a free open-source engine that is currently developed by an entire community. Stockfish was based on a chess engine created by Tord Romstad in 2004 that was developed further by Marco Costalba in 2008. Joona Kiiski and Gary Linscott are also considered founders.

2.1.3 Chess Library: The chess module is a pure Python chess library with move generation, move validation and support for common formats. We can play chess with it. It will help us to move the king queen, pawn, bishops and knights. We just have to import the chess library and with it, we can play chess. When we will import the chess library we have to call the function named board so that we can see the status of the chess board.

2.1.4 PYTHON IDE: There are lots of IDEs for python. Some of them are PyCharm, Thonny, Ninja, Spyder etc. Ninja and Spyder both are very excellent and free but we Used the python idle which is downloaded along with the python software. For more advanced development, PyCharm is one of the most popular and feature-rich IDEs available. It offers both a free Community version and a paid Professional version with additional features like support for web development frameworks such as Django and Flask. PyCharm provides intelligent code analysis, debugging tools, testing frameworks, and seamless version control integration, making it a great choice for professional developers and larger Python projects. However, it can be resource-heavy and may feel overwhelming for beginners.

Another great IDE for beginners is **Thonny**, which offers a simple and clean interface, making it ideal for educational purposes. It includes a built-in debugger, variable explorer, and an easy-to-use interface, allowing newcomers to quickly understand how Python works. However, it lacks the advanced features necessary for larger, professional-scale projects.

2.2 HARDWARE REQUIREMENTS

A. Laptop with 8 GB RAM or above.

B. Standard keyboard and mouse for user interactions.

C. Monitor with a minimum resolution of 1280x720 for a clear view

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

3.1 PURPOSE

The purpose of this document is to define the requirements for the development of a chess game GUI. This system is designed to automate the process of playing chess on computer.

3.2 SCOPE

The Chess game GUI will include the following features:

* Game mode-Human vs Human, Human vs AI.

* GUI for engaging experience.

* AI integration (Stockfish engine).

* Proper game state management.

3.3 FUNCTIONAL REQUIREMENTS

A.User Interface(UI):

* The presence of an interactive graphical user interface (GUI) with chess pieces and menus.

* Button menus contain functions for new game initiation, resignation, draw offer and mode change.

B. Game Modes:

* Human vs Human – 2 players can play versus each other locally on the same device.

* Human vs AI – It is possible to compete against an AI engine (Stockfish for example), which operates as an intelligent game player

C. Timer and Time controls:

* There is a timer for both opponents which acts as a countdown for the time left for them.

D.AI Integration:

* Incorporating the Stockfish engine so the AI can make moves in Human vs AI mode
* The AI must option available and should be able to make an action based on the board position in pre-defined time duration.

Figure. 3.1 Chess with traditional timer

3.4 NON-FUNCTIONAL REQUIREMENTS

A.Performance:

*Application should run smooth at 60 FPS to deliver smooth gaming and responsiveness

*AI movements should be calculated within a reasonable time, say in a few seconds to maintain a smooth user experience.

B.Usability:

*User interface has to be intuitive and easily navigable for both first-time and advanced users.

C.Reliability:

The system shall be available at least 50% of the time during working hours.

Timers must work, and pop-ups must disappear when they're done.

D.Usability:

The GUI shall be intuitive and user-friendly.

E.Compatibility:

The system shall be compatible with common operating systems . The application should be compatible with Windows (for Stockfish integration).

F.Scalability:

The game should be designed to allow future additions, such as saving/loading game states, enhancing the AI, or adding multiplayer over the network.

G. Constraints:

The system is constrained by the hardware specifications of the camera and processing capabilities of the host machine.

H. Maintainability:

*The code should be well-commented and organized to ensure easy maintenance and updates.

*The project should be modular to facilitate future feature additions or bug fixes.

CHAPTER 4

APPLICATION ARCHITECTURE

The Chess Game application begins with the instantiation of the ChessGame class, which initializes the necessary components for the game. The pygame library is used to create a display window (1000x800 pixels) and sets up a chessboard and menu using surfaces and rectangles. The chess piece images are loaded from a specified directory and scaled to fit each square on the board. Once initialized, players can start to set up the game. They can choose between Human vs Human or Human vs AI modes from the menu. Time controls are adjustable through a dropdown menu, and the chessboard is initialized using the python-chess library to manage the game's state. The application enters a continuous loop, which is the heart of the game's operation. In this loop, events such as mouse clicks and drags are processed, and the board and UI are updated accordingly. The screen is refreshed at 60 frames per second (FPS) to ensure smooth gameplay. When a player makes a move, they can interact with the pieces by clicking and holding to drag them to a valid square, and upon releasing the piece, the move is validated using python-chess. The move gets processed and reflected on the board. If the game is set to Human vs AI mode, the Stockfish engine determines the best move for the AI after the player's turn. The AI makes its move, and the game state is updated accordingly.

It monitors each player's time in order to manage the flow of the game. The timer starts each time when a player's turn begins. The remaining time in terms of minutes and seconds is also indicated for that player. It includes options for the players in its menu to initiate a new game, make a draw offer, or resign. They can change between game modes and control the time parameters. Pop-up messages indicate to players when they resign or draw offers, which appear only briefly to alert them to what's happening. The game is concluded if a player resigns, a draw is proposed and accepted, or checkmate or stalemate occurs on the board. At the end of the game, players can play the game anew using the "New Game" button. All the resources used such as the AI engine are cleaned up and closed when their job is done. The application, through the whole game, has used event-driven programming to handle all user inputs and updates in real time. It has used the python-chess library to enforce rules of chess and check legality.

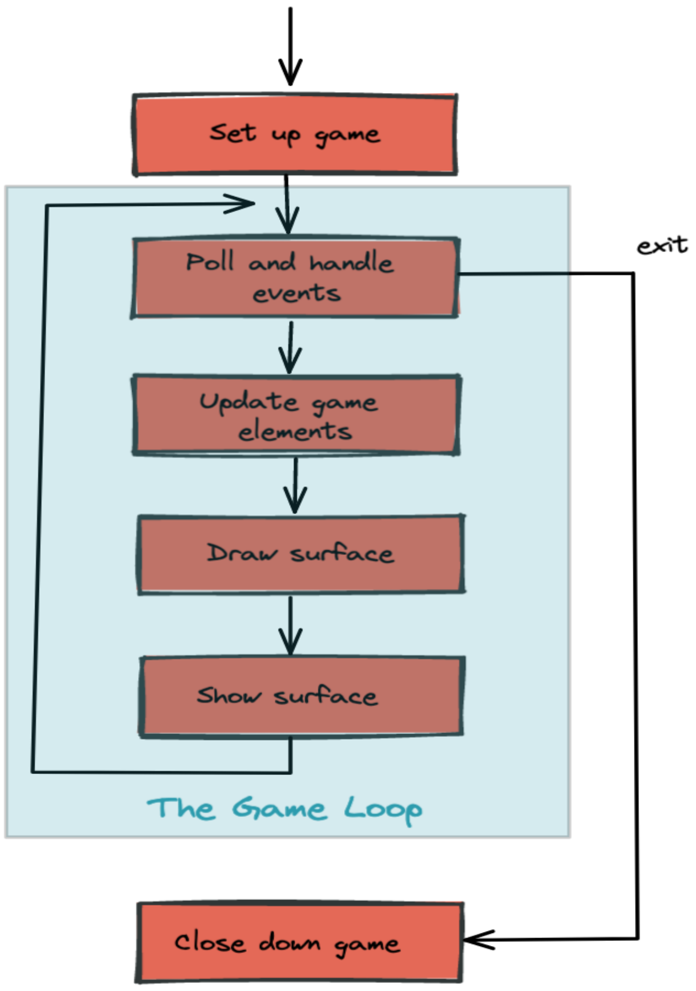


Figure. 4.1 Proposed System

4.1 INPUT PART

Input part is prerequisite for chess engine like path for stockfish and proper moves of chess through mouse. Ask for time controls and . Moves are interpreted and algorithm is implemented to find best moves.

4.2 ALPHA-BETA ALGORITHM

The **Alpha-Beta** algorithm (Alpha-Beta Pruning, Alpha-Beta Heuristic) is a significant enhancement to the minimax search algorithm that eliminates the need to search large portions of the game tree applying a branch-and-bound technique. Remarkably, it does this without any potential of overlooking a better move. If one already has found a quite good move and search for alternatives, **one** refutation is enough to avoid it. No need to look for even stronger refutations. The algorithm maintains two values, alpha and beta. They represent the minimum score that the maximizing player is assured of and the maximum score that the minimizing player is assured of respectively.

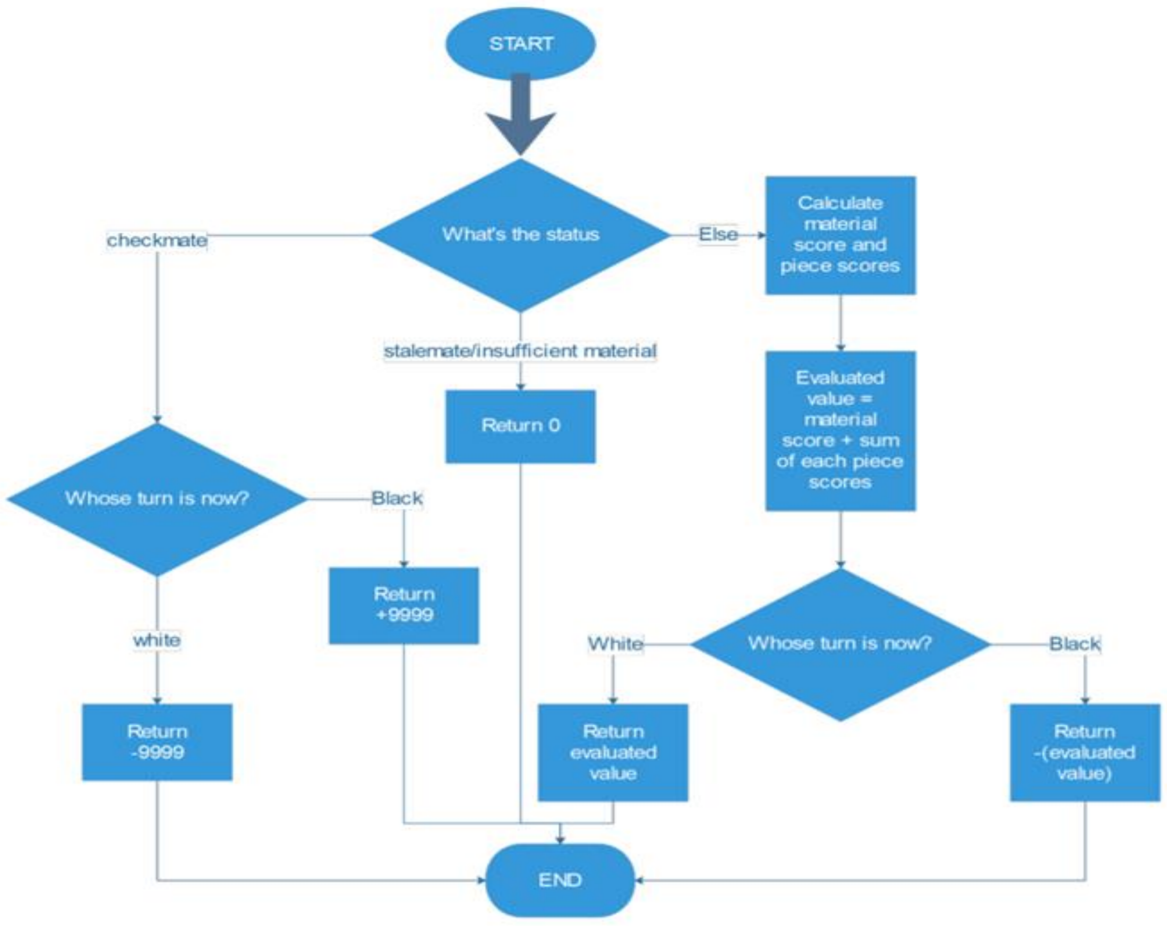

START
What's the status
checkmate
Else
Calculate material score and piece scores
stalemate/insufficient material
Return 0
Evaluated value = material score + sum of each piece scores
Whose turn is now?
Black
Return +9999
white
Return -9999
White
Whose turn is now?
Black
Return evaluated value
Return -(evaluated value)
END

Figure. 4.2 Algorithm for engine evaluation

The flowchart represents a decision-making process in the assessment of chess positions, Key Steps in the Algorithm:-

A. Game Status Evaluation

The process starts by checking whether the game has ended in a checkmate, stalemate, or has insufficient material. There are possibilities:

- **Checkmate**: A decisive outcome where one player wins.
- **Stalemate/Insufficient Material**: The game is terminated, and the algorithm assigns a score of 0.

If the game has ended, there is no need to evaluate further moves. This prioritization ensures that the AI immediately recognizes the end of the game, saving computational resources and ensuring quicker decision-making.

B. Checkmate Handling

Checkmate results in a score based on whose turn it is:

- **White's Turn**: A negative score of -9999 means White has lost.
- **Black's Turn**: A positive score of +9999 indicates a win for Black.

This step ensures that checkmate is given the highest priority, with extreme scores indicating the decisive nature of the result. The engine doesn't waste time evaluating other positions once a checkmate condition is met.

C. Material and Position Scoring

For ongoing games, the algorithm calculates:

- **Material Score**: The total value of remaining pieces based on their standard values (e.g., pawns, rooks, queens).
- **Positional Score**: A strategic estimate of how well the pieces are positioned and controlling the board. Total score is their sum.

This reflects the combination of material advantage (e.g., more pieces or higher-value pieces) and positional advantage (e.g., controlling the center, piece coordination). These factors together give the engine a well-rounded assessment of the game.

D. Turn Adjustment

The algorithm takes into account whose move it is:

- **White's Move**: The returned value of the evaluation is simply returned as-is.
- **Black's Turn**: The engine gives the **negative** of the scored value.[5]

4.3 METHODOLOGY FOR IMPLEMENTATIONS

4.3.1 CHESS GUI METHOD:

These are another methods to implement chess GUI:-

4.3.1.1 Object-Oriented Programming (OOP) :-

Description: Object-Oriented Programming can be used to break down your code into more manageable,self-contained classes. For example, you could have a Board class, a Player class for human and AI players, and a GameController class to handle the game flow.

Benefits :-

- Modularity: Easily extendable by adding more features like multiplayer or custom rules Maintainability: Easier to maintain and debug due to clear class responsibilities.
- Challenges: Requires more upfront planning in terms of class design.
- Can be overskill for simple projects, increasing complexity unnecessarily.

4.3.1.2 Procedural Programming Description :-

In procedural programming, application flow is controlled by Functions. Each function has a certain function to perform and game state is maintained using global variables or simple data structures.

Advantages :

- Simple: Easier to implement for small projects or prototypes.
- Quick Setup: Not so time-consuming in terms of design compared to object-oriented .

Challenges :

- Scalability: Becomes difficult to manage as the project grow and it leads to messy code.
- Maintenance: Introducing new features without interfering with the existing code is not an easy task.

4.3.1.3 Event-Driven Programming with State Machines Description :-

This method uses states to handle different phases of the game (for example, “Idle”, "Dragging Piece", "AI Thinking"). Every state deals with particular user actions or system events. The game can easily move between these states depending on the player's actions.

Advantages :

- Clarity: Makes the game flow clear and well-defined.
- Efficient State Management: Provides a clean way to handle different game state

Challenges :

- Overhead: Designing and managing state transitions can become complex for more sophisticated games.
- Initial Setup: Requires careful planning of states and events.

4.3.1.4 AI Integration with Neural Networks Description :-

Developing an advanced AI player through the integration of a neural network or reinforcement learning algorithm like AlphaZero. Libraries such as TensorFlow or PyTorch could be used in developing the AI.

Advantages :

- Advanced AI: Trains and learns, adjusting the strategy over time.
- More effective AI opponent that grows over time.

Difficulty Level: Computational Resources: Extensive computational power needed. Complexity: Developing the game using machine learning and reinforcement learning involves deep knowledge of AI techniques.

4.3.1.5 Web Implementation with Flask/Django :-

You can develop the game as a web application using a framework like Flask.The backend would be Python, which would manage game logic, and frontend. Would be HTML/CSS/JavaScript, which would render the board and handle user Interactions.

Advantages :-

Cross-Platform: Players can access the game from any device that has a web browser. Realtime player interactions can be attained through WebSocket in case multiplayer support is required. Easy scaling with cloud services for storage of game data.

Problems Latency :-

Multiplayer games are highly prone to delays, more so with real-time update systems.

Complexity: The backend is Python with real-time interactions, whereas the frontend is JavaScript and both should know development on the client-side as well as server[6][7]

CHAPTER 5
SOFTWARE FEATURES

5.1 FEATURES:

5.1.1 GUI (GRAPHICAL USER INTERFACE)

This project ,chessboard contains 8*8 chessboard with light white and black squares providing a clear and aesthetically pleasing board. The board dynamically displays chess pieces in different transformation from game state to another state. It enhances the experience of user playing the chess game with these interactive screen on computer screen.

5.1.2 SIDEBAR MENU

There is a dedicated area to the right of chessboard screen. With following features:-

A.)Start game:-It initializes the chessboard and timers.

B.)Resign:-It terminates the game with current player resigning.

C.)Offer draw:-Sends a draw offer ,ending the game.

D.)New game:-Resets the game to its initial state.

E.)Game modes :-AI vs Human, HUMAN v HUMAN

Interactive timers:- When we set timer and select mode for the game, timers come on the right side below the side bar menu. Timers are running whose player chance currently is .User can Select multiple time controls like 5+2,10+2.

Pop-up notifications:-Temporary popup for events like Draw ,Resign and game-end conditions and other conditions. It auto-disappear after 3 seconds after popup over the center of window.

5.1.3 AI INTEGRATION

The STOCKFISH engine support connects to via UCI protocol to get best moves for AI. It gives reaction to user move in quick 2 seconds. It analysis the best or optimal move based on factors like material advantage, king safety etc . It works on Alpha beta pruning.[8][9][10]

Figure 5.1 stockfish

5.1.4 MISCELLANEOUS FEATURES

Cross Platform compatibility:-It runs on systems with python and Pygame support with configurations for stockfish engine. Efficient Frame Rate management for fast gameplay.

5.1.5 FUTURE POTENTIAL ENHANCEMENTS

A.)Move history display:- a log showing the PNG files for games played.

B.)Save/load game for to save and resume games.

C.)Customizable AI difficulty

D.)Online multiplayer mode

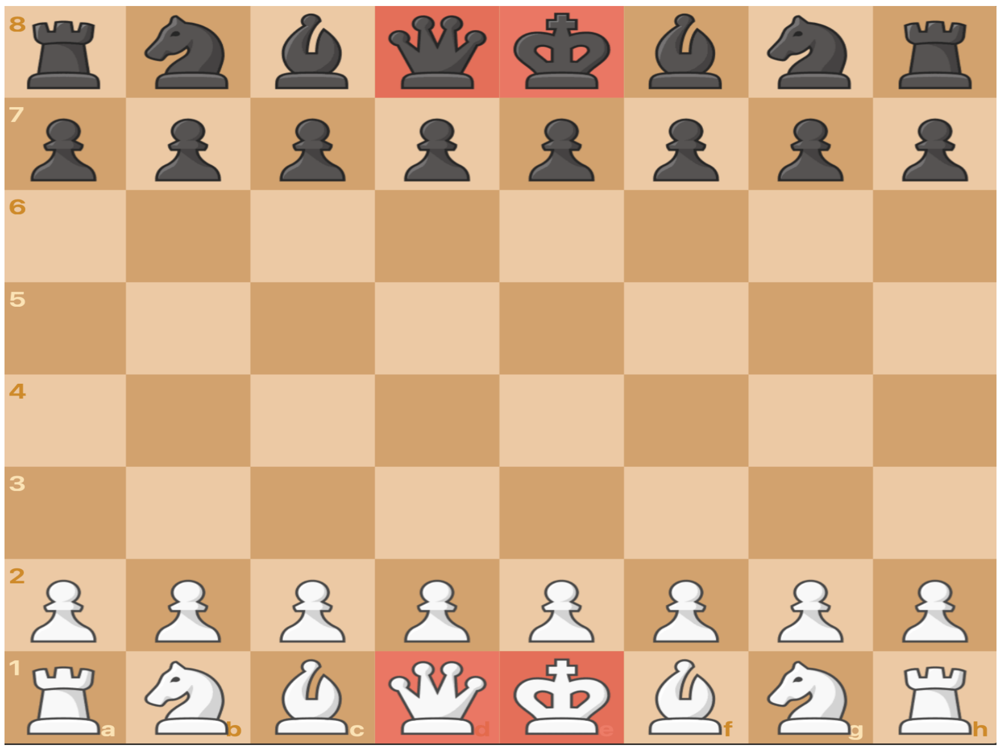


Figure 5.2 Chess Board layout

Figure 5.3 AI vs Human

CHAPTER 6

PROJECT SNAPSHOTS

6.1 SAMPLE IMAGES :

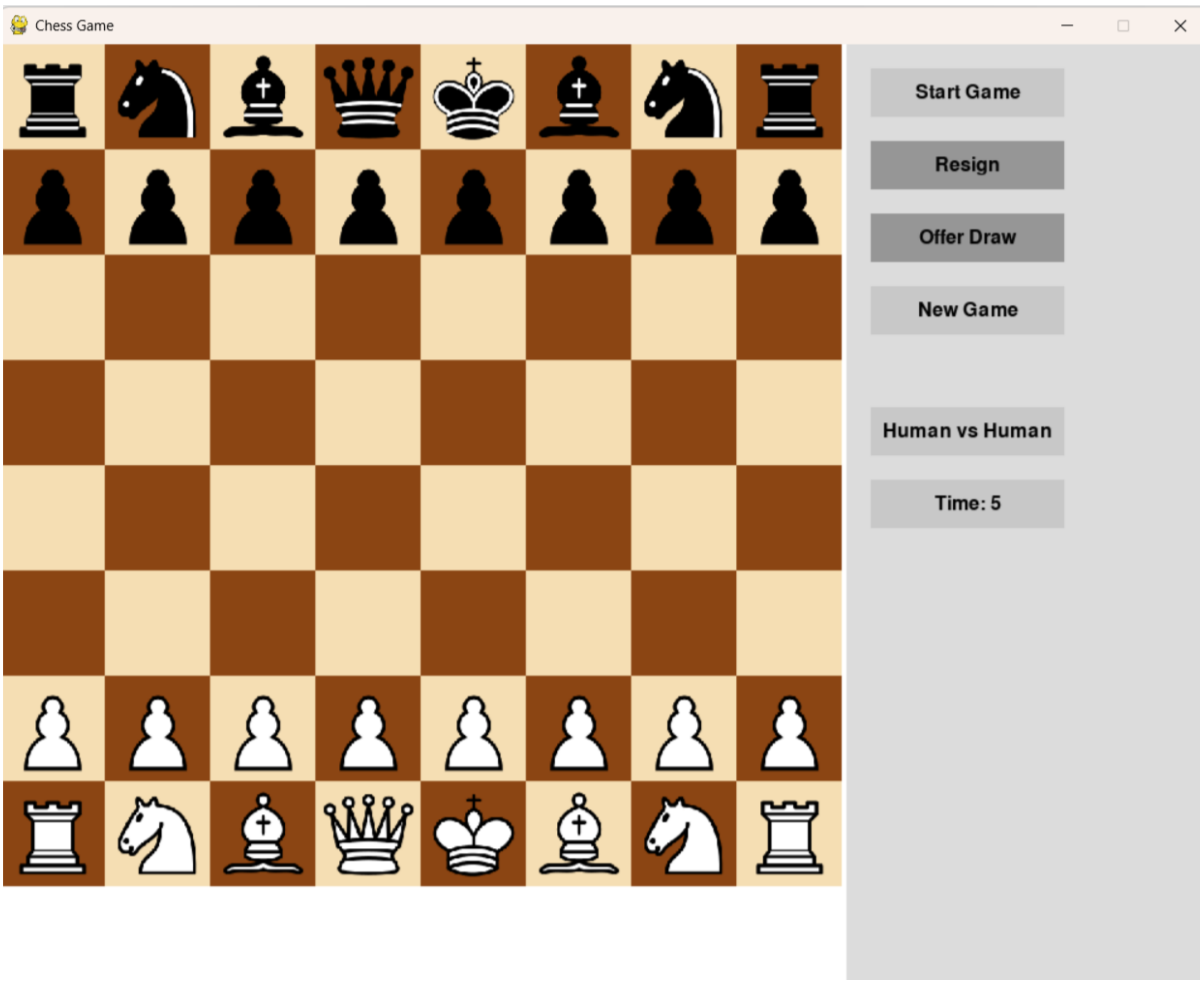
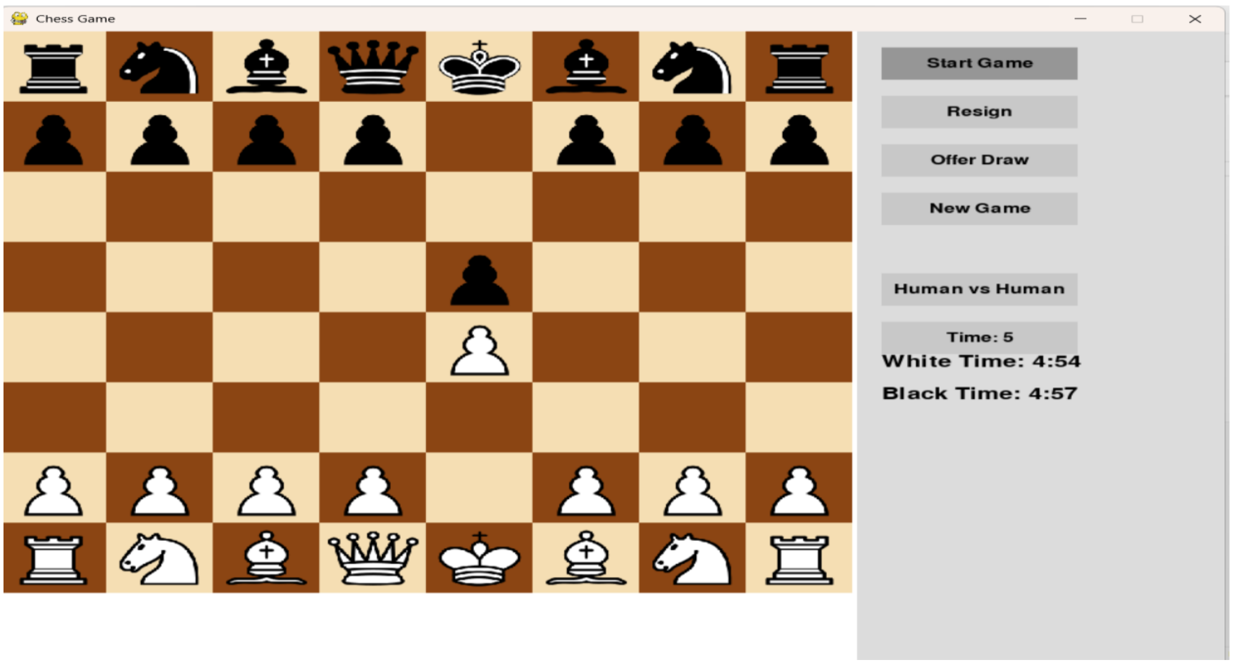


Figure 6.1 Sample Image

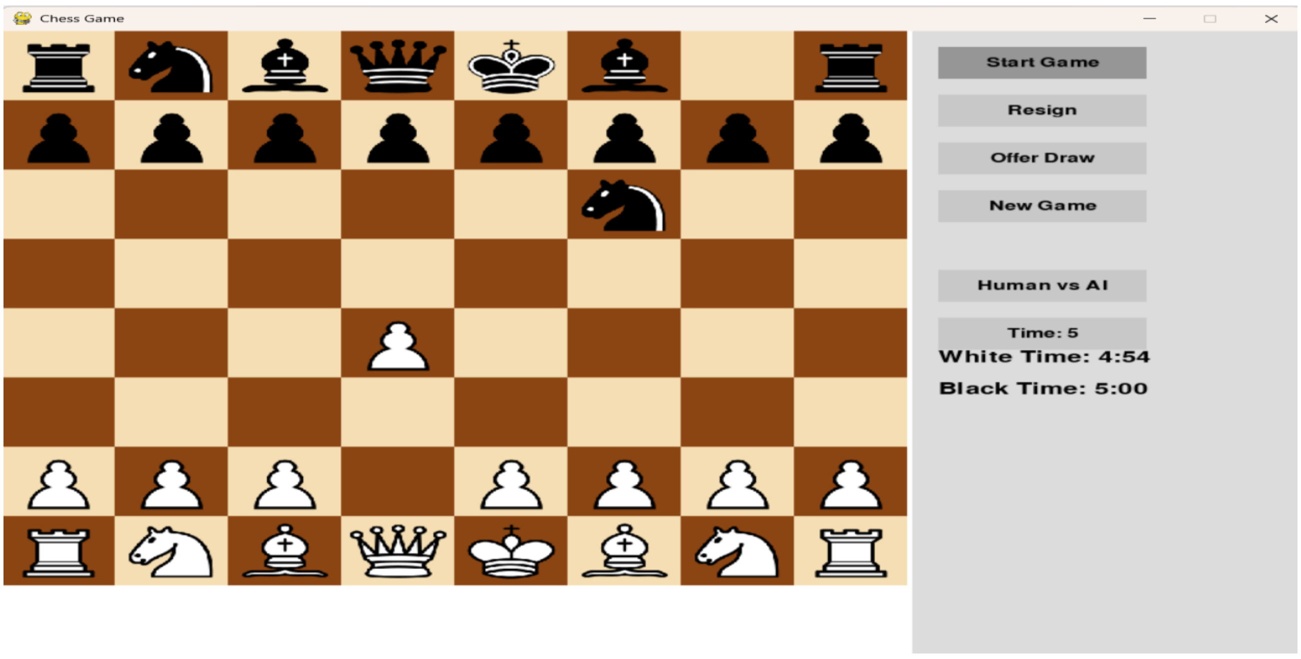
6.2 SAMPLE WORKING:



White Time: 4:54
Black Time: 4:57

Figure 6.2 Sample Working

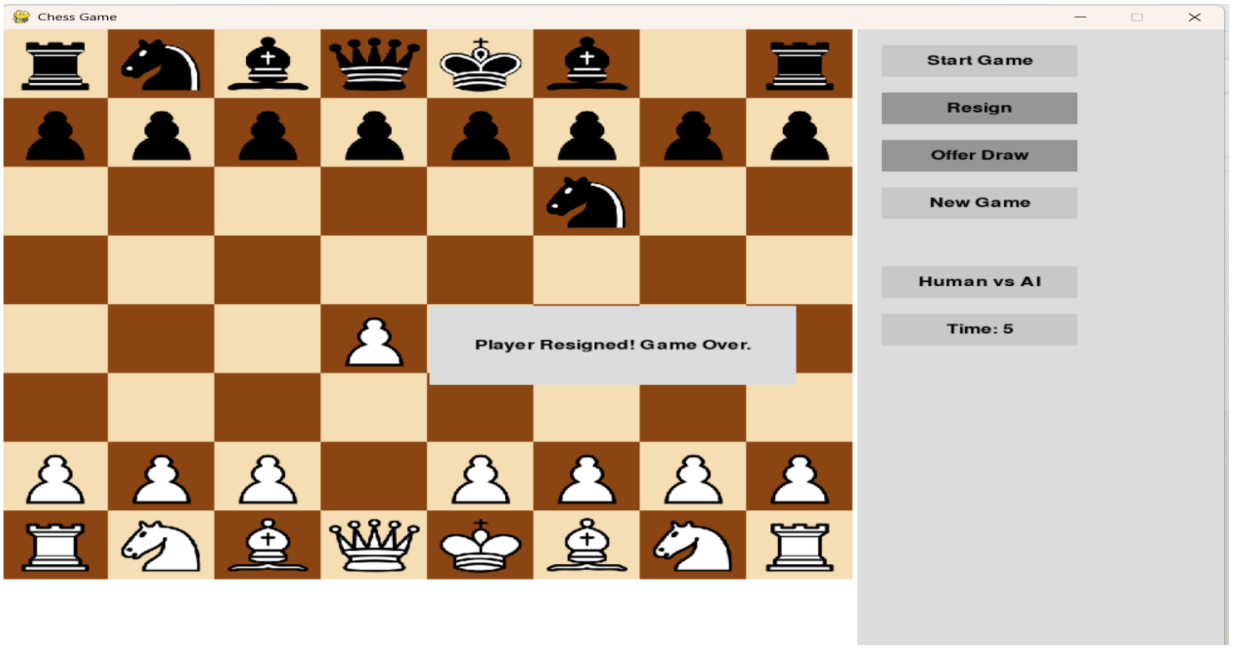
6.3 AI vs Human:



White Time: 4:54
Black Time: 5:00

Figure 6.3 AI vs Human

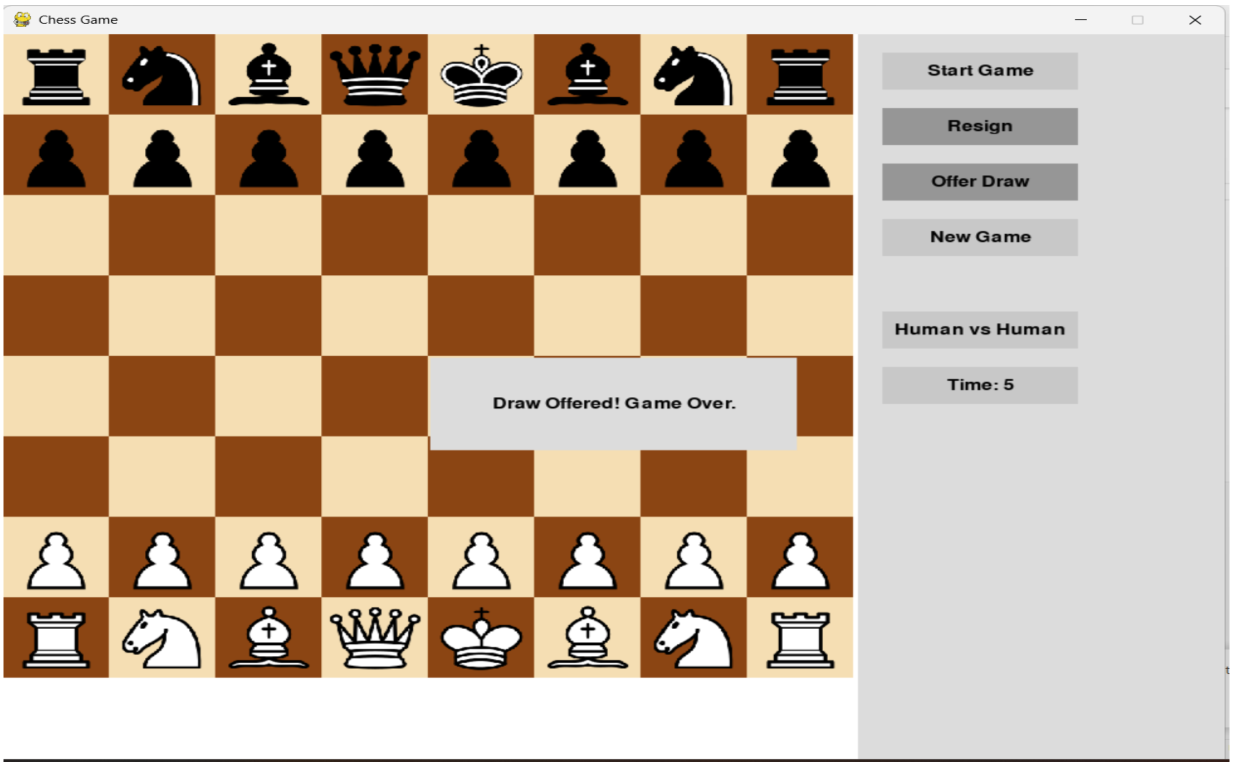
6.4 RESIGN WINDOW:



Player Resigned! Game Over.

Figure 6.4 Resign Window

6.5 DRAW WINDOW:



Draw Offered! Game Over.

Figure 6.5 Draw Window

CHAPTER 7

LIMITATIONS

7.1 User Experience :

One of the main challenges of chess project is the user experience in beginner’s understanding of rules, timer management, chess knowledge. These factors can make it difficult for the algorithms to generalize and cope with different scenarios and conditions. For example, a new user does not know about rules of chess . Users with disabilities like Visually impaired require high contrast , different input method other than mouse . Software requires seamless animation , sounds , 3D chessboard, indication of illegal move during game for new users. Endgame feedback , celebratory messages after conclusion of game . Touch screen devices might not be able to integrate drag and drop feature seamlessly.

7.2 Future Considerations :

Another challenge of chess project is the diversity of the features that are used in it. Different datasets may have different characteristics, such as resolution, format, annotation, distribution, and bias. Rendering high-res Graphical user interface may require high rendering. Moreover, AI features require some proper configurations and resources .This is a big problem in integrating all features modules and ensure it works efficiently. Such AI-based enhancements will help in refining the datasets. Then once the datasets are gathered, they'll need to be sufficiently complex covering all aspect. More features put pressure on system processor speed. Extensibility of new modules must me handled properly and coordinated properly so that the application run properly. Multiplayer real-time matching requires strong hold on networking concepts and proper configuration needed. Large database needed to store the details of users like username, password, PNG files , dictionary of previous high level professional game played historically during world championship , candidates etc.

7.3 Complexity of algorithm :

A third challenge of chess project is the complexity and trade-off of the algorithms that are used to implement the tasks. On one hand, the algorithms need tobe accurate and robust enough to handle the variability. On the other hand, the algorithms need to be efficient and scalable enough to run on different devices and platforms, such as mobile phones etc. Therefore, the algorithms need to balance between speed, memory, power, and accuracy, which can be hard to achieve and optimize. With the advent of AI in chess, the cost of access and processing power will reduce over time.

On adding new features like chess960,leaderboard ,we have to handle large datasets, resources, network management. The more datasets is gathered,the more data will push low end system to failure. Increasingly such a datasets will need to be managed by AI systems rather than humans, but the accuracy aspect is a concern. Online features also include new vulnerabilities in code like data breach, hack etc. Regular maintenance and improvements will need to be done.[11][12]

7.4 Cheating in chess :

A fourth challenge of Computer chess with engine is online cheating possibilities in it. Technology has been used by chess cheaters in several ways. The most common way is to use a chess program while playing chess remotely, such as on the Internet or in correspondence chess. Rather than play the game directly, the cheater simply inputs the moves so far into the program and follows its suggestions, essentially letting the program play for them. For ex-

- In April 2015, Georgian grandmaster Gaioz Nigalidze was banned from the Dubai Open Chess Tournament after officials discovered him consulting a smartphone with chess software in the washroom during a game. He was later stripped of his grandmaster title and banned from competition for three years, though he was allowed to keep his International Master title.

- In February 2016, Sergey Aslanov was expelled from the Moscow Open, for a smartphone in the toilet, hidden under a loose tile behind a drainpipe. He declared himself to be guilty of error but not a crime, and was only suspended for one year.

7.5 Future directions

A fifth challenge of Chess GUI and engines is the future direction and innovation of the research and development of the tasks. Chess GUI are still evolving and improving fields that have many open problems and opportunities. For example, some of the future directions include improving the robustness and adaptability of the algorithms to new and unseen domains and scenarios, enhancing the interpretability and explainability of the algorithms to understand their decisions and errors, and integrating the algorithms with other modalities and tasks.

Figure. 7.1 Limitations

CHAPTER 8

FUTURE SCOPE

It can be easily implemented in multiplayer modes over with others players in real-time. A method could be proposed to illustrate robustness against the variations that is, in near future we could build a system which would be robust and would work in undesirable conditions too. It is proposed for casual chess lovers but we can integrate leaderboard for ranking and interaction with chess communities like chess.com. In future there is possibility to offer subscription plans for user to use some key features like AI. On the other hand, our system can be used in a completely new dimension of chess GUIs, cross platform, which can be an easy way for everyone to play on the go. A game archive for PNG files of the games can be stored and later analyzed for improving chess understanding for chess learners. A player stats tracking for strengths and weakness can be a new approach for players to prepare according against them. A 3D chess board approach can be set up for more immersive experience for users. Real-time analysis of chess games played with best suggestions along with ideas behind it. Different themes and piece skins for user to modify according to their taste.

Figure. 8.1 Future scope

CONCLUSION

This project implements both traditional chess with new features like AI mode to increase user experience, timers, interactive GUI. Implementations of chess are creating interactive experience for user to play chess with human or engines.The chess GUI can be used as implementation to play online also. Chess GUI has been envisioned for the purpose of making engaging gameplay with competitiveness of engine with drag and drop features. This project integrates event driven programming approach to provide a chess GUI with seamless user experience. It shows benefits of new chess GUI over old chessboard game

The efficient and accurate method of chess GUI with AI that can replace the old manual playing methods. This method is secure enough, reliable and available for all. Proposed algorithm is capable of playing at very high level than human cognitive ability so player can test them against it, and performance of system has acceptable good results.

REFERENCES

[1] Pygame contributors, "Pygame Documentation," [Online]. Available: https://www.pygame.org/docs/. [Accessed: Dec. 10, 2024].

[2] T. Romstad, "Stockfish," [Online]. Available: https://stockfishchess.org/. [Accessed:Dec. 10, 2024].

[3] Python Chess Contributors, "python-chess," [Online]. Available: https://pythonchess.readthedocs.io/en/latest/. [Accessed: Dec. 10, 2024].

[4] Wikipedia contributors, "History of chess," Wikipedia, The Free Encyclopedia, [Online].Available:https://en.wikipedia.org/wiki/History_of_chess#:~:text=The%20earliest%20precursor%20of%20modern,checkers%20and%20Go)%2C%20and%20victory. [Accessed: Dec. 10, 2024].

[5] Author(s), "How chess algorithm works?" Medium, [Online]. Available: https://medium.com/analytics-vidhya/how-chess-algorithm-works-69e8ae165323. [Accessed: Dec. 10, 2024].

[6] "GUI," [Online]. Available: https://www.chessprogramming.org. [Accessed: Dec. 10, 2024].

[7] Author(s), "Minor project proposal - A real-time classic chess game," ResearchGate, [Online]. Available: https://www.researchgate.net. [Accessed: Dec. 10, 2024].

[8] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 4th ed., Pearson, 2020. [Online]. Available: https://www.pearson.com/store/p/artificial-intelligence-a-modern-approach/P100000474388. [Accessed: Dec. 10, 2024].

[9] M. G. H. M. T. G. S. S. A. H. B. J., *Programming Game AI by Example*, Wordware Publishing, 2003. [Online]. Available: https://www.amazon.com/Programming-Game-Example-Wordware-Publishing/dp/1556228474. [Accessed: Dec. 10, 2024].

[10] A. Karpathy, "Convolutional Neural Networks for Visual Recognition," Stanford University, [Online]. Available: http://cs231n.stanford.edu/. [Accessed: Dec. 10, 2024].

[11] D. E. Knuth, *The Art of Computer Programming, Vol. 3: Sorting and Searching*, 2nd ed., Addison-Wesley, 1998. [Online]. Available: https://www.pearson.com/store/p/the-art-of-computer-programming-volume-3/P100000474446. [Accessed: Dec. 10, 2024].

[12] S. Thon and R. V. Mayer, "Chess Programming: Algorithms, Data Structures, and Artificial Intelligence," in *Proceedings of the International Workshop on Computational Intelligence*, 2018, pp. 100-105. [Online]. Available: https://www.researchgate.net/publication/327462857_Chess_Programming_Algorithms_Data_Structures_and_Artificial_Intelligence. [Accessed: Dec. 10, 2024].

[13] P. Williams, "Building AI-Powered Chess Bots with Stockfish and Python," Medium, 29 Jan. 2019, [Online]. Available: https://medium.com/@petewilliams_22065/building-ai-powered-chess-bots-with-stockfish-and-python-12b8da4b9aaf. [Accessed: Dec. 10, 2024].

[14] M. K. Z. S. M. A. H. A. B., "Real-Time Strategy Game AI Using Search Algorithms," in *Proceedings of the International Conference on Artificial Intelligence and Game Development*, 2017, pp. 87-95. [Online]. Available: https://www.acm.org/publications/proceedings. [Accessed: Dec. 10, 2024]

[15] Google, "Google Images," [Online]. Available: https://www.google.com/imghp. [Accessed: Dec. 10, 2024].