1. **Purpose**

The purpose of this project is to create a functioning Reversi game. Specifically, this Reversi game should provide the user the ability to play against an AI or to pit two AI against each other over the network. The Reversi game should have a functioning GUI on the client side to display the game state and provide game options such as undo, new game, and AI difficulty. The game should also have a server side that any game player will connect to. The AI and all of the game logic will be on the server side so that the client will be as thin as possible.

Beyond the immediate deliverables of this project, the purpose is to become familiar with a SCRUM development environment. SCRUM is used quite frequently as a code development technique thus it is important to be familiar with it. Additionally, iterative development methods like SCRUM can be more powerful than the traditional waterfall method used especially when coding as it provides a much more flexible development environment.

1. **High Level Entities (Descriptions few sentences and Interaction between two and roles)**

The interaction between the high level entities begins most logically between the server and the client. The server will contain all of the game code and will execute all of the game logic as well as keeping the state of the game. The client will solely be responsible for providing input to the server to drive the game. The server then, or rather the individual threads of the server, each of which will be responsible for serving one client, will simply be waiting for an input and then will calculate the new game state and execute a move and then send the move back to the client.

For the server to calculate a game state and to execute a move it will rely on the other two major entities, the game and the AI.

The client’s interaction with the game and AI entities is relatively limited in that it never directly interacts with either entity, but acting through the server the client entity does interact with both the game and AI. The input the client gives to the server will modify the game entity which will in turn cause the AI entity to “react” differently in that a different optimal move will be calculated. Thus the client’s action propagate to both the game and the AI and actually drive both of them.

* + **Server**
    - **Sockets**
      * Sockets will be used to communicate with the client.
    - **Child Processes**
      * Each individual child process will handle
  + **Client**
    - **GUI**
  + **Game**
    - **Board**
    - **State**
  + **AI**
    - **Search Algorithm**

1. **Low Level Entities (Usage (Interface, Config -> If needed, Model (UML) Diagram, Interaction Diagram)**
   * **Server**
     + **Sockets**
     + **Child Processes**
   * **Client**
     + **GUI**
   * **Game**
     + **Board**
     + **State**
   * **AI**
     + **Search Algorithm**

1. **Benefits, Assumptions, and Risks (5-6 top benefits of the design, a list of ALL known risks/issues and a list of ALL assumptions.)**
   * **Benefits**
     + Game design is significantly abstracted due to the use of classes. This also increases the ease of maintainability and debugging.
     + Provides a GUI for an easier more intuitive user interface.
     + Common command protocol enables the Reversi server to communicate with other reversi servers and pit the AI against each other.
     + Different forms of searching and different search parameters allow for easy to set game difficulties.
     + Providing multi-threading on server for performance reason as well as allowing simultaneous play.
   * **Assuptions**
     + Assumed that the server is running in a Unix environment so that it will be capable of producing multiple threads.
     + Assumes that the program will be running in a graphical environment.
   * **Risks**
     + Providing a GUI removes the ability for the game to be played purely from a command line interface. Users without a functioning windowing system will not be able to play the game.
     + Server relies on fork() command which is only available on \*nix based operating systems.
     + Communication between user and server will be unencrypted making the game session vulnerable to cheating or interference.