Software Requirements Specification

for

Online Multiplayer Board Game

Version 1.0

Prepared by

Group Name: Wolverines

Ryan Paulos	**11354677**	**Ryan.paulos@wsu.edu**
Damon Willingham	**11747649**	**Damon.willingham@wsu.edu**
Yousef Kitali	**11536847**	**Yousef.Kitali@wsu.edu**
Jingyu Cao	**11696853**	**Jingyu.Cao@wsu.edu**

Date: 11/6/2020

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
1.0	Ryan Paulos Yousef Kitali Damon Willingham Jingyu Cao	This SRS version encapsulates all key details of the product. Any further revisions will be in accordance with the Agile method of discovering and addressing problems dynamically.	11/6/2020

1 Introduction

1.1 Document Purpose

As a requirement for progressing through the computer science program at WSU Vancouver, students are required to successfully complete a semester-long project whereby sound software engineering principles are applied toward the development of a software product. These principles include the documented enumeration of software requirement specifications found herein.

1.2 Product Scope

The Online Multiplayer Board Game shall be a web-based game server that allows players to compete against one another in a game of tic tac toe. The system will provide a basic matchmaking system whereby players are paired together and a game instantiated, maximizing the ease with which the world's tic tac toe enthusiasts can pursue their passion.

The system should support and require each user to create a set of login credentials. Such a feature would foster community by providing each player with a public-facing alias and profile page.

1.3 Intended Audience and Document Overview

This document is intended for consumption by the CS320 TA and/or our professor. The remainder of the document contains sections providing an overall description of the product, specific requirements providing for the function of the software product, and a final section describing the products non-functional requirements. Each section stands alone in providing information as labeled, though reading the document in order provides the best context for understanding subsequent sections.

1.4 Definitions, Acronyms and Abbreviations

API	Application Programming Interface.
CSS	Cascading Style Sheets
FAQ	Frequently Asked Questions
GB	Gigabyte
GUI	Graphical User Interface
HTML	HyperText Markup Language
IEEE	Institute of Electrical and Electronics Engineers.
JS	JavaScript
RAM	Random Access Memory
REST	Representational State Transfer
SRS	Software Requirement Specification. This document.
SQL	Structured Query Language
TA	Teaching Assistant
TDD	Test Driven Development
WSU	Washington State University

1.5 Document Conventions

This document conforms to IEEE formatting conventions: Use Arial font size 11 throughout the document for text. Document text is single spaced and maintains 1” margins.

Conventions	Content
Font	Arial 11
Margin	1 inch
Line spacing	Single
Citation	IEEE style

1.6 References and Acknowledgments

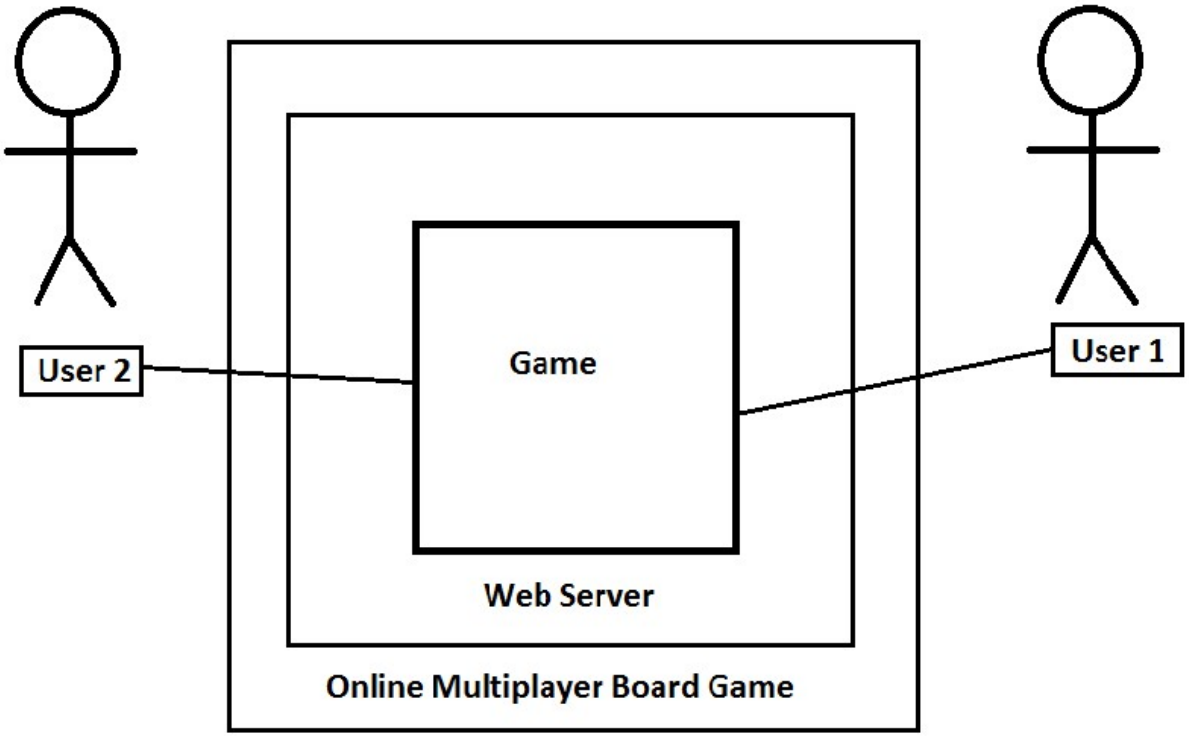
Django Software Foundation, *Django*. [Online]. Available: https://www.djangoproject.com/. [Accessed: 05-Nov-2020].

MySQL, *MySQL*. [Online]. Available: https://www.mysql.com/. [Accessed: 05-Nov-2020].

React, React – A JavaScript library for building user interfaces. [Online]. Available: https://reactjs.org/. [Accessed: 05-Nov-2020].

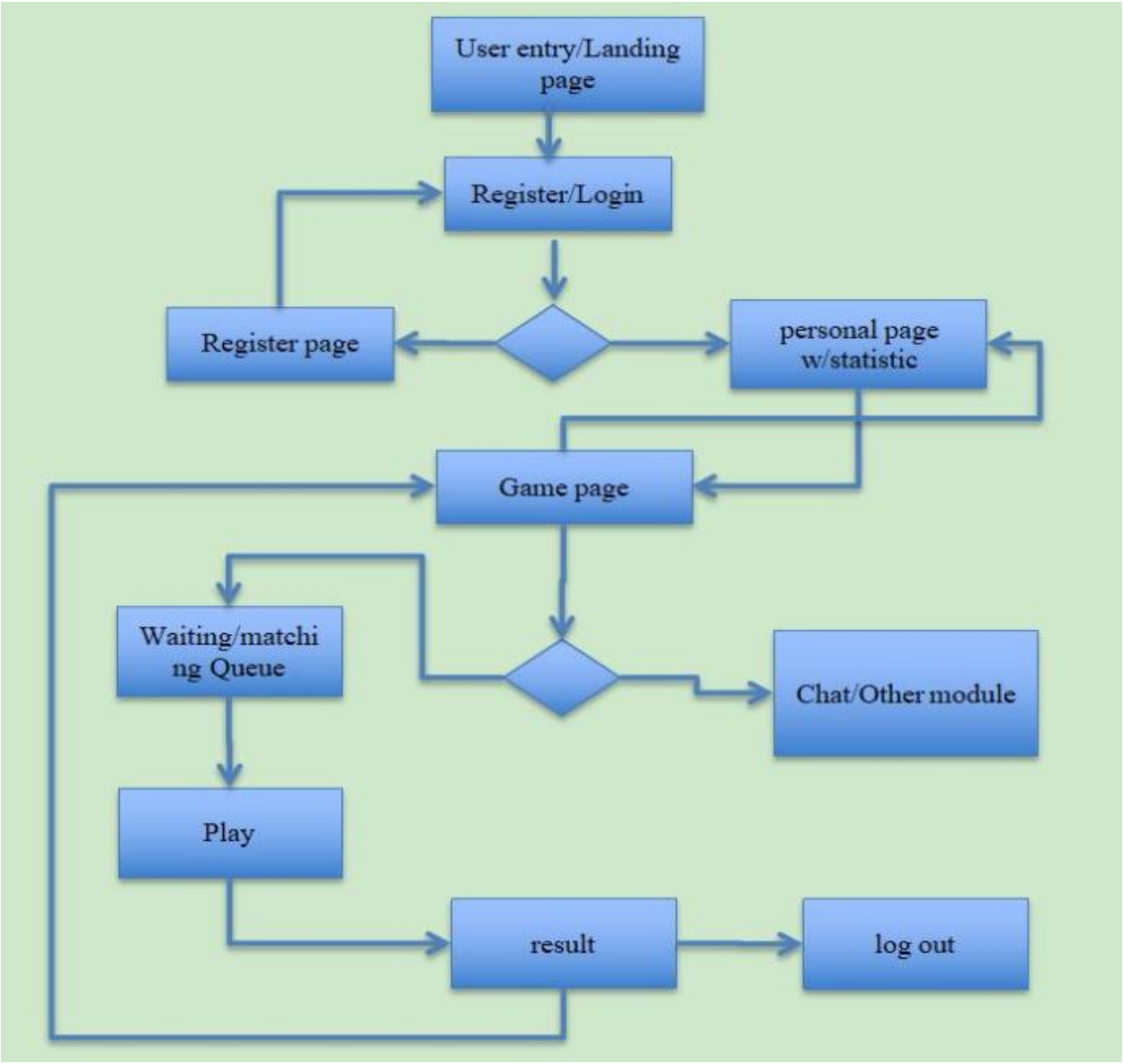
2 Overall Description

2.1 Product Perspective



The Online Multiplayer Board Game shall have two active actors and one system. User 1 and User 2 shall access the website through the Internet. The Users must sign-in to the website through a username and password. After logging into the system users will be placed into a queue. The Users are matched using a web server. The Users can then play the game together. The results of the match will be saved to a database. The Users can then queue for another game.

2.2 Product Functionality



* Shall be Browser-based GUI
* Will provide basic matchmaking to pair prospective players and initialize a match.
* Will utilize a credentialing system allowing each user a unique,persistent identity within the system.
* Shall require users to connect to the internet
* Will track Wins/Losses Statistics
* Should allow Interaction Between Users
* Shall allow Users to be placed into a Queue
* Should allow Users to view Personal Statistics

2.3 Users and Characteristics

The users are expected to be Internet literate and be able to navigate the web page. All users are equally important in the context of our product. Product Service and function will be homogeneous across the user base. The only pertinent characteristic of our user base is their knowledge of the tic tac toe ruleset. All users are expected to understand English.

2.4 Operating Environment

The back end of our system will be running on the WSU Vancouver engineering servers under the Ubuntu operating system. Minimum requirements for the server reflect the hardware available to us: An Intel Xeon CPU E5-2650 and 8 GB of RAM. Our product will rely on a Python-based back end utilizing the Django web framework to facilitate API calls with the front end. Account information will be stored within a SQL database.

The front end of our system will be written using Javascript, CSS, and html. The requirements for the front end is any system able to run <Browser chosen by TA>

Index	Description
Server hardware	Intel Xeon CPU E5-2650 and 8 GB of RAM
Client hardware	Support html access
Data management	SQL database with management software
Server application	Django web framework
Client application	JS+CSS+HTML
Network	Required

2.5 Design and Implementation Constraints

Relying on the WSUV server to host our product limits our ability to configure the operations environment should the need arise.

The knowledge/skill set of the product team will limit which tools, technologies, and languages the development team can choose from. The Online Multiplayer Gaming system has a time constraint of six weeks. The team is limited by the functions of Django and the SQL database.

2.6 User Documentation

A single FAQ-like page to inform our users of all relevant details. It would contain exposition on how to create an account, how to navigate to the user profile page and matchmaking sections, and a section describing the rules of tic tac toe.

2.7 Assumptions and Dependencies

*We are assuming our product will be supporting no more than 10 concurrent users.
* We are assuming the storage costs (disk space) of user data will not exceed 1 GB
* We are assuming Django will provide an easy way to create API endpoints for our javascript front end to interact with and a good way to interface with a SQL database. Unanticipated difficulty implementing our design with Django could negatively impact the scope of our product.
* We are assuming the school server is configured adequately and provides sufficient computing power to successfully deploy our product.

3 Specific Requirements

3.1 External Interface Requirements

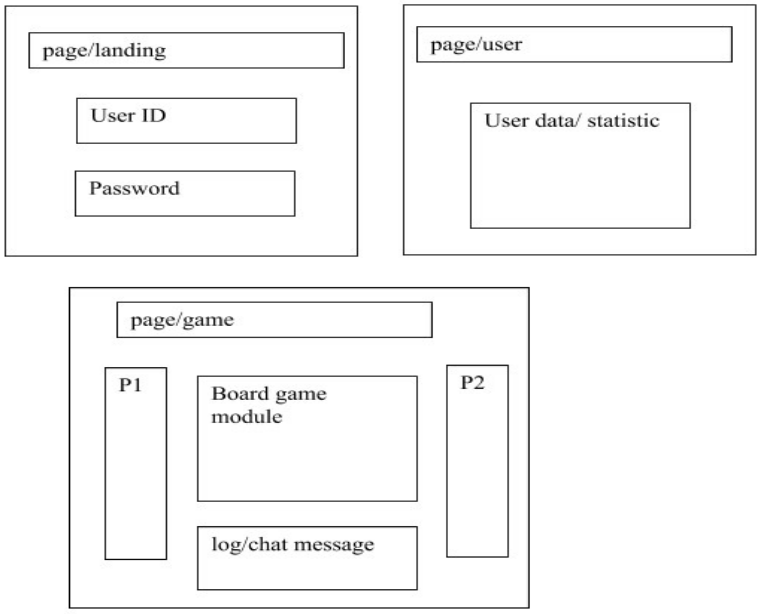
3.1.1 User Interfaces

There will be a login screen for users that allows them to login to an existing account, or sign up for a new account. This page will include two separate fields for typing in a username and password for either signing up or registering. There will also be a sign-up and register button.

There will be a landing page once a user has logged into their account, on this screen the user will be shown some basic statistics about their profile, they will have a play button to begin matchmaking, and a logout button to enable users to log out of their account. There will also be a button to navigate the user into the games page.

There will also be a FAQ-like page with rules and simple instructions on how to play tic tac toe. This page will also have a back button to redirect them onto the original landing page.

There will lastly be a game board screen where the game being played is held, a tic tac toe title, an indicator telling both players whose turn it currently is and how much time left they have to take their turn. Lastly there will be a concede button for any user who wants to quit playing. This will result in a winner and a defeat pop-up, which gives both players a button to return to their home landing page.



3.1.2 Hardware Interfaces

Online Multiplayer Board Game will take inputs from the user's keyboard and mouse to interact with each of the pages. The website will then transmit those actions to our server to make changes to their screen or update an existing game board.

3.1.3 Software Interfaces

The django web server/python interface will be interfacing with the Ubuntu linux distribution to act as the site's backend logic. The Django web server will be responsible for generating and updating all of the user's pages by interfacing with their browser.

3.1.4 Communications Interfaces

Our website will utilize https, meaning that all communication between the user and our server is encrypted using Transport Layer Security (TLS). The web page will interact with our server by means of a REST API. The game playing logic, matchmaking service, and user credentials/profile will all be processed server side. The web client is purely a user interface.

3.2 Functional Requirements

* Browser-based GUI

This is a simple website where each page has no more than three buttons. This is true for all four pages our website provides. The GUI will also present the user with stats, instructions, login prompts, and the tic tac toe game itself.

* Basic matchmaking to pair prospective players and initialize a match.

This is the ability for our website to pair two logged in users to the server side game, it will pull two queued users and put them into a game together.

* Credentialing system allowing each user a unique,persistent identity within the system.

This is the creation of a personal account by means of a user-created username. This will allow for a display name and also provide a username for logging into the website.

* Connect to the internet

A stable connection to the internet is needed to enable each user to communicate with the server. Actions taken on the game need to be translated to the opponent, which isn't possible without an internet connection.

* Wins/Losses Statistics Tracked

The website will keep track of the amount of wins and losses that have occurred for each individual based on their account. This information is then displayed to the user on their homepage.

* Allow Interaction Between Users

The website should allow users to look up other users via their username and see their stats. We should also have this in the form of some premade messages such as "Good Game" or "Close One!" which can be sent while in-game.

* Allow Users to connect to Game

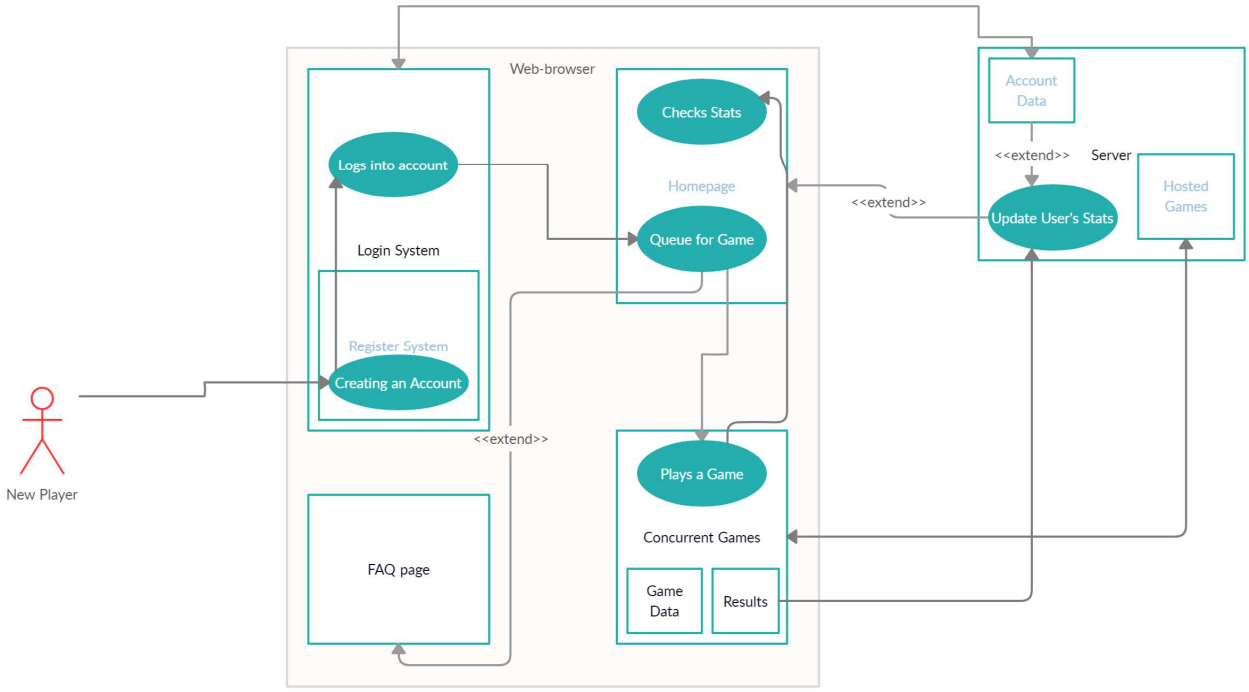
The website shall find a game for our user, or tell them no other users are available within five minutes of being queued to play a game.

* Allow Users to placed into Queue

The website shall allow the user to queue up for a match, and see how long they have been waiting to be placed in one.

3.3 Behavior Requirements

3.3.1 Use Case View



The new player wants to play a game of tic tac toe. They must first register an account, then log into their account. Then they will proceed to their homepage where they can then queue for a game. Once they have matched with an opponent, they will then play a game of tic tac toe. When it is complete they are directed back to their homepage.

4 Other Non-functional Requirements

4.1 Performance Requirements

Once placed within a game, user input shall be processed and propagated back to the user's browsers in less than three seconds. The live, interactive nature of the product necessitates the system to provide a reasonable level of responsiveness. Anything further degrades customer experience. Once a game is completed the user shall be redirected back to their home screen after a maximum of ten seconds to reduce wasted server resources.

4.2 Safety and Security Requirements

The system shall be resilient against SQL injection. Inadequate safeguards could result in partial or total user credential loss or compromise. The client browser shall communicate with the server in a secure fashion to prevent the interception of user credentials. Our server shall be adequately protected since it stores user's passwords. This shall be accomplished by the standing configuration of the WSUV school servers e.g. strict firewall rulesets.

4.3 Software Quality Attributes

Our code plan shall implement TDD to minimize the amount of bugs we occur. We shall utilize the git versioning system to separate production and development code. This should reduce the amount of errors we allow into our final prototype. In order to stop users from wasting large amounts of time in the queue, if any user is unable to find a game within five minutes, they shall receive a message saying that there are currently no other users queued and to queue again later.

Appendix A – Data Dictionary

TLS	Transport Layer Security. A set of cryptographic protocols to provide secure transmission over a computer network.
Django	A high-level web framework for the Python programming language.
TDD	Test Driven Development. Software Engineering methodology whereby development progress is intrinsically linked with good testing practices.
Ubuntu	A Linux operating system.
Linux	An open-source, unix-like operating system and kernel.
Kernel	The functional core of an operating system.
SQL	Structured Query Language. Used for programming and managing data held within a relational database management system.
Database	A collection of data arranged for ease and speed of search and retrieval.
SQL Injection	An attack on computer systems that attempts to take advantage of poorly coded SQL queries in order to perform actions on the database they would otherwise be disallowed from making.
API	Application Programming Interface. The official means of interacting with a computer program that supports such functionality.
REST	Representational State Transfer. This is an architectural style common across the web.
RAM	Random Access Memory. This is where active, running program information is stored while the computer is running.
TA	Teaching Assistant. A likely grader for this project. Determines which web browser our project will be optimized for.
CSS	Cascading Style Sheets. This technology is largely responsible for generating the layout and look of web pages. Usually paired with HTML.
HTML	HyperText Markup Language. Web technology used to manage the layout of a web page. Usually paired with CSS.
FAQ	Frequently Asked Questions. An often-included section in websites to provide answers to common questions.
GB	Gigabyte. A unit of compute storage capacity. Equal to one billion bytes.
GUI	Graphical User Interface. This is in contrast to a text-based, terminal style interface.

IEEE	Institute of Electrical and Electronics Engineers. A respected organization in computer and electrical engineering that develops standards in those fields. Their prescribed document formatting informed the style of this document.
Back End	The program logic that runs and is executed on the web server.
Front End	The program logic that runs and is executed by the clients web browser.
Web Server	A computer that processes and responds to incoming traffic from those visiting the web page.

Appendix B - Group Log

10/20/2020 – Ryan and Yousef conducted the initial group meeting. Ryan and Yousef reviewed the requirements for project milestone 1, developed a methodology for developing the SRS document, and determined a general agenda for the following meeting on 10/27/2020.

10/27/2020 - Ryan, Yousef, and Damon met to develop an outline for the SRS document. In this meeting we divided the document into three parts, one for each of us to complete. The process of incorporating each section will commence at the next meeting. The sections were assigned as follows:

Ryan: Section 1
Yousef: Section 2
Damon: Section 3

11/3/2020 - Attended by Ryan, Yousef, and Damon. Reviewed and integrated our individual sections into the first draft of a completed SRS document.