Module 5- Computer Systems (2021-22)

Project

Software Design Document Template

Project Name: ChessMate	Team ID: 17
Team Members: Hein Huijskes, Julia van der Geest, Ujjwal Dodeja, Jose Gavilanes, Maouheb Bessi, Mengmeng Li	Mentor(s): Ramish Bhutto, Wenjie Zhao

Activity diagram: Maouheb
Class diagram: Joe
Use case diagram: Ujjwal
Data flow diagram: Mengmeng
Wireframing: Julia
Leftovers: Hein

Instructions:

i) You must explain all the given sections clearly and concisely.

ii) You must fill in the basic information about your projects such as Project Name, Team Members, Team ID, and Mentor(s).

iii) Make sure to consider the checklist of the Design phase provided in the Security by Design document.

iv) The length of the document should be 4 to 8 pages.

1. Introduction

ChessMate is an application that allows the user to play chess and complete challenges using only their voice. This means that the pieces on the board can be moved using easy voice commands using voice recognition software.
ChessMate allows for both private and online matches, using peer-to-peer for the latter. Alternatively a user can play against an AI.
There are already similar applications in existence, however ChessMate improves upon these in multiple fields. First off it allows for more easy voice commands to control the game. Furthermore it adds challenges, as well as leaderboards to track a player's progress or elo rating. Finally the app allows for undoing moves, in case the voice command was inaccurate or a private opponent maliciously made a move in his opponent's turn.

2. Functional/Non Functional Requirements

2.1 Functional requirements:

1. The system should be able to record voices using a microphone
2. The system should be able to relay messages using a speaker
3. The system should be able to host offline games
4. The system should be able to connect to another system online to allow for matches between players via the

internet

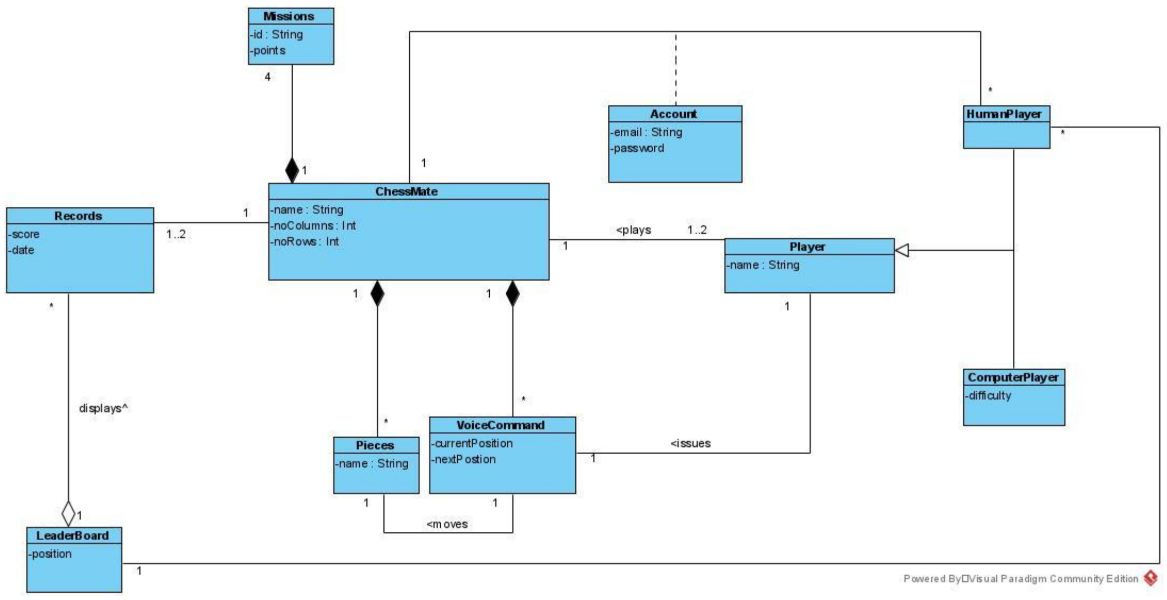
2.2 **Nonfunctional requirements:**

5. The system should be able to connect to an online elo rating system for global leaderboards by sending match results to a server/database
6. The system should be able to convert voice commands into moves in the games
7. The system should include chess challenges for users to complete
8. The system should include an extensive list of voice commands for the player to use to play the games
9. The system should include a nice interface on which to show leaderboards and games
10. The system should include multiple levels of AI difficulty to play against
11. The system should include secure player accounts protected by passwords
12. The system should be able to undo moves (in case they were made maliciously by the opponent)

3. Architectural Design

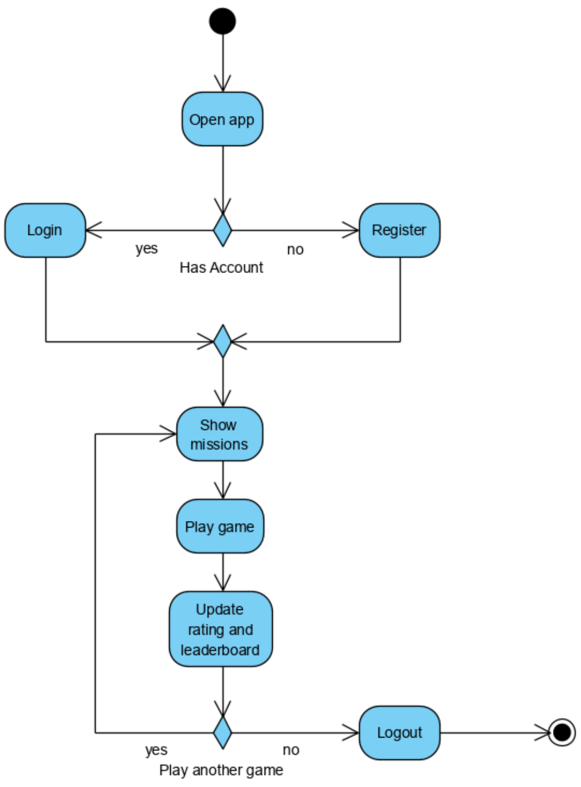
Class Diagram:

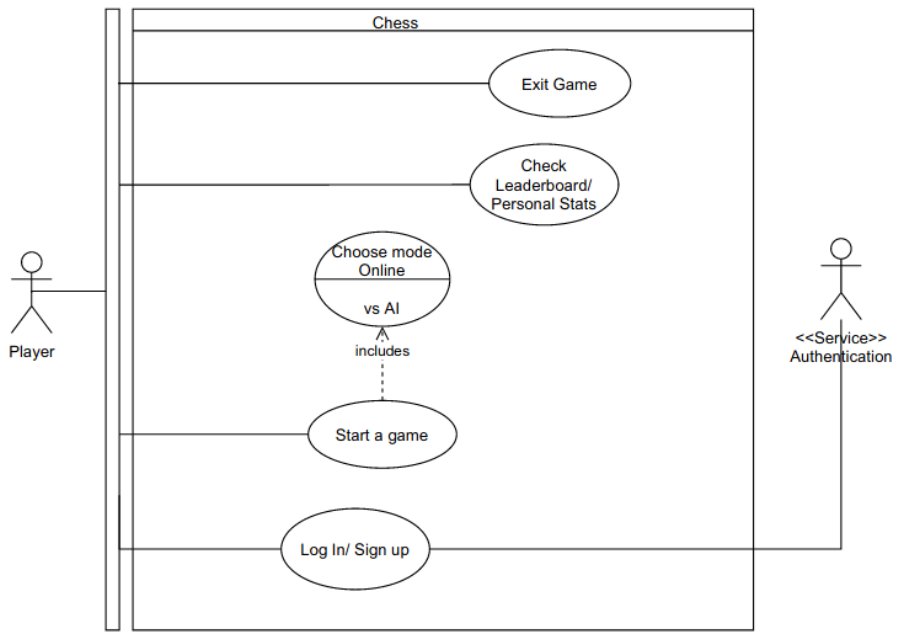
The Platform (ChessMate) is the main class where the magic occurs. A player can access the platform by creating a ChessMate account. ChessMate will have two types of players (Human players and Computer players) for which only the scores of Human players will be stored and shown in the Leaderboard. The Player issues voice commands to move the pieces, only one voice command is performed at a time which will move only one piece. Plus, the game has Missions which grant the player extra points when the mission is completed successfully.

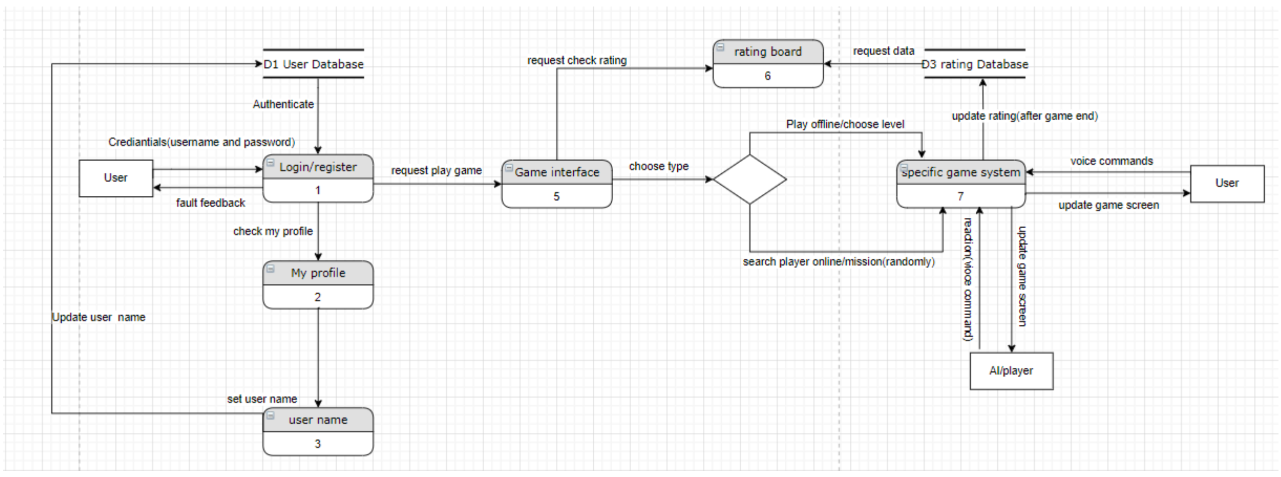


Activity Diagram:

The activity diagram shows the states of the system. Upon opening the app, the user has the option to either login or register. After this is done, the system will show a set of missions. The user can then start a game, after the game is done the system will update the leaderboard and the rating. If the user wishes to play another game, the system will go back to showing the missions, if not the system will logout.







(Create diagrams to depict a high-level impression of your system using UML diagrams that should include Use case, Class, Activity, and Data flow diagrams (at least one diagram for each category). The diagrams can be drawn using online tools, i.e. draw.io (free tool), etc.

Write the motivation for all your UML diagrams to highlight the project features and requirements. This section aims to understand the system and its basic components, how the components interact with each other, data repositories, security requirements in design such as trust boundaries, input/output points, swapping/updating firmware for fixing bugs, and adding new features to the product, etc.).

4. Product User Interface

*(In this section, you should **create illustrations using Wireframing tools** (Pencil Project, Mockplus, etc.), based on the Software Requirement Specification (SRS) of your application, **for example**, the menus required for the interface, buttons, and its tasks (Toolbar), any other functionality like events, sub-events, controls, status bar, etc.)*

https://www.figma.com/

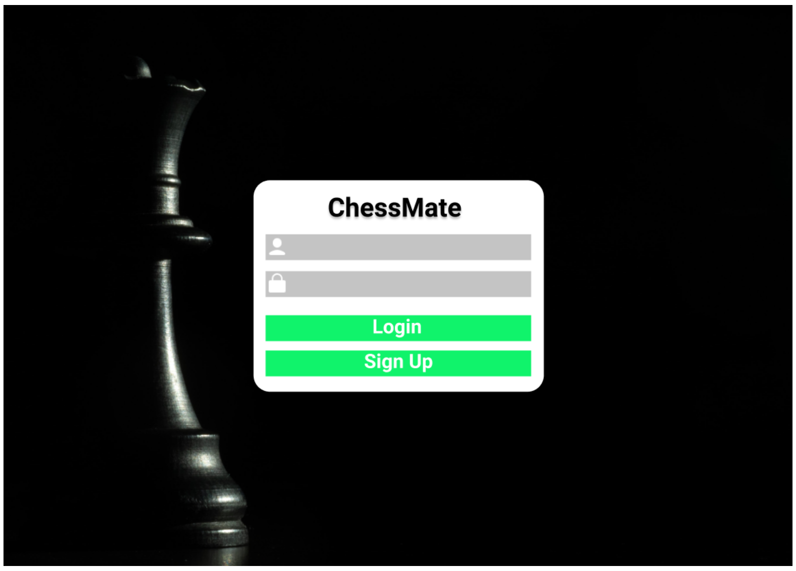


Fig 1, Login page.

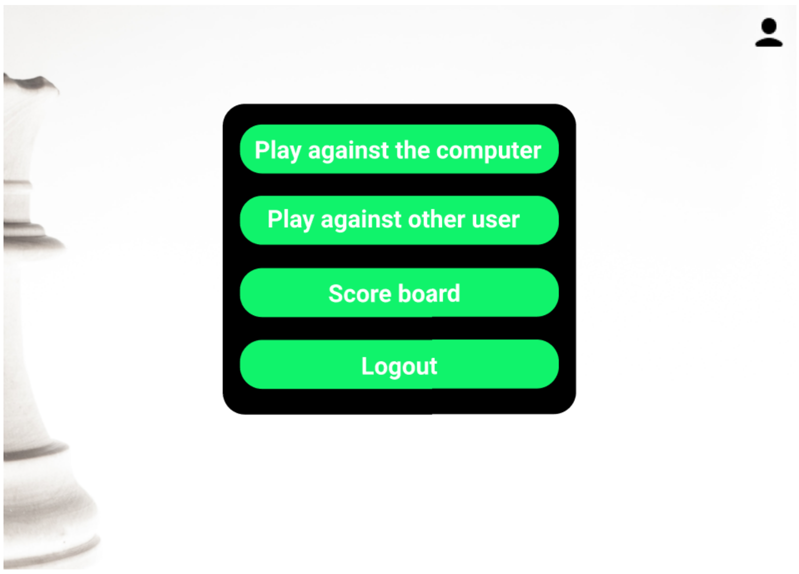


Fig 2, Home page.

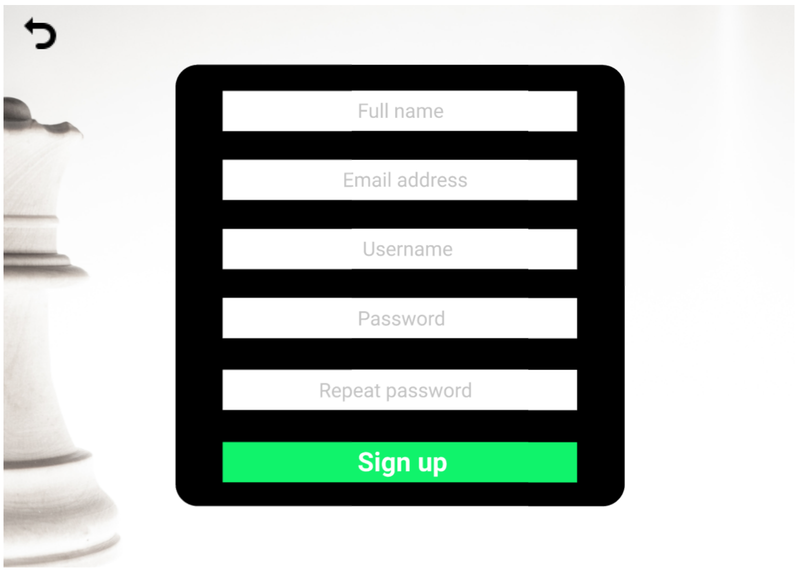


Fig 3, Sign up page.

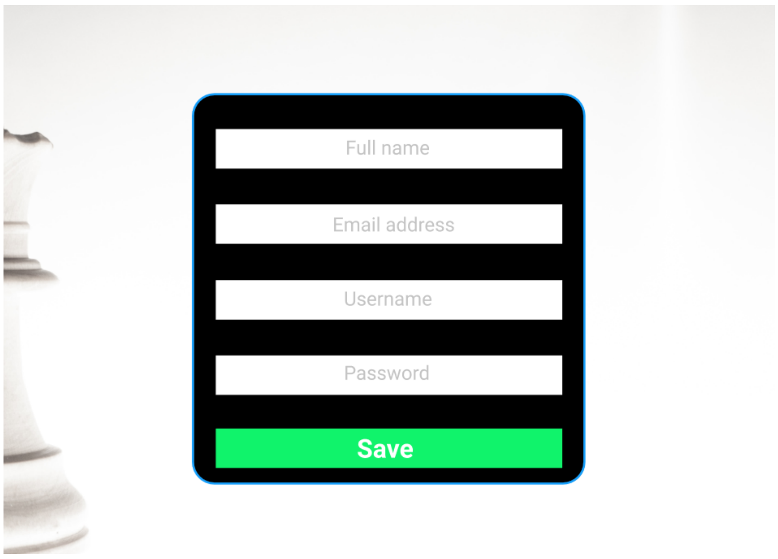


Fig 4, Profile page.

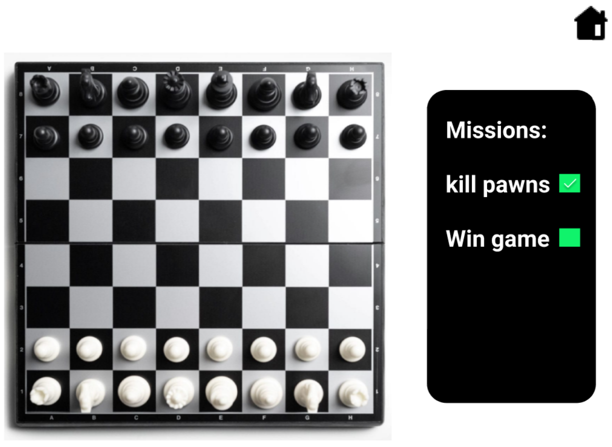
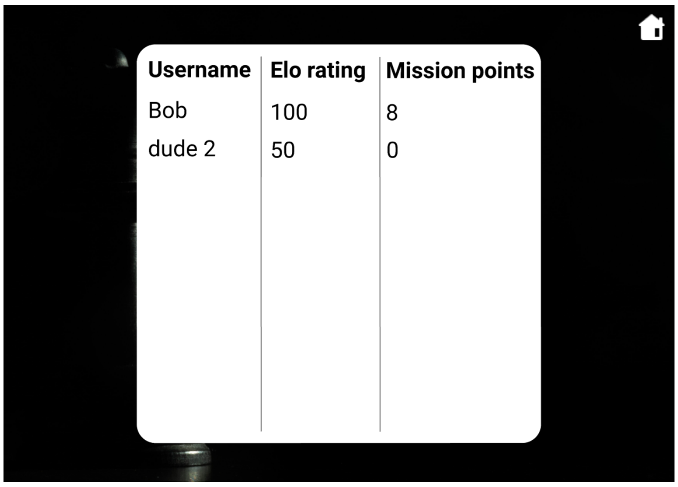


Fig 5, Actual game page.



Username	Elo rating	Mission points
Bob	100	8
dude 2	50	0

Fig 6, Scoreboard page.

https://www.figma.com/file/TmMXeoRLYKChXWq5KPfLgl/Untitled?node-id=0%3A1 This link will let you access the interactive interface. When running the interface the buttons are functional and will take you through the game.

5. Prevention/Mitigation Criteria (Security Controls)

(In this section, you should specify the criteria to avoid/ remove the risks identified in the design.)

The only two parts which might form a risk are the login and the update of the leaderboard. We must ensure that the passwords are safely stored and that the user is not able to alter the elo rating system.

6. The cost involved (if any):

The system requires a raspberry pi, connected to an interface, a microphone and a speaker.
The time constraint for this system is that it will be built within the timeframe of the project, which means around 8-10 hours a week per team member for 10 weeks.

7. Conclusion:

This project has gone through quite some iterations of different ideas already. At the core is always the voice recognition component, which allows for a completely different way of playing games. Furthermore the challenge system will introduce a new edge to chess, which the game normally wouldn't have.
Integrating the voice recognition might be a challenge, although for this mostly existing libraries or APIs will be used. A more interesting problem is implementing the entire game of chess, as well as writing a decent AI for it. This is quite a challenge that will require some time to figure out completely. Finally, getting players to connect to each other online and play games against each other could also be challenging depending on the implementation.

References:

- Sanjeev, A. (2018, 23 march). The best voice recognition software out of three we tested, and how to set it up on Raspberry Pi. maker. https://maker.pro/raspberry-pi/tutorial/the-best-voice-recognition-software-for-raspberry-pi
- E. (2021, 9 februari). *Setup a Raspberry Pi MYSQL Database*. Pi My Life Up. https://pimylifeup.com/raspberry-pi-mysql/
- Corke, T. (2021, 15 augustus). Building Your Own Team Rating System. Matter of Stats. http://www.matterofstats.com/mafl-stats-journal/2013/10/13/building-your-own-team-rating-system.html