Software Requirements Specification

for

CodeDuel

Version 1.0 approved

prepared by Ajay Raj Nelapudi

Team Shazam:

Ajay Raj Nelapudi

Raghu Mylapilli

Amrutha Lankshmi

Sushma Manjeti

31.01.2019

1. Introduction

1.1 Purpose

The objective of this project, CodeDuel is to provide a platform for coding competitions and Hackathon events where the competition is done between two coders. Each contestant can bring their own set of problems to challenge the other contestant. The challenge thrown by one contestant should be solved by the other within a given time limit and the code should be pushed to the server where it will be run against the test cases. The purpose of this SRS is to specify the requirements of this project, CodeDuel and is intended to be the online judge for Cursors 2019 event “Code Avadhan”.

1.2. Document Conventions

Page margin: normal
Top 1” bottom 1”
Left 1” right 1”

Page header
Header left: Software Requirements Specification
Header right: Page Nos
Footer left: Anil Neerukonda Institute of Technology and Sciences

Main heading:
Font style: times new roman, bold
Font size: 24pt
Numbering: numeral
Alignment: center

Sub heading:
Font style: times new roman, bold
Font size:16pt
Numbering: numeral.numeral

Alignment: left

Sub-sub heading:

Font style: times new roman, bold

Font size:14pt

Numbering: numeral.numeral.numeral

Alignment: left

Content:

Font style: times new roman,

Font size:12pt

Alignment: justified

Figures:

Alignment: center

Border: thick

Caption:

Font style: times new roman,bold

Font size:8

Alignment: center

numbering: fig.number: fig name

Tables:

Alignment: center

Border: thick

Caption:

Font style: times new roman, bold

Font size:8

Alignment: center

numbering: table.number: table name

Table content:

Numerals: center alignment

Text: left aligned

Biblography:

Font style: times new roman

Font size:10pt

Numbering: [numeral].

Alignment: justified

Book name, author.

Referred websites:

Hyperlinks

Font style: times new roman

Font size:10pt

Numbering: arrow bullets.

Alignment: justified

1.2 Intended Audience and Reading Suggestions

Students and the instructors are the intended audience of this SRS.

Initially, one may go through the introduction and project scope.
Then, proceed to reading the hardware and software requirements.
Followed by the in-depth details of the project such as design, coding and testing in this document.

1.3 Product Scope

CodeDuel is a platform for code battles or similar competitions. It has an online judge along with the capability to host code battles for many contestants. Given the rules of the event “Code Avadhan”, there exists no platform to host it except to manually run the program files and verify the outputs. This project will provide a solution by being the platform for the event.

The various modules involved in the project are described as below:

Login Module: The login module will help the contestant to log into their accounts. They cannot register by themselves since this is a closed event and requires a great deal of pre-processing. Hence, only the contestants having access to their id and password can log in.

Scoreboard Module: Each contestant is allowed to see two contestants score. One is themselves and the other is their opponent's score. The scope of the scoreboard is only limited to two contestants. Without this platform, there would be no means to know the score except by manually asking. The scoreboard also has a refresh button to update the scores.

Accept Challenge Module: Since this is a duel event, the questions can only be seen after accepting a challenge thrown by the opponent. Here the opponent will just mention the key of his problem and the contestant will enter into this module to view the challenge.

Run Tests Module: After solving a problem, the contestant can push his code to the server. Once this is done, the corresponding test cases and their scores will be fetched and the tests are run. The database is updated according to whether the test case has passed or not.

Errors and Warnings Module: Since this is a coding event, the files pushed to the server might have compilation or runtime errors, these errors have to be reported back to the client, else he will not know what is cause of the failed test cases. This module will collect all errors and warnings and send it to the client. The client side software is responsible for displaying it to the contestant.

1.5 References

Referred Websites

- https://docs.python.org/3.6/
- https://dev.mysql.com/doc/refman/8.0/en/

- https://pyftpdlib.readthedocs.io/en/latest/
- https://www.geeksforgeeks.org/socket-programming-python/

2. Overall Description

The only purpose of building this application is to use it as a platform for the event "Code Avadhan". In this event, there exists contestants who challenge each other with coding problems. The opponents solve the problem and push it to the server where the code will be tested against the test cases provided. The contestants should register prior to the event and should submit their question, solution and test cases as .txt, .c/.cpp/.java/.py and .txt file respectively. These files are put into the server's file system which is uniquely designed for this platform.

Initially the scores will be zero for all. When the event starts, the contestant will enter the problem key to pull it from the server to a client machine. The contestant is then free to use any external IDE's or text editors to write code and test it. Once, he has solved it, he will upload it using the client side software. This file is first validated against given set of valid file types. Then the program is compiled and run against the test cases.

Depending on the test cases passed or not the score is updated in the database. This score is shown on the scoreboard of the client side software.

2. 1 Product Functions

The proposed application software is designed for Code Avadhan. This application can be classified into two parts.

1. Server
2. Client

Server

The server consists of the following modules:

1. Server
2. Runtime
3. Database

The server will have all the socket programming related code. It also has a dedicated format to receive the messages from the client system. Only the inputs in this format received by the server are accepted else it will be discarded. The received words tell the server to perform either of the following tasks:

1. Authenticate login.
2. Run tests for the file uploaded.
3. Fetch scores for the contestant and his opponent.

The runtime module is responsible for running the submitted program and catching either the output, compilation errors or runtime errors. It opens a subprocess and runs the program so that any harmful code executed via the subprocess is kept safe from the application. The runtime module will also update the scores in the database depending on the test cases passed.

The database module is a separate class which acts like a connector to the database. This will have a set of functions which can be called to perform a certain task. This can be used as an API for the CodeDuel application. This method of clean implementation means that the process will only be connected to the database and the connection is passed as argument. New connections are only spawned for new threads. Other than that no unnecessary connections are made.

Client

The client software consists of two modules:

1. Client
2. ClientUI

The client will have all the code related to the socket connections. It also has a CLI that the contestant can use to communicate with the client. The module will create a metadata.json file with the contestant id and password and will store it on the client's current working directory. It sends messages in the format set earlier to the server.

The clientUI module will have all the tkinter components to build the GUI for the contestant. The UI consists of scoreboard, accept challenge section, question section and push the code section with a background image. This module will import the client module and make necessary calls to communicate with the server.

2.2 User Classes and Characteristics

The contestant should have the knowledge to this competition. The UI is simple and clean such that a well informed contestant will be able to use it with ease. No specific user training is required. However, a demo might be necessary to all the contestants. There are two types of users in this application.

1. Admin
2. Contestant

Administrator:

The administrator is the person conducting the event. He/She will have access to the database and the file system on the server. He has the privileges to add or remove contestants, add or remove problems, add or remove test cases and also to insert all the data into the database.

Spec Files:

The spec files consists of the problem statement or specification. The permissions to upload these files or remove them are only assigned to the admin. The admin does not user the client software instead has a web application which he uses to login and perform the necessary tasks.

Test Files:

These files are the test cases where each file consists either of the input or expected output. Each file is entered into the database with a unique path and consists of a predefined no of marks. The admin reserves the right to upload these files to the server and update the database as well.

Scores:

The admin unlike the contestants will be able to view the scores of all the contestants. He will have a scoreboard and a UI on the web application that is unlike the scoreboard on the client side. The scoreboard will sort the leaders according the points earned and timestamp of their submissions.

Contestant:

The contestant will be able to perform the following operations.

View Score:

The contestant will be able to view his score and his opponent’s score only. He will have a refresh button which will update the scoreboard based on the last entry in the database at the server side.

Pull Spec File:

The contestant will be able to pull the spec file from the server to the client system. Then he will view the question in the spec file and try to solve it.

Push Source File:

After solving the problem he will push the code to the server using this class. Then the tests are run on the server side and the database is also updated. The scores are also refreshed after pushing the source file.

2.3 Operating Environment

Server

Hardware Requirements:

Processor: Intel Core i5 or higher

RAM: 8 GB or higher

HDD: 5 GB or higher

Network: LAN

Client

Hardware Requirements:

Processor: Intel Core Duo or higher

RAM: 2 GB or higher

HDD: 5 GB or higher

Network: LAN

2.4 Design and Implementation Constraints

Server

Software Requirements:

Operating System: Ubuntu Linux

Runtime Environment: Python 3.6 or above

Database: Mysql 8.0 or above

Client

Software Requirements:

Operating System: Ubuntu Linux

Runtime Environment: Python 3.6 or above

2.5 Assumptions and Dependences

1. All roles of and tasks are predefined.

External Interface Requirements

2.6 User Interfaces

The Graphical User Interface is clean and simple. Contestants with very less knowledge of the system will be able to use the client software.

Since, the users are contestants who write code, they are also well versed with using the command line. The command "clash help" will let the contestants know the commands on how to use it via CLI.

2.7 Hardware Interfaces

Monitor: 15" screen or more

Mouse: Scroll or optical mouse

Keyboard: Standard 110 key keyboard

2.8 Software Interfaces

The connection between the client and server are done using socket connections. The port number for sending requests is 32757 and for sending files is 6000.

2.9 Communication Interfaces

The server and the client are connected over sockets and the communication protocol is unique to the application.

3. Other Non Functional Requirements

3.1 Performance Requirements

The proposed system is one of a kind. It will be put to use during one of the events of the fest. Considering that the fest will also be attended by other college students, any drop in performance will effect the reputation of the college and will show that the college is incapable of conducting this event. Hence, the performance is a critical factor.

3.2 Safety Requirements

The data in the databases is very sensitive. Any manipulation to this data can cause the winner to change and the prize money will go to the wrong person. Since the prize money is high the data becomes more valuable. Hence, it should be protected at all costs.

3.3 Security Requirements

It is possible that the data can be modified for any reason. To overcome this factor we are using a secure database. The file system by default is protected by the operating system. Additionally we will change the permissions to the directories specific to the server side application.

3.4 Software Quality Attributes

Since, the project uses open source dependencies only, it is easier to run this application on any server. The code is written such that every variable is properly named and function defined exact to its name. Due to this, it is very easy to maintain the software in the future.

3.5 Business Rules

The spec and test files should be submitted to the host at least 2 days in prior to the event.

The metadata.json file generated during the usage of client software should be preserved.