Software Requirements Specification

for

Intersectional Insights

**Version 1.0 approved**

**Prepared by <authors>**

**The A-Team**

**Table of Contents**

**Table of Contents** [**ii**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.ihv636)

**Revision History ii**

**1. Introduction** [**1**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.32hioqz)

1.1 Purpose [1](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.1hmsyys)

1.2 Document Conventions [1](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.41mghml)

1.3 Intended Audience and Reading Suggestions [1](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2grqrue)

1.4 Product Scope [1](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.vx1227)

1.5 References [1](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.3fwokq0)

**2. Overall Description** [**2**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.1v1yuxt)

2.1 Product Perspective [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.4f1mdlm)

2.2 Product Functions [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2u6wntf)

2.3 User Classes and Characteristics [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.19c6y18)

2.4 Operating Environment [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.3tbugp1)

2.5 Design and Implementation Constraints [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.28h4qwu)

2.6 User Documentation [2](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.nmf14n)

2.7 Assumptions and Dependencies [3](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.37m2jsg)

**3. External Interface Requirements** [**3**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.1mrcu09)

3.1 User Interfaces 3

3.2 Hardware Interfaces [3](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.46r0co2)

3.3 Software Interfaces [3](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2lwamvv)

3.4 Communications Interfaces [3](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.111kx3o)

**4. System Features** [**4**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.3l18frh)

4.1 System Feature 1 [4](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.206ipza)

4.2 System Feature 2 (and so on) [4](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.4k668n3)

**5. Other Nonfunctional Requirements** [**4**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2zbgiuw)

5.1 Performance Requirements 4

5.2 Safety Requirements [5](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.1egqt2p)

5.3 Security Requirements [5](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.3ygebqi)

5.4 Software Quality Attributes [5](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2dlolyb)

5.5 Business Rules [5](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.sqyw64)

**6. Other Requirements** [**5**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.3cqmetx)

**Appendix A: Glossary** [**5**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.1rvwp1q)

**Appendix B: Analysis Models** [**5**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.4bvk7pj)

**Appendix C: To Be Determined List** [**6**](https://docs.google.com/document/d/1XLK5EBHvjhy90Gryccacyln5Z1l8KxTo/edit#heading=h.2r0uhxc)

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Samson Cournane | 10/22 | Finished Document | 0.0.1 |
| [Samson Cournane](mailto:samson.cournane@maine.edu) | 10/27-29 | Fully Updated System Features and NF Requirements |  |

# Introduction

## Purpose

This document is a requirements specification for the initial version of Intersectional Insights The document covers specifications for several systems, including the user authentication method, feedback gathering system, and direct communication channels.

## Document Conventions

This Document was created based on the IEEE template for System Requirement

Specification Documents.

The following conventions in the document were used as follows:

| **Convention** | **Description** |
| --- | --- |
| Intersectional Insights | The Company Name: Intersectional Insights |
| React | Front-end Javascript library |

## Intended Audience and Reading Suggestions

This document is targeted for the product manager, developers, designers, and end-users. It dives deep into the complexities of the product, documenting its functionality, the interaction with auxiliary software components, critical product features, and other non-functional necessities.

End-users are urged to familiarize themselves with Sections 1, 2, and 3 for a full knowledge of the product's essence.

For a comprehensive grasp, the product manager, developers, and designers are advised to examine the whole text completely.

## Product Scope

Intersectional Insights is a React-driven website where students anonymously express class-related inquiries or exchange professor feedback. Aimed specifically at students suffering with impostor syndrome, this application encourages inclusion, potentially benefiting academic achievements and promoting University reputation. This project promises mutual benefits: heightened student engagement and enhanced University metrics.

## References

Team A GitHub:

<https://github.com/COS420-Fall23/The-A-Team>

# Overall Description

## Product Perspective

Intersectional Insights is a new platform, blending features reminiscent of Reddit and Discord, specifically tailored for the University of Maine students. Recognizing the inhibitions many students face due to impostor syndrome, this platform offers anonymous channels within class-specific groups. Here, students can voice concerns, ask questions, or seek help without the fear of judgment.

Extending its reach, Intersectional Insights also allows connections with UMaine alumni, granting students insights from those who've navigated similar academic terrains. Integrated with the University's email system, the platform ensures authenticity and exclusivity, making it a dedicated, trusted space for UMaine students. In essence, Intersectional Insights is an open-source hub, designed for candid and collaborative academic interactions.

## Product Functions

* The system must allow users to create a registered account using their student email
* The system must allow users to search for their course that the college offers
* The system must allow users to give feedback or ask questions openly with other students
* The system must allow users to view the groups based on the courses their in
* The system must allow users to like and dislike posts in each group
* The system must allow users edit their own reviews in markdown and add files if necessary
* The system must allow users to privately message their professors and TA’s

## User Classes and Characteristics

The software is tailored to cater to three distinct user classes, each pivotal in their own right, but with varying priorities and functionalities.

University students, who undeniably form the software's backbone. They are endowed with a plethora of features: from profile creation and accessing the same, to posting pertinent queries about their courses, offering feedback to their educators, and engaging with posts from peers. Their additional capabilities to upvote, downvote, and join academic groups further enriches their user experience, making them the most active and prioritized users of this software.

However, ensuring the smooth functioning of this platform are the system administrators and developers. Although their interactions differ starkly from that of students, their role is undeniably paramount. They don't delve into course reviews. Instead, their focus is on maintaining the platform's operational integrity. Endowed with superior security clearances, they can modify website content and access critical system data, ensuring the platform remains up-to-date and secure.

On the other side of the spectrum are professors, TAs, and alumni. Their engagement with the software is primarily as respondents and observers. They are centered on perusing course feedback, which offers them valuable insights into student sentiments. While their interaction is confined to responding to posts and receiving direct messages, their presence completes the software's feedback mechanism.

## Operating Environment

Modern web browsers like Chrome, Firefox, Microsoft Edge should be compatible with Intersectional Insights. Any operating system that furthermore supports any browser software that can support Javascript will also be able to use it.

## Design and Implementation Constraints

Given the sensitive nature of the application – covering personal feedback data as well as particular class-related information supplied by instructors – it is important to adopt rigorous security and privacy protections. While we attempt to apply core protective measures, it's vital to accept our limits in terms of needs and experience for complete data protection. Consequently, specialized knowledge will be essential in strengthening the application against potential security breaches and maintaining optimal data privacy.

## User Documentation

Intersectional Insights will contain a dedicated "How-to" page, which offers users with a full instruction on utilizing the platform's functions. This page will detail the step-by-step methods of each feature, ensuring users to maximize the application's capabilities with ease. Accessible from the main menu or sidebar, this guide will be built for straightforward navigation, delivering clear instructions backed by pertinent graphics or illustrations.

In addition, every feature on the platform will have an accompanying "Info" icon. Clicking on this symbol will provide a brief description and fast suggestions on how to utilize that specific function, offering rapid support for those unfamiliar with any area of the platform.

Moreover, for users who might have issues or have special inquiries, a "Help & Support" link will be present on every page, leading to a contact form for direct connection with our support team.

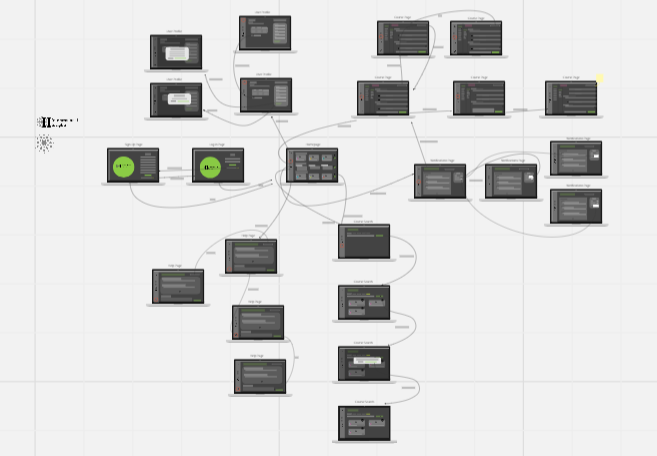
## Assumptions and Dependencies

Given the sensitive nature of Intersectional Insights – covering personal feedback data as well as particular class-related information supplied by instructors – it is important to adopt rigorous security and privacy protections. While we attempt to apply core protective measures, it's vital to accept our limits in terms of needs and experience for complete data protection. Consequently, specialized knowledge will be essential in strengthening the application against potential security breaches and maintaining optimal data privacy.

# External Interface Requirements

## User Interfaces

[Link to Mockups](https://miro.com/app/board/uXjVNY9FHAw=/)



Sign Up Page:

* Primary purpose: To allow new users to create an account.
* Fields: First Name, Last Name, Email, Password, Confirm Password, University Name, Purpose
* Actions: Sign Up, Link to Login.

Login Page:

* Primary purpose: For returning users to access the platform.
* Fields: Email, Password
* Actions: Login, Forgot Password, Link to Sign Up.

Homepage:

* Primary purpose: Central dashboard for users to navigate the platform.
* Features: Quick access to profile, notifications, course search, and recent courses.
* Actions: Select specific courses or functionalities from the dashboard.

User Profile

* Primary purpose: Display and manage personal information.
* Fields: User profile picture, name, email, password change option, and other personal details.
* Actions: Edit profile, Log out, Toggle Anonymous

Course Page:

* Primary purpose: To provide details and content for a specific course.
* Features: Course title, description, categories, chats, and related courses.
* Actions: Start/Continue course, mark as complete, access specific lessons/modules.

Notifications Page:

* Primary purpose: To display notifications, updates, or alerts to the user.
* Features: List of notifications with date and brief description.
* Actions: Click to view detailed notification, clear all notification, filter notifications.

Course Search:

* Primary purpose: To search for courses based on specific criteria.
* Fields: Search bar, filter options (e.g., category, level, duration).
* Actions: Enter keywords, apply filters, click on search results.

Help Page:

* Primary purpose: Provide assistance or information to users.
* Features: FAQs, contact form or chat option, links to guides or tutorials.
* Actions: Search for help topics, submit inquiries, access resources.

## Hardware Interfaces

There are precise logical and physical interfacing requirements between the system's software and hardware components in order to ensure seamless functionality of our web application. A reliable internet connection is a must, as it serves as a vital conduit for communication between the user's hardware and the software infrastructure. This consistent connection enables the browsing capabilities required to access and engage with our web application. This connection must support the operation of JavaScript, a basic technology used for developing interactive user interfaces and performing complicated actions on the client side of a web application.

The hardware requirements for the client include a visual interface, such as a monitor or screen, to show the information supplied by the web application. This visual depiction of data is vital for users to successfully engage with the application's functionalities. To move through the application, you'll also need a pointing device like a mouse and/or a keyboard. These devices serve as a link between the user and the software, allowing for the necessary control and interaction to input data and execute commands within the application.

The application's operating functionality is housed on the server side. The server is hosted on a personal server managed by one of the developers in this scenario, representing the physical part of the backend architecture.

The application's backend is built in Java, a sturdy and adaptable programming language appropriate for developing sophisticated server-side logic. The server processes the client's requests, executes the necessary logic, and returns the appropriate responses to be presented on the client's visual interface. The client-server communication is governed by well-established internet protocols, which ensure that data and control interactions are secure and dependable.

## Software Interfaces

The design of this software product is to ensure seamless interactions between various software components, delivering an efficient and user-friendly experience. At the core of the system lies React for the front-end development, Node.js and npm for managing dependencies and runtime, Java for database management, and a Linux backend for robust server-side operations. The front-end aesthetics and interactions are further enhanced using CSS, TypeScript, and HTML, ensuring a visually appealing and intuitive user interface.

The interactions between these software components are facilitated through well-defined protocols, ensuring a smooth data flow and service orchestration across the system. For instance, the Java database interacts with the Linux backend, processing and storing data generated or requested by the users, while the front-end, developed using React, CSS, and HTML, renders this data to the users in a structured and user-friendly manner.

The system is designed to handle various messages and data items coming in and going out, each serving specific purposes in the application's ecosystem. On the incoming side, messages include requests to create a new user account, email verification confirmations, requests for joining or creating groups, upvoting/downvoting requests, login and logout confirmations, and requests to add comments, update user information, or post content. Each request triggers specific services and data processing routines on the backend, ensuring the desired outcomes are achieved and reflected on the user interface.

On the outgoing side, messages generated by the system include notifications of new login attempts from different IPs, emails sent out for new user confirmation, and important informational emails regarding security breaches or updates to the Terms of Use. These messages serve to keep users informed, verify their actions, and enhance the security and transparency of the platform.

## Communications Interfaces

Several critical infrastructure considerations have been outlined in order to ensure the secure and efficient operation of our online application. To begin, our server will host a dedicated email account through Google, which will be used to send out verification emails to our user base. This account will serve as a "no reply" entity, with an automatic process that will delete any inbound emails, allowing it to focus solely on outward verification correspondence.

In addition, as a helper bot, a conversational assistant powered by OpenAI's cutting-edge API will be integrated to deliver an engaging and helpful user experience. This bot will be programmed to help users navigate the application, answer questions, and provide relevant insights.

The complete operational configuration will be housed on the same system, including the web server and database.

The web server will communicate with the database, saving and retrieving data as needed. Anonymous reviews will be managed systematically by tying them to a generic dummy user account, ensuring no linkage with real people and therefore maintaining anonymity.

Our application's backend is built to be strong and versatile, with Java used to construct server-side functionality and JSON used for data formatting and transfer between client and server. This configuration not only ensures a systematic and organized data management system, but also a solid framework for future expansions, such as moving the database to a different server for increased security and scalability.

# System Features

## Feedback with Emotional Context

[Link to Feedback with Emotional Context Document](https://docs.google.com/document/d/1a4bQlN0szPqVtlyXOhOtP2QOLeH2sWe6vxvGgWUQpq4/edit)

* SRS Document formatting set aside for the time being, all information for Deliverable 1 is in linked document above

4.1.1 Description and Priority

This feature allows users to provide feedback with an emotional context, enabling a deeper understanding of the user's sentiments regarding a particular topic.

Priority: High

4.1.2 Stimulus/Response Sequences

User selects the option to give feedback.

System provides an interface to capture feedback and select emotional context.

User submits feedback.

System acknowledges and saves feedback.

4.1.3 Functional Requirements

REQ-1: The user shall be able to track their given feedback through their profile.

REQ-2: The user shall be able to see previous feedback that has finished the review process and has been resolved by the separate user.

REQ-3: The system shall offer a feedback form that is available to all the user categories including students, professors, and administrators.  
REQ-4: This feedback form should facilitate the capture of screenshots to provide visual evidence or context.

REQ-5: Users should be able to depict their emotional context either by selecting from predefined emotional tags (like frustrated, confused, satisfied) or by entering their feelings in a free-text format.

REQ-5: To assist users in expressing emotions, the feedback form will feature visual aids such as emoticons and sentiment sliders.

REQ-6: An auto-correct tool will be available within the feedback form to ensure the feedback is grammatically accurate.

REQ-7: The feedback system should have an anonymous option, however, message sending should be restricted to users possessing a UMaine domain.

REQ-8: A "Feedback" tab should be readily accessible on a user's public profile.

REQ-9: Users should have the choice to input feedback directly or utilize the predetermined structure offered by the feedback form.

REQ-10: Users should be equipped to thread or categorize different feedback types.

REQ-11: After the user submits their feedback, the system shall provide a confirmation to the user within 3 seconds.

REQ-12: For consistency and clarity, the system should be able to reference prior feedback to address recurrent or similar feedback items.

## Commenting System

* + 1. Description and Priority

Description: The commenting system provides users the ability to comment on threads and interact with other users' comments

Priority: Medium

* + 1. Stimulus/Response Sequences

User navigates to a feedback or discussion.

System displays comments associated with that thread.

User inputs a comment and submits.

System displays the comment under the feedback or discussion.

* + 1. Functional Requirements

REQ-13: The user shall be able to find threads by navigating the homepage.

REQ-14: The user shall be able to remove comments in a chain which may not be helpful or come across as rude to the user.

REQ-15: The user shall be able to report questions or comments left by any profile if they don’t fit the terms and conditions of the application.

REQ-16: The system should securely store feedback submissions, maintaining both technical specifications and the emotional tone.

REQ-17: To maintain efficiency and modern standards, feedback data submission and retrieval should be managed using REST API.

## Account Management

* + 1. Description and Priority

The account management feature allows users to manage their profiles, including adding classes, toggling anonymity, and viewing feedback.

Priority: Medium.

* + 1. Stimulus/Response Sequences

User logs into their account.

System displays user profile and associated options.

User makes changes to their profile or settings.

System saves and confirms the changes.

* + 1. Functional Requirements

REQ-6: The user shall be able to toggle whether their profile is anonymous in posting questions or messaging other users.

REQ-7: The system shall provide the user with the ability to add their classes to a profile, which will be displayed publicly.

REQ-8: Upon a first-time login, the system shall show the user other public profiles which may include the same class as the user.

REQ-18: Should any section of the feedback form remain incomplete or if the feedback fails to reach the server, the system shall alert the user with a red notification bar.

REQ-19: Upon accurate and successful feedback submission, users should receive a confirmation message.

REQ-20: The system should offer a readily accessible user guide or manual detailing the steps to effectively incorporate emotional context in feedback.

# Other Nonfunctional Requirements

[Link to Non-Functional Requirements Document](https://docs.google.com/document/u/0/d/1aOgIL0o7vSpzT5GZXFSTjAPFGBnHsUZVCiJS6EuPK5A/edit)

* SRS Document formatting set aside for the time being, all information for Deliverable 1 is in linked document above

## Performance Requirements

* The app should ensure an uptime availability of 99% across all platforms: web, IOS, and Android.
* Real-time messaging capabilities should be integrated with minimal latency to ensure swift communications between users.
* The app shall be optimized to efficiently utilize system resources including CPU, memory, and storage.
* Professors and users should receive immediate notifications for new messages or updates.
* Monitoring tools shall be in place to continuously evaluate system performance, user metrics, and engagement in real time.

## Safety Requirements

* The system will combine robust data breach prevention safeguards while aligning with data protection standards like GDPR and HIPAA.

## Security Requirements

* The app will emphasize security by adhering to OWASP standards, using HTTPS and OAuth 2.0 standards for authentication, and ensuring GDPR-compliant data management.
* The app will implement authorization checks, ensuring only appropriate individuals access direct communication channels.

## Software Quality Attributes

* The app shall comply with WCAG 2.1 Level AA accessibility standards.
* The system shall be compatible with 95% of devices.
* All text and user interfaces shall be easily translatable, and region-specific formats shall be automatically adjusted based on the user's locale.
* The app should integrate seamlessly with third-party APIs and platforms.

## Business Rules

* Utilize topic tagging and advanced search for efficient discussion organization. Enhance communication with read receipts and instant notifications.
* The platform will prioritize community moderation and emotional context analysis to understand user sentiments better.
* Users can tailor their privacy settings, fostering easy networking opportunities with peers, alumni, and mentors.
* The platform's unique selling point will be its academic-centric features, setting it apart from generic platforms.

# Other Requirements

* Utilize tagging for efficient discussion searches and enhance communication with read receipts and advanced search tools.
* Integrate immediate notifications, community moderation, and emotional context analysis to boost user engagement and understanding.
* Prioritize fostering student networking, group formations, and connections with alumni and mentors.
* Ensure customizable user privacy settings and differentiate with unique academic-centric features not found in generic platforms.

**Appendix A: Glossary**

Team A: The Company Name

Intersectional Insights: the app name

React: A library for building interactive UIs.

Node.js: A JavaScript runtime for scalable applications.

OpenAI API: Provides access to advanced text-generation models.

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**