

# Final Report

---

## COMP3000 Computing Project 2022/2023

---

Callum Reece Organ  
10693074  
BSc (Hons) Computer Science



# GitHub Repository

The repository for this project has been split into *Documentation* and *Code*, under their respectively named folders. The repository can be accessed at the following link:

- <https://github.com/ORG4N/ChessAI>

## Word Count

The total word count of this report is: TEXT

## Abstract

Text

# Table of Contents

GitHub Repository.....	2
Word Count.....	2
Abstract.....	2
1. Introduction .....	5
2. Background .....	6
2.1 Literature Review.....	6
2.1.1 Existing Applications .....	7
2.1.2 Technology.....	7
2.1.2 Conclusion of: Background .....	8
2.2 Aims and Objectives.....	9
3. Requirements.....	11
3.1 Requirements Gathering.....	11
3.2 Constructing User Stories .....	12
3.3 Prioritising Requirements .....	13
3.4 Product Backlog .....	14
3.5 Conclusion of: Requirements Analysis.....	18
3.6 Chosen Development Technologies .....	15
4. Project Management .....	19
4.1 Method of Approach.....	19
5. System Architecture, Design, Implementation.....	21
6. Development and Stages .....	21
7. Testing.....	21
8. Output Documentation.....	21
9. End of Project Report.....	21
10. Project Post-Mortem .....	21
11. Conclusion.....	21
12. References .....	23
13. Bibliography .....	23
14. Appendices.....	24
14.1 Project Initiation Document .....	24
14.1.1 Project Vision .....	24
14.1.2 Project Goals .....	24
14.1.3 Project Objectives .....	24
14.1.4 Critical Success Factors .....	24
14.1.5 Roles and Responsibilities.....	25
14.1.6 Proposed Gantt Chart .....	25
14.2 Risk Register and Assessment Matrix .....	26
14.2.1 Risk Register.....	26
14.2.2 Risk Matrix .....	29
14.3 Requirements Analysis Document.....	30
14.3.1 Stakeholders .....	30
14.3.2 Technology.....	31
14.3.3 High Level Requirements .....	32
14.3.4 User Stories.....	33
14.3.5 MoSCoW Prioritisation .....	35
14.3.6 Product Backlog .....	37
14.4 Wireframes .....	39
14.4.1 Index.....	39
14.4.2 Login/Create Account .....	40

14.4.3 Match Creation .....	41
14.4.4 In Match .....	42
14.4.5 Profile (History) .....	43
14.4.6 Profile (Medals).....	44
14.4.7 Profile (Friends).....	45
14.4.8 Learn (Basics) .....	46
14.4.9 Learn (Openings).....	47
14.4.10 Learn (Key Advice) .....	48
14.5 Kanban Board.....	49
14.5.1 Starting Board .....	49
14.5.2 Chronological Progression of Boards.....	49
14.5.3 Final Board .....	55
14.6 Sprint Schedule .....	56
14.7 Sprint Plans and Retrospections .....	57
14.7.1 Sprint Zero.....	57
14.7.2 Sprint One .....	58
14.7.3 Sprint Two .....	59
14.7.4 Sprint Three .....	60
14.7.5 Sprint Four .....	61
14.7.6 Sprint Five .....	62
14.7.7 Sprint Six .....	63
14.7.8 Sprint Seven .....	64
14.7.9 Sprint Eight.....	65
14.7.10 Sprint Nine .....	66

# 1. Introduction

The first section of this report introduces a literature review that discusses existing applications within the chess community and the technologies they use. The benefits of these technologies will be discussed.

The second section introduces the requirements of the application and discusses the technology planned to be used.

The fourth section of the report discusses project management and how sprints were planned and navigated.

The fifth section of the report displays the designs made for the User Interface of the application.

The sixth section discusses the challenges of development and keeping within sprints.

## 2. Background

Learning Chess can be a daunting adventure for some individuals, but this is merely a stigma that has been fabricated by the stereotype that Chess is a game for only intelligent people, and that to play Chess a person must be fully committed to learning a game that is perpetuated as complex. However, as with any other game or sport, Chess can be played casually, and indeed without knowledge of strategy.

The main method of learning a game or sport is through a repetitive cycle of practice and revision. For example, athletes in physical sports must perfect their form by repeatedly going through new experiences and experimenting and then ultimately deciding whether those experiences were positive or negative. This act of building knowledge via experience is otherwise known as practice. Within the game of Chess this is applied when a player moves a piece and then realises that a positive effect has occurred wherein they take an opponent's piece or win via checkmate. Likewise, a player may realise that a move has negative consequences, and they allow the enemy to gain an advantage over them.

The ChessAI application provides this environment within which a user can develop their Chess skills as it provides them a range of computer-operated opponents to select from, with each representing a different skill level of Chess. ChessAI is an application ideally suited for beginner and novice chess players. Through the app, these players can play games against AI opponents that aren't designed to play the best possible move, but rather determine which move the average player of the AI's designated skill level. Therefore, new players can play against a simulation of a human player without the pressure of playing against a real human player.

In Chess, '*elo rating*' is the value which represents the skill level of a player (*Elo rating system - Chess Terms, no date*). This value can increase or decrease, which is determined by the outcome of each game they play. ChessAI uses the elo rating system to classify the different computer opponents. Each AI model is trained on the data of players in different elo brackets, ranging from between 1100 to 1900. Within the elo system a higher number represents a higher level of skill.

The idea of implementing Artificial Intelligence within Chess is not revolutionary. Different applications do exist and lots of new applications are in the process of being created. The following Literature Review investigates these solutions.

### 2.1 Literature Review

This Literature Review was undertaken to answer two research questions:

- Which solutions currently exist within the Web for persons to learn Chess and develop their skills?
- Which technologies are currently widely employed to implement Artificial Intelligence within Chess?

The first section of this review investigates the first question, whereas the second section explores the second question. Answering these questions and carrying out this review highlighted issues with current applications that the Chess AI can innovate upon as well as differentiate itself and tap into a much more specific segment of the Chess community.

## 2.1.1 Existing Applications

The biggest contenders within the Chess community are Chess.com and Lichess. These two platforms both allow users to play Chess in a manner of ways. Common features that both these applications share is:

- Player vs Player.
- Player vs Computer.
- Casual, non-traditional game modes.
- Puzzles.
- Lessons to learn all about Chess (rules, pieces, openings, advice).
- Forums and Chess news.
- Tools to review moves made and highlight blunders.

These platforms are not just platforms that enable users to play Chess but encourage the community to interact with each other and get involved within discussions via the forum and in-game chat box and friend functionality. These platforms can be described as social medias, with Chess as the common theme.

Within both applications, post-game analysis features are implemented that allow players to review the game that they just played. Both applications use the Stockfish engine to evaluate the board position which returns what is perceived to be the best possible move, as well as evaluating the move that the player has made to determine whether it is a blunder or perfect move.

### Limitations

These two applications' limitations are very few as these are global platforms used by millions of active users (*Rapid leaderboard - chess rankings, no date*) each month. However, whilst both sites have a great sense of community, they do not sufficiently focus on the individual user. Within both applications it is not easy for the user to find their statistics. The focus of these platforms is to simply play Chess and win or lose. Within these applications it is difficult to monitor progress, aside from seeing the elo rating increase or decrease.

Also, Chess.com is designed around its membership scheme. Some of the features most significant in developing a player's proficiency are hidden behind this paywall and access is limited. This paywall limitation prevents players who want to get better at Chess from getting better via the Chess.com platform.

A platform where a user can freely complete unlimited puzzles and track their stats is desirable within the Chess community. Neither platform offers a fun progressive experience as a novice learns how to play the game, nor do they reward the players in a fun personal way. Rewarding new players is essential as the satisfaction they gain from being rewarded keeps them motivated and invested into using the platform, and overall getting better at Chess.

## 2.1.2 Technology

Both existing applications are served as Web applications (for online play), and Android/iOS applications (for online and offline play). Both applications also use the Stockfish Chess engine to generate computer moves and to analyse board positions. An alternative to Stockfish is the Leela Chess Zero (LC0) engine.

## Stockfish

Stockfish is an open-source, free Chess engine that is compatible across Desktop, iOS, and Android. Stockfish is rated as “the strongest CPU chess engine in the world” (*Stockfish (chess), 2023*). This engine uses a variation of a minimax algorithm which essentially creates a tree of nodes, where each node represents a possible position on the board. The algorithm will cut off branches of nodes that are deemed worse if a better move to be played has been found. This will occur until the algorithm can determine that it has found the best possible move out of all moves within the tree.

## Leela Chess Zero

LC0 is another open-source, free Chess engine. This engine uses deep neural networks alongside the Monte Carlo search tree algorithm to evaluate board positions. Different networks can be given to the engine as a parameter. This makes LC0 versatile as different network models can be used depending on the hardware or need of the application (*Selecting network to use, no date*).

## Maia Chess models

The Maia Chess models are an example of networks that have been made and are open-source and free to use (*Maia chess, no date*). These models have been specifically designed so that the move that the chess engine returns is a move like one played at a specific elo rating. For example, the Maia 1100 model will play a move that has factored in the average playstyle of a 1100 real human player. These models are trained on the open-source datasets that Lichess publishes every month, which contain millions of games. These datasets are filtered and compressed and used to train the model. The model can be ran using the Leela Chess Zero engine.

## Python

Python is at the forefront of Artificial Intelligence research and development due to the very high range of libraries to choose from and the readability of code. Within these libraries there are also various frameworks that can be used to communicate with Chess engines and provide the functionality of a virtual chessboard. The Python-Chess library is one example. The benefit of these libraries is that code can be imported into another application and reused without developers having to create and maintain their own libraries.

## 2.1.2 Summary of: Background

Chess.com and Lichess, the two most popular Chess platforms, are underwhelming when it comes to personal progress and turning Chess into a fun learning experience. Afterall, Chess is a game and if a player wins or achieves milestones they should be rewarded and enticed to keep playing. These platforms lack a reward system.

The Literature Review has also found that there exist open-source models that are able to determine how a human at a specific elo would play. These models can be run on the open-source Chess engine Leela Chess Zero. Theoretically, a multitude of different Chess engines could be used in conjunction for different reasons. Where the LZ0 engine running Maia models could show the move a human would play, Stockfish could be used to then show the user the overall best move to play.

Overall, the first research question has been answered and Chess.com and Lichess have been highlighted as the two largest platforms within the Chess community and they enable players to learn via lessons and



puzzles (of which may be hidden behind paywalls). Players can also learn, through statistical analysis at the end of each match via the Stockfish engine, where they blundered and the move they should have played instead.

Finally, the second research question has also been explored generally. To implement Artificial Intelligence in Chess, different engines have been made using various algorithms. The Stockfish engine creates trees based on combinations of moves and possible board positions and evaluates future positions to determine the strongest long-term plays. The Leela Chess Zero engine uses neural networks to evaluate board positions and make moves. LCO can be given differently trained neural networks, such as the various Maia models. Overall, this makes LCO more versatile.

## 2.2 Aims and Objectives

As expressed within the Project Vision ([Appendix 14.1.1](#)) the aim of the ChessAI application is to make a Chess practice tool wherein users can monitor their stats. The application will provide its own set of AI opponents that the players can versus. Each match should end with the user being shown an evaluation of all moves played and provide them with resources to improve. The main priority of this application is to enable the end-user to be able to play Chess games against computer opponents that play like a human player of chosen rating. End-users should be able to then analyse these matches and see how often they win or lose, and the application should highlight this for them and provide methods of helping to improve their performance.

The Project Vision was outlined at the start of the project within the Project Initiation Document ([Appendix 14.1.1](#)).

The Project Objectives were also outlined within the Project Initiation Document. These objectives provide meaningful targets that the project should achieve, such as:

- The AI developed must be accurate in determining the move that is expected of a computer player at assigned elo rating. If the AI is making irregular moves, or moves that are terrible, then this could have a negative impact on a player's satisfaction. Players don't want to play against opponents that are clumsy unless they have chosen the easiest level of computer.
- Another objective of the ChessAI project is to be fast. Players do not enjoy waiting for their opponents to make moves – especially if their opponent is a computer. Although a small amount of time is beneficial as it lets the player think about their next move, consistently waiting long amounts of time for the computer to move will make the player bored and they will leave the application.
  - Likewise, one such limitation in Player versus Player Chess on sites like Chess.com and Lichess is the possibility of opponents disconnecting mid-game and sabotaging the match. With an AI opponent, this cannot happen as the AI will always seek to play the best possible move it can find.
- The final objectives of the project are to ensure that the UX is clear and concise. One such moment where this is essential within the application is when highlighting blunders and perfect moves. It is important to highlight clearly when a player has made a mistake, so that they can learn and correct themselves.

To ensure that these Aims are met, each objective must be fulfilled. The next section details the Requirements behind these objectives and what features will be needed to ensure that the project is usable.

Alongside, the Aims and Objectives, some general Critical Success Factors ([Appendix 14.1.4](#)) were also identified within the Project Initiation Document. These Critical Success Factors highlight that the following deliverables must be maintained throughout the project:

- Kanban board ([Appendix 14.5](#))
- Risk Register ([Appendix 14.2](#))
- Gannt chart ([Appendix 14.6](#))

## 3. Requirements

From observing the Chess.com and Lichess websites and noting common features within Chess apps, and via thinking of Chess practice tools from a beginner Chess player's perspective, the project's requirements were extracted. All requirements analysis was carried out and outlined within the Requirements Analysis Document ([Appendix 14.3](#)) that was created during sprint zero, following the Product Initiation Document ([Appendix 14.2](#))

### 3.1 Requirements Gathering

The first step in gathering the requirements of the application was to identify all components that the application could have. These components were written into the Functional Requirements ([Appendix 14.3.3](#)) table. Figure 1 is a short extract from the Functional Requirements table.

No.	Description
1	User can play Chess against an AI.
2	User can choose to start with White or Black pieces.
3	User can select difficulty range of AI.
4	User can see their personal win rate against AI.
5	User can see statistical analysis at the end of each match.

**Figure 1: Functional requirements examples**

The requirement examples within Figure 1 show how the functional requirements are written as actions that the user can take within the application. Other requirements were written to describe the behaviour of other systems within the application, such as the database or the application. For the database, requirements described what information would be stored, in a general manner. Likewise, requirements that described the application in a whole specify the ways in which the application should be accessible.

Non-Functional Requirements were also written into a separate table ([Appendix 14.3.3](#)). These requirements differ from functional requirements, as whilst the latter describes actions that should be able to be made, non-functional requirements establish how these actions can be measured (where applicable).

For example, one non-functional requirement describes how many invalid attempts a user has before they are disabled from making more account sign-in attempts. Other requirements specify how long it should take for a page to load, or for an action to occur.

Gathering both functional and non-functional requirements helped identify the overall functionality that should be implemented within the ChessAI application. However, at this stage of requirements analysis, there was no comprehension of which users will be using the application. Likewise, some of these requirements are broad and describe a solution, rather than a 'usable' product, where usable is a term that describes something that consumers can interact with repeatedly as it is reliable, useful, and desirable.

Therefore, to better articulate the requirements and describe them as usable feature they were further processed into User Stories (see section: *3.2 Constructing User Stories*).

## 3.2 Constructing User Stories

The second step within Requirements Analysis ([Appendix 14.3](#)) was to generate User Stories ([Appendix 14.3.4](#)) from the high-level requirements that were defined in the previous section ([3.1 Requirements Gathering](#)).

User stories are descriptions of requirements from the perspective of a stakeholder. These descriptions are written in a specific structure in the following format:

As a [stakeholder name] ...  
I want to [specific action/requirement] ...  
So that [justification or benefit of action/requirement]

This format is beneficial to both developers and stakeholders as features are emphasised; developers can directly implement these requirements as features. This focus on features is desirable in a development environment because it incorporates the user into the approach to development. In-fact, when implementing the user stories as features within the application, the end-user is a central figure, and the developer is conscious of this. The developer can prioritise creating an application suitable for an end-user, rather than themselves or the visions of a superior.

One benefit of writing user stories is their simplicity. They are clear, concise, and structured. The advantage this offers is that they directly communicate an end-user's need to the developer. Another advantage is that they can be easily understood by everyone. The relevancy of this within this project specifically is that the lead developer can provide the project supervisor with the list of user stories to easily convey the core features and users of the project, and why a user would want these features.

Additionally, when referring to user stories, such as within the Kanban board ([Appendix 14.5](#)) it is easy to understand what the desired outcome of each user story is. For individuals with no familiarity with the project they can readily perceive the expected effect on a user. User stories are written in a way that minimises technical jargon – users do not care about the specifics of databases or html elements or buttons, all they know are the specific interactions they want the app to provide. Minimal jargon increases the readability of user stories and further justifies why they are important within requirements analysis.

Figure 2 presents a short extract of User Stories from the Requirements Analysis Document. More user stories can be found in the appendix ([Appendix 14.3.4](#)).

As a Chess Player...	I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.
	I want to learn new openings so that I can improve at the game.
	I want to choose to start with white or black pieces, so that I can practice different openings.
	I want to select the rating of my opponent, so that I can train against easier or harder opponents.
	I want to view my personal statistics such as win rate so that I know if I am improving or not, and gain confidence to keep playing chess.
	I want to review a match that I just played, so that I can see where I made mistakes/blunders.
	I want to select and play different match types (bullet/blitz/rapid), so that I can play either short or long games.
	I want to play against chess professionals, so that I can challenge myself.
	I want to abandon the match, so that I can stop playing and do something else, or because I feel like the game is lost.
	I want to view my match history so that I can review past games / see if I'm on a loss or win streak.

**Figure 2: User Stories from the perspective of a Chess player.**

## 3.3 Prioritising Requirements

The third step of processing requirements was to categorise ALL user stories by their priority. Prioritisation is a beneficial activity within requirements analysis as it can be used to sort requirements from most necessary to least necessary. This activity is useful in highlighting the core requirements of the application and the features that the application must contain for it to be considered 'successful'.

Within this project, the MoSCoW ([Appendix 14.3.5](#)) technique was used to apply prioritisation. With this technique, each user story was classified as either:

- Must have.
- Should have.
- Could have.
- Won't have.

Figure 3 shows an example of the Chess Player user stories being separated into the above categories. The full list of prioritised requirements can be found within the appendix ([Appendix 14.3.5](#)).

MUST	I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.
	I want to learn new openings so that I can improve at the game.
	I want to choose to start with white or black pieces, so that I can practice different openings.
	I want to select the rating of my opponent, so that I can train against easier or harder opponents.

**Figure 3: Sample of some user stories being prioritised as Must Haves.**

The advantage of prioritising the application's requirements is that the user stories can be balanced and the most critical factors to the product's success are identified as more urgent, whilst less significant features are either delegated to be completely unnecessary within the project, or still relevant but not essential.

'Must have' requirements are obviously essential to the project. They are the features that the application requires to fulfil the baseline of a customer's needs. Examples of these within the ChessAI application include requirements relating to Chess matchmaking, such as being able to set the length of the match, the starting side (white/black), and the elo-rating of the opponent so that the difficulty of the match can be controlled. Without these the project will not be successful.

'Should have' requirements succeed the core requirements. Whilst these components are not essential to the application, they add a lot of additional features to the product that the customer will want. These requirements are intended to be the next set of features for the developer to implement when all core 'must have' requirements have been finished and tested. Examples of these include the ability of being able to unlock badges and customize Chess board themes. These are important to the user's experience, but they are not integral to helping a player learn how to play Chess.

'Could have' requirements are like 'Should have' requirements as they are also not vital to the application's viability. However, they are identified as less important and desired than the 'Should have' requirements. Examples of these requirements include being able to change the native application of the language and developing a mobile application alongside the web application. For the effort of these requirements, there is minimal gain, and they will not likely be implemented due to the minimal time designated to the project.

As seen in the next section (3.4 *Product Backlog*), the ‘Won’t have’ requirements are completely absent from the final backlog to reduce clutter, making it easier to view the core features of the application. Within this project all user stories were deemed to be important to the project in some manner and therefore, no requirements fall under this category.

### 3.4 Product Backlog

The Product Backlog ([Appendix 14.3.6](#)) is the final deliverable of the Requirements Analysis phase. The backlog is a table that includes all items and their priorities. Furthermore, the backlog has been expanded to include fields for date added and the components which the item can be categorized into. Figure 4 is a short extract of the backlog table.

Date	Component	Item	Priority
15/11/22	Accessibility	I want to play chess on my desktop/laptop so that I can have the match full screen.	High
15/11/22	Accessibility	I want to play chess on my smartphone so that I can play wherever I am (outside, in bed, on the sofa).	Low
15/11/22	Accessibility	I want to use my screen reader so that I can read the text on the screen because I have a visual impairment.	High
15/11/22	Accessibility	I want to change the language of the application so that I can use it in my native languages.	Low
15/11/22	Accessibility	I want to input my chess moves using my voice because I have a motor or vision disability.	Low
15/11/22	Accessibility	I want to be able to change the colour of the chess board and/or pieces so that I can differentiate them because I am colour blind.	Medium

**Figure 4: Extract from Project Backlog.**

All previously defined user stories that have been prioritised are included within the backlog as items. Since the ‘Won’t have’ user stories don’t need to be included within the backlog, priorities have been described in concise terms:

- Must have → High priority.
- Should have → Medium priority.
- Could have → Low priority.

Renaming each category of priority only provides the simple benefit that it is easier and more intuitive to convey items on a scale of low to high priority, rather than could have to must have. The former scale requires no additional explanation to what each priority level entails.

Items do not necessarily need to all be user stories. For example, within the backlog there are items for creating different deliverables, such as wireframes and diagrams for system architecture. Whilst these items could be written as user stories, it is not necessary as they are not functionality of the app, but rather items that document and explain the approach behind design, implementation, and testing stages within the project.

The component field of the backlog is necessary as it maps each item to a specific part of the project. For example, within Figure 4, there are items that describe Accessibility features. These requirements can generally be understood as features to make the application inclusive and easier for a broader range of individuals to use.

Other components that are highlighted within the backlog are: Account, Profile, Learn, Matchmaking, and Design/Plan. All but the lattermost component of the project can be mapped to a distinct part of the application, such as the specific webpages that will be required. Likewise, these components also align with the Project Aims established at the start of the project (see section: *2.2 Aims and Objectives*).

The data field within the backlog records when the item was added to the backlog. Whilst not fully necessary, it provides the additional benefit of showing the backlog being built up throughout the project. Through testing and user feedback, the requirements of the application are expected to evolve, potentially growing in scope, or reshaping how a user story is approached. This is covered in greater detail within section: *4 Project Management*.

## 3.5 Chosen Development Technologies

Upon concluding the Requirements Analysis phase, the technology to manage and implement the project was decided upon. These initially agreed upon technologies can be found within the Requirement Analysis Document ([Appendix 14.3](#)) under the heading 'Technology' ([Appendix 14.3.2](#)). This section of the project was supported by the background investigation (*see section: 2.1 Literature Review*) conducted within the project initiation phases.

It is important to note that the lead developer of this project did not want to learn an abundance of new technologies through the course of this project, and where different technologies or frameworks were to be learned it was important that they were easy to learn as well as potentially desirable for future careers. This is due to the project not just being about creating a project but further enhancing the studies and skillset of the lead developer.

Within the Technology table ([Appendix 14.3.2](#)) numerous tools, frameworks and languages were highlighted. This section of the report will cover each by defining them, describing how they were used in the project, and if not then why weren't they. This section will conclude with an evaluation on how the technologies performed.

### **Project Management Tools.**

This category of tools was used to manage the project and each tool performs a different task. Using these tools built upon the lead developer's management skills as they had to fulfil the role of 'Project Manager' ([Appendix 14.1.5](#)) and efficiently schedule tasks to maintain progression throughout the project and implementation of user stories. Each tool is covered in detail below.

*GitHub*. This tool is a hosting service wherein the source code and documentation of the project are uploaded to. The importance of storing the project within an online repository is that the code/documentation is backed up as a copy from the development machine(s). For example, were the local computer to have technical issues, the uploaded code could still be recovered and worked on from another machine. Likewise, GitHub offers version control so that the changes made to the repository can be easily tracked and reverted. GitHub is an invaluable tool to this project and gaining experience with it through *Git* is important for building career skills.



Whilst GitHub can be used for a variety of reasons, in this project it was simply used to backup code and track 'commits', which are records of the changes made to the project since the last time it was uploaded. Tracking the commits was important because, from the perspective of Project Manager, it is necessary to determine whether features are being implemented routinely and frequently. A lack of commits would suggest insufficient project planning and management.

*Trello.* This project management tool was used within this project to create a Kanban board ([Appendix 14.5](#)). This tool is used to visualize the progress of each item within the Product Backlog. Kanban boards achieve this by having multiple columns, such as:

- To-do
- In Progress
- Testing
- Complete

An item will chronologically move throughout each stage (column) as it is worked on by a developer, and the Project Manager can determine whether there are too many or not enough items being implemented and as such take measures to increase performance and ensure that items from the product backlog are developed in an efficient manner by the end of the project. The stages listed above are not conclusive and may differ based upon how the project is being carried out.

Trello was therefore used in this project by the lead developer and project manager to control which backlog items (written as user stories) were to be implemented, in progress, or done.

## **Programming Languages and Frameworks.**

*Python.* This is a high-level programming language that has many libraries and frameworks that can be imported into the project. This advantage of this is that premade components can be used to develop the application very quickly and easily. Likewise, the lead developer of this project already had previous experience with the Python language. Python's viability for this project was investigated within the Literature Review ([see section: 2.1 Literature Review](#)).

*Flask.* This is a Python framework that is used to create Web applications. This framework was chosen over other existing web development frameworks because Flask is very lightweight, compared to its alternative Django. Flask is also a very popular framework and has lots of online support and a cohesive documentation to follow and get started with – which was useful when it came to learning how to make apps in flask for the first time.

It was decided that a web application would be created as opposed to a mobile or desktop application as the application would be more available to a range of different users. The web application would still be accessible from multiple different devices that have Internet access. The lead developer had never used Flask before and therefore this would be one technology that they had to learn. Flask was justifiably a good framework choice as it was simple and had lots of documentation to learn from.

*Javascript.* As a web application was being made, JavaScript (JS) was also determined to be a necessary programming language. JS would be used to implement webpage behaviour. JS also has various libraries that were useful when creating the frontend dynamics of the application, such as button presses and navigation between webpages. The lead developer had experience with this technology and was comfortable using it within the project.

Overall, it was decided that a Flask web application would be developed. Python would be used to implement backend functionality, and JavaScript for front end behaviour.



## Database Technology.

*SQL Server.* The Technology table ([Appendix 14.3.2](#)) highlighted that the was originally intended to integrate SQL Server as the database management system. It was chosen for the reasoning that SQL Server is an industry standard tool and learning this tool would increase future employability and skillset. Additionally, the lead developer had experience with this tool from other modules during their degree.

However, during the implementation phase of the project SQLite was determined to be a better tool to use within the project. Therefore, before implementing any code using SQL Server, it was replaced with SQLite.

*SQLite.* This technology is also used as a database server. However, SQLite is much more lightweight than any other database alternatives. SQLite was very easy to setup and implement within the project and integrates very efficiently with the Flask framework. Additionally, as the project in this stage is expected to be deployed as a Minimum Viable Product, whilst SQLite may eventually not scale with the project's size and capacity, it is perfect to create an initial application.

## Python Libraries.

*Python-chess.* This is a Python library that implements Chess functionality such as move generation and validation. The library was used within the project to work in conjunction with the computer AI and generate the computer's move. The advantage of using this library was that to implement a bespoke Chess engine would take an extensive amount of time.

## JavaScript Libraries.

*Chessboardjs.* This is the GUI library that was used to represent the board object within HTML. This library allows the user to interact with a chessboard. These interactions are implemented as JavaScript events, and each action on the board can be caught by these events. For example, when a user clicks on a Chess piece, the library implements its own methods to let the user move that piece to another square on the board. However, this library does not implement its own Chess engine and must therefore be used alongside another component that can provide features such as move validation and the ability to determine when the game is concluded.

*Chess.js.* This library is like Python-chess as it implements similar functionality – move generation and validation, and the ability to determine when the game is finished. This library is used alongside its Python counterpart because it is needed to determine whether the moves that they player makes on the chessboard are legitimate and can be used alongside chessboardjs to prevent a move from occurring.

## Other.

*CSS and SASS.* Whilst not as important as the other technologies/tools used within the project, CSS was used to design the webpage. To increase workflow, SASS was setup within the work environment. SASS is essentially a tool that lets developers write CSS in a more structured way. SASS will then compile the developer's SASS code into CSS code automatically. Writing CSS this way increases workflow and speed of writing stylesheets.

*HTML.* This markup language is used to implement the structure of the webpages. HTML is the core of every webpage on the Web. The project was to be split into multiple webpages, that the User Stories outlined, such as: Play, Profile, Sign in, and Learn.

## 3.6 Summary of: Requirements Analysis

The Requirements Analysis Document (14.3) is an important deliverable that details the requirements analysis phase within the project. This phase was essential to starting the project off as it highlights a set of initial core features that the application will require.

The first step in this phase was to directly highlight all possible features (functional) and the criteria to measure the performance (non-functional) of these features. Whilst the analysis phase could end here, with these requirements forming the product backlog (Appendix 14.3.6), these requirements may be too broad and the reason for why these requirements exist is not obvious. Who do these requirements benefit?

Therefore, the second step was to rewrite these requirements into a meaningful format – user stories. Defining user stories was essential within this project as it uncovered and defined the end-user's needs. Where the functional requirements were too broad, they were decomposed into multiple user stories. Through this analysis there were three groups outlined:

- Disabled users
- Chess players
- General users

It was determined that the application should have lots of inclusivity features, such as allowing users to customize the colour theme of the chessboard, as well as supporting a variety of methods to interact with the application like being able to input chess moves through text-to-speech or algebraic keyboard input.

For the typical Chess player, the user stories describe how they should be able to create matches and play against bots. These user stories highlight how and why users want to play Chess. The common theme of these reasonings is to improve their skills at the game and challenge themselves.

The general user is a separate type of end-user because it describes the common user of applications, not just specifically chess applications. For example, these users want customizable profile pages, with badges, and the ability to use the application across multiple different devices (desktop, mobile etc). It was important to establish this group of users as separate to Chess players as some of the requirements established for this end-user are not relevant to playing and improving at Chess but do instead however make the application more personalized and customisable – incentivising users to keep playing Chess.

However, the requirements analysis phase was still not complete as another question arose: which of these requirements are the most crucial for the application's success?

The solution to this question was prioritisation. The process of prioritising the user stories established a starting point within the development of the project's implementation. Prioritisation also determined that whilst being inclusive to a variety of users is desirable, these features are not important in creating a Minimum Viable Product. For this application, these features should be implemented lastly and built upon an already fully tested and working codebase. Likewise, the customization requirements are also not high priority as they are not core features that will fulfil the Product Vision, Goals, and Objectives (Appendix 14.1), which were to create an application where users can play Chess and train via matchmaking against bots and statistical analysis.

Finally, the initially determined technologies were highlighted and explained. However, some of these technologies, such as the database server, were changed and these changes were justified. Finally, the technologies that were uncovered throughout the development process were also covered.

## 4. Project Management

### Kanban Board.

Using Trello, a Kanban board was made to track which Product Backlog items were currently being worked on ([Appendix 14.5](#)). The board was broken down into sections for:

- Backlog
- To do
- Doing
- Testing
- Done

These columns distinctly show the stage of each item. This visualization of tasks makes it clear to see how many items are being currently worked on, how many are left, and how many are completed. Items within the Kanban board were coloured according to their priority which was calculated within the Requirements Analysis phase.

### GitHub.

GitHub was used as the repository for all the source code and documentation. GitHub is highly necessary as it allows for development to occur on multiple different machines, which is useful for when the developer needs to access the code from another device. GitHub also tracks all changes made to the project. GitHub could have been used more effectively through branches to separate different version releases of the project.

### Bi-weekly Meetings.

Bi-weekly meetings were arranged with the Project Supervisor. These meetings involved the lead developer of this project, and other students within the same module coming together and presenting the results of their two-week sprints ([see section: 4.1 Method of Approach](#)).

Attending the meetings allowed for the developer to compare their progress

## **4.1 Method of Approach**

An agile approach was taken within this project wherein sprints of two weeks were carried out consecutively. For each sprint a sprint plan was conducted (Appendix 14.7) alongside a retrospective blog type piece of text concluding what had happened that sprint. Alongside the sprint plans and retrospections, biweekly meetings with the project supervisor were also carried out.

## **5. System Architecture, Design, Implementation**

The front end of the application was heavily designed within Figma, an online application where the developer could create shapes and structure webpages. Figma pages were made for each page needed on the application, which were:

- Index
- Login/Signin
- Learn
- Profile – History, Friends, Badges
- Play
- In game

The Figma files were converted into jpg images which can be found in the appendices (Appendix 14.4)

## **6. Development and Stages**

There were nine sprints overall across the course of the academic year, with the first four sprints (including sprint zero) were dedicated to requirements analysis, design and research. For sprints four through to nine, development and testing were the main focus of the stages. The developer would iteratively develop a piece of functionality and test it and then if the tests passed, the item would be moved along on the kanban board, and a new one selected. The sprint plans detailed each item to carry out within each two week sprint period.

At the start of Semester 2 the developer noticed that productivity in semester 1 was very low and therefore, a new overall sprint schedule was made to map the rest of semester two to sprints. This included special dates such as holidays and other coursework deadlines.

Overall, the productivity as compared to semester one and semester two wherein the developer was following a strict set schedule was greater. This is largely due to the clash of schedules with other modules assignments.

## **7. Testing**

## **8. Output Documentation**

## **9. End of Project Report**

## **10. Project Post-Mortem**

## **11. Conclusion**



## 12. References

*Elo rating system - Chess Terms* (no date) *Chess.com*. Available at: <https://www.chess.com/terms/elo-rating-chess#rating-ranges> (Accessed: 2023).

*Maia chess* (no date) *Maia Chess*. Available at: <https://maiachess.com/> (Accessed: 2023).

*python-chess: a chess library for Python* (no date) *python-chess*. Available at: <https://python-chess.readthedocs.io/en/latest/>.

*Rapid leaderboard - chess rankings* (no date) *Chess.com*. Available at: <https://www.chess.com/leaderboard/live/rapid> (Accessed: 2023).

*Selecting network to use* (no date) *Selecting Network to use - Leela Chess Zero*. Available at: <https://lczero.org/play/bestnets/> (Accessed: 2023).

*Stockfish (chess)* (2023) *Wikipedia*. Available at: [https://en.wikipedia.org/wiki/Stockfish\\_\(chess\)](https://en.wikipedia.org/wiki/Stockfish_(chess)) (Accessed: 2023).

## 13. Bibliography

Dyouri, A. (2021) *How to use an sqlite database in a flask application*, *DigitalOcean*. Available at: <https://www.digitalocean.com/community/tutorials/how-to-use-an-sqlite-database-in-a-flask-application> (Accessed: 2023).

Herbert, A. (2021) *How to add authentication to your app with flask-login*, *DigitalOcean*. Available at: <https://www.digitalocean.com/community/tutorials/how-to-add-authentication-to-your-app-with-flask-login#step-1-installing-packages> (Accessed: 2023).

*Lichess.org open database* (no date) *lichess.org open database*. Available at: <https://database.lichess.org/> (Accessed: 2023).

*Portable game notation* (2023) *Wikipedia*. Available at: [https://en.wikipedia.org/wiki/Portable\\_Game\\_Notation](https://en.wikipedia.org/wiki/Portable_Game_Notation) (Accessed: 2023).

*Quickstart* (no date) *Quickstart - Flask Documentation (2.2.x)*. Available at: <https://flask.palletsprojects.com/en/2.2.x/quickstart/> (Accessed: 2023).

## 14. Appendices

### 14.1 Project Initiation Document

#### 14.1.1 Project Vision

To develop a fast Machine Learning application wherein Chess enthusiasts can practice with AI trained against the data of professional players (i.e., Magnus Carlsen) or amateurs within set Elo rating ranges; ChessAI strives to be a practice tool that will challenge players, transforming them into stronger opponents.

#### 14.1.2 Project Goals

- Lead developer grows proficiency in:
  - Designing and developing desktop applications.
  - Optimizing software
- Contribute to helping novice chess players improve at the game.
- Develop a working MVP by: 23/04/2023
- Develop and release a final product by: 01/05/2023

#### 14.1.3 Project Objectives

- AI must be highly accurate and reflect the data that it is being trained on.
- AI must be fast and efficient and make moves in a timely fashion (within seconds).
- UX must be clear and separate the Chess from the evaluations.
- UX must present evaluations of player playstyle to help identify blunders

#### 14.1.4 Critical Success Factors

- Lead developer attends and is actively involved within bi-weekly meetings.
- Lead developer dedicates approximately 15 hours per week to project.
- Lead developer maintains Gantt chart, Kanban board, and risk register.
- Lead developer develop familiarity with technology – practice coding.
- Diverse client involvement when testing deliverables.
- Stay agile!

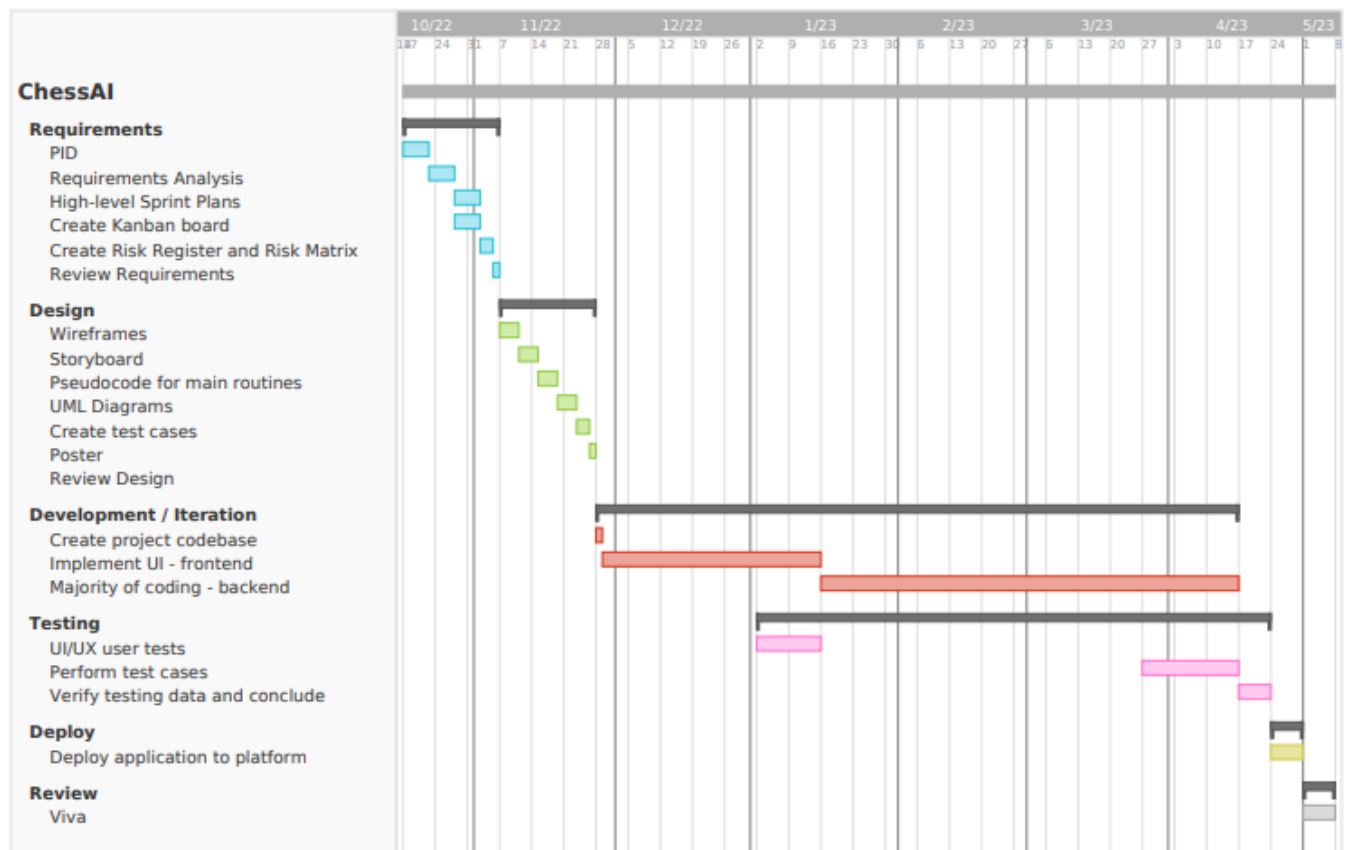


## 14.1.5 Roles and Responsibilities

It is the duty of the Project Lead Developer to ensure that they fulfil the following roles and responsibilities:

Role	Responsibility
Risk Manager	Identify and describe risks within Risk Register. Categorize risks within Risk Assessment Matrix. Define mitigation/response strategies.
Project Manager	Plan the execution of the project. Schedule tasks via Kanban Board. Keep track of progress of project by being aware of deadlines and comparing work done to initial Gantt chart.
Developer	Produce deliverables (prototypes, meeting reports, code) and present them to Project Supervisor. Ensure that deliverables are correct and well-tested (where possible) by reviewing them.

## 14.1.6 Proposed Gantt Chart



## 14.2 Risk Register and Assessment Matrix

### 14.2.1 Risk Register

- All the following risks will have Responsibility assigned to the project Lead Developer.
- Detailed Mitigation and Contingency plans can be found on the GitHub repository [in progress].

ID	Date Raised	Description	Category	Likelihood	Impact	Priority
R1	20/10/2022	User stories badly defined – scope too great, not concise.	Planning	High	High	High
R2	20/10/2022	User stories prioritised incorrectly – main functionality is not the core focus.	Planning	High	High	High
R3	20/10/2022	Test cases aren't detailed enough.	Testing	High	Medium	Medium
R4	20/10/2022	Documents / code stored on local storage becomes corrupted.	Technical	Low	High	Low
R5	20/10/2022	Unplanned work – requirements are identified later within the project.	Planning / Development	High	Medium	Medium

R6	20/10/2022	Technical issues preventing code/documents to be pushed to remote repository.	Technical	Low	Low	Low
R7	20/10/2022	Lead developer unable to attend bi-weekly meetings.	Schedule	Low	Medium	Low
R8	20/10/2022	Project supervisor is unavailable.	Schedule	Low	Low	Low
R9	20/10/2022	Behind on schedule – deadline has passed but deliverable is not complete.	Planning / Schedule	Medium	High	Medium
R10	20/10/2022	Lead developer has insufficient knowledge on relevant subjects.	Planning / Development	High	Low	Low
R11	30/01/2023	Assignment deadlines for other modules are approaching and conflict with sprints.	Planning / Schedule	High	High	High
R12	30/01/2023	Project member is having personal life difficulties and finding it hard to focus on or start on sprint items.	Personal	High	High	High
R13	30/01/2023	Vacations are upcoming and may conflict with sprints.	Planning / Schedule	High	Medium	Medium
R14	30/01/2023	Unknown assessment dates stop planning decisions from	Planning /	High	High	High

		being made effectively, with justification.	Schedule			
R15	30/01/2023	Changes to technology being used; technology needed is not well defined.	Technology	Medium	High	Medium
R16	30/01/2023	No access to computer so can't work on code.	Technical	Low	High	Low
R17	30/01/2023	No access to computer so can't work on documentation.	Technical	Low	Medium	Medium

# 14.2.2 Risk Matrix

- Find additional information about each risk by finding corresponding ID within the Risk Register.

		Impact		
		Low	Medium	High
Likelihood	High	R10	R3    R5    R13	R1    R2    R11 R12
	Medium			R9    R15
	Low	R6    R8	R7    R17	R4    R16

## 14.3 Requirements Analysis Document

### 14.3.1 Stakeholders

Stakeholder Interest and Impact Table			
Stakeholder	Interests	Estimated Project Impact	Estimated Priority
End-users	Use application to improve at chess.	High +	1
	Use application for entertainment.	High +	
Project Supervisor(s)	Ensure project is on track.	High +	5
	Provide advice to primary stakeholders.	High +	
Project Manager(s)	Improve soft skills.	Medium +	3
	Expand project management knowledge.	Medium +	
	Make sure deadlines are met.	High ?	
Project Owner(s)	Improve soft skills.	Medium +	4
	Ensure prototypes meet user needs.	High +	
Project Developer(s)	Improve hard skills, expand technologies.	Medium +	2
	Expand project portfolio.	Low +	
	Ensure code is not buggy.	High +	

Within the *Stakeholder Interest and Impact table* it is identified that the end-user is the most important stakeholder – this is because they are the ones that will interact with prototypes throughout the delivery of the project, and they are the target audience for the completed chess application.

## 14.3.2 Technology

Minimum Viable Product (MVP)		
Item	Category	Justification
Python 3	Language	Has many open-source libraries and tools that make implementing AI and Machine Learning easier.
Javascript	Language	Implement the interactivity of the web app. Frontend development.
HTML CSS	Language	Used to implement the structure/design of the web pages. How content is displayed. Necessary for web development.
Flask	Web app framework	Very lightweight, compared to its alternative Django. Flask is one of the most popular web-app frameworks – lots of support and increase employability.
Atom	IDE / text editor	Easy to use, easy to configure, GitHub integration, multiple files open at once, open projects.
GitHub	Repository	Store documentation and code online within Cloud storage. Easy to access and maintain – version control.
SQL Server	Database	Industry standard relational database management system. Vast number of resources, and lead developer already has experience.
Trello	Kanban	Kanban board used for project management - track tasks.

## 14.3.3 High Level Requirements

Functional requirements describe features and system behaviour. The end-user should be able to use these features to perform specific actions. On the other hand, non-functional requirements identify criteria that the system can be judged upon – often to describe a systems capability.

### Functional Requirements

No.	Description
1	User can play Chess against an AI.
2	User can choose to start with White or Black pieces.
3	User can select difficulty range of AI.
4	User can see their personal win rate against AI.
5	User can see statistical analysis at the end of each match.
6	User can select match type (rapid, blitz).
7	User can play against AI mimicking selected professional player.
8	User can Abandon or Surrender the game (counts as loss).
9	User can view history of previous games played.
10	User can create profile or be a guest visitor.
11	Application will be a Single Page Web Application.
12	Application will have homepage.
13	Application will have login / create account page.
14	Application will have Play Chess page.
15	Application will have profile page.
16	Profile will display username, picture, statistics, match history, badges.
17	User can earn Achievements/Badges by beating AIs at different ratings.
18	Database will store user profile information.
19	Database will store user login information.
20	Application will be accessible via desktops.
21	Application will be accessible via smartphone.
22	Strong password checking when creating account.



23	Application locks user out when they attempt too many incorrect logins.
----	---

## Non-Functional Requirements

No.	Description
1	Application must be accessible and online 24/7.
2	Application will be native English.
3	Application should support hundreds of people accessing it at the same time.
4	Pages should load within 5 seconds.
5	Customer should log in within 10 seconds.
6	Customer should be locked out after 5 invalid login attempts.

## 14.3.4 User Stories

As a General User...	I want to play chess on my smartphone so that I can play wherever I am (outside, in bed, on the sofa).
	I want to play chess on my desktop/laptop so that I can have the match full screen.
	I want to change the language of the application so that I can use it in my native languages.
	I want to create a profile so that I can see all my statistics and match history in one place.
	I want to customise my profile's display name and profile picture so that my profile is personalised.
	I want to be able to share my profile with other people so that they can my statistics.
	I want to make my profile private so that only I can see it.
	I want to play as a guest so that I can remain anonymous and don't have to create an account or sign in.
	I want to unlock badges and display them on my profile, so that I can show off my achievements and progress.
	I want to change and update my password so that my account is secure.

As a person with a disability ...	I want to use my screen reader so that I can read the text on the screen because I have a visual impairment.
	I want to be able to change the colour of the chess board and/or pieces so that I can differentiate them because I am colour blind.
	I want to be able to navigate the website using keyboard because I have a motor disability.
	I want to be able to play chess matches using algebraic input so that I can use my keyboard instead of my mouse because I have a motor disability.
	I want to input my chess moves using my voice because I have a motor or vision disability.

As a Chess Player...	I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.
	I want to learn new openings so that I can improve at the game.
	I want to choose to start with white or black pieces, so that I can practice different openings.
	I want to select the rating of my opponent, so that I can train against easier or harder opponents.
	I want to view my personal statistics such as win rate so that I know if I am improving or not, and gain confidence to keep playing chess.
	I want to review a match that I just played, so that I can see where I made mistakes/blunders.
	I want to select and play different match types (bullet/blitz/rapid), so that I can play either short or long games.
	I want to play against chess professionals, so that I can challenge myself.
	I want to abandon the match, so that I can stop playing and do something else, or because I feel like the game is lost.
	I want to view my match history so that I can review past games / see if I'm on a loss or win streak.

1. The persona General User has been explicitly identified as their needs/desires are not solely those of a Chess Player and could reflect the wants of a user that has no idea what Chess is.
2. Chess Players could have been further decomposed into Beginners and Experienced, however they share the same goal: to improve at the game, and therefore their needs are similar.

## 14.3.5 MoSCoW Prioritisation

MUST	I want to play chess on my desktop/laptop so that I can have the match full screen.
	I want to create a profile so that I can see all my statistics and match history in one place.
	I want to customise my profile's display name and profile picture so that my profile is personalised.
	I want to be able to share my profile with other people so that they can my statistics.
	I want to make my profile private so that only I can see it.
	I want to play as a guest so that I can remain anonymous and don't have to create an account or sign in.
	I want to change and update my password so that my account is secure.
	I want to use my screen reader so that I can read the text on the screen because I have a visual impairment.
	I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.
	I want to learn new openings so that I can improve at the game.
	I want to choose to start with white or black pieces, so that I can practice different openings.
	I want to select the rating of my opponent, so that I can train against easier or harder opponents.
	I want to view my personal statistics such as win rate so that I know if I am improving or not, and gain confidence to keep playing chess.
	I want to review a match that I just played, so that I can see where I made mistakes/blunders.
	I want to select and play different match types (bullet/blitz/rapid), so that I can play either short or long games.
	I want to abandon the match, so that I can stop playing and do something else, or because I feel like the game is lost.
	I want to view my match history so that I can review past games / see if I'm on a loss or win streak.

SHOULD	I want to unlock badges and display them on my profile, so that I can show off my achievements and progress.
	I want to be able to change the colour of the chess board and/or pieces so that I can differentiate them because I am colour blind.
	I want to be able to navigate the website using keyboard because I have a motor disability.
	I want to be able to play chess matches using algebraic input so that I can use my keyboard instead of my mouse because I have a motor disability.
	I want to play against chess professionals, so that I can challenge myself.
COULD	I want to play chess on my smartphone so that I can play wherever I am (outside, in bed, on the sofa).
	I want to change the language of the application so that I can use it in my native languages.
	I want to input my chess moves using my voice because I have a motor or vision disability.
WONT	None

## 14.3.6 Product Backlog

Date	Component	Item	Priority
15/11/22	Accessibility	I want to play chess on my desktop/laptop so that I can have the match full screen.	High
15/11/22	Accessibility	I want to play chess on my smartphone so that I can play wherever I am (outside, in bed, on the sofa).	Low
15/11/22	Accessibility	I want to use my screen reader so that I can read the text on the screen because I have a visual impairment.	High
15/11/22	Accessibility	I want to change the language of the application so that I can use it in my native languages.	Low
15/11/22	Accessibility	I want to input my chess moves using my voice because I have a motor or vision disability.	Low
15/11/22	Accessibility	I want to be able to change the colour of the chess board and/or pieces so that I can differentiate them because I am colour blind.	Medium
15/11/22	Accessibility	I want to be able to navigate the website using keyboard because I have a motor disability.	Medium
15/11/22	Accessibility	I want to be able to play chess matches using algebraic input so that I can use my keyboard instead of my mouse because I have a motor disability.	Medium
15/11/22	Accessibility	I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.	High
15/11/22	Account	I want to create a profile so that I can see all my statistics and match history in one place.	High
15/11/22	Account	I want to play as a guest so that I can remain anonymous and don't have to create an account or sign in.	High
15/11/22	Account	I want to change and update my password so that my account is secure.	High
15/11/22	Profile	I want to customise my profile's display name and profile picture so that my profile is personalised.	High
15/11/22	Profile	I want to be able to share my profile with other people so that they can see my statistics.	High
15/11/22	Profile	I want to make my profile private so that only I can see it.	High
15/11/22	Profile	I want to view my personal statistics such as win rate so that I know if I am improving or not, and gain confidence to keep playing chess.	High


15/11/22	Profile	I want to unlock badges and display them on my profile, so that I can show off my achievements and progress.	Medium
15/11/22	Match history	I want to view my match history so that I can review past games / see if I'm on a loss or win streak.	High
15/11/22	Matchmaking	I want to choose to start with white or black pieces, so that I can practice different openings.	High
15/11/22	Matchmaking	I want to select the rating of my opponent, so that I can train against easier or harder opponents.	High
15/11/22	Matchmaking	I want to select and play different match types (bullet/blitz/rapid), so that I can play either short or long games.	High
15/11/22	Matchmaking	I want to abandon the match, so that I can stop playing and do something else, or because I feel like the game is lost.	High
15/11/22	Learn	I want to review a match that I just played, so that I can see where I made mistakes/blunders.	High
15/11/22	Learn	I want to learn new openings so that I can improve at the game.	High
15/11/22	Design/Plan	Wireframes	High
15/11/22	Design/Plan	Storyboard	High
15/11/22	Design/Plan	UML Diagrams	High
15/11/22	Design/Plan	Risk Register and Matrix	High
15/11/22	Design/Plan	Technology poster	High
15/11/22	Design/Plan	Use-case scenarios	High

# 14.4 Wireframes

## 14.4.1 Index




# 14.4.2 Login/Create Account



CHESS . AI

Go back...



Create Account

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.

Username

Email

Password

Confirm Pass


Username...

Email...

Password...

Confirm Password...

Create Account



Login

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.


Username

Password

Username...

Password...

Login



Continue as Guest...

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo.

Click here to continue



# 14.4.3 Match Creation



14.4.4 In Match

<player\_black>

<player\_white>

Bot Difficulty

1000

▼

Game Time

5 min

▼

⏮

⏪

⏩

⏭

#	Black	White
1.	<move>	<move>
2.	<move>	<move>
3.	<move>	<move>
4.	<move>	<move>
5.	<move>	<move>
6.	<move>	<move>
7.	<move>	<move>
8.	<move>	<move>
9.	<move>	<move>
10.	<move>	<move>
11.	<move>	<move>
12.	<move>	<move>
...		

Enter move...

RESIGN

<player\_black>

00:00

8								
7								
6								
5								
4								
3								
2								
1								
	a	b	c	d	e	f	g	h

<player\_white>

00:00

### 14.4.5 Profile (History)



### 14.4.6 Profile (Medals)



# 14.4.7 Profile (Friends)



# 14.4.8 Learn (Basics)



CHESSEI









Learn the Basics



1. King



2. Pawn



3. Rook



4. Bishop



5. Queen



6. Knight



7. Castling



8. En Passant



9. Check and Checkmate



10. Stalemate

Learn

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi.

1

Learning the Basics

2

Explore Openings

3

Key Advice

# 14.4.9 Learn (Openings)

CHESSEI

Explore Openings

1. e4 Sicilian Defense

1. e4 French Defense

1. e4 Ruy López Opening

1. d4 Queen's Gambit

1. d4 Slav Defense

1. d4 King's Indian Defense

1. ...

1. ...

1. ...

Learn


Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi.

1 Learning the Basics


2 Explore Openings


3 Key Advice


# 14.4.10 Learn (Key Advice)




# CHESS . AI









## Key Advice

1.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.

2.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.

3.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.

4.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi. Aliquam non suscipit nisl. Praesent posuere ipsum felis, sed porttitor elit aliquet in.

## Learn

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla ut vehicula leo. Sed placerat nunc quis erat tempor semper. Nulla facilisi.

1

Learning the Basics

2

Explore Openings

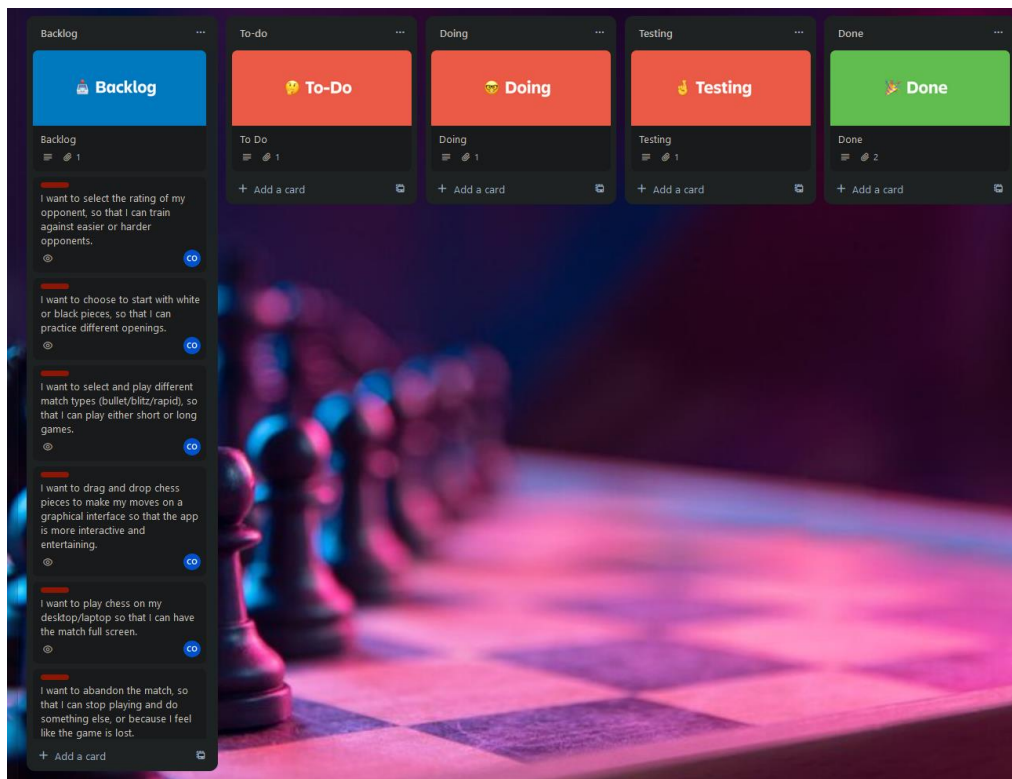
3

Key Advice



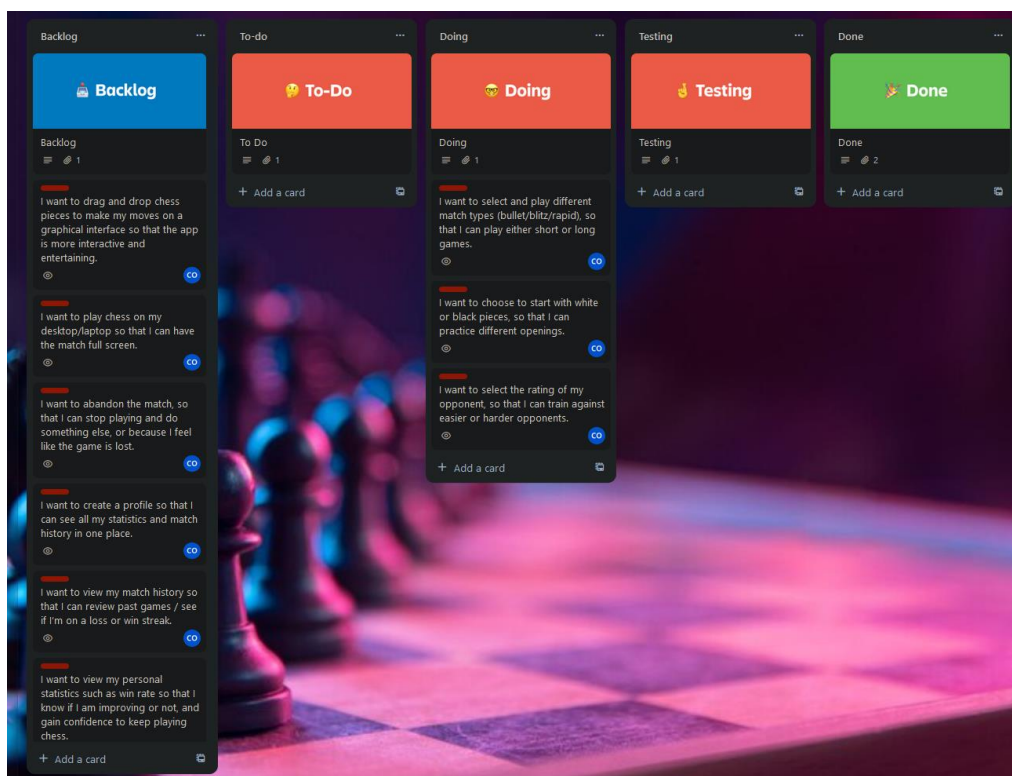
## 14.5 Kanban Board

### 14.5.1 Starting Board



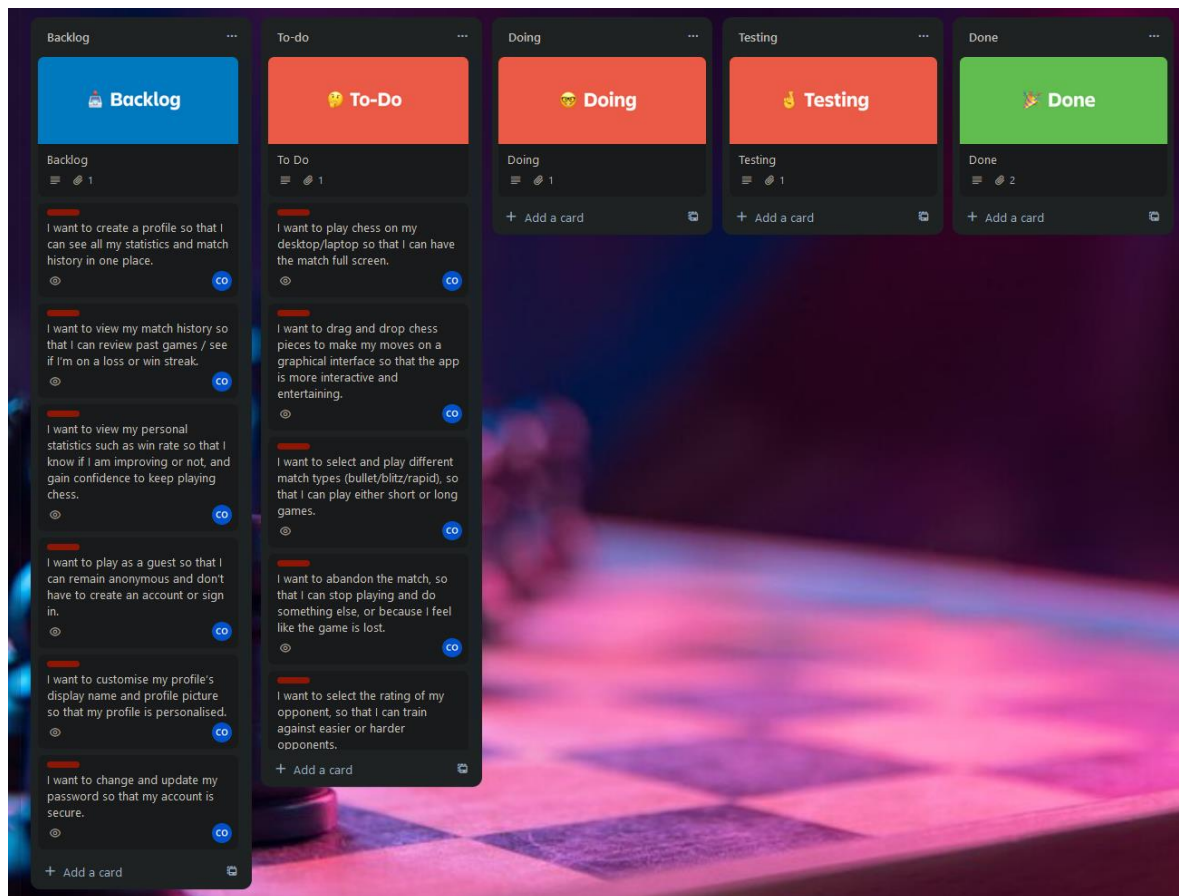
### 14.5.2 Chronological Progression of Boards

Implementing match control features.

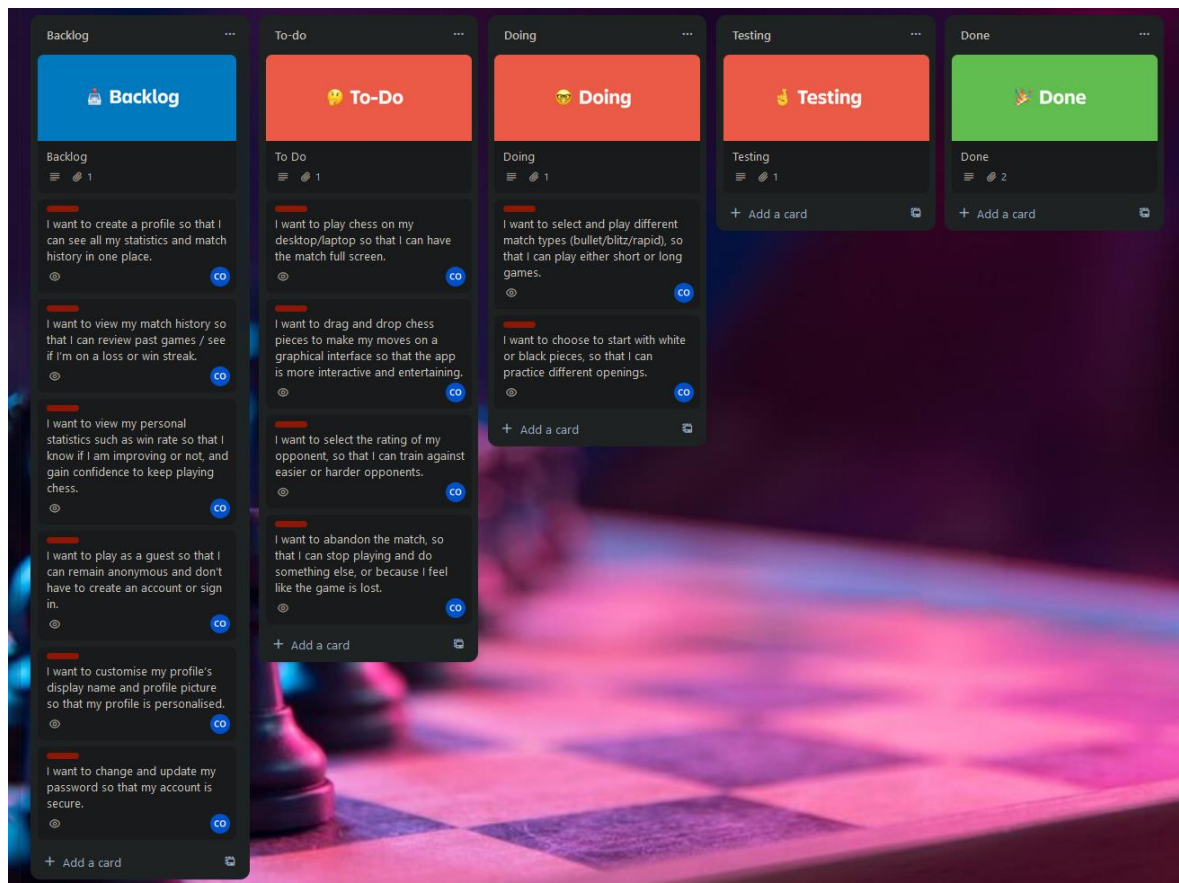




Gathering Chess game relevant items.

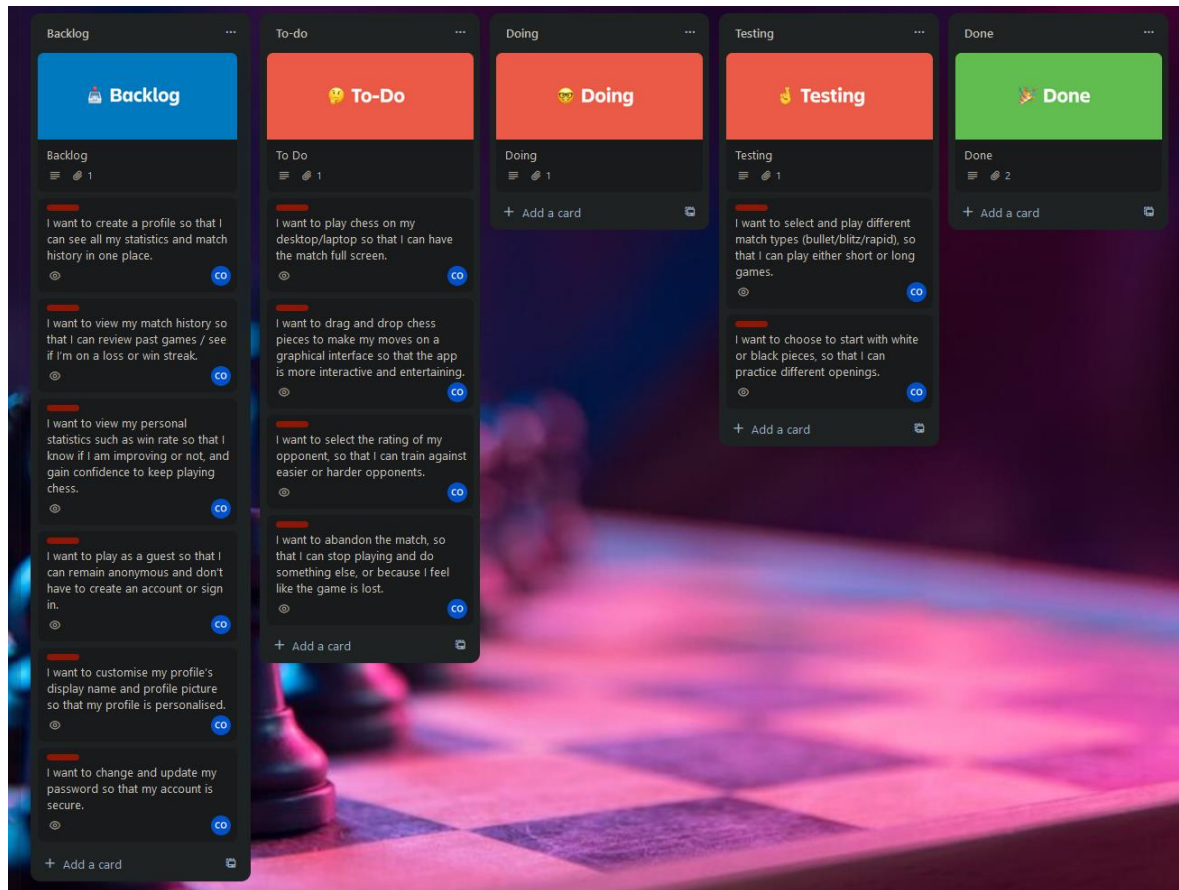


Implement match control features. Player can select Black or White. Player can select time control.

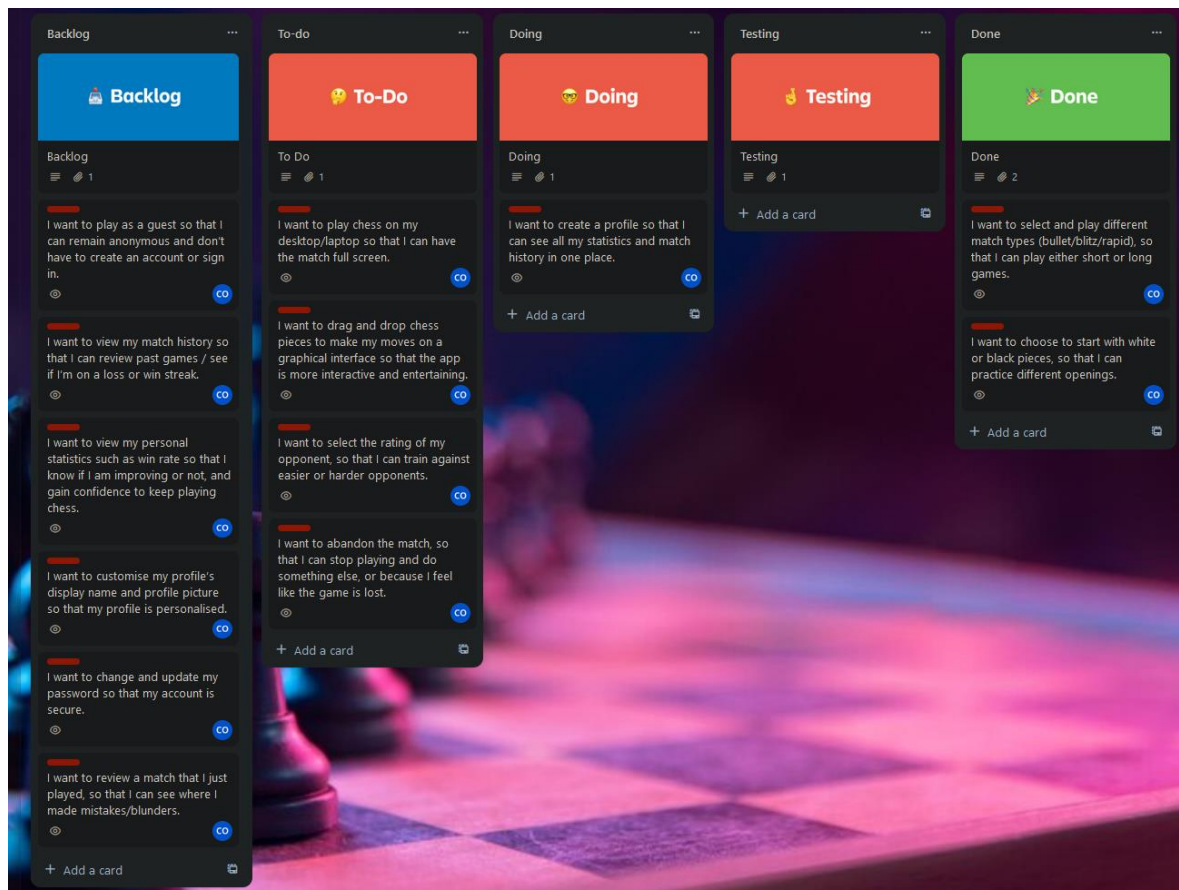




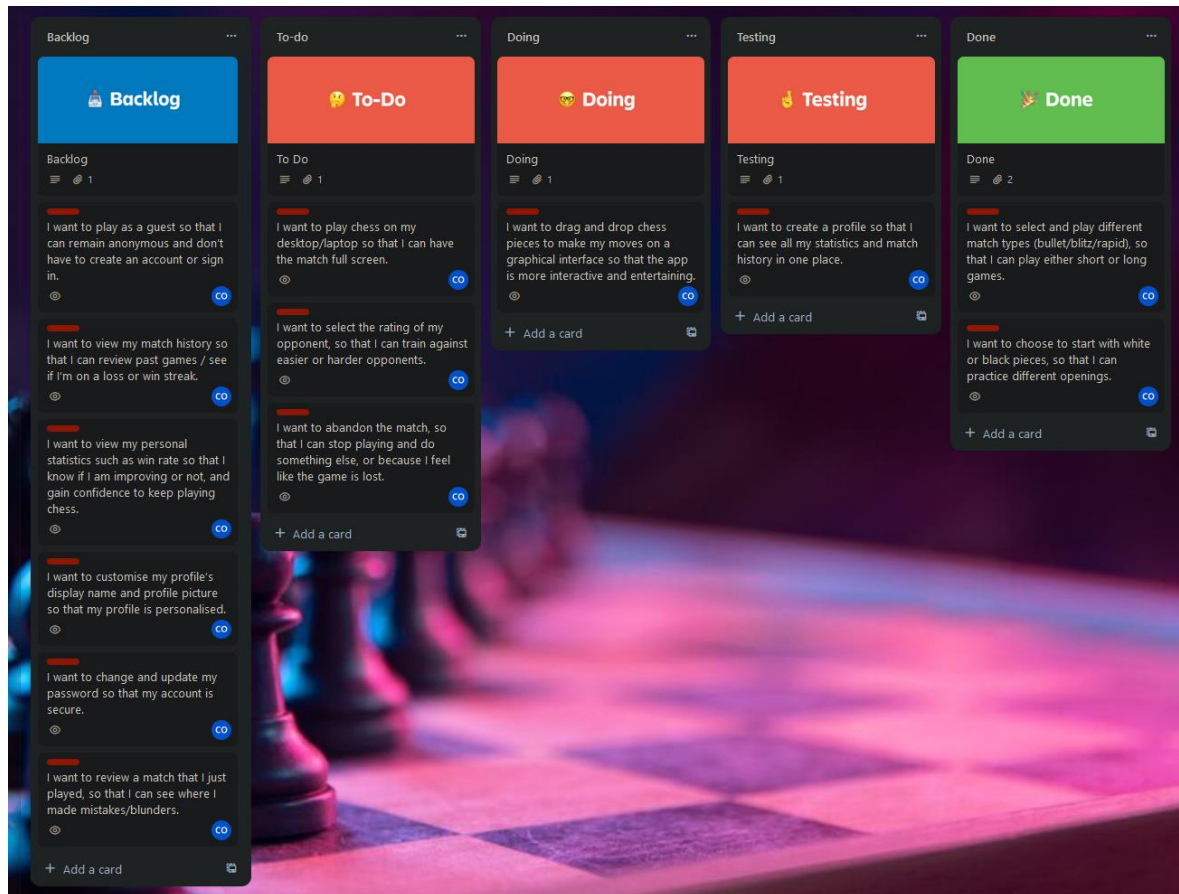
## Testing Chess match control criteria.



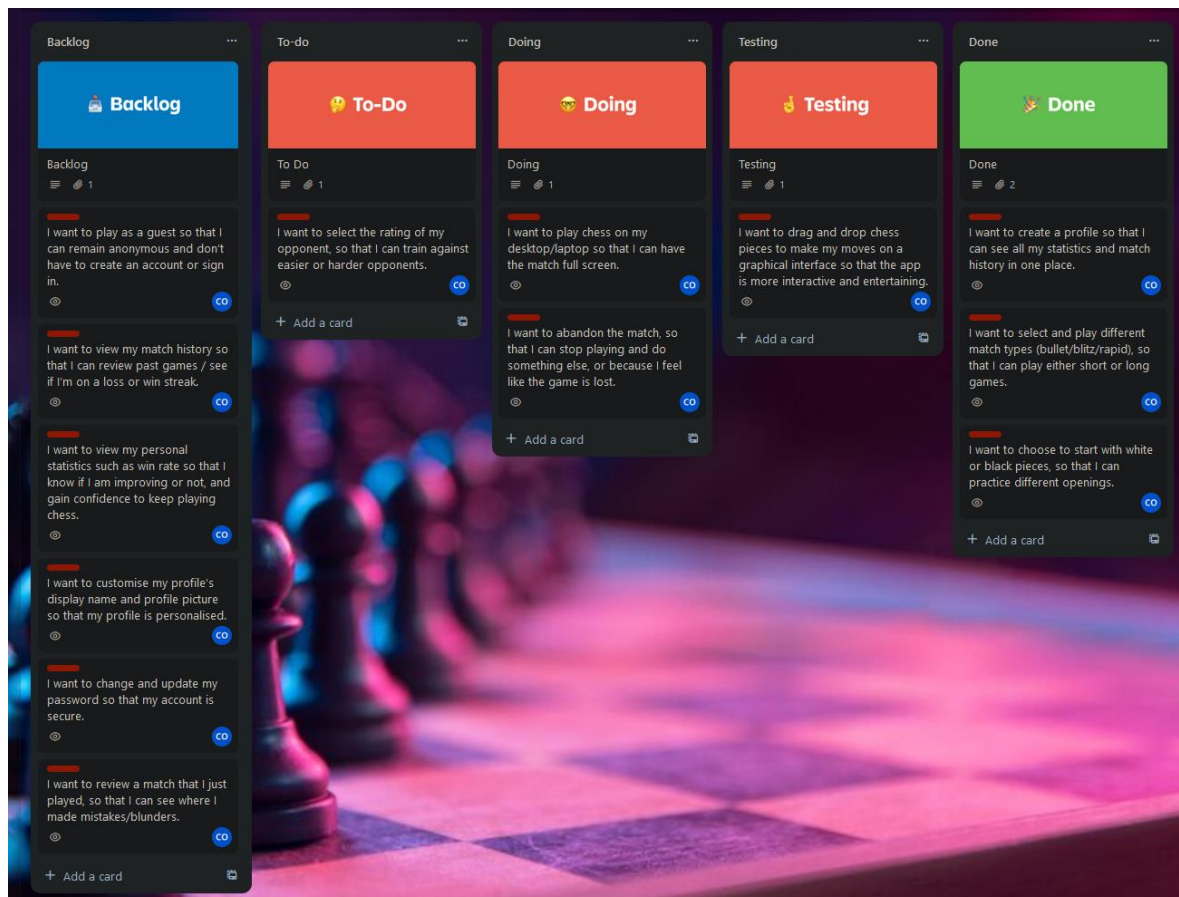
## Match control done! Implementing profile creation/login/authentication.



Testing authentication. Start implementing Chess frontend GUI.

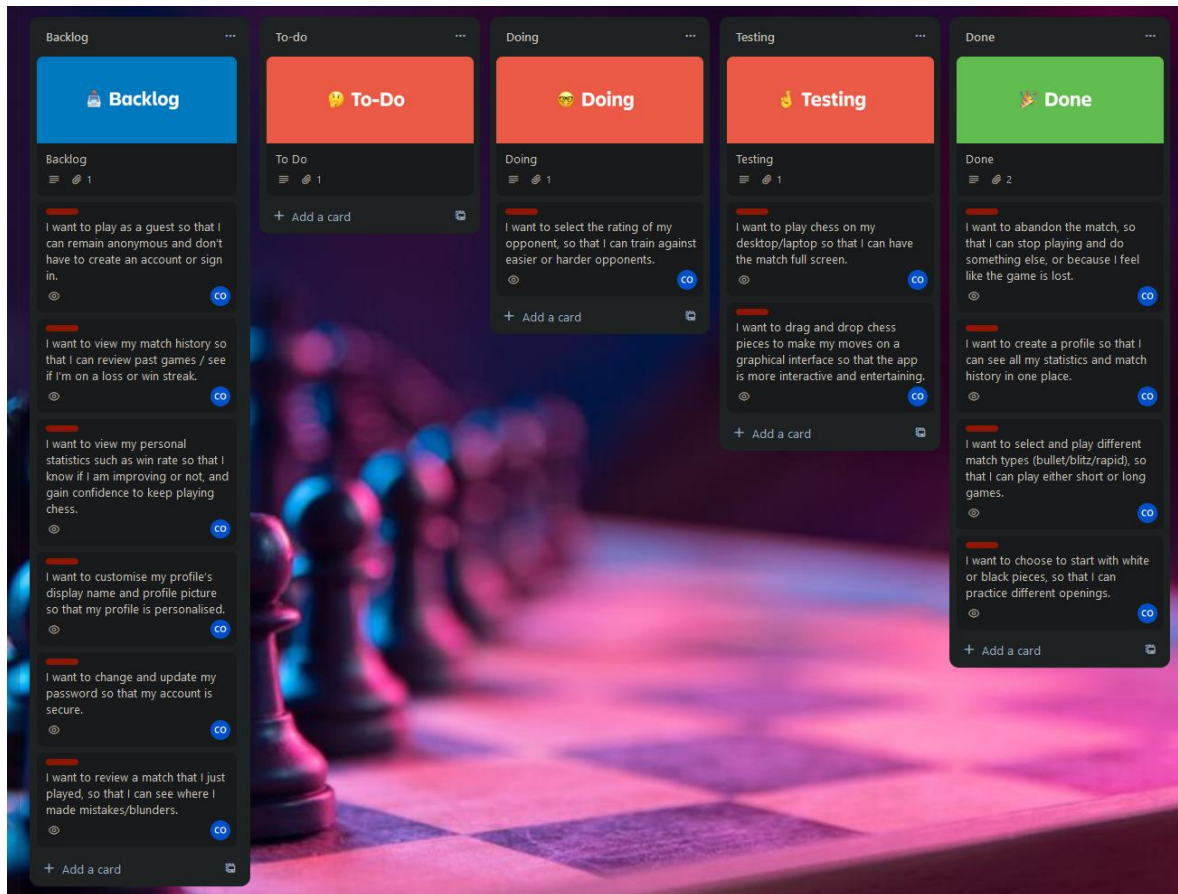


Testing GUI. Start implementing Chess backend and game state. Win/Lose/Draw/Resign.

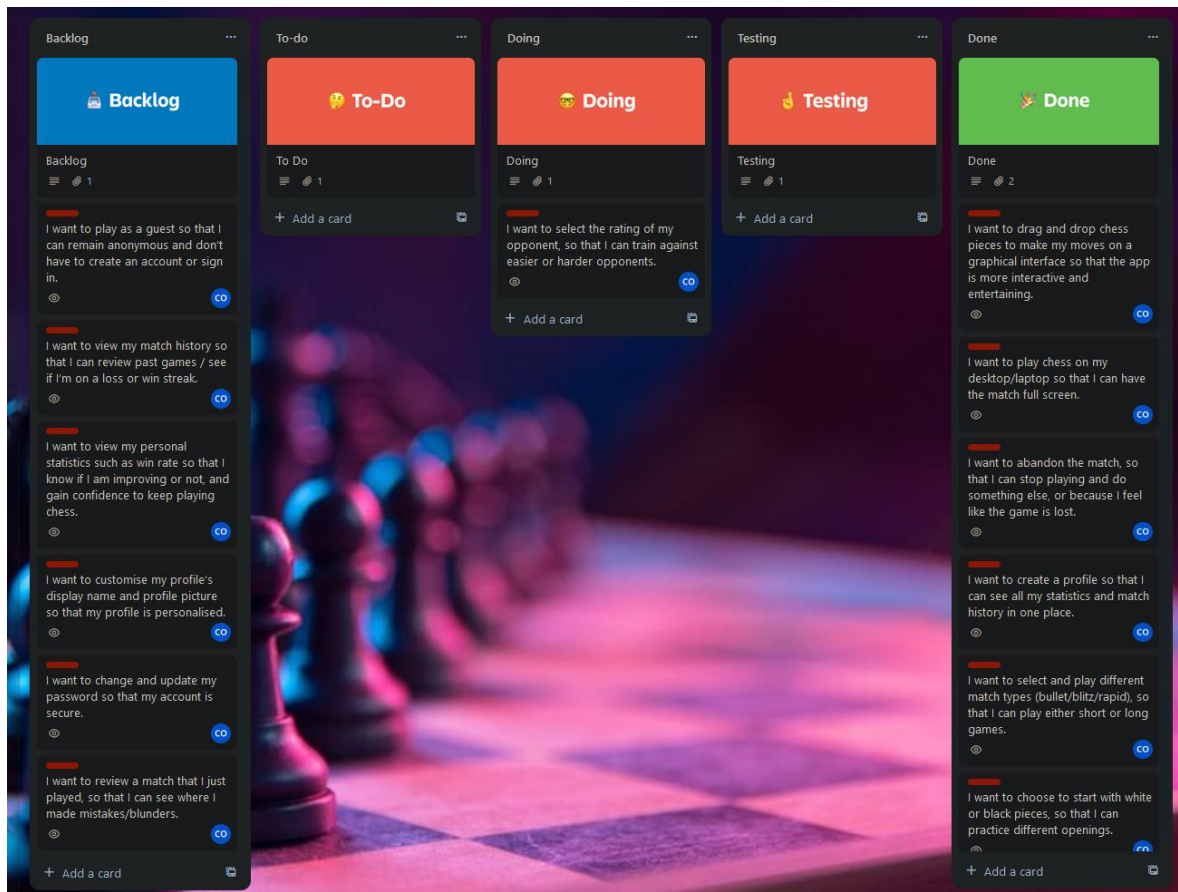




Developing AI opponent move. Meanwhile testing GUI and move validation.

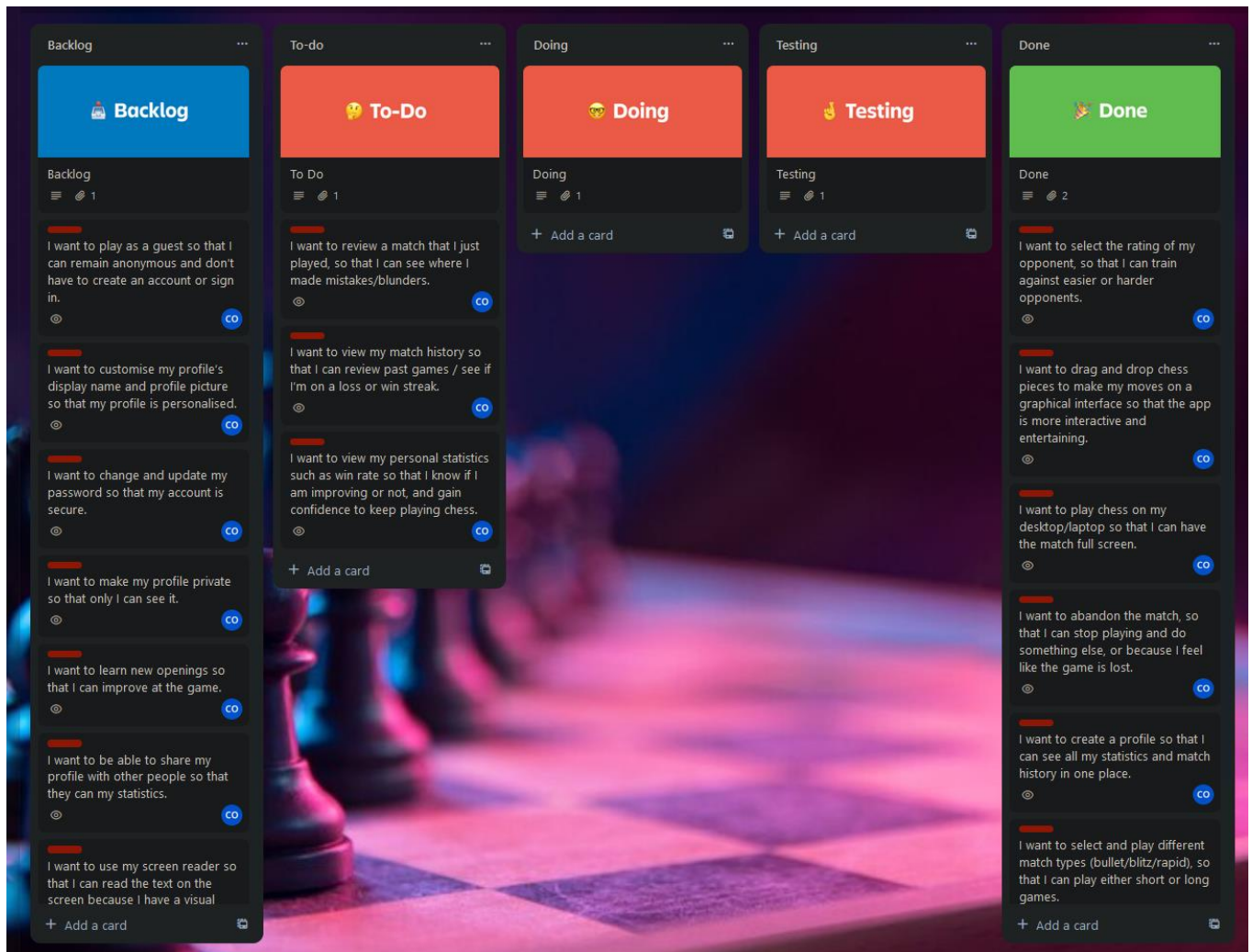


GUI and move validation correct. Checkmate etc working. Still working on AI.

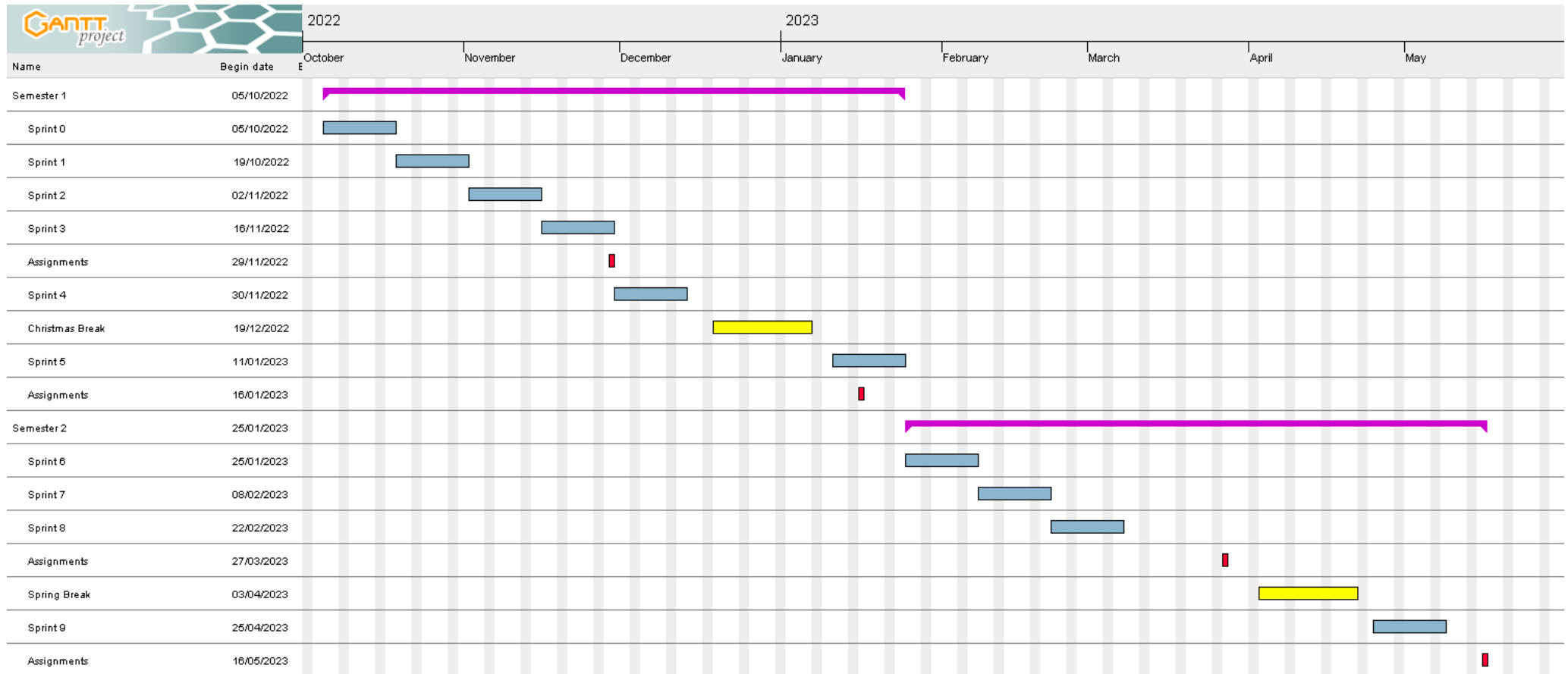


## 14.5.3 Final Board

AI opponents implemented.



## 14.6 Sprint Schedule



This figure shows a roadmap for all sprints that occurred within the entire academic year, where:

- Blue boxes represent sprints.
- Red boxes represent assignment deadlines.
- Yellow boxes represent vacations/breaks.
- Purple boxes represent semesters.



## 14.7 Sprint Plans and Retrospections

### 14.7.1 Sprint Zero

#### SPRINT 0

Start: 5/10/2022

End: 18/10/2022

##### Sprint Goal:

Prepare for the project.

##### Sprint Work:

1. Project Initiation Document.
2. Create GitHub repository.

##### ITEM 1 TASKS:

- Project vision, goals, and objectives.
- Risk assessment, matrix, and register

##### ITEM 2 TASKS:

- Make a private GitHub repository.
- Add supervisor as collaborator.

---

##### Additional Notes:

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

## 14.7.2 Sprint One

### SPRINT 1

---

Start: 19/10/2022

---

---

End: 01/11/2022

---

#### Sprint Goal:

Start investigating the initial requirements analysis.

#### Sprint Work:

1. Create Requirements Analysis Document.
2. Create Kanban board.

#### ITEM 1 TASKS:

- Identify high level requirements.

#### ITEM 2 TASKS:

- Create Kanban board on Trello.
- Create following columns: backlog, to-do, doing, review, testing, done.

---

#### Additional Notes:

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

## 14.7.3 Sprint Two

SPRINT 2

Start: 02/11/2022

End: 15/11/2022

Sprint Goal:

Continue and complete investigating the initial requirements analysis.

Sprint Work:

1. Finalize Requirement Analysis Document.

2. Add tasks to Kanban board.

ITEM 1 TASKS:

• Write out high level requirements as user stories.

• Prioritise user stories – ~~MoSCoW~~.

• Write out Product Backlog.

ITEM 2 TASKS:

• Move Product Backlog items onto the Trello Kanban Board.

• Colour items based on their priority.

Additional Notes:

• Kanban board: <https://trello.com/b/TubtD2KW/chessai>

• GitHub repo: <https://github.com/ORG4N/ChessAI>

### Retrospection:

Accomplished everything that I planned for within the sprint plan. Next sprint I should split the work over the two weeks instead of rushing to finish last minute. Also, try to plan workloads around other module's coursework deadlines.

## 14.7.4 Sprint Three

### SPRINT 3

Start: 16/11/2022

End: 29/11/2022

**Sprint Goal:**  
Start designing and developing MVP chess prototype (simple).

**Sprint Work:**

1. Define technology to be used.
2. Create initial codebase.
3. I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.

**ITEM 1 TASKS:**

- Research the most optimal language.
- Choose frameworks and justify them within Requirements Analysis Document.

**ITEM 2 TASKS:**

- Create project within appropriate editor.
- If possible, start with a template to get basic file structure.
- Push code to GitHub.

**ITEM 3 TASKS:**

- Front end development of chess board.
- Create sketches/storyboard for web page design and interaction.
- Create necessary UML diagrams.
- Create use-case scenarios.
- Test functionality against use-cases when implemented.

---

**Additional Notes:**

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>
- Assignment deadlines on: 29/11/22

**(RED ITEMS PUSHED TO SPRINT 4 BECAUSE OF APPROACHING DEADLINES – REDUCED TASKS TO ALLOW MORE TIME FOR OTHER MODULES' ASSIGNMENTS)**

### Retrospection:

Only 1 of 3 items completed. Other 2 items pushed ahead to sprint 4. Next sprint, a doable amount of work should be assigned rather than overloading the sprint with too many jobs. Also, workload should be influenced by ongoing events, such as approaching coursework deadlines. Plan ahead of time by making an abstract overall sprint plan that considers these special dates and the schedule of sprint plans across the academic year.

## 14.7.5 Sprint Four

SPRINT 4

Start: 30/11/2022

End: 13/12/2022

Sprint Goal:

Start designing and developing MVP chess prototype (simple).

Sprint Work:

1. Create initial codebase.

2. I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.

ITEM 1 TASKS:

Create project within appropriate editor.

If possible, start with a template to get basic file structure.

Push code to GitHub.

ITEM 2 TASKS:

Front end development of chess board.

Create sketches/storyboard for web page design and interaction.

Create necessary UML diagrams.

Create use-case scenarios.

Test functionality against use-cases when implemented.

Additional Notes:

Kanban board: <https://trello.com/b/TubtD2KW/chessai>

GitHub repo: <https://github.com/ORG4N/ChessAI>

Approaching Christmas break between: 19/12/22 and 08/01/23

(RED ITEMS PUSHED TO SPRINT 5)

### Retrospection:

Only 1 of 2 items completed. The uncompleted item was pushed ahead to sprint 5. Both items for this sprint were carried over from previous sprint, therefore not completing them sets me back. Productivity this sprint was low due to just completing assignments and needing to take a short break, however, I started to feel burned out and distressed. Personal life issues also happened and these issues I faced will be added to the risk register.

## 14.7.6 Sprint Five

# SPRINT 5

Start: 11/01/2023

End: 24/01/2023

**Sprint Goal:**  
Start designing and developing MVP chess prototype (simple).

**Sprint Work:**  
1. I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.

**ITEM 1 TASKS:**

- Front end development of chess board.
- Create sketches/storyboard for web page design and interaction.
- Create necessary UML diagrams.
- Create use-case scenarios.
- Test functionality against use-cases when implemented.

---

**Additional Notes:**

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

(RED ITEMS PUSHED TO SPRINT 6)

### Retrospection:

1 of 1 item partially completed. Some sketches, diagrams, and use cases of the application were made. Main issues within this sprint were a continuation of those from the other sprints wherein a mixture of personal life issues and upcoming deadlines. In the next sprint the work will need to be caught up to very quickly due to the past few sprints lacking. In the next sprint the MVP prototype will need to be developed.

## 14.7.7 Sprint Six

### SPRINT 6

Start: 25/01/2023

End: 07/02/2023

**Sprint Goal:**  
Start designing and developing MVP chess prototype (simple).

**Sprint Work:**

1. I want to drag and drop chess pieces to make my moves on a graphical interface so that the app is more interactive and entertaining.

**ITEM 1 TASKS:**

- Front end development of chess board.
- Test functionality against use-cases when implemented.

---

**Additional Notes:**

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

### Retrospection:

1 of 1 item completed. Implemented Chess board via chessboardjs and chessjs on Javascript backend. Chessboardjs displays board whilst chessjs validates positions. Chessboard was tested by me to see if it works – such as preventing illegal moves and ending when checkmate. Very little sprint work assigned here to cutdown on overloading with different tasks.

## 14.7.8 Sprint Seven

### SPRINT 7

Start: 08/02/2023

End: 21/02/2023

#### Sprint Goal:

Start developing the match control settings.

#### Sprint Work:

1. I want to choose to start with white or black pieces, so that I can practice different openings.
2. I want to select and play different match types (bullet/blitz/rapid), so that I can play either short or long games.
3. I want to select the rating of my opponent, so that I can train against easier or harder opponents.

#### ITEM 1/2/3 TASKS:

- Buttons to select starting colour. Black/White/Random.
- Buttons to select length of game. 3/5/10 minutes.
- Buttons to select the opponent difficulty rating. 500/600/700 ... 1800/1900/2000.
- Validation for each selection.
- Implement each range of possible values within database tables.

#### Additional Notes:

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

### Retrospection:

3 of 3 items completed. Player can choose whether they want white or black, which time category, and which opponent rating. The latter two choices are loaded dynamically from the database by the flask framework. Buttons are added dynamically. The input of each button is validated, and the program makes sure that the submitted values exist within the database. Prevents users from cheating by changing values on their browser. Lots of work done this sprint.



## 14.7.9 Sprint Eight

### SPRINT 8

Start: 22/02/2023

End: 08/03/2023

#### Sprint Goal:

Start developing and designing the database.

#### Sprint Work:

1. Design database for:
  - a. Users and Profile pages.
  - b. Chess matches.
  - c. BOT AI opponents
2. Create database.
3. Test database / Use to create fill in match control options.

#### ITEM 1 TASKS:

- Research database technology and choose one.
- Create ERD diagram.
- Design each table.

#### ITEM 2 TASKS:

- Create tables for each entity.
- Link to Python Flask backend.
- Test if GET works.
- Test if POST works.

#### ITEM 3 TASKS:

- Javascript GET for all values of TIME and BOT DIFFICULTY.
- These values are variable whereas BLACK/WHITE is static.

#### Additional Notes:

- Kanban board: <https://trello.com/b/TubtD2KW/chessai>
- GitHub repo: <https://github.com/ORG4N/ChessAI>

### Retrospection:

3 of 3 items completed. Database implemented in SQLite. Did lots of coding and designed database. Database used to create users and matches. Followed guides on how to implement SQL database.

## 14.7.10 Sprint Nine

SPRINT 9

Start: 25/04/2023

End: 08/05/2023

Sprint Goal:

Develop the chessboard and AI.

Sprint Work:

1. I want to select the rating of my opponent, so that I can train against easier or harder opponents.

2. Chess opponents implemented as AI that are trained by data within that Elo range

ITEM 1 TASKS:

Ratings of opponents are ranged between 400 and 1700, in brackets of 100.

User can select the rating when creating a game.

ITEM 2 TASKS:

Find a dataset.

Train model.

Implement within opponent's turn within game.

Test for when AI is playing as white and black.

Additional Notes:

Kanban board: <https://trello.com/b/TubtD2KW/chessai>

GitHub repo: <https://github.com/ORG4N/ChessAI>

### Retrospection:

2 of 2 items completed. User can play against an AI at different elo ratings. However, ranges are much narrower than previously stated. Also, did not train my own model or use my own data. Instead, used the already developed Maia models just to prove concept. Future development could involve creating own models and creating new AI options alongside Maia models. Did lots of work this week.