1. State the purpose of your project/sub-system

1.1 Purpose

The purpose of this document is to describe the implementation of the BubbleChess software system described in the BubbleChess software Design. BubbleChess is a program designed to bring users together to play a game of chess . The creation of this system will allow players to reach much more diverse competition from their usual opponents. With the ability of the history and user system players will have the ability to learn from past mistakes, keep track of their record as a player, and build a profile as a player across the community. With this consistent information, any average player will get the essential practice and statistics needed to become one of the greats.

1.2 Scope

This document describes the initial design of the BubbleChess software system. While this document provides a baseline design for the system, as requirements change so will the design of the system. This document will not describe the playing and rules of the actual game itself, rather it will outline the structure and design of the system used to represent the game in the virtual environment as well as the GUI objects necessary for creating an enjoyable user experience.

Description of Problem

A common problem that chess players face when looking for an opponent is that they can not physically meet in person. Additionally, the standard chess game does not keep track of information such as game history and user data. This information can be useful in a players growth in the game of chess. With BubbleChess these problem no longer exist. Using a simple internet connection users will be able to play against another opponent anywhere in the world. Additionally, they will be able to keep track of game history and statistics.

2. Define the high level entities in your design

Figure 1 depicts the high-level system architecture. The application will consist of many distinct components. These components are listed below:

- **User(1 and 2)**: Consists of two separate players that will be play the game via the application
- **Client User Interface**: The graphical interface that the user will interact with. It will communicate directly with the client backend
- **Client Backend**: The backend client code to the client that will handle the game state, pieces, board, and calculate the moves. This will also communicate with the server to send game history, player moves, and login information.
- **Java Web Server**: The backend server code that will handle the object from the client backend to achieve the login, game history, and player move tasks from one cient to another.
- **Server Processing**: The layer that will communicate with the database directly converting the information from the webserver to valid sql statements
- **Database**: The interface for storing and receiving data for the server

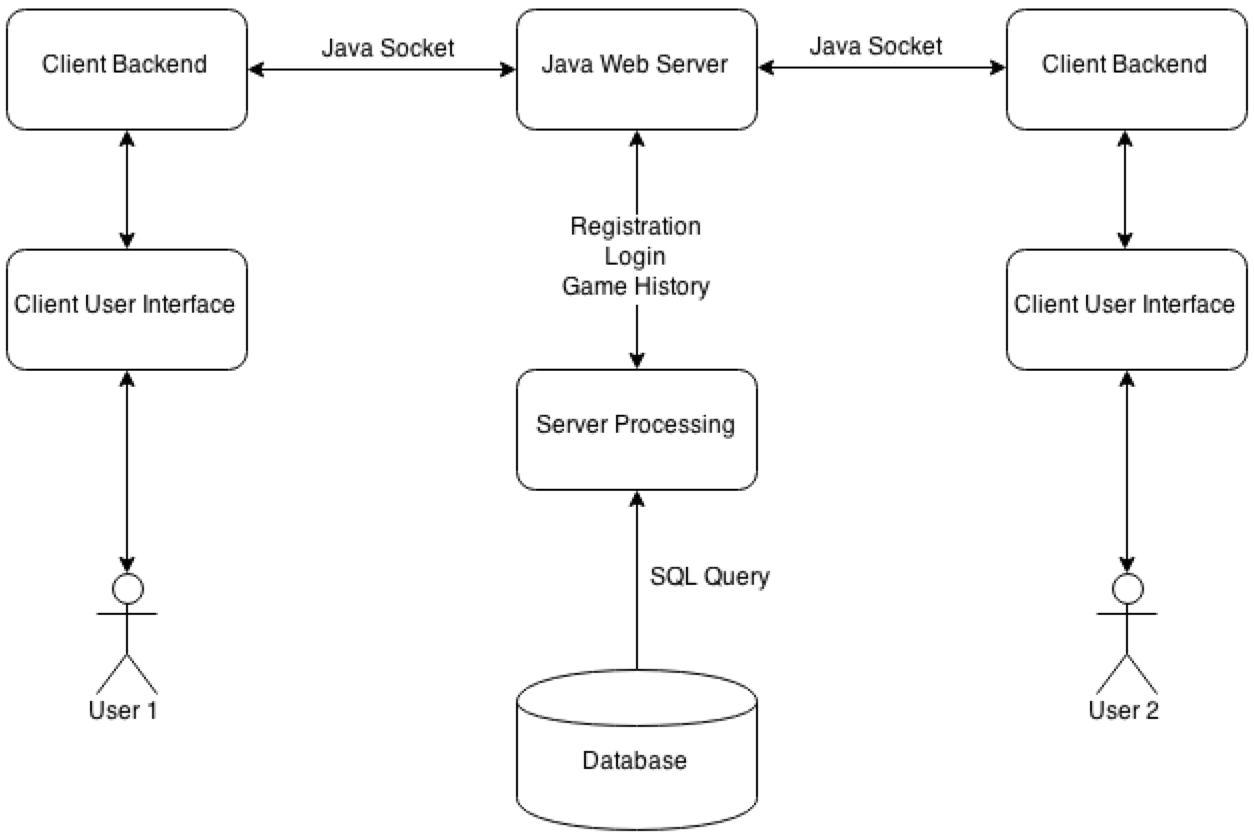

Figure 1

In the figure above you can see the basic flow of how the system works. The user interacts directly with the Client User Interface, that interface will then interact with the client backend and calculate the information needed to send to the Java Web Server. Here the server will make requests to the Server Processing interface which will get/input information from the Database through a SQL Query. This information will then be sent back to the webserver and to the respective User through the reverse process.