I need to first do the project definition. I need around 2 paragraphs but they must be very concis­­­­e and focussed. I will first look at the exemplar.

I need to define the problem:

* Chess is broadly played by adults but can be played by children of the age 8+
* Chess bots allow users to play a chess game against a computer. This could serve to:
  + Demonstrate some new AI technique etcetera
  + To allow people to practice
  + To entertain as a pass time

How are chess bots used:

Can be used for practice

Used as a prompt in a user vs user game to help you practise by showing you what moves you should have made. Players can also potentially play against AI for practise.

But the most important function of a chess program is entertainment

* Many people just play chess for entertainment

So my purpose is entertainment:

Many people play chess. According to chess.com they have over 15 million users. While they do provide computer chess games for ‘training’ the majority of users play for entertainment. This is why part of there offering includes puzzles.

People enjoy playing chess as it is a challenge as it requires them to think ahead and anticipate the relative cost vs benefits of any given move.

**Who are the stake holders:**

* The only stakeholder is the user
* Anyone over the age of 8+
* Any age due to no gore
* Mainly adults
* Anyone looking for entertainment

Specifically

* My dad, he is tech savvy and so could suggest new features. He also regularly uses a chess program on his phone
* My mum, she is not tech savvy but will be able to help me with the user experience and ensuring that my user interface is self-explanatory and easy to use
* George could give feedback: could be very technical and specific

**How is it solvable by computational techniques:**

**How can I use abstraction:**

* **Previous moves abstracted:** The chess engine only needs to be given the current board state in order to determine the next best possible move. For example the history of the last few moves the user and then computer made are irrelevant. This is because all the information needed to make the next move is on the chess board and not hidden. The program will not analyse the users moves as it will assume they will play optimally to reduce complexity
* **Pieces are abstracted in the backend:** A chess piece can be abstracted as how it looks, its design and exact colour are irrelevant. The only relevant attributes of a non-king piece is:
  + **Who owns the piece:** Which player it belongs to
  + **What possible moves can it make:** The set of vector
  + **What is the relative value of each piece**
* **The board is abstracted:** The board can be abstracted to a coordinate grid in terms of x and y or vector space. The colours of the squares can also be ignored.

**Thinking Ahead:**

Inputs:

* Menu options in navigating the website and for example selecting a difficulty setting. This could be in the form of a dropdown or radio buttons
* A click on a piece and then on a square on a chess board to indicate that the user wishes to move that piece there.

Outputs:

Preconditions:

**Decomposition:**

The 2 core parts of the solution are:

* A chess engine that has a API interface: e.g. web sockets. This program will not take inputs from the user but will be responsible for determining the next move to make
* A web interface in the form of a website that will allow users to interact with the chess board. The main function of this is to provide a GUI

Chess engine:

Website GUI interface: