**Software Requirements Specification**

**for**

Senior Project 499

**Prepared by James Brennan**

**SUNY Fredonia**

**9/4/2023**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

**2. Overall Description 1**

2.1 Product Perspective 1

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

**3. External Interface Requirements 3**

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

**4. System Features 3**

4.1 Functional Requirements 3

4.2 Non-Functional Requirements 5

4.3 Output 5

**5. Other Nonfunctional Requirements 6**

5.1 Performance Requirements 6

5.2 Security Requirements 7

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |

# **Introduction**

## **Purpose**

Our purpose in this project is to create a web based application that students of SUNY Fredonia can use to submit their code and receive feedback. This feedback will be exported back to the students as a PDF, as well as a way for the professor to see the feedback as well.

## **Document Conventions**

We are working on this project with the assumption that we have most of the back end work done, as we are improving, or revamping a previous version. We are adding a GUI for the web-based application, which interacts with a C++ back end functionality.

## **Intended Audience and Reading Suggestions**

Our intended audience and use-cases are meant for incoming freshmen taking beginner computer science classes. This program is meant to act as a buffer between the students and the professor. Users will be encouraged to read the brief description and tutorial provided by the professor.

## **Product Scope**

Our program is going to take in C++ files, and our program will compare the syntax of the submitted code to the syntax in the libraries containing the correct forms of syntax. It will then put visual boxes around the sections of syntax, in either a green or red box. Good syntax in green boxes, bad syntax in red boxes. These boxes and their original code will be exported to the user as a PDF file.

# **Overall Description**

## **Product Perspective**

This project is just one of many steps in the direction of helping teachers with the strenuous workload of grading homework, as well as automated grading for code. Our iteration this year is an improvement on the previous year version, and we are sure that future iterations will out-perfrom ours. However, our goal this year is to make one that our professors can actually use in their classes the following semester.

## **Product Functions**

Our program will have many functions, all pointing to the end-goal of helping students and professors understand code more efficiently. The program will be simple and easy for the students to use on the front end, yet quite advanced and functional on the back end. Our goal is for the program to be easy for the incoming freshmen, since they may have never written code, or used a program like this.

## **User Classes and Characteristics**

Our program will only have two different user classes, professor and student. The student class will have very few permissions, as there are not many things the students will need to use other than the core functions. Professors, however, will have access to the database where copies of the students’ code will be stored.

## **Operating Environment**

Our program will be able to work from everywhere, as it is a web based application. Any data that will be stored will be stored in a separate database.

**2.5 Design and Implementation Constraints**

Working on a project for a college university, there are a few constraints that we are faced with. The university itself has a duty and responsibility to keep all of its students' information safe and secure. With this in mind, we will not be able to store sensitive information about the students, such as email, password, grades, etc.

# **External Interface Requirements**

## **User Interfaces**

## Our GUI for our program is going to be very simple, as the functionality on the front end is very simple. Previous versions of this program had very few input options, containing a “submit” button and two drop boxes, one for the students’ code, and one for the output in a PDF form. Our plan is to create a visually pleasing and simple interface, with as little input as possible. Our’s will also contain a submit button, after the student either clicks and drags the file into the drop box, or browse the computer for the file. The program will then prompt the user to select where to save the output in the form of a PDF file.

## **Hardware Interfaces**

## Due to the nature of our program being web based, it will be available to most devices, regardless of operating system.

## **Software Interfaces**

The first thing the user will see of our program will be our webpage. This will be written in HTML. This front end then interacts with our c++ programming, which in succession accesses the libraries contained within them to make comparisons. Our web interface will also communicate with our mySQL database, which will store the name, email, and previous assignments of the students.

## **Communications Interfaces**

Our program will have communication between the web-based, HTML, GUI and the C++ based back end functionality. Since our interface will be web based, we can also store information within a database, which will be where the name, email, and code that the students submit will be stored, so that the professor using the application can see the code that was submitted.

# **System Features**

**4.1 Functional Requirements**

* Basic GUI Requirements - connection with C++ back-end
  + The web based interface
  + Web-based interface to use a C++ based back-end
  + C++ file(s) to be uploaded to the GUI which invokes the back-end C++ code for SCA, on the file
  + (out of scope) C++ back-end executes and generates HTML file corresponding to input C++ files
  + Convert the SCA generated HTML files into PDF and display in the browser
* Additional Mandatory Requirements
  + Assignments
    - Create DB Schema
    - Ability to create assignments
      * Each assignment will have a randomly generated URL that will be store in our database, to later be used for organization
    - A unique random URL for each assignment that students can use to upload their files
      * This is will provide both the professor and student using the program an easy and organized way to keep track of assignments
    - Store random student URLs for each assignment in a table in DB
    - Store randomly assigned student ID in the table as identifier
    - Each student gets a random ID at the beginning of the semester
      * *202330-ALPHANUMERIC (len 5)*
    - Dashboard for professor to view student assignments and their corresponding PDFs
  + Authentication
    - Auth0 login for professor to view dashboard
  + Database Connections
    - We will have a database, created by MySQLWorkbench, that will store unique information about each student across multiple tables
      * Student Table
        + random\_ID INT: this will hold the student’s randomly generated ID number, also this table’s Primary Key
        + mapping\_ID VARCHAR(45): this element will hold the user’s email
      * Assignments Table
        + assignment\_Num INT: will store the user’s assignment number for each unique assignment, also this table’s Primary Key
        + random\_ID INT: Foreign Key from Student table
        + assignment\_Title VARCHAR(45): will store the title for each assignment, i.e. Homework1
        + course\_ID VARCHAR(45): this will store the course number and type for each assignment, i.e. CS231
        + semester\_ID VARCHAR(45): will store the year and semester of when the assignment was submitted, i.e FALL2020
        + marks INT:
        + file1 & 2 VARCHAR(45): this will hold links to the files uploaded
      * URL Table
        + serial\_Num INT: will store the serial number of the randomly generated URL given to each student at the beginning of the year, also the Primary Key of this table
        + assignment\_Num INT: Foreign Key from Assignments table
        + assignment\_URL VARCHAR(45): will store each student’s randomly generated URL
        + sub\_folder VARCHAR(45): will store any additional organization needed for the files
      * Parameters Table
        + serial\_Num INT: same as above table, also the Primary Key of this table
        + IP\_Address VARCHAR(45): will store the IP address of each student when they access the program
        + assignment\_Folder VARCHAR(45): will store any additional folders needed for organization, as determined by the professor
      * Permissions Table
        + permission\_alias INT: a numerical identifier to show whether this user has access to a certain webpage in not (teacher’s view)
        + assignment\_URL VARCHAR(45): same as tables above, stores URL of the user to hold the identifiers for the permissions
        + random\_ID VARCHAR(45): same as tables above, stores the user’s ID to hold the identifiers

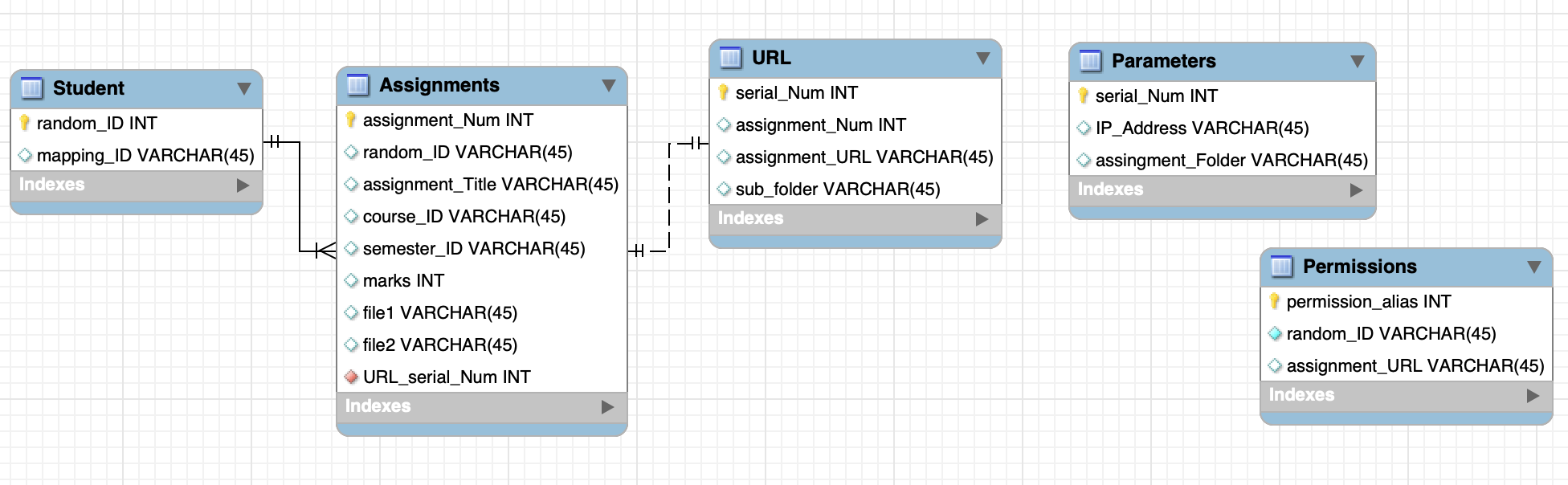


Fig 1: DB Schema

## 4.2 Non-Functional Requirements

* Web Based GUI Requirements
  + Clean, smooth look and feel to the webpage
  + Limited amount of buttons
    - Log-In screen
    - Dropbox/Button to upload file
    - Location to save exported PDF file
  + Teacher’s View
    - Separate web page for the professor(s) who use the program for their class
    - Will have separate functionalities to check submissions of students
      * Student ID search: teacher will be able to search a student’s ID and see the assignments they submitted, as well as the actual files that were uploaded
      * Assignment Number search: optional feature, professor will be able to search for a specific Assignment Number if they so choose, and will only show that assignment uploaded by the students they searched for. By default, a search for just
* Back-End Functionality Requirements
  + Libraries of correct syntax in storage

## 4.3 Output

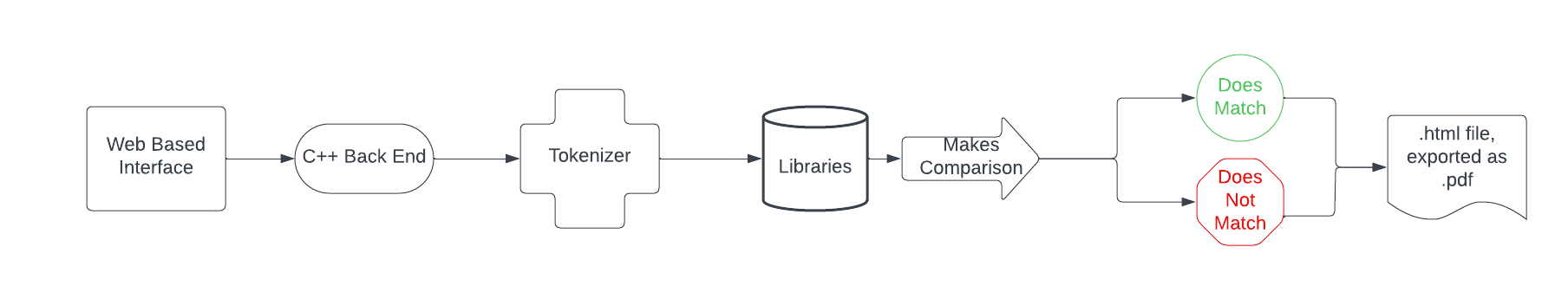
1. Our program will provide a PDF file that will highlight the correct and incorrect parts of their code.
2. Students will have to select where on their computer to save this output file. **

Fig 2: Workflow

# **Other Nonfunctional Requirements**

## **Performance Requirements**

Our program will be very simple, and does not have a lot of requirements. It will require a stable internet connection, and the storage space for the exported file.

## **Security Requirements**

This software’s security is to its users, the students. We will not be storing any sensitive personal data about the students, which will prevent any information being compromised. We will also have a built in permissions system, which only allows certain users, the professors, to be able to see specific web pages within the program.