Software Requirements Specification

for

EduSense

Version 1 approved

**Prepared by Alex Gignac, Brianna Thomas,
Christian Biehn, Cody Donahue,
Dillon Sapp, Hannah Gill,
and Zemi Gebreyohannes**

Old Dominion University

November 24, 2025

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this document is to provide an in-depth description of the scope and functionality of the EduSense tutor software product The following sections provide a detailed reference for developers, testers, and stakeholders to ensure the system meets the needs of students, instructors and administrators.

1.2. Scope

An increasing number of students nowadays rely on Artificial Intelligence tools to complete their assignments, leading to a decline in critical thinking and problem-solving skills. EduSense is a web application that transforms Large Language Models (LLMs) into an intentional educational tool for students and educators. It encourages students to think first by using guided questions, reflective prompts, and challenge modes to limit or delay AI input, preventing over-reliance on AI. EduSense provides educators with essential oversight features, such as usage tracking and assignment monitoring, allowing them to review student LLM interactions and effectively manage assignments to ensure that AI is supporting, not subverting, the learning process.

[This space intentionally left blank.]

1.3. Definitions, Acronyms, and Abbreviations

Artificial Intelligence (AI) - A commonly used term encompassing any machine learning algorithm designed to train from a given input to provide an expected output.
Large Language Model (LLM) - An advanced machine learning algorithm trained on massive text datasets to understand and generate human-like language.
Canvas LMS - A learning management system used by educators to manage course content, assignments, and communication with students.
Challenge Mode – A setting that encourages learners to try on their own before getting help. It limits access to answers to encourage thinking through the assignment first.
Guided Prompts - Targeted questions or hints created to help students think critically and come up with their own solution.
MFCD (Modified Functionality Component Diagram) - A diagram showing the major hardware and software components of the product and how they interact.
Usage Tracking - The process of recording how users interact with the system, such as which features they use or how they engage with LLM prompts.

[This space intentionally left blank.]

1.4. References

1. Farhan, Hind N. “The Impact of AI-Powered Writing Tools on Students’ Writing Performance: A Content Analysis and Future Prospects.” ResearchGate, 1 Mar. 2025, http://www.researchgate.net/publication/389458566_The_Impact_of_AI-Powered_Writing_Tools_on_Students.
2. Freeman, Josh. “Student Generative AI Survey 2025 - HEPI.” HEPI, 26 Feb. 2025, www.hepi.ac.uk/2025/02/26/student-generative-ai-survey-2025/.
3. Ju, Qirui. “Experimental Evidence on Negative Impact of Generative AI on Scientific Learning Outcomes.” Research Square (Research Square), 21 Sept. 2023, https://doi.org/10.21203/rs.3.rs-3371292/v1.
4. M. Helena Vasconcelos, et al. “Explanations Can Reduce Overreliance on AI Systems during Decision-Making.” ArXiv (Cornell University), 13 Dec. 2022, https://doi.org/10.48550/arxiv.2212.06823.
5. Rastogi, Charvi, et al. “Deciding Fast and Slow: The Role of Cognitive Biases in AI-Assisted Decision-Making.” Proceedings of the ACM on Human-Computer Interaction, vol. 6, no. CSCW1, 30 Mar. 2022, pp. 1–22, krvarshney.github.io/pubs/RastogiZWVDT_cscw2022.pdf, https://doi.org/10.1145/3512930.
6. Zhai, Chunpeng, et al. “The Effects of Over-Reliance on AI Dialogue Systems on Students’ Cognitive Abilities: A Systematic Review.” Smart Learning Environments, vol. 11, no. 28, 18 June 2024, pp. 1–37, slejournal.springeropen.com/articles/10.1186/s40561-024-00316-7, https://doi.org/10.1186/s40561-024-00316-7.
7. Team Emerald. “Software Requirement Specification for EduSense.” Sharepoint.com, Microsoft, 29 Oct. 2025, http://olddominion-my.sharepoint.com/EduSense. Accessed 29 Oct. 2025.

1.5. Overview

Section 2 of this SRS provides a brief overview of EduSense.
Section 3 of this SRS, organized by feature, contains the specific requirements.

[This space intentionally left blank.]

2. Overall Description

Section 2 and the subsections therein describe the functionality, customer, and end users of EduSense.

2.1. Product Perspective

EduSense is an innovative web application designed to help students and educators use Artificial Intelligence (AI) tools more resourcefully and effectively in academic environments. It addresses the growing concern that over-reliance on AI can weaken essential skills such as critical thinking and problem-solving. EduSense allows instructors to upload assignments and monitor student interactions with generative AI, providing oversight and helping identify areas where students need assistance. Simultaneously, it encourages students to engage with their assignments independently by utilizing guided prompts and leading questions rather than direct answers, ensuring that the LLM is solely an aid in the learning process. Through this intentional integration of AI, EduSense aims to empower students to develop lifelong critical thinking skills while still benefiting from modern technology.

2.2. Product Functions

EduSense is designed as a highly scalable application with a polished and responsive UX/UI, ensuring an intuitive experience across mobile and web devices. The platform offers customizable LLM integration that supports multiple LLMs, allowing for tailored guided prompts on a per-assignment basis rather than direct answers. It features robust security (OAuth and encryption) and provides full API sync for seamless Canvas LMS integration. Additionally, the system includes full analytics (detailed and exportable) for educators to monitor student usage, track progress, and evaluate the effectiveness of AI integration.

2.3. User Classes and Characteristics

EduSense has two main classes of users: Students and Educators (or Instructors). Students are the primary beneficiaries, using the application to access an LLM for support with their assignments. Their core requirement is a tool that assists their learning via guided prompts and reflective questions without providing direct answers. Educators utilize the platform to upload assignments, set learning parameters, and monitor student-AI interactions. They need to be able to effectively manage classroom activities and have clear oversight to ensure AI solely assists their students rather than doing all the work for them.

2.4. Design and Implementation Constraints

N/A

2.5. Assumptions and Dependencies

N/A

[This space intentionally left blank.]

3. Specific Requirements

3.1. External Interface Requirements (O: Biehn)

This section details the specific requirements for how EduSense shall interact with the external world, including the user, hardware, other software systems, and communication networks.

3.1.1. User Interfaces (O: Biehn)

EduSense Web Application:

- Browser Compatibility: EduSense shall render and function correctly on the latest stable versions of major web browsers, including Google Chrome, Mozilla Firefox, Microsoft Edge, and Apple Safari
- Student Interface: EduSense shall provide a clean, focused environment for accessing assignments and interacting with the LLM. Key elements include the course and assignment selection screen, the primary work area, the LLM chat/prompting panel, and the chat history.
- Educator Interface: EduSense shall provide administrative and monitoring capabilities. Key elements include the assignment creation screen, where the course, assignment, level of AI assistance, and optional features such as challenge mode can be selected, as well as the analytics dashboard for monitoring student usage, including assignment completion, average score, question performance, and student progress.
- Responsive Design: EduSense shall be optimized for all modern screen sizes and orientations (portrait and landscape), including desktop, laptop, tablet, and mobile phone displays.

3.1.2. Hardware Interfaces (O: Biehn)

End-User Devices:

- Client Devices: EduSense shall function correctly on any modern laptop, desktop, tablet, or mobile device that is capable of connecting to the internet.

[This space intentionally left blank.]

3.1.3. Software Interfaces (O: Biehn)

External Integration (Canvas LMS):

- Data Exchange: EduSense shall be capable of transmitting and receiving metadata, student interaction logs, and usage metrics to/from Canvas.
- API: EduSense shall integrate with the Canvas API to facilitate seamless synchronization of assignments and the tracking of student-AI interactions. The interface shall handle secure authentication and data transfer mechanisms required by the Canvas API.

Internal Frameworks:

- Front-end Runtime: The system utilizes HTML, CSS, JavaScript, and React executed within the client's web browser environment.
- Database: The system interacts with the SQLite database management system for storing user data, assignments, and interaction logs.
- Server Stack: The system employs the Django framework for managing backend logic, user requests, and LLM interaction.

3.1.4. Communications Interfaces (O: Biehn)

- Web Protocol: All client-server communication shall be secured using HTTPS with current TLS encryption standards.
- API Communication: All communication with the LLaMA model and the external Canvas API shall utilize secure, asynchronous API calls
- Network Requirements: Client devices shall maintain a stable connection to the public internet. The system shall be designed to tolerate minor network latency but requires persistent connectivity for real-time LLM interaction and usage tracking.

[This space intentionally left blank.]

3.2. System Features

This section details EduSense's system features.

3.2.1. Account Management

3.2.1.1. Introduction/Purpose of Feature

The purpose of this feature is to manage EduSense account details for students, teachers, and administrators. This provides teachers with the ability to customize each student's experience.

3.2.1.2. Stimulus/Response Sequence

Stimulus: A new user navigates to the account creation page and begins the account creation process.
Response: Once registration is complete, the user's form entries are processed, inspected for accuracy, encrypted, and securely stored on the EduSense database. An account creation confirmation message is provided to the user, and a verification email is sent to the email address on file to activate the account.

3.2.1.3. Associated Functional Requirements

3.2.1.3.1. Account Registration Form (O: Sapp)

The system shall provide the new user with an account registration form that gathers the following user information: Name, email address, school, account type, course numbers, date of birth, password, and Canvas authentication token.

3.2.1.3.2. Account Registration Validation *(O: Sapp)*

The system shall verify that the username, email address, and authentication token are unique to the new account. The account type is validated with a code provided by the system administrator or teacher. Course numbers will be optional for administrators.

3.2.1.3.3. Confirmation Email (O: Sapp)

The system shall provide the new user with a confirmation email that includes an activation link in the text of the email.

3.2.1.3.4. Account Activation *(O: Sapp)*

The system shall monitor the status of account activation, informing the user of any restrictions to the account due to an inactive status and prompting the user to perform the activation. Once active, all restrictions are lifted, and the status of account activation is updated.

3.2.1.3.5. Username Blacklist *(O: Sapp)*

The system shall maintain and regularly update a blacklist of words, word fragments, and other known identifiers that are not allowed to be included in whole or in part of a given username. If a user provides a new username with a token that is listed in the blacklist, a prompt will be issued to the user, and the change to the account is denied.

3.2.1.3.6. Encryption *(O: Sapp)*

The system shall encrypt all Personally Identifiable Information (PII) and store it securely in accordance with ISO / IEC 27701:2025 and the GDPR.

3.2.2. Login

3.2.2.1. Introduction/Purpose of Feature

The purpose of this feature is to provide secure access to a personalized experience on the EduSense program.

3.2.2.2. Stimulus/Response Sequence

Stimulus: A user interfaces with the login webpage, entering their credentials into the form.
Response: After the user enters their credentials, the system will process the given credentials and validate them. Then, the user will be prompted to complete a two-factor authentication process using the account's stored email.

Stimulus: A user interfaces with the login webpage by clicking on the forgot password link.
Response: The user is redirected to the forgotten password webpage and prompted to enter their email address to identify the locked account and spur a password reset email to be sent.

Stimulus: A user interfaces with the login webpage by clicking on the new account link.
Response: The user is redirected to the account creation webpage and prompted to enter their information and begin the activation process.

3.2.2.3. Associated Functional Requirements

3.2.2.3.1. Credential Validation (*O: Sapp)*

The system shall validate the credentials provided via the login form. If the credentials do not match any known account, a login failure notification will be presented to the user. Otherwise, the user will be directed to their account dashboard.

3.2.2.3.2. Password Recovery *(O: Sapp)*

The system shall provide a method of recovering a user's password that relies on the security of the user's email account provider. When a user navigates to the forgotten password webpage, they will be prompted to enter the email address associated with their account. They must then navigate to that email and locate the password reset link that was provided by the system.

3.2.2.3.3. Account Creation *(O: Sapp)*

The system shall provide a webpage and a link on the login page that will navigate the user to the account creation webpage. There, the user will be able to enter their account information and start the activation process.

3.2.3. Large Language Model Interfacing

3.2.3.1. Introduction/Purpose of Feature

The purpose of this feature is to provide the student with an interface that is both conducive to learning and encourages a sense of enjoyment.

3.2.3.2. Stimulus/Response Sequence

Stimulus: The user communicates with the EduSense LLM via the chat interface.
Response: The system receives the prompt and converts it to markdown format to increase the readability of the content for the LLM. Then, the LLM receives the prompt, reviews its restrictions, and alters the prompt based on those restrictions. Once all other context has been gathered, the LLM will provide a response. This response is then reviewed by the LLM to verify it does not violate any of its restrictions and hand the response off to the system. At which point, the system will finally provide the user with the LLM's response.

3.2.3.3. Associated Functional Requirements

3.2.3.3.1. Prompt Validation *(O: Sapp)*

The system shall review the prompt for attempts to subvert the system guardrails and alter the prompt to maintain its meaning, but worded in such a way that it does not violate any of the LLM's directives.

3.2.3.3.2. Response Validation *(O: Sapp)*

The system shall review the response for attempts to provide the user with information that subverts its given directives. If the response does so, the LLM will be forced to rephrase the response such that it no longer violates those directives.

3.3. Performance Requirements (O: Gebreyohannes)

This section defines the quantitative performance criteria for the EduSense prototype. The performance metrics cover response time, throughput, execution time, and storage capacity to ensure that the system operates efficiently under expected user loads.

3.3.1. Speed of Response (O: Biehn)

- LLM Response Time: Responses from the LLaMA model (including guided prompts or hints) shall be returned to the student interface within a maximum of 5 seconds under normal load.
- Dashboard Loading Time: The student dashboard, educator assignment creation screen, as well as the educator analytics dashboard shall load within a maximum of 5 seconds, even with large class sizes (up to 100 students)
- Account Creation Time: Account creation from the user's submission to receipt of confirmation shall be completed within a maximum of 7 seconds.

3.3.2. Throughput (O: Biehn)

- Usage Concurrency: The backend shall support at least 500 concurrent active users without degradation in the LLM response time requirement.
- Account Creation Processing: The backend shall support at least 100 account registrations per minute.

3.3.3. Storage Capacity (O: Gebreyohannes)

- Prototype Capacity: The SQLite prototype database shall store a minimum of 10,000 conversation records and 50,000 prompt and response pairs without exceeding 75 MB of total file size.
- Data Retention: All assignment, usage, and log data shall persist for at least 90 days to support instructor analytics and audit requirements.
- Logging and Backup: These storage requirements shall align with the daily backup and logging policies defined in section 3.6.

[This space intentionally left blank.]

3.3.4. Execution Time (O: Gebreyohannes)

- Prompt Processing: Each complete prompt and response cycle, including validation, model generation, and database write, shall complete within 15 seconds for 95 percent of requests under normal load.
- Database Transactions: CRUD operations executed through the Django ORM shall be completed within 200 milliseconds per query.
- Survey Startup: The system's backend shall initialize and accept requests within 10 seconds of startup.
- API Request Timeout: The local LLM API endpoint shall enforce a timeout limit of 300 seconds to prevent system hangs and resource overload.

3.4. Design Constraints (O: Donahue)

3.4.1. Standards Compliance (O: Donahue)

- The system shall comply with institutional IT security policies (e.g., FERPA).
- The system shall adhere to IEE 830-1998 documentation and software engineering standards.
- The backend shall follow RESTful API and JSON requirements.

3.4.2. Hardware and Software Limitations (O: Donahue)

- The local LLM (Ollama) shall operate on developer-provided hardware with limited GPU memory (6 GB VRAM).
- The system shall run on a standard university lab machine (Windows 10/11, Linux) without external cloud services.
- The database shall use SQLite for local storage.

3.4.3. Development Environment Constraints (O: Donahue)

- All backend components shall be implemented in Python 3.13.9 and Django 5.2.6.
- The AI integration module shall use Ollama API v0.12.7.
- Front-end development is limited to HTML/CSS/JavaScript with Django.

3.4.4. Regulatory, Ethical, and Academic Integrity Constraints (O: Donahue)

- The LLM shall not provide full answers or direct solutions to any prompts.
- The LLM shall not generate graded content.
- All user data is stored and processed locally (no cloud transfer).
- The system shall log interactions for transparency and auditing.

[This space intentionally left blank.]

3.4.5. Interface or Integration Constraints (O: Donahue)

3.4.5.1. Ollama Local API (O: Donahue)

- EduSense relies on the Ollama REST API for local model use.
- All LLM interactions occur via the HTTP endpoint http://localhost:11434/api.
- Network communication is local only; no external API keys are permitted.
- The model selection is limited to pre-loaded local models.

3.4.5.2. Django REST Framework (O: Donahue)

- EduSense exposes internal REST endpoints through the Django REST framework.
- The front end and AI modules communicate only through these endpoints.

3.4.5.3. Canvas LMS Integration (O: Donahue)

- EduSense will access Canvas data via the Canvas REST API v1 through the canvasapi Python wrapper v3.3.0.
- Only authorized endpoints (assignments, grades, users) may be used.
- The system shall comply with FERPA requirements when transmitting any student data.

3.5. Software System Attributes (O: Biehn)

This section defines the non-functional requirements that the EduSense application shall satisfy to ensure proper operation, security, and long-term viability.

3.5.1. Security (O: Biehn)

- LLM Interaction: The backend shall sanitize all student prompts before relaying them to the LLaMA model to prevent prompt injection or denial-of-service (DoS) vulnerabilities.
- Data Protection: All stored user data, assignment content, and usage logs shall be encrypted both in transit (via HTTPS as per 3.1.4) and at rest.
- Access Control: The system shall enforce strict role-based access control:
    - Students: Can only access their own assignments and LLM interaction history.
    - Educators: Can upload assignments, configure class settings, and monitor usage data for their enrolled students.
    - Authentication: All users must be authenticated before accessing any system features.

[This space intentionally left blank.]

3.5.2. Reliability/Availability (O: Biehn)

- Availability: EduSense shall aim for 99% uptime availability during standard operating hours. Maintenance windows shall be scheduled outside these hours.
- Assignment Delivery: Assignments uploaded by educators shall be available to students immediately and consistently without corruption.
- Database Integrity: The system shall guarantee accurate logging of all student-AI interactions, including time spent, prompts used, and Challenge Mode attempts. No usage data shall be lost due to system failures.

3.5.3. Maintainability (O: Biehn)

- Testability: The system shall support unit and integration testing to allow developers to verify the functionality of individual components easily.
- Code Standards: The Python/Django backend and JavaScript/HTML/CSS shall adhere to established industry coding standards and be thoroughly documented.
- Modularity: The core components shall be modularly designed to allow updates to one component without affecting others.

3.5.4. Portability (O: Biehn)

- Platform Independence: Since EduSense is a web application using standard browser technologies (HTML, CSS, JavaScript), it is inherently independent of the client device's operating system (Windows, Linux, Android, macOS, etc.).
- Database Migration: While SQLite is specified for the initial development, the database layer shall be designed to allow seamless migration to a more scalable relational database system (e.g., PostgreSQL or MySQL) in the future without major application code changes.

[This space intentionally left blank.]

3.6. Other Requirements (O: Donahue)

3.6.1. Backup and Data Recovery (O: Donahue)

- The system shall automatically backup its SQLite database and log files once every 24 hours.
- In case of system failure, data shall be restorable from the latest backup within 30 minutes.
- Backup archives shall be stored locally on the server.

3.6.2. Logging (O: Donahue)

- EduSense shall maintain logs of authentication attempts, API calls, and AI interactions for auditing purposes.
- Logs shall include timestamps, user IDs, and operation types.
- Logs shall be maintained for a minimum of 90 days and only accessible to administrators.