SE 4G06: Software Requirements Specification
Team #12 - CodeChamp

Kanugalawattage, Anton Subedi, Dipendra
Rizkalla, Youssef Leung, Tamas Zhao, Zhiming

September 28, 2022

Contents

List of Tables

List of Figures

Table 1: **Revision History**

Date	Version	Notes
Sept. 27, 2022	0.1	Initial Version

1 Project Drivers

1.1 The Purpose of the Project

Practicing coding the traditional way can be daunting. The current most popular method to learn is to use a problem database site like LeetCode. This method of learning can often feel tiring and endless. There are over 2000 problems on the website which can be intimidating to many new coders. CodeChamp will introduce a collaborative and fun way to interact with your friends while improving your algorithmic skills.

1.2 The Stakeholders

1.2.1 The Client

- Dr. Spencer Smith
- TAs of SE 4G06

1.2.2 The Customers

- Learners who wants to practice and improve on their data structures & algorithmic skills.
- Existing users of popular coding practice sites like Leetcode and DMOJ.
- Groups looking to compete and practice their coding skills.

1.2.3 Other Stakeholders

- Developers on the project

1.3 Mandated Constraints

1.3.1 Solution Constraints

- System shall be accessible through the internet to any device with a browser.

1.3.2 Budget Constraints

- For the scope of the project, deployment and hosting shall cost $0.

1.4 Naming Conventions and Terminology

-

1.5 Relevant Facts and Assumptions

- User's have and will maintain a steady internet connection throughout the game.
- User's will have input devices to type out their solutions to the problems presented.
- User is able to navigate a computer or a smart device.

2 Functional Requirements

2.1 The Scope of the Work and the Product

2.1.1 The Context of the Work

- This application will allow 2-20 users in a lobby to compete against each other across multiple browsers.

2.1.2 Work Partitioning

- Server will maintain all match data
- Server will compile user's submissions and evaluate the correctness of the submission.
- User clients will request match data and match state from server
- User clients will communicate with server to update match state

2.1.3 Individual Product Use Cases

- User creates an account
- User logins into application
- User creates a lobby
- User creates invite code to invite friends
- User joins lobby with an invite code
- User finds lobby to join
- User writes code in editor
- User submits code for compilation
- User returns to main menu after winning/losing
- User views previous match history

2.2 Functional Requirements

FR.1 The system shall allow users to sign up using a username, email and password.

FR.2 The system shall allow users to log in using a username and password.

FR.3 The system shall display an error if the user fails to log in.

FR.4 The system shall allow users to create a lobby link.

FR.5 The system shall allow users to join a lobby using a link.

FR.6 The system shall match-make users into a match.

FR.7 The system shall start a match when 20 players are match-made.

FR.8 The system shall have 3 rounds per match.

FR.9 The system shall display one coding problem per round.

FR.10 The system shall allow users to submit code through the web interface.

FR.11 The system shall compile the user's code.

FR.12 The system shall detect malicious code.

FR.13 The system shall prevent malicious code from compiling.

FR.14 The system shall display the result of the code's compilation and report any errors to the user.

FR.15 The system shall run the user's code against the test cases for a problem.

FR.16 The system shall track the code's running time in seconds and its memory usage in megabytes.

FR.17 The system shall time-out any code that exceeds the time or memory limit for a coding problem.

FR.18 The system shall display the result of running the user's code against the test cases to the user.

FR.19 The system shall qualify the first 10 users in the first round to solve the coding problem into the second round and shall disqualify the remainder.

FR.20 The system shall qualify the first 5 users in the second round to solve the coding problem into the third round and shall disqualify the remainder.

FR.21 The system shall declare the first user to solve the coding problem in the final round as the winner and shall disqualify the remainder.

FR.22 The system shall allow users to view a history of their previous matches.

FR.23 The system shall display the user's win percentage.

FR.24 The system shall display the rounds in which the users qualified or disqualified for a match's history.

FR.25 The system shall display the coding problem for each round in a match's history.

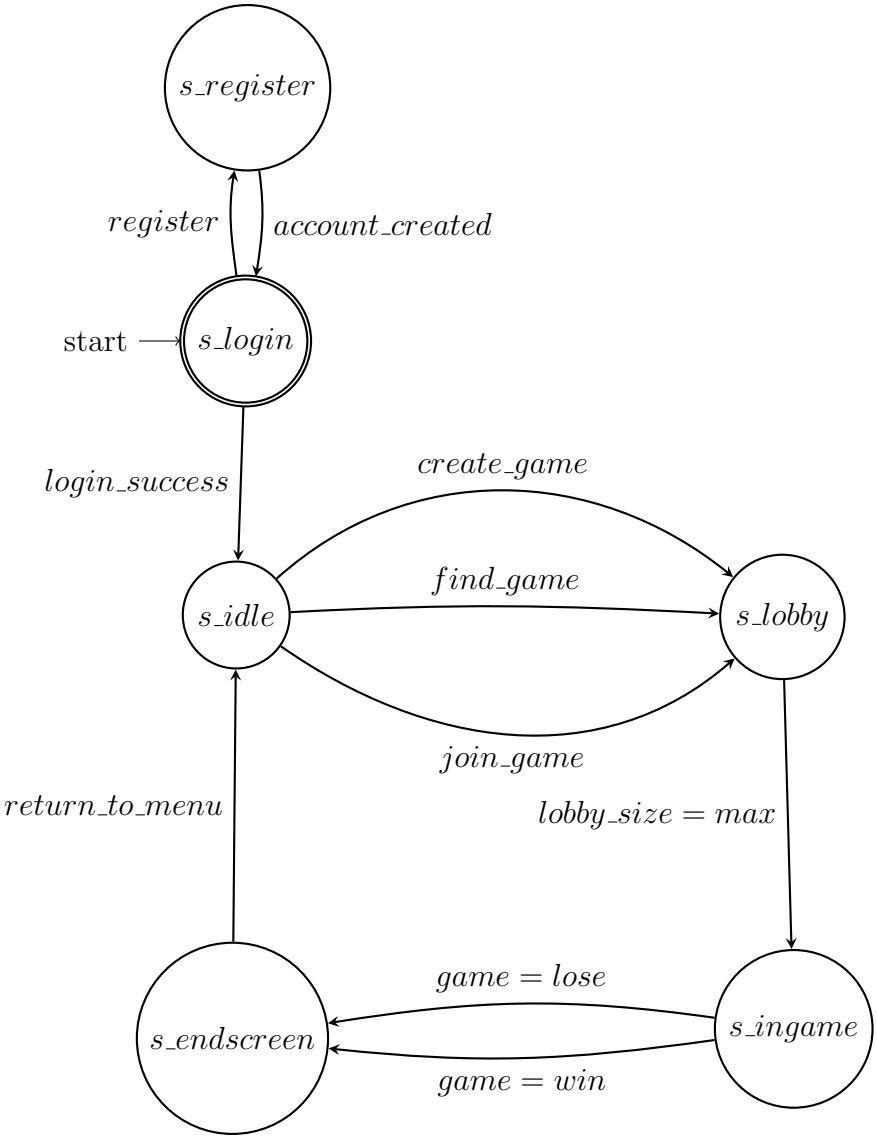
FR.26 The system shall track and display the top 100 users with the largest number of wins.

FR.27 The system shall allow developers to create, modify and delete coding problems.

FR.28 The system shall allow developers to create, modify and delete test cases for a coding problem.

FR.29 The system shall allow developers to define a time in seconds and a memory limit in megabytes for each coding problem.

2.3 User Lobby State Flow



3 Non-functional Requirements

3.1 Look and Feel Requirements

NFR.1 The interface of the system shall be easy to read and the elements are not clumped together.

NFR.2 The elements on the interface shall be organized and placed in a logical way.

3.2 Usability and Humanity Requirements

NFR.3 People with basic computer understanding shall be able to navigate the system.

NFR.4 The system shall be simple to understand and user should spend less than 5 minutes to understand what each part of the system does.

NFR.5 The system should be easily accessible so that the user can navigate around the system without the use of mouse.

3.3 Performance Requirements

NFR.6 The system shall respond to a user's action within 2 seconds.

NFR.7 The system shall process and give results base on user's inputs within 10 seconds.

NFR.8 The system shall be operating 24 hours a day, 7 days a week.

3.4 Operational and Environmental Requirements

NFR.9 The system should run on major browsers and devices that support these browsers.

3.5 Maintainability and Support Requirements

NFR.10 The system shall provide the ability for developers to add questions and test cases when needed.

3.6 Security Requirements

NFR.11 The system shall require users to register their own accounts in order to use the system.

NFR.12 The system shall deny the user access if the user fails to login.

3.7 Cultural Requirements

NFR.13 The system shall not have anything that is or suggests cultural inappropriate content.

3.8 Legal Requirements

NFR.14 The system shall be protected by GNU General Public License (GPL).

3.9 Health and Safety Requirements

NFR.15 The system shall not have any flashing lights that can put the user under the risk of epilepsy or seizure.

4 Project Issues

4.1 Open Issues

- Judging and evaluation of a user submission
- Safety, managing malicious code submissions

4.2 Off-the-Shelf Solutions

N/A

4.3 New Problems

N/A

4.4 Tasks

- Back-end architecture design
- Front-end UI design
- Communication schema design between front-end and back-end
- Database schema design
- Unit tests for front-end and back-end
- End to End integration tests

4.5 Migration to the New Product

N/A

4.6 Risks

- Multiple concurrent communications between front-end and back-end
- Back-end server may not be able to handle multiple concurrent of compilation
- Distributed Denial-of-Service attacks
- Malicious code injections

4.7 Costs

- Back-end hosting server costs
- Front-end hosting server costs
- Database hosting costs

4.8 User Documentation and Training

N/A

4.9 Waiting Room

N/A

4.10 Ideas for Solutions

For concurrent communications and concurrent code compilations, vertical scaling for hardware is an easy fast method to remove the limitations of concurrency. However this leads to future higher costs. Load balancing can be used to allow for horizontal scaling of system.

For costs, many options exists for free/hobby tier project hosting, allowing for cost savings.

5 Appendix

This section has been added to the Volere template. This is where you can place additional information.

5.1 Symbolic Parameters

The definition of the requirements will likely call for SYMBOLIC_CONSTANTS. Their values are defined in this section for easy maintenance.