**Software Requirements Specification**

**for**

Intersectional Insights

**Version 1.0 approved**

**Prepared by <authors>**

**The A-Team**

**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

1.5 References 1

**2. Overall Description 2**

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

**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 4**

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

**5. Other Nonfunctional Requirements 4**

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

**6. Other Requirements 5**

**Appendix A: Glossary 5**

**Appendix B: Analysis Models 5**

**Appendix C: To Be Determined List 6**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Samson Cournane | 10/22 | Finished Document | 0.0.1 |
|  |  |  |  |

# **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**

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## **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.

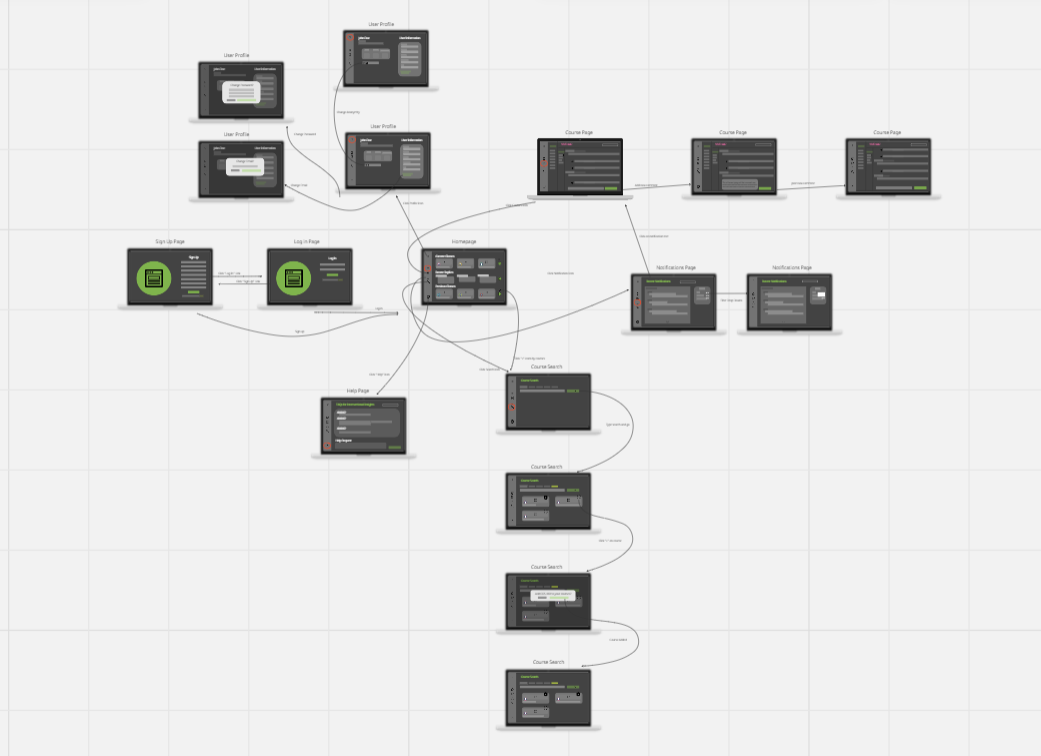
## **Assumptions and Dependencies**

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 meticulously crafted to ensure seamless interactions between various software components, delivering an efficient and user-friendly experience. At the core of the system lies a diversified tech stack, including 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, JavaScript, 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 a variety of 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 of these requests 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**

*<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>*

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

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

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

REQ-#: 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-#: 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.

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

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

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

# **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 shall have an availability of 99% to ensure users can access it anytime they need.
* The system should support real-time messaging with minimal latency.
* Professors should receive prompt notifications for new messages.
* Monitoring and analytics tools shall be deployed to continuously track and analyze system performance, user engagement, and other critical metrics in real-time.
* The app shall be compatible across web, IOS, and Android platforms, ensuring a consistent user experience.
* The system shall be designed to consume resources (such as CPU, memory, and storage) efficiently.

## **Safety Requirements**

* The system shall have safeguards to prevent data breaches and unauthorized access.
* The system shall comply with data protection laws and regulations like GDPR or HIPAA.

## **Security Requirements**

* The app shall have security measures that adhere to OWASP standards.
* The app will use HTTPS for secure data transmission.
* Authentication and authorization will adhere to OAuth 2.0 standards.
* User anonymity shall be maintained, and personal information shall be securely stored and managed following the GDPR guidelines.

*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**

* Topic tagging will be in questions to organize discussions and make searching easier.
* Read receipts will be implemented to inform users when their messages have been viewed.
* There will be robust search functionality to help users find relevant discussions and answers.
* The system will notify users of new messages or responses.
* The platform will facilitate community moderation to maintain a positive environment.
* The system shall have capabilities to analyze the emotional context provided by users.
* The platform should facilitate networking and group formations among students.
* Allow users to control their privacy settings.
* The platform should offer academic-focused features distinct from general-purpose platforms.
* The app should provide easy access to connect with alumni and mentors.

# **Other Requirements**

* Topic tagging will be in questions to organize discussions and make searching easier.
* Read receipts will be implemented to inform users when their messages have been viewed.
* There will be robust search functionality to help users find relevant discussions and answers.
* The system will notify users of new messages or responses.
* The platform will facilitate community moderation to maintain a positive environment.
* The system shall have capabilities to analyze the emotional context provided by users.
* The platform should facilitate networking and group formations among students.
* Allow users to control their privacy settings.
* The platform should offer academic-focused features distinct from general-purpose platforms.
* The app should provide easy access to connect with alumni and mentors.

**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**