Software Requirements Specification

for

ProfInsights

Version 1.0

Prepared by

Group Number: 8

Niharika Attem	**Se22ucse038**	**Se22ucse038@mahindrauniversity.edu.in**
Niharika Mallavarapu	**Se22ucse182**	**Se22ucse182@mahindrauniversity.edu.in**
Sanjana Arroju	**Se22ucse309**	**Se22ucse309@mahindrauniversity.edu.in**
Siri Vennela Potluri	**Se22ucse212**	**Se22ucse212@mahindrauniversity.edu.in**
Deepak B	**Se22ucse077**	**Se22ucse077@mahindrauniversity.edu.in**
Ananya Gandla	**Se22ucse025**	**Se22ucse025@mahindrauniversity.edu.in**
Divya Reddy	**Se22ucse083**	**Se22ucse083@mahindrauniversity.edu.in**

Instructor: ***Vijay Rao***

Course: **Software Engineering**

Lab Section: ***CSE***

Teaching Assistant: **Sravanthi Acchugatla**

Date: **10 March 2025**

Contents

Revisions

Version	Primary Author(s)	Description of Version	Date Completed
Draft Type and Number	Full Name	Information about the revision. This table does not need to be filled in whenever a document is touched, only when the version is being upgraded.	00/00/00

1 Introduction

1.1 Document Purpose

*This document specifies the software requirements for the **ProfInsights**, a platform where students can anonymously rate and review professors based on teaching quality, clarity, responsiveness, and other criteria. The document outlines the functional and non-functional requirements, design constraints, and other relevant details necessary for the development of the system.*

1.2 Product Scope

ProfInsights is designed to provide students with a platform to share their experiences with professors anonymously. The system will allow users to search for professors, view their ratings, and submit reviews without the need for creating an account. The platform will also include features such as real-time review updates, a moderation system for detecting spam or fake reviews, and a reporting system for inappropriate content. The primary goal is to create a transparent and user-friendly environment for students to share feedback while ensuring privacy and security.

1.3 Intended Audience and Document Overview

This document is intended for the following audiences:

- ***Developers**: To understand the system requirements and design constraints.*
- ***Project Managers**: To oversee the development process and ensure alignment with the requirements.*
- ***Clients/Professors**: To review the system's functionality and provide feedback.*
- ***Testers**: To create test cases based on the requirements.*

The document is organized as follows:

- ***Section 1**: Introduction and purpose of the document.*
- ***Section 2**: Overall description of the product, including functionality and constraints.*
- ***Section 3**: Specific requirements, including functional and non-functional requirements.*
- ***Section 4**: Other non-functional requirements, such as performance and security.*
- ***Section 5**: Additional requirements, if any.*
- ***Appendices**: Data dictionary and group log.*

1.4 Definitions, Acronyms and Abbreviations

- ***SRS**: Software Requirements Specification*
- ***UI**: User Interface*
- ***API**: Application Programming Interface*
- ***NoSQL**: Non-relational database*
- ***SQL**: Structured Query Language*
- ***AI**: Artificial Intelligence*

1.5 Document Conventions

*This document follows the IEEE formatting requirements. The text is written in **Arial font size 11**, with section titles following the template format. Italics are used for comments, and the document is single-spaced with 1-inch margins.*

1.6 References and Acknowledgments

- ***IEEE SRS Template**: Used as the basis for this document.*
- ***React.js Documentation**: For frontend development.*
- ***Node.js Documentation**: For backend development.*
- ***MongoDB Documentation**: For database design.*

2 Overall Description

2.1 Product Overview

*The **Professor Rating Website** is a web-based platform that allows students to rate and review professors anonymously. The system will be accessible via web browsers on both desktop and mobile devices. The platform will include features such as search and filtering, real-time review updates, and a moderation system to ensure the quality of reviews.*

***Context Diagram**:*

- ***Users**: Students, Faculty, Administrators*
- ***System**: Professor Rating Website*
- ***External Systems**: Database (MongoDB/PostgreSQL), AI Moderation System*

2.2 Product Functionality

The major functions of the system include:

- *Anonymous reviews (no user tracking).*
- *Search and filter professors by name, university, course, or rating.*
- *Professor profile pages with overall ratings and individual reviews.*
- *User reviews with the ability to add, edit, or delete reviews within a time limit.*
- *Ratings system with a 1-5 star breakdown by criteria.*
- *Moderation system for detecting spam or fake reviews.*
- *Reporting system for inappropriate reviews.*
- *Real-time review updates.*
- *Responsive UI for both mobile and desktop.*

2.3 Design and Implementation Constraints

- *Frontend: React.js with Tailwind CSS for styling.*
- *Backend: Node.js with Express.js for REST API development.*
- *Database: MongoDB (NoSQL) or PostgreSQL (SQL) for storing professor and review data.*
- *Caching: Redis for faster search results.*
- *AI Moderation: Integration of AI-based spam/fake review detection.*
- *Cross-Browser Compatibility: The website must work on Chrome, Firefox, Edge, and Safari.*

2.4 Assumptions and Dependencies

- *The system assumes that users will have access to a modern web browser.*
- *The system depends on the availability of the MongoDB/PostgreSQL database and Redis for caching.*
- *The AI moderation system is assumed to be accurate in detecting spam or fake reviews.*

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface will include:

- ***Home Page**: Search bar, trending professors, and navigation bar.*
- ***Professor Listing Page**: List of professors with sorting and filtering options.*
- ***Professor Profile Page**: Professor details, overall rating, and individual reviews.*
- ***Review Submission Page**: Rating (1-5 stars) and detailed review text.*

3.1.2 Hardware Interfaces

Nil: The system is web-based and does not require specific hardware interfaces.

3.1.3 Software Interfaces

- *Frontend: React.js for UI rendering.*
- *Backend: Node.js with Express.js for API handling.*
- *Database: MongoDB/PostgreSQL for data storage.*
- *Caching: Redis for faster search results*

3.2 Functional Requirements

F1: The system shall allow users to submit anonymous reviews.
F2: The system shall provide a search and filter functionality for professors.
F3: The system shall display professor profile pages with overall ratings and individual reviews.
F4: The system shall allow users to add, edit, or delete reviews within a time limit.
F5: The system shall include a ratings system with a 1-5 star breakdown by criteria.
F6: The system shall include a moderation system for detecting spam or fake reviews.
F7: The system shall include a reporting system for inappropriate reviews.
F8: The system shall provide real-time review updates.
F9: The system shall have a responsive UI for both mobile and desktop.

3.3 Use Case Model

3.3.1 Use Case #1: Submit Review (U1)

- *Author: Ananya Gandla*
- *Purpose: To allow users to submit anonymous reviews for professors.*
- *Requirements Traceability: F1, F4, F5*

- *Priority: High*
- *Preconditions: User must be on the professor's profile page.*
- *Post Conditions: Review is submitted and displayed on the professor's profile page.*
- *Actors: Student*
- *Extends: Nil*

Flow of Events:

- *User clicks "Write a Review" button.*
- *User selects a rating (1-5 stars).*
- *User writes a detailed review.*
- *User clicks "Submit" button.*
- *Review is stored in the database and displayed on the professor's profile page.*
- *Includes: Nil*
- *Notes/Issues: Nil*

3.3.2 Use Case #2: Search Professor (U2)

- *Author: Siri Vennela Potluri*
- *Purpose: To allow users to search for professors by name, university, or course.*
- *Requirements Traceability: F2*
- *Priority: High*
- *Preconditions: User must be on the home page or professor listing page.*
- *Post Conditions: List of professors matching the search criteria is displayed.*
- *Actors: Student*
- *Extends: Nil*

Flow of Events:

- *User enters search criteria (name, university, or course).*
- *User clicks "Search" button.*
- *System displays a list of professors matching the criteria.*
- *Includes: Nil*
- *Notes/Issues: Nil*

4 Other Non-functional Requirements

4.1 Performance Requirements

- *P1: The system shall load search results within 2 seconds.*
- *P2: The system shall update reviews in real-time without requiring a page refresh.*
- *P3: The system shall handle up to 1 million users concurrently.*

4.2 Safety and Security Requirements

- *S1: The system shall not store any personal information of users.*
- *S2: The system shall use AI-based moderation to detect and remove spam or fake reviews.*
- *S3: The system shall allow users to report inappropriate reviews, which will be reviewed by an admin.*

4.3 Software Quality Attributes

4.3.1 Reliability

- *The system shall be available 99.9% of the time, with minimal downtime for maintenance.*

4.3.2 Usability

- *The system shall have a responsive UI that works seamlessly on both mobile and desktop devices.*

4.3.3 Scalability

- *The system shall be designed to handle up to 1 million users concurrently without performance degradation.*

5 Other Requirements

***Nil**: No additional requirements at this time.*

Appendix A – Data Dictionary

Appendix B - Group Log

Meeting 1: Project Kick-off

- ***Time Spent****: 2 hours*
- ***Task****: Discussed project scope, assigned tasks, and finalized the technology stack.*

Meeting 2: Initial SRS Draft

- ***Time Spent****: 3 hours*
- ***Task****: Drafted Sections 1 (Introduction) and 2 (Overall Description), and outlined functional requirements.*

Meeting 3: Use Case Model

- ***Time Spent****: 2.5 hours*
- ***Task****: Finalized use case models (U1: Submit Review, U2: Search Professor) and completed Section 3 (Specific Requirements).*

Meeting 4: Non-Functional Requirements

- ***Time Spent****: 2 hours*
- ***Task****: Discussed and finalized Section 4 (Non-Functional Requirements) and reviewed the entire document.*

Meeting 5: Final Edits

- ***Time Spent****: 1.5 hours*
- ***Task****: Proofread the document, made final edits, and prepared for submission.*