Software Requirements Specification (SRS)

GradeFast

Version 1.0 | Date: 10/30/23

Celine Chiou, Will Eand, Mihai Siia, Aravind Sripada

Stakeholder: Fatma Serce

Software Requirements Specification

GradeFast: An Efficient Grading Tool

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Group	10/26/2023	Began drafting SRS, started introduction	0.5
Group	10/30/2023	Finished drafting SRS	1.0

1. Introduction

The rise of online learning has fundamentally transformed the education landscape, especially in recent years. The convenience and accessibility of online courses have made education more inclusive and adaptable to various schedules and circumstances. However, this shift is not without their own set of unique challenges as well.

Computer Science as a discipline plays a pivotal role, enabling the seamless delivery of online education and development of learning tools and platforms. Students have access to more resources today than ever before, KhanAcademy, Udemy, and various other online self-learning services; however, in pursuit of accessibility for students, professors have been left behind.

Professors have found themselves grappling with the taxing nature of grading assignments, particularly in computer science, where code evaluation can be time-consuming and complex. To address these challenges and improve the quality of life for computer science professors, there is a growing emphasis on creating user-friendly grading software that streamlines assessment process benefitting online webspace, educators, and students alike.

1.1 Document Purpose

The purpose of this document is to specify the requirements our team has created for an interactive online web application GradeFast. This Software Requirements Specification document defines the requirements set by the development team in collaboration with the primary stakeholder for the code grading tool.

Additionally, this document will serve as a basis and guideline for developers seeking to produce similar applications. This document will contain functional requirements, nonfunctional requirements, and constraints that will help developers grasp a generalized overview of our design process and goals. This document will also provide use case diagrams to provide visual descriptions for users and use case narratives to provide further technical information for developers.

1.2 Product Scope

The Minimum Viable Product (MVP) of this application will encompass three primary categories of functionality: submission management, applying comments, and exporting graded submissions. Submission management aims to develop a system that enables instructors to upload individual assignment submission batches as a single zip file. The application will separate, extract, and organize the files to a per-student basis. Applying comments will allow the instructors with access to a toolkit that provides prewritten comments and allows for editing of the comments for a quicker, yet thorough grading process. Exporting graded submissions enables teachers to export auto-generated reports to each student.

Additional core functionalities will be implemented following the completion of the MVP. A system that will store the comment bank will be introduced. This function will allow professors to reuse previous comments on the same assignment with a drag-drop shortcut, thereby reducing redundancies. A code review interface will be designed so that the platform is intuitive and user-friendly for reviewing student submissions, while ensuring that code files are displayed in a readable format with syntax highlights. The final feature will be for report generation. This will allow the instructor to generate a report with comments from the instructor for the student and reports on class performance for the professor to review.

Some features will be excluded as they fall outside the scope of this project. These exclusions consist of automatic student generation from text files, mobile compatibility for grading from mobile devices, and animations. These exclusions have been deliberately omitted to maintain a clear project focus and minimize complexity.

1.3 Definitions, Acronyms and Abbreviations

Acronym/Key Term	Definition	Description
—	Grader	An individual responsible for grading student assignments. These individuals can be teachers, professors or teaching assistants.
DBMS	Database Management System	A computerized data-keeping system.
FERPA	The Family Educational Rights and Privacy Act	Is a federal law enacted in 1974 that protects the privacy of student education records. It applies to any public or private elementary, secondary, or post-secondary school.

1.4 References

GradeFast Press Release and FAQ

GradeFast Project Plan

FERPA Government Regulations

1.5 Intended Audience & Document Overview

The intended audience of stakeholders for this specification are for our project manager. This specification contains documentation regarding the application's specific requirements, non-functional requirements, and constraints for the application that is being developed.

The parties below may also have an interest in this document:

- Computer Science faculty & related interests
    - Universities may see design intention behind each feature and how it aims to best support professors in their grading procedures.
    - Computer Science students that aim to work on educational tools as a reference to benefit their departments.
- For general purpose users, to understand the goals and intention behind creation of this application software.
- Client (owner)
    - Understand the scope of the project and understand the specifications between what the functional and nonfunctional requirements are for the intended project.
- Developers
    - Understanding the framework of our design and requirements to gather the scope of the project's capabilities and constraints.

1.6 Document Conventions

This document will be using Times New Roman font size 12 for all body text and Times New Roman font size 16.5 bold for headings. Subheadings will be 16 bold. Please also see section 1.3 to see the written out acronyms or jargon that are used specifically for this document.

2. Product Overview

The envisioned product is a comprehensive grading system that is designed to enhance the efficiency of the instructor's grading process. This system will offer a range of features essential to streamline the management of student submissions and grading. This product will promote a more user friendly and productive experience for educators. The major functions of the application are tailored for the introductory CS courses at Bellevue College.

Various factors may influence the product such as User Requirements, Scalability, and User Experience. User Requirements emphasizes the importance of meeting the needs and expectations of the user. This means that user feedback will be a priority for the success of the product. The product must also be able to handle increasing data loads or user demand. The User Experience stresses the design of the product as this will impact usability of the product and overall user satisfaction.

2.1 Product Perspective

GradeFast's origin stems from computer science professors' need for a fast, intuitive, and specialized tool to accelerate code reviews. A bespoke design approach to such a tool addresses these concerns best, tailored to suit the needs of individual departments. To that end, GradeFast is a self-contained, isolated application that takes the tedium out of navigating slews of interfaces to perform code reviews while retaining the features professors already know and love. GradeFast also adds features many professors wish they had, such as comment differentiation, formatting statistics, report generation, and many more.

2.2 Product Functions

1. User Account Creation:
    - Users can create credentials for their class, allowing them to access the system securely.
2. Batch Student Upload:

    - Users can conveniently upload a batch zip file containing their students' work onto the web application.
3. Student Management:
    - The system will organize students based on the uploaded files into separate categories or buckets for easy access and evaluation.
4. Code Readability:
    - Users will have the ability to view student code with specific keywords being highlighted, improving code readability
5. Navigation Shortcuts:
    - Users can use shortcuts to efficiently navigate between each student's code, enhancing workflow and productivity.
6. Commenting System:
    - Users can input comments and feedback directly into the system, streamlining the communication and evaluation process.
7. File Export:
    - Users will be able to export all student files individually, conveniently separated into folders for efficient management and grading.

2.3 Product Constraints

- Web applications need to be compatible with various browsers.
    - Browsers: Mozilla, Chrome, Safari, Edge
- Network and Latency
    - The speed and reliability of the user's internet connection can affect the web application performance.
- File formatting compliance
    - .zip or similar file formats must be supported
- Security
    - Protecting student data from outside attacks
    - Maintain privacy laws pertaining to FERPA
- Internet Access
    - Users need to be able to get online to access this tool

- Concurrency
    - Handling multiple users simultaneously with data consistency

2.4 User Characteristics

Professors:

As the main users of GradeFast, professors have experience working with other productivity apps. Full visibility into GradeFast's source code is needed. Should have access to the application on more than one computer.

Teacher Assistants:

Less experienced than professors, teacher assistants don't need visibility into how the application works. Accessing student files requires that submissions be sanitized to remove identifying information. Application access is limited to one computer.

2.5 Assumptions and Dependencies

Assumptions

1. User Proficiency:
    - Users should have a basic level of computer literacy and familiarity with web-based applications
2. Internet Connectivity:
    - Users should have reliable internet connectivity to access and use the system
3. Availability of Data:
    - Data, such as student assignments, should be accessible
    - Data should be in submitted in a consistent format
4. Scalability Planning:
    - There should be planning for increase in data and growing user base
5. User Feedback:

- Users will provide feedback for application updates

Dependencies

1. Access to Libraries and Frameworks:
    - Project is dependent on the availability and compatibility of specific libraries and frameworks
2. Availability of Software Developers:
    - Progress of the project is dependent on the schedule of the development team members
3. DBMS:
    - The project may depend on the data storage and retrieval method of DBMS

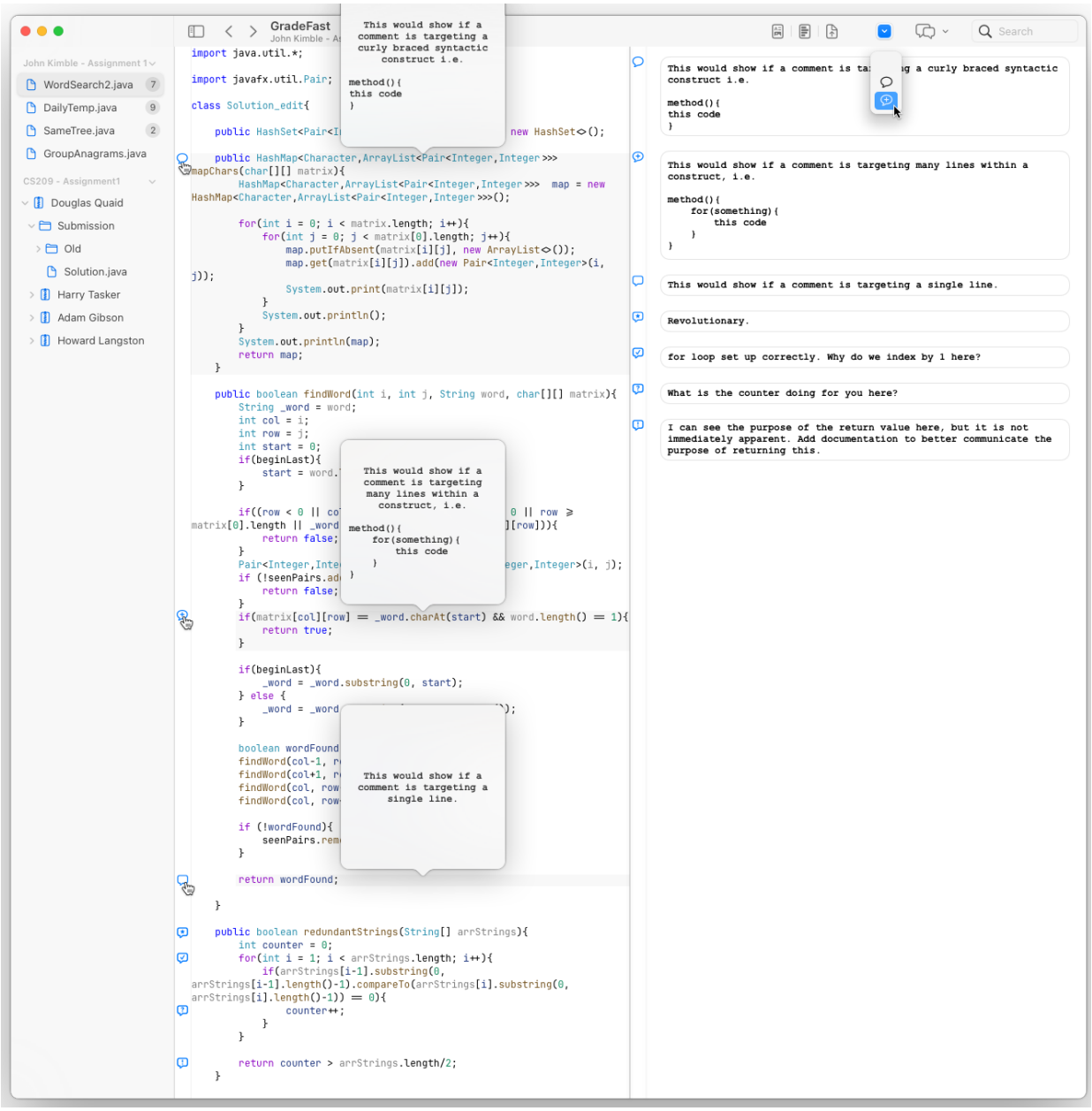
3. Requirements

3.1 External Interfaces

This subsection defines all the inputs into and outputs requirements of the software system. Each interface defined may include the following content:

- Name of item
- Source of input or destination of output
- Valid range, accuracy, and/or tolerance
- Units of measure
- Timing
- Relationships to other inputs/outputs
- Screen formats/organization
- Window formats/organization
- Data formats
- Command formats
- End messages

3.1.1 User interfaces

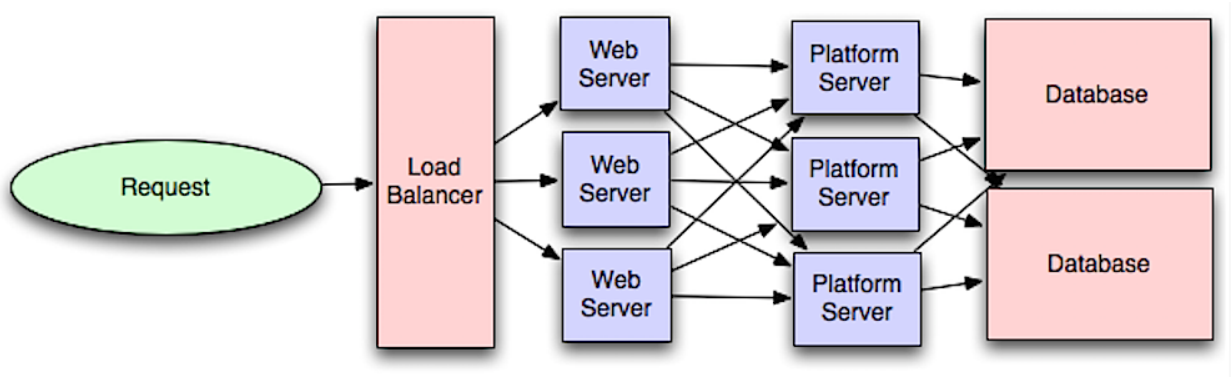


3.1.2 Hardware interfaces

GradeFast does not have any hardware interfaces.

3.1.3 Software interfaces

1. Cloud services
    - Utilize cloud platforms (AWS, Azure) for scalable hosting and resources.
2. Java Compiler API calls
    - Enable data exchange between Java Compiler API and GradeFast platform to recognize Java syntax and highlight words.
3. Continuous Integration Service
    - Automate code integration, testing, and deployment for improved development workflow (TravisCI).
4. Database Interface
    - Interface for storing, retrieving, and managing application data in databases.



3.2 Functional

1. As a user, I want to create accounts with email/name and password so that I can access the grading platform.
2. As a user, I want to log in with email/name and password so that I can securely access my account.
3. As a user, I want the ability to reset my password through a reset password email sent to my verified email address to regain access to my account if I forget my password.
4. As a user, I want to submit compressed folders containing multiple student assignments for grading.
5. As a user, I want the system to save my submitted student code after uploading.

6. As a user, I want the platform to separate submitted compressed folders into individual assignments and display each assignment clearly.
7. As a user, I want an interface to interact with student code and write comments to provide feedback.
8. As a user, I want the grading platform to be color-coded to improve the readability of the code.
9. As a user, I want a comment bank system that stores previously written comments for easy reuse.
10. As a user, I want the ability to recycle and reuse stored comments when providing feedback.
11. As a user, I want comments to identify the lines of code they reference for clarity and context.
12. As a user, I want to edit written comments to make adjustments or corrections.
13. As a user, I want the platform to generate a general report summarizing comments made for all assignments and an individualized report specific to one student assignment.
14. As a user, I want the generated reports to be sent to the correct email addresses, ensuring students receive their feedback.

3.3 Nonfunctional

3.3.1 Performance

1. The system should be able to handle a minimum of 100 concurrent user connections.
2. The system should be able to scale to accommodate 10,000 users.
3. GradeFast should use system resources responsibly. It should not demand more than 2GB of RAM at any time.
4. Autosaves should not result in perceptible slowdowns of the system or GradeFast itself.
5. The platform must load all non-zipped student assignments within 2 seconds.
6. The web application shall load webpages in less than 2 seconds for a typical user on a standard internet connection.
7. The platform should be able to load 70 submissions at a time.
8. A save file should be less than 25MB.

3.3.2 Security

1. GradeFast must be secure from data breaches.
2. The platform must be secure against credential brute force attacks.

3.3.3 Reliability

1. The system must have a maximum downtime of 1 hour per month.
2. Data must be regularly backed up and restorable within 6 hours.

3.3.4 Availability

1. The system must be available 99% of the time.

3.4 Design and Implementation

3.4.1 Portability

Being a Javascript application, GradeFast will be largely platform agnostic. Performance is largely dictated by whether a given platform can run Chromium/V8 Engine. As a rule of thumb, if it can run Chrome, it can run GradeFast.

3.4.2 Deadline

GradeFast 1.0 is slated for release in May 2024.

4. Appendices

GradeFast's Github