Software Requirements Specification

Student Code Online Review and Evaluation

A terminal program and web application for use in Florida Tech's CSE department to facilitate the submission of code for professor-created assignments.

Team Members:

Michael Komar - mkomar2021@my.fit.edu
Charlie Collins - ccollins2021@my.fit.edu
Logan Klaproth - lklaproth2021@my.fit.edu
Thomas Gingerelli - tgingerelli2021@my.fit.edu

Faculty Advisor / Client:

Dr. Raghuveer Mohan - rmohan@fit.edu
09/20/2024

Table of Contents

Introduction

1.a Purpose

The purpose of this document is to outline the requirements for the features of the Student Code Online Review and Evaluation (SCORE) application for the purposes of development.

1.b Scope

The scope of the SCORE application is to provide a more streamlined submission platform for the CSE department to submit code. This application also intends to allow more feedback to be given to the students and provide a more efficient grading platform for professors.

1.c Definitions

- Auto Test: A submission will be run with the provided input and compared against the expected output.
- Test case: Expected input and output
    - Visible: Shown to the student
    - Hidden: Not shown to the student
- MOSS: Measure of Software Similarity algorithm
- Grading portal: The page where a professor can view and modify grades
- SSH: Secure Shell
- MD: Markdown
- Code01: Florida Tech CSE's server

1.d References

- MOSS: https://theory.stanford.edu/~aiken/moss/
- Canvas Api Documentation: https://canvas.instructure.com/courses/785215
- Python Canvas Package: https://canvasapi.readthedocs.io/en/stable/getting-started.html#

Overall Description

2.a Project Perspective

SCORE will consist of a backend server with which students can submit assignments, and professors can view assignments. This application will also include configurable auto tests that will be performed on each submission, and will be integrated with Canvas for seamless grade submissions.

2.b Product Functions

SCORE is a robust, full stack, code submissions platform. The main features of the application will be assignment submission, configurable auto tests, test case specific feedback, and Canvas integration

2.c User Characteristics

For this application there are two intended users: students and professors. Students will be able to view assignments, submit to assignments, and view feedback. Professors will be able to create assignments, set up auto tests and feedback, and view student submissions.

2.d Constraints

There are several constraints on the SCORE application, the first being user authentication. We can't rely on any authentication, but specifically one that will allow only Florida Tech students and professors to use the application. The next big constraint is on computational power of the server SCORE will be hosted on. This will constrain how quickly we can auto test submissions and how many student users the application can serve at a given time.

2.e Assumptions and Dependencies

SCORE has two main dependencies. The first is on user authentication, in which there will be a dependency onto the TRACKS CAS system. The second dependency is onto the Canvas API, as it will be utilized to upload grades from the grading portal to Canvas.

2.f Apportioning of Requirements

Moving forward, once these requirements are met, we hope to be able to allow SCORE to accept a greater number of programming languages and have more robust similarity detection using clustering.

Requirements

3.a Functional Requirements

3.a.1 Immediate Feedback

Upon submission of an assignment, the auto-grading system will automatically test the assignment against a predefined set of test cases. Depending on the configuration of the assignment, students will be able to view pass/fail results on a certain number of test cases. Each test case will have an associated amount of feedback, if that test case was a failure, then the associated feedback will direct students toward correcting common errors. Professors will have access to view all provided feedback to each student, per student submission. In addition to the immediate feedback present on submission, after the due date students will be able to view the feedback from any hidden test cases that were used for grading but not revealed during the development process.

3.a.2 Auto Testing

Upon submission of each submission, the auto-grading system will automatically compile, run, and test each submission in a unique testing container. These test cases will be documented in the database, and user feedback will be shown to the user. Professors will receive a detailed report including any errors, failures, and potential grade report when this testing is completed.

3.a.3 Grading Portal

Once the due date is reached, each student's highest graded submission will be applied to any defined curve, if any exist. Then, these grades will be uploaded to the class's Canvas page. If during the grading process a student's submission was flagged for potential plagiarism, then the professor will be notified and grades will be withheld from Canvas until the plagiarism is manually reviewed by the professor. During the assignment creation process, a professor can configure a flag to enable manual review of grades before they are uploaded to Canvas.

3.a.4 MOSS Integration

- Once the due date for an assignment has passed, all of the submissions will be checked for similarities using MOSS (Measure of Software Similarity). The file submissions will be sent to Stanford's MOSS server, and the application will receive an HTML report on the similarities. This report will then be interpreted by the application and provide visual indication on the grading portal for professors about which student submissions have been flagged for similarities. The HTML report that the application receives will also be available to download from the professor client.
    - Within the portal, flagged students will have their name underlined in red, with a warning symbol next to it. Upon clicking on the student's name, the application will display the other submission(s) that have been suspected of being copied from.
    - Sample Input (Valid): A list of student-submitted files
    - Sample Output: Similar submissions will be underlined in red
    - Sample Output: A downloadable HTML report

3.a.6 Shell Client-Server connection

- The basic functionality of the program will all be available through the connection between a client and the submission server.
    - Example connection:
        - "ssh jsmith9999@code01.fit.edu" - enter into code01.
        - "score connect" - from code01 connect to SCORE.
    - Sample Input
        - "list classes"
    - Sample Output
        - "CSE4101 - Computer Science Project"
        - "CSE4020 - Database Systems"
    - Sample Input
        - "select CSE4020"
        - "list assignments"
        - "exit"
    - Sample Output
        - "4020 selected"
        - "Homework 1 - Relational Algebra"
        - "exited 4020 successfully"
- Assignments created, removed, or edited on the shell client will be cached and changes will be automatically synchronized to the submission server.

- The server will accept a large number of concurrent connections but will prioritize accurate time-stamping of submissions (rather than auto-grading) in the event of the system being overwhelmed.
- When the server receives a submission, assuming sufficient resources are available, it will automatically create a virtual environment to execute the code submission on, receive the output from the virtual environment, and then recycle the pre-allocated memory and processing power.

3.a.7 Web Application-Server connection

- Assignments created on the web application will be stored on the server along with any associated files or details.
- Files uploaded to the web application by students will be stored and tested on the backend.
- Student submissions stored on the backend will be visible to the professor.
- Details of auto-testing will be displayed on the web application for the professor to review.
    - The number of test cases passed or failed will be visible to the professor
    - Students will see results if the professor configures so.
        - Feedback attached to test cases will be visible to students
- Details of the MOSS report will be displayed on the web application for the professor to review.

3.a.8 Assignment Creation

- The professor can create and edit an assignment by providing the following information: assignment name, assignment description, number of allowed attempts, assignment due date, assignment test cases, and auto-test configuration.
    - Assignment name: This is the name of the assignment as it will appear on both the student dashboard and the professor dashboard.
        - Sample Input (Valid): Joy With Centrality.
        - Sample Output: An assignment with the above.
    - Assignment description: This is the written description of the assignment uploaded as a file. Accepted file types are .doc, .docx, .pdf, .txt, or md. In the client shell, the assignment description can be written directly in the application. This description can include images and any media supported by the above file types. If an unsupported file type is provided, the application will display an “Unsupported file type” error message.
        - Sample Input (Valid): A .pdf file.
        - Sample Output: An assignment with the above .pdf file as its description.

        - Sample Input (Invalid): A .png file
        - Sample Output: Unsupported file type error message
    - Number of allowed attempts: This is the number of times that a student can resubmit an assignment. By default, this field is set to unlimited, meaning students can resubmit as many times as they wish. This can be changed to a positive integer representing the number of allowed attempts.
        - If a non-positive integer number is inputted, the field will default back to unlimited, and the application will display “Invalid Input” error message.
        - Sample Input (Valid): 10
        - Sample Output: Number of allowed attempts is set to 10
        - Sample Input (Invalid): -2
        - Sample Output: Invalid Input error message
    - Assignment due date: This is the due date upon which no more submissions will be allowed from students. If auto-grading is configured, once this date has passed, the grades will automatically be uploaded to canvas or be available as a .csv.
        - Sample Input (Valid): 9/22/2024
        - Sample Output: The due date is set to the above date
    - Assignment test cases: The user can provide as many test cases as they wish. By default, there will be no test cases, but clicking the “New Test Case” button or using the “New Test Case” command will create a new test case. Each test case has the following fields: input, output, feedback, and visibility.
        - Input: This is the input that will be supplied to the student submission. This can be entered as text or uploaded as a text file.
        - Output: This is the expected output of the programming given the corresponding input, which will be used to determine the auto-test score of a submission.
        - Feedback: This is an optional field that allows the user to attach feedback to this specific test case. The user can also attach feedback to be triggered only by certain output. *Ex. If the submission outputs a negative integer, provide feedback A, else supply feedback B.* This field is blank by default and can be left blank by the user.
        - Visibility: This field defines whether or not the student user will be able to view this test case. Visibility can be set to either **Visible** or **Hidden**. All test cases marked as **Visible** will be shown to the

student as a part of the assignment details. All test cases marked as **Hidden** will never be shown to the student user.

3.a.9 Assignment Submission

- The student user can submit a valid assignment within the classes that they are a part of before the due date has passed. A submission requires the student to upload a file or multiple files. By default, the submission is a single file, but the student can increase the number of files submitted by clicking the “Add Another File” button within the web app or by using the “Add Another File” command in the shell client.
    - Accepted file types: python, java, C++, and C
    - Unaccepted file types (not limited to): Byte code, executables, folders, or compressed archives
        - Upon uploading a non-accepted file type, the application will reject the file and display an “Unsupported file type” error message.
    - Sample Input (Valid): A Python file
    - Sample Output: The above file is submitted to the assignment
    - Sample Input (Invalid): A zipped archive
    - Sample Output: Unsupported File Type error message
- After a submission is graded, the student may upload a new submission so long as they have not exceeded the number of allowed attempts.
    - Upon exceeding the number of allowed attempts, the “Submit” button on the assignment details page will be grayed out, and the “Submit” command in the shell will no longer be allowed.

3.a.10 Assignment Deletion

- The professor user can delete any assignment for their class. This can be done by using the “Delete Assignment” button located within the professor dashboard of the web client, as well as the “Delete Assignment” command from within the shell client. Upon doing these actions, the application will ask the professor to confirm that they wish to delete the assignment and inform the professor that this action can not be undone.
    - When an assignment is deleted, all student submissions, as well as associated auto-testing scores, will also be deleted.
    - Sample Input (Valid): User selects delete, then confirm
    - Sample Output: The assignment and associated submissions are deleted

3.b Interface Requirements

3.b.1 Shell Client

- The student will be able to access the code submission application through the submit shell on code01. Functionality available to the student will be displayed using shell commands. Classes that the student is a part of will be populated in the shell client and this list can be viewed at any time. To view a specific assignment for a class, the student will first select that class before being prompted with class specific commands.
    - The student can choose a class to specifically filter by. Commands entered within the class filter will only show information about that class.
    - The student can select an assignment to make a submission towards.
    - The student can view detailed information about the selected assignment.
- The professor will be able to access the code submission application through the submit shell on code01. Functionality available to professors will be displayed using shell commands with the added functionality of adding, editing, and removing assignments as well as viewing student submission information for assignments.
    - The professor can add, remove, or edit classes that enrolled students will join by running the appropriate command.
    - The professor can add, remove, or edit assignments using the appropriate command.
    - The professor can view at any time three different levels of information, class specific, assignment specific, and user specific.

3.b.2 Web Application

- The student will be able to access an online dashboard populated with enrolled courses and navigate between them through an interactive web application. Each class page will display abbreviated cards for past, current, and future assignments with corresponding details and an option to display detailed feedback. Each assignment will have a dedicated page displaying the assignment details, results of test cases, potential feedback, and a section for file submissions.
    - Students can interact with assignment cards for more details regarding assignment submissions.
    - Students can upload files for submission on an assignment’s page in the submission section.
- The professor will be able to access an online dashboard populated with classes the professor teaches. The professor can navigate between multiple class pages. The professor will see brief cards of past and current assignments per course and have the ability to create and post new assignments.
    - The professor can assign a name, description, due date, and suite of test cases. The professor can modify the details of existing assignments.

        - The professor can specify input, output, feedback, and visibility per test case.
    - The professor will have access to a page displaying student submissions and other details for the purpose of grading
        - The professor can view each student's submission and the number of test cases each submission has passed.
        - The professor can assign grades and comments to each submission.
        - The professor will see students who have been flagged by MOSS for plagiarism detection.

3.c Security Requirements

3.c.1 User Authentication

When registering or logging in, students will use their Florida Tech login credentials to gain access to the server. We will use industry standard security processes to verify that it is the correct user accessing the site.

3.c.2 Containerization