

---

# **Software Requirements Specification**

**for**

## **Vets 2 Tech**

**Version 1.6**

**April 3, 2025**

**Prepared by Connie Rodriguez, John McDurmon,**

**Beth Gallatin**

**Dr. Radana Dvorak**

**CSC 482 Senior Project**

**Hal and Inge Marcus School of Engineering**

**Saint Martin's University**

# Table of Contents

<b>Table of Contents .....</b>	<b>1</b>
<b>Revision History .....</b>	<b>1</b>
<b>1. Introduction.....</b>	<b>2</b>
1.1 Purpose .....	2
1.2 Scope .....	2
<b>2. System Overview.....</b>	<b>4</b>
2.1 Product Perspective .....	4
2.2 Product Features .....	12
2.3 User Classes and Characteristics .....	15
2.4 Design and Implementation Constraints.....	19
2.5 User Documentation .....	22
<b>3. Use Cases.....</b>	<b>24</b>
3.1-3.10 Individual Use Cases .....	24
<b>4. Non-Functional Requirements.....</b>	<b>31</b>
4.1 User Interface Design .....	31
4.2 Software Interfaces.....	35
4.3 Performance Requirements.....	38
4.4 Safety Requirements.....	39
4.5 Security Requirements.....	40
4.6 Software Quality Attributes.....	42
4.7 Professional Standards/IEEE/ACM/ADA .....	44
<b>Appendix A: Glossary.....</b>	<b>45</b>
<b>Appendix B: Issues List.....</b>	<b>47</b>
<b>Appendix C: Incident Response Plan (IRP).....</b>	<b>49</b>
<b>Appendix D: Wireframe.....</b>	<b>52</b>

## Revision History

Name	Date	Reason For Changes	Version
Beth Gallatin Connie Rodriguez	10/15/24	This template is used from previous project authored by Beth Gallatin and Connie Rodriguez	1.1
Connie Rodriguez	10/20/24	Updated Section 1	1.2
Connie Rodriguez	10/29/24	Updated Section 2	1.3
Connie Rodriguez	11/4/24	Updated Section 3	1.4
Connie Rodriguez	11/25/24	Updated Section 4 and Appendix	1.5
Connie Rodriguez	2/17/25	Updated Section 1, 2	1.6
Connie Rodriguez	3/25/25	Updated Section 3, 4	1.7

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to provide a comprehensive description of the Washington Vets 2 Tech (WAV2T) platform, an educational system designed to support prospective, current, and graduated students in their academic and professional journeys. This system provides tools for skills assessment, adaptive practice exams, study materials, and entrance exams tailored to three different learning pathways known as cohorts. Additionally, it integrates a chatbot for study assistance and resources for professional certification preparation.

This document details the system's purpose, features, interfaces, functionality, and constraints. It also explains how the system will interact with external inputs and environments. The document serves as a reference for the WAV2T development team, Dr. Dvorak, and the Engineering Advisory Board (EAB), ensuring alignment with the project's objectives and securing approval for further development.

## 1.2 Scope of Project

The WAV2T platform is an educational system that empowers students at all stages of their journey—prospective, current, and graduated—by offering tools and resources designed to foster success. The platform includes the following features:

- Secure Login System: Protects user data and provides access to tailored resources.
- Entrance Exams: Customized for each of the three pathways to determine readiness and placement.
- Skills Assessment: Evaluates user abilities to guide their learning path.
- Adaptive Practice Exams: Dynamically adjusts to the user's skill level for more effective learning.
- Study Materials: Comprehensive resources for entering the program, succeeding within it, and achieving professional certifications.
- Chatbot Study Buddy: An AI-powered assistant to support users with studying and resource navigation.
- Math Evaluation Exam: Ensures students meet pre-requisite requirements for program success.

- **Admin Data Dashboard:** Provides insights for the WAV2T team to assess program impact and secure ongoing funding.
- **Multi-Tabbed Layout:** Organized interface guiding users through the various educational tools and resources.

The WAV2T platform aims to:

1. **Streamline the Onboarding Process:** Provide prospective students with the tools to succeed in entrance exams and math prerequisites.
2. **Enhance Student Success:** Offer adaptive study materials and assessments for current students.
3. **Support Graduated Students:** Deliver resources to help alumni achieve industry-recognized certifications.
4. **Enable Data-Driven Decision-Making:** Equip administrators with analytics for continuous program improvement.

### 1.2.1 Key Technologies

The development of the WAV2T platform will leverage several key technologies to ensure a secure, and user-friendly system:

- **Python and Django Framework:** Provides a secure and scalable foundation for backend development, enabling efficient handling of user authentication, data handling, routing, adaptive quiz logic, and application programming interfaces (API) integration. Selected for its flexibility, security, and ease of integration with frontend and database tools.
- **Web Technologies:** Using Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript to power the frontend with a responsive, intuitive, and accessible user interface compatible across devices.
- **SQLite Database:** Offers lightweight, efficient storage for user data, entrance exam information, and study materials, ensuring quick access and simplified deployment.
- **Local Hosting (Django Development Server):** The current version of WAV2T is hosted locally during development and testing, ensuring quick deployment and debugging prior to full deployment on cloud infrastructure.
- **OpenAI GPT 3.5:** Powers the chatbot functionality for intelligent, natural-language student support.

- **Cybersecurity Protocols:** Integrates advanced security protocols, including secure login, encrypted cookies, role-based access control, data encryption, secure API communication, and future-ready SSL configuration, to protect student information and system integrity.
- 

### 1.2.2 Key Features

The WAV2T platform is designed with essential features to support students at every stage of their educational journey:

- **User Authentication and Role Management:** Django-powered login with secure user roles and session handling.
- **Entrance and Math Exams:** Includes randomized, timed entrance exams for three distinct pathways, Server and Cloud Application (SCA), Cloud Application Development (CAD), and Cybersecurity Administration (Cyber), ensuring proper placement and readiness evaluation. Math exam comprises of the MTH 101 curriculum that students must show comprehension of prior to entering the program.
- **Study Materials Repository:** Offers a curated repository of study resources, including videos, written materials, and practice exams. These resources are designed to support entrance preparation, ongoing coursework, and professional certifications.
- **Adaptive Practice Exams:** Leverages a machine learning model to dynamically tailor exam difficulty based on the user's performance and learning patterns. By analyzing factors such as accuracy, response time, and progression, the system provides personalized question sets and targeted feedback, optimizing learning outcomes and engagement.
- **Admin Panel:** Allows program staff to manage content, monitor quiz results, and export data for analysis.
- **Chatbot Study Buddy:** Real-time AI assistance using OpenAI API, integrated with Django views and routes, to help users with study-related queries and navigation.
- **Modular Interface Design:** Facilitates seamless navigation with organized tabs for each educational pathway, ensuring intuitive access to exams, study materials, and resources.

## **2. System Overview**

### **2.1 Product Perspective**

The WAV2T system is a secure educational platform designed to support prospective, current, and graduated students by providing a centralized location for entrance exams, study materials, adaptive practice exams, and certification preparation. The system employs a machine learning model to deliver personalized assessments, enhancing user engagement and optimizing learning outcomes.

The platform maximizes user engagement by offering structured, adaptive study materials and ensuring the secure handling of sensitive user data. Students are guided through the educational process with targeted resources and assessments tailored to their respective pathways (SCA, CAD, Cyber), with support from the integrated chatbot.

This platform addresses the need for a seamless and secure integration of exam administration, personalized study resources, and efficient user data management. It incorporates push notification functionality to alert WAV2T personnel when tests require grading, ensuring timely feedback. By adhering to the latest cybersecurity protocols, the WAV2T system delivers an intuitive, tailored experience while safeguarding personal and academic information.

#### **2.1.1 Relationship to Existing Systems**

The WAV2T platform is currently being developed as an independent, locally hosted system for proof of concept (PoC) purposes. The backend is built using Django and Python, with frontend communication utilizing standardized data formats such as JavaScript Object Notation (JSON). This ensures that the system is well-prepared for future integration with other educational or student management systems when deployed at scale.

While the platform is not currently connected to external systems, its modular architecture and use of RESTful APIs will support seamless integration in the future — enabling secure data exchange such as user records, exam results, study material usage analytics, and adaptive learning metrics.

Though not deployed on AWS at this stage, the system is being designed with future scalability in mind. When moved to a production environment, WAV2T can be transitioned to cloud infrastructure like AWS or other hosting solutions to provide high availability, data redundancy, and enterprise-level security controls.

#### **2.1.2 Major Components and Subsystem Interconnections**

The WAV2T system is built with a layered architecture comprising the following components:

## 1. User Interface (UI) Layer:

- **Description:**
  - The UI layer provides components for interacting with the platform, allowing students and administrators to navigate the entrance exams, study materials, and account management sections.
- **Components:**
  - **Student Dashboard:** Interface for users to access study materials, view exam results, and manage personal information.
  - **Admin Dashboard:** Interface for administrators to manage user accounts, entrance exam questions, and monitor the system.
  - **Pathway-Specific Pages:** Individual pages for each educational pathway (SCA, CAD, Cyber) with associated resources, study materials, and exams.
  - **User Profile Management:** Interface allowing users to update personal information, such as recovery emails, and change passwords.
- **Technologies:**
  - **Django Templating Engine (HTML/CSS/JS):** Renders dynamic content using Django views and context.
  - **Bootstrap:** Provides responsive and consistent UI styling.
  - **JavaScript:** Enhances interactivity with the site through things such as quiz toggles, or timers.
  - **ARIA/Web Accessibility:** Improves usability across assistive technology.

## 2. Security Layer:

- **Description:**
  - The security layer manages authentication, role-based access, authorization, and the protection of sensitive data.
- **Components:**

- **Authentication:** Secure login and logout functionalities via Django Allauth with role enforcement.
- **Authorization:** Role-based access control for administrators, prospective students, current students, and graduates.
- **Secure Communication:** Local hosting uses HTTPS for encrypted transport (TLS ready).
- **Data Encryption:** User passwords stored using Django's built-in password hashing.
- **Technologies:**
  - **Django Allauth and Middleware Security Settings:** Handles secure user authentication, login/logout, password management, and email verification using Django's built-in authentication system and Allauth for social and email login options.
  - **Session Management:** Tracks authenticated user sessions securely and expires sessions upon logout or inactivity to reduce unauthorized access risks.
  - **TLS/SSL (Configured for Production Readiness):** Ensures encrypted data transmission between client and server. Although currently in local development, the system is ready for TLS/SSL configuration upon deployment.
  - **Role-Based Access Control (RBAC):** Restricts access to features and data based on user roles (admin, student, graduate), ensuring only authorized users can access sensitive areas.
  - **Password Hashing (Django default):** Protects user passwords by storing them as salted, hashed values rather than plain text, significantly enhancing credential security.
  - **Login Attempt Limiting (via Django-Axes):** Helps prevent brute force attacks by limiting repeated failed login attempts and locking accounts after too many failures.

### 3. Presentation Layer:

- **Description:**



- This layer handles user requests, processes input and presents data to the user.
- **Components:**
  - **Views:** Handles logic and routes for displaying quizzes, materials, and dashboards.
  - **Templates:** Renders HTML content based on user role and context.
  - **Forms/Context Processors:** Manage input and data exchange between frontend and backend.
- **Technologies:**
  - **Django Views:** Handle HTTP requests, determine logic flow, and prepare data for templates.
  - **Django Templates:** Generate the HTML content served to users, dynamically injected with context variables.
  - **Django Forms & Context Processors:** Manage user inputs and pass shared data to all templates.
  - **HTML, CSS, JavaScript:** Create a responsive and accessible user interface, ensuring compatibility across devices and browsers.
  - **Web Accessibility Features (ARIA):** Enhance usability for individuals with disabilities by providing semantic roles and properties to dynamic web elements.
  - **Bootstrap:** Streamline responsive design and ensure consistency across UI components.

#### 4. Application Layer:

- **Description:**
  - This layer contains the core business logic, handling interactions between the presentation and domain layers.
- **Components:**
  - **Exam Management Service:** Handles logic for creating, managing, grading, and submitting entrance exams and practice exams. Integrates machine

learning for adaptive practice exams, dynamically adjusting question difficulty based on user performance.

- **User Management Service:** Manages registration, authentication, and role-based access controls. Supports operations such as password recovery, user profile updates, and role assignment.
- **Notification System:** Manages push notifications to WAV2T personnel, alerting them when exams require grading. As well as users to their user account when exams have been graded or new study material becomes available.
- **Study Material Service:** Provides access to curated study materials, including videos, documents, and other resources tailored to each educational pathway (SCA, CAD, Cyber). Tracks resource usage analytics for potential insights.
- **Pathway Management:** Guides users through their specific pathway, ensuring they access the relevant resources and exams based on their chosen track.
- **Technologies:**
  - **Python (Django Logic):** Powers backend processes and service functions across the platform.
  - **Django Signals:** Listens for backend events (e.g., quiz submission) to trigger actions like notifications.
  - **Adaptive Logic:** Uses quiz performance data to dynamically adjust question difficulty.
  - **Machine Learning Integration:** Employs a custom-trained model to manage adaptive learning in practice exams.
  - **Dependency Injection (DI):** Manages service lifetimes and dependencies efficiently for modular design.

## 5. Domain Layer:

- **Description:**
  - The domain model represents the core entities and business rules for WAV2T.

- **Components:**
  - **User:** Represents users (students, administrators) with attributes like name, ID numbers, pathway, and role.
  - **Exam:** Represents entrance exams with attributes like questions, time limits, and scores.
  - **Adaptive Logic Practice Exam:** Represents dynamically tailored exams with attributes like question difficulty, user performance metrics, and skill progression tracking, encapsulating the business rules for machine learning-driven adaptive practice exams.
  - **Study Materials:** Represents the resources available to students, including videos, texts, and practice exams.
- **Technologies:**
  - **Django Models:** Represent domain entities and structure the database.
  - **POPOs (Plain Old Python Objects):** Represent business logic independently of Django's ORM.
  - **Business Rules:** Define how objects interact (e.g., quiz difficulty increases after success).
  - **SQLite Database:** Lightweight efficient storage of domain data.

## 6. Persistence Layer:

- **Description:**
  - This layer handles data storage and retrieval from the database.
- **Components:**
  - **Django ORM:** Maps model objects to relational database tables.
  - **Data Queries:** Retrieves or filters information from tables.
  - **Migration System:** Tracks changes to models and updates schema accordingly.
- **Technologies:**
  - **SQLite:** Lightweight relational database used for local development and PoC testing.

- **Django ORM:** Maps models to tables and automates database queries and updates.
- **Migrations:** Tracks and applies changes to the database schema.
- **Querysets and Q Objects:** Allow complex filtering and querying of data using Python.

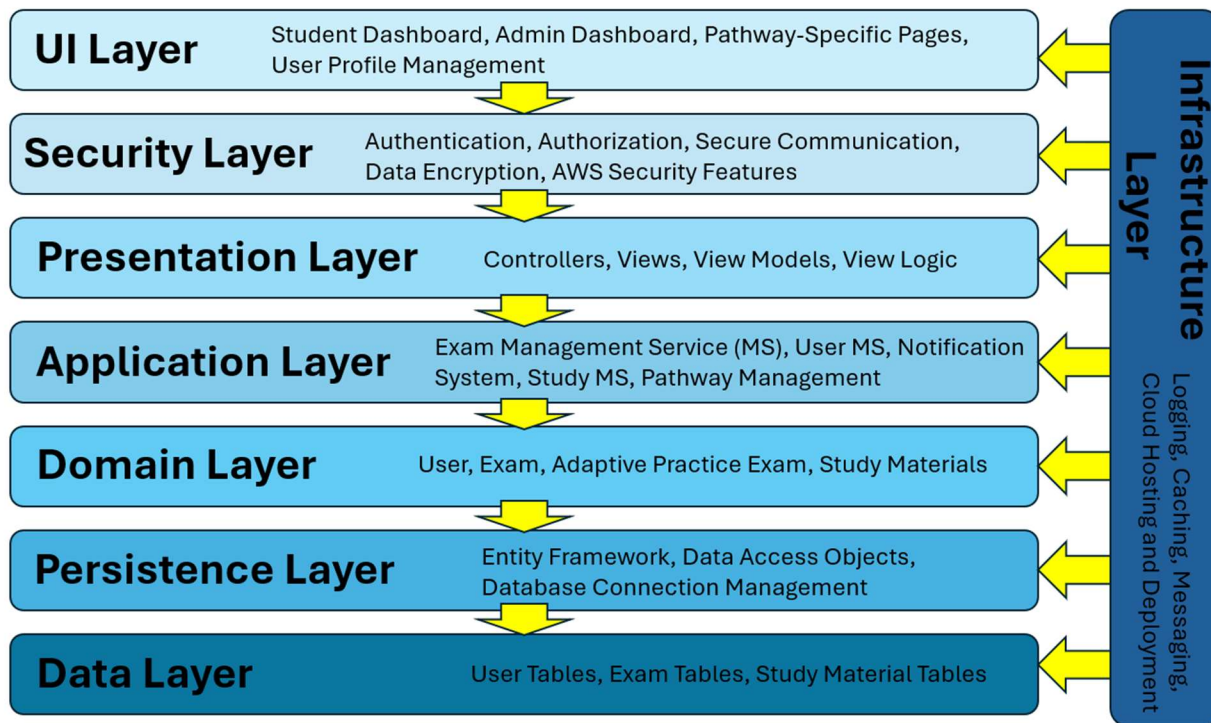
## 7. Data Layer:

- **Description:**
  - The data layer represents the actual database where information about users, exams, and study materials is stored.
- **Components:**
  - **User Tables:** Store information about users, including personal details, roles, and pathway selections.
  - **Exam Tables:** Store entrance exam questions, answers, results, and performance logs for adaptive learning.
  - **Study Material Tables:** Contain metadata and links for study resources like videos, texts, and practice exams.
  - **Progress Table:** Tracks student's performance over time.
- **Technologies:**
  - **SQLite Database:** A lightweight, file-based database chosen for its simplicity and efficiency in small to medium-scale applications.
  - **Encrypted Fields:** Ensures sensitive information (e.g., passwords) is stored securely.
  - **Django Migrations:** Automatically generates database tables and schema updates from model changes.

## 8. Infrastructure Layer:

- **Description:**
  - This layer provides supporting services and utilities for the system.
- **Components:**
  - **Logging:** Captures logs for monitoring, security audits, and debugging.

- **Caching:** Stores frequently accessed data (e.g., study materials and user session data) to enhance performance and reduce database load.
- **Push Notification Logic:** Implements push notifications for WAV2T personnel to alert them about pending exam grading.
- **Deployment (PoC):** Currently hosted locally using Django's development server. Future versions will be containerized or deployed via a cloud server.
- **Technologies:**
  - **Python Logging Module:** Records backend events for debugging and auditing.
  - **Django Caching Framework:** Stores frequently accessed data to improve performance.
  - **Static Files Middleware:** Serves CSS, JS, and media files locally during development.
  - **Local Hosting (Django Dev Server):** The platform is currently hosted on a local machine for PoC testing, with cloud deployment planned for production.
  - **TLS/SSL Ready:** System is designed to support secure HTTPS once deployed in production.



### 2.1.3 External Interfaces

1. **Student Management Systems:** WAV2T platform is designed to integrate with external student management or academic systems using APIs and standardized data formats (e.g., JSON and XML). This enables the synchronization of user data, such as enrollment status, exam scores, and user roles (e.g., prospective, current, or graduated students). These integrations facilitate the seamless import and export of student information, ensuring the WAV2T system remains accurate and up to date.
2. **Email Systems:** While email system integration is planned for a future release, it is not currently implemented in this version of WAV2T. Notifications will be delivered internally through the platform's user interface (push-to-account only), including alerts for completed grading and updates to study materials. When implemented, email functionality will allow for automated delivery of exam results, account updates, and administrative alerts.
3. **Authentication Services:** The current system uses Django Allauth for local authentication, managing secure user login, registration, and session handling. Advanced security features such as single sign-on (SSO) and two-factor authentication (2FA) are not yet implemented but are planned for future

development. The system is structured to allow integration with external identity providers like AWS Cognito or Auth0 in production environments.

## 2.2 Product Features

The WAV2T system is designed to provide educational support through a range of features that cater to prospective, current, and graduated students. Below is a high-level overview of the key functionalities:

### 2.2.1 Product Overview

#### 1. User Account Management

- **User Profiles:** Allows users to create and manage their profiles, specifying details like their name, chosen pathway, semester of enrollment, current certifications, and ID numbers (student or application ID). Profiles also allow users to update their recovery email and personal preferences for a personalized experience.
- **Role-Based Access Control:** Implements role-based access to distinguish between prospective students, current students, graduates, and administrators. Each role is granted access only to the resources and features applicable to them.
- **Authentication and Security:** Login system powered by Django Allauth with encrypted password storage and CSRF protection. While 2FA is not yet active, the system is designed with expansion to include advanced authentication options.

#### 2. Entrance Exam Management

- **Exam Creation and Administration:** Allows administrators to create and manage three distinct entrance exams, tailored to each educational pathway (Server and Cloud Applications, Cloud Application Development, and Cybersecurity Administration). The exams pull questions from a bank of items.
- **Exam Randomization and Timing:** Ensures that each student receives a randomized set of 65 questions, consisting of multiple-choice and short-answer formats, and is given 90 minutes to complete the exam.
- **Results and Feedback:** Admins can manually review and grade exams. Students receive internal push notifications on their account dashboard when results are posted.

### 3. Study Materials and Resources

- **Study Material Repository:** Offers students access to a wide range of study resources, including videos, texts, and practice exams, organized by pathway. The system provides materials to help students prepare for both entrance exams and certifications (e.g., CompTIA Network+, Security+).
- **Adaptive Practice Exams:** Features machine learning-powered adaptive practice exams that adjust question difficulty based on user performance. Students can choose between timed and untimed options, with instant feedback to enhance learning outcomes.
- **Study Tools:** Includes interactive and engaging study tools to assist students in mastering content for entrance exams and certifications.

### 4. Notification and Reminder System

- **Internal Push Notifications:** The platform currently delivers internal notifications only (no email). Admins are notified when grading is required, and students are alerted when results are posted or new resources become available.
- **Progress Tracking:** Users can monitor their quiz history, see their current skill level by topic, and track upcoming study goals and milestones within their dashboard.

### 5. Reporting and Analytics

- **Performance Dashboards:** Displays metrics such as exam completion rates, average scores, and user engagement with study materials. Dashboards are available for both administrators and students to track performance.
- **Custom Reports:** Administrators can generate detailed reports on user performance and engagement. Filters by date range, pathway, and user group support data-driven decision-making.
- **Real-Time Analytics:** Provides live insights into user activity and system usage, helping administrators optimize the system and assess the effectiveness of exams and resources.

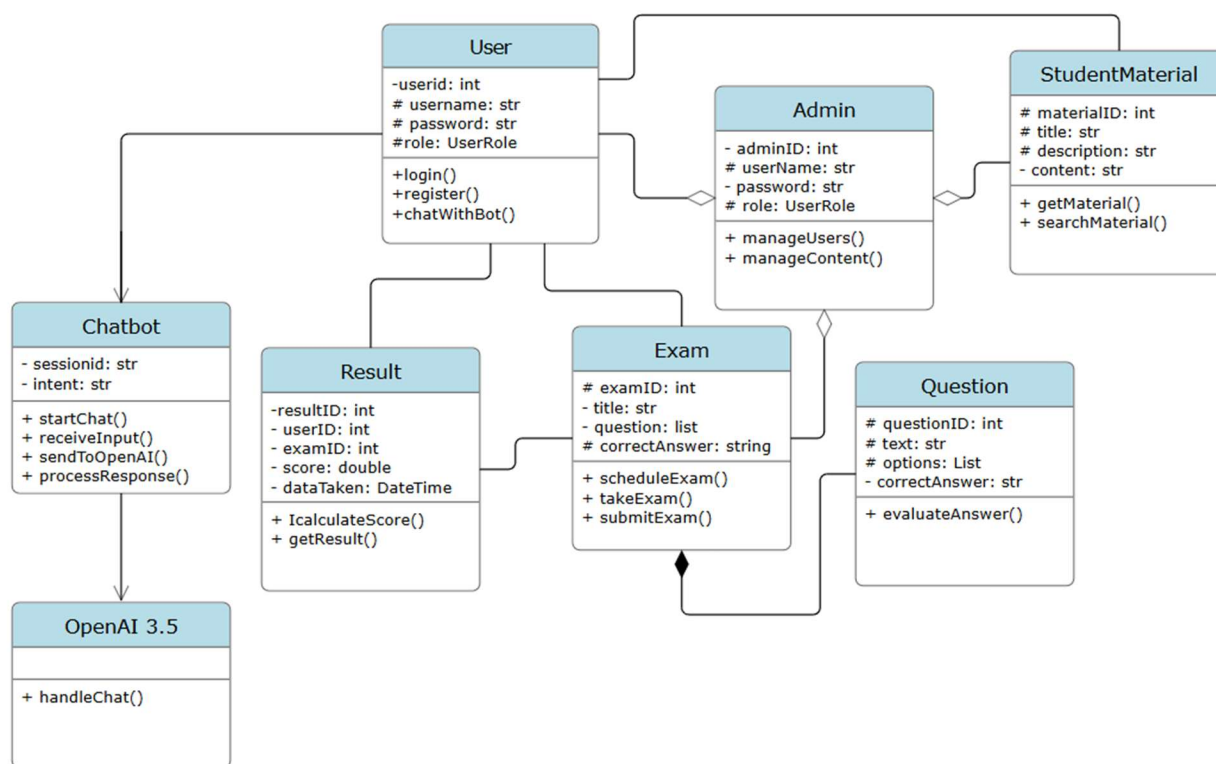


## 6. Integration Capabilities

- API Integration (Planned):** While not currently implemented, the WAV2T backend supports future API integration using JSON/XML. This will allow two-way syncing of user data, exams, and learning analytics with institutional systems.
- Data Import and Export:** Admins can manually import bulk user and exam data for onboarding or archival purposes. Exports will support performance reports and research evaluations in the future.

### 2.2.2 System Interaction Diagram

To visually represent how these features interact within the WAV2T system, a top-level system interaction diagram is provided below. This diagram illustrates the modular structure of the system, showing the relationships between the key components, including the user interface, exam management, study materials, notifications, and the underlying database architecture.



## 2.3 User Classes and Characteristics

The WAV2T educational platform is designed to support various user classes, each with distinct roles, responsibilities, and interactions with the system. These user groups are

defined by their purpose within the educational process, frequency of use, security access levels, and the system features they utilize. Below is a detailed description of the main user classes expected to interact with the WAV2T system:

## **1. Administrators (Admins)**

- **Characteristics:**

- Staff members or educators responsible for managing the overall system, including user accounts, entrance exams, and study materials.
- High technical expertise, with a comprehensive understanding of the platform's functionalities and security protocols.
- Frequent users of the system, primarily focusing on exam management, user data, study materials, and monitoring system activity.

- **Privileges:**

- Full access to all system modules, including account management, exam creation, study material management, and analytics dashboards.
- Ability to manage user roles, configure system settings, and oversee chatbot interactions and integrations with Amazon Lex.
- Can configure system settings, manage user roles, and maintain integration with external systems.
- Notifications to manage exam grading and platform maintenance tasks.

- **Requirements:**

- Advanced tools for creating and managing exams, study materials, monitoring system logs, and sending targeted notifications.
- Access to detailed audit logs for tracking platform activity and ensuring compliance with security protocols.

## **2. Prospective/Incoming Students**

- **Characteristics:**

- Students applying to the institution, primarily interacting with the system to take entrance exams and access preparatory study materials.
- Low to moderate technical expertise, requiring simple, intuitive interfaces for navigation.

- Frequent users during the application process, with an emphasis on exam preparation and completion.
- **Privileges:**
  - Access to pathway-specific entrance exams (SCA, CAD, Cyber) and related preparatory study materials.
  - Ability to view exam instructions, complete exams, and receive exam results.
- **Requirements:**
  - Clear navigation to exams and study materials, with guidance on using the platform.
  - Secure exam environments to maintain academic integrity and user trust.
  - Notifications for exam-related updates, including results and deadlines.
  - Secure exam environments to maintain academic integrity.

### 3. Current Students

- **Characteristics:**
  - Enrolled students using the platform to access study materials, take practice exams, and manage academic progress.
  - Moderate technical expertise with regular platform interactions for learning and receiving system notifications.
  - Frequent users of study resources and practice exams, particularly for certifications.
- **Privileges:**
  - Access to pathway-specific study materials (e.g., videos, texts, practice exams) for entrance exams and certifications (e.g., CompTIA Security+).
  - Ability to track progress, receive feedback on practice exams, and manage learning pathways.
  - Notifications for certification deadlines, study material updates, and reminders.
- **Requirements:**

- A well-organized repository of study resources and adaptive practice exams tailored to individual performance.
- Notifications to assist with study planning and deadlines.
- Practice exams with customizable settings for feedback and time limits.
- Cross-platform accessibility for ease of use on multiple devices.

#### **4. Graduated Students**

- **Characteristics:**

- Former students accessing the platform for professional development and certification preparation.
- Varied technical expertise, with occasional use for study materials and certification practice exams.
- Infrequent interaction, focused on certification preparation and professional growth.

- **Privileges:**

- Access to all certification study materials, including adaptive practice exams and professional development resources.
- Ability to track preparation progress and receive feedback on practice exams.

- **Requirements:**

- Simplified access to certification resources and tools to help with ongoing professional development.
- Cross-platform support to allow access from multiple devices, making it easy to study on the go.

#### **5. Support Staff**

- **Characteristics:**

- IT staff or platform administrators responsible for ensuring system performance, security, and troubleshooting.
- High technical expertise, with deep knowledge of the platform's backend and database architecture.

- Frequent interaction for maintaining system operations, resolving issues, and implementing updates.
- **Privileges:**
  - Full access to system configuration, database management, and monitoring tools.
  - Ability to troubleshoot technical issues, manage chatbot interactions with Amazon Lex, and ensure compliance with security protocols.
- **Requirements:**
  - Advanced administrative tools for backend maintenance, user support, and platform updates.
  - Real-time monitoring and logging tools to ensure system reliability and identify potential security risks.

## 2.4 Design and Implementation Constraints

The development and deployment of the WAV2T educational platform are subject to various constraints that influence design choices and implementation strategies. These constraints ensure the system remains secure, scalable, and user-friendly for all types of students and administrators within a college setting.

### 1. Regulatory and Institutional Policies

- **Data Privacy Regulations:**
  - Compliance with the Family Educational Rights and Privacy Act (FERPA) and any local/state regulations governing student data privacy. This ensures strict handling and protection of student data, including exam results and personal information.
- **Institutional Security Policies:**
  - Adherence to institutional protocols for encryption, access control, and auditing to meet security and privacy requirements, for future possible integration.

### 2. Hardware and Infrastructure Limitations

- **Local Hosting for Proof of Concept:**

- The current version of WAV2T is hosted locally for development and testing. Future deployment will require migrating to a scalable cloud environment, such as AWS or an institutional server, with secure HTTPS configuration.
- **Client Device Compatibility:**
  - The platform must be fully responsive and support cross-platform compatibility on desktops, laptops, tablets, and smartphones, ensuring seamless user experiences across devices.

### 3. Technological Constraints

- **Programming Languages and Frameworks:**
  - The platform is built using the Django framework with Python as the primary language. This limits compatibility with tools or libraries built for other ecosystems such as .NET or Java.
- **Database Management:**
  - SQLite is currently used as a lightweight, file-based database for local development. While sufficient for the proof of concept, future expansion may require migration to a more scalable database system (e.g., PostgreSQL or Amazon RDS).

### 4. Interface Requirements

- **User-Centric Design:**
  - Interfaces must prioritize simplicity and accessibility to accommodate users with varying levels of technical expertise. Students and administrators should easily navigate features such as exams, study materials, and notifications. Stakeholder feedback will be incorporated iteratively to refine usability and meet user expectations.
- **Communication Protocols:**
  - The system must support standard internet protocols, such as HTTPS for secure communication and SMTP for email notifications related to exam results, reminders, and updates.

### 5. Parallel Operations

- **Concurrency Management:**

- Efficient transaction and concurrency control mechanisms are required to handle simultaneous interactions by multiple users, such as students taking exams and administrators managing resources, without data conflicts or performance degradation.

## **6. Adaptive Testing Model Development:**

- **Dynamic Testing Framework:**

- A machine learning-powered adaptive testing model will dynamically adjust question difficulty based on user performance. The development of this feature must address AI complexity while ensuring that the system remains intuitive and user-friendly.

- **Feedback Integration:**

- Real-time feedback from users and stakeholders will guide iterative improvements to the adaptive testing framework.

## **7. Accessibility Standards**

- **WCAG Compliance:**

- The platform must meet Web Content Accessibility Guidelines (WCAG) standards to support users with disabilities. This includes keyboard navigation, screen reader compatibility, and adequate color contrast for readability.

## **8. Security Considerations**

- **Authentication and Authorization:**

- The system implements secure authentication via Django Allauth and enforces role-based access control (RBAC) to restrict data access according to user roles (e.g., admin, student, graduate).

- **Data Encryption:**

- All Sensitive data such as passwords is encrypted at rest using Django's default hashing (PBKDF2), and HTTPS will be enforced in production to encrypt data in transit.

## **9. Design Conventions and Programming Standards**

- **Code Maintainability:**

- The codebase will follow Django and Python PEP8 coding standards, using clear naming conventions, modular architecture, and documentation to support long-term maintainability and collaboration.
- **Software Development Lifecycle:**
  - The project will follow an Agile development methodology to ensure iterative progress, continuous feedback, and timely delivery with frequent testing to maintain quality. Stakeholder feedback will be actively integrated during development cycles to ensure the platform meets evolving requirements.

## 2.5 User Documentation

This documentation will cater to the different user classes, providing detailed guidance specific to each group's interactions with the WAV2T platform. Below is a list of the key user documentation components and their delivery formats.

### 1. User Manuals

- **Administrator Manual:**
  - Detailed instructions for administrators on configuring the platform, managing user roles, creating and reviewing entrance exams, uploading and managing study materials, and troubleshooting common system issues.
  - Includes guidance on generating analytics reports, managing notifications, and monitoring system activity.
  - Section dedicated to integrating and managing chatbot functionality via Amazon Lex.
- **Student Manual:**
  - Instructional guidance for prospective and current students on completing entrance exams, using adaptive practice exams, accessing study materials, viewing exam results, and updating profile information.
  - Instructions for interacting with the chatbot study buddy for study assistance and navigation support.
- **Support Staff Manual:**
  - Technical guide for IT support personnel, including instructions on maintaining system performance, managing data backups, resolving user issues, and implementing security updates.



- Covers advanced troubleshooting techniques, system monitoring, and database maintenance procedures specific to SQL and AWS integration.

## 2. Online Help

- **Context-Sensitive Help:**

- Integrated help features, such as tooltips and clickable help icons, that provide users with relevant guidance based on the section of the system they are interacting with (e.g., during exam-taking or profile management).
- Step-by-step walkthroughs for key tasks, such as exam submission or profile updates, available within the interface.

- **FAQ Sections:**

- Searchable repository of frequently asked questions addressing common issues and queries for all user classes.
- Includes troubleshooting tips, best practices, and links to relevant sections of the user manual for additional detail.
- Categorized by user class (e.g., students, administrators, support staff) for easy navigation.

## 3. Tutorials

- **Video Tutorials:**

- Step-by-step instructional videos that guide users through common processes, such as logging in, taking exams, accessing study materials, and checking exam results. These videos will provide visual and auditory learning aids.

- **Interactive Tutorials:**

- Interactive modules that walk users through the system's core features, such as exam preparation and submission, allowing them to practice tasks in a controlled environment.

## 4. Quick Start Guides

- **Printable Quick Reference Cards:**

- Simple, easy-to-follow guides that provide quick tips for frequently performed tasks. These guides will be available for download as PDFs and can be printed for easy reference.

## 5. Technical Support Documentation

- **Troubleshooting Guide:**

- A guide designed to help users and IT staff diagnose and resolve common technical issues, such as login problems or issues accessing exams.

- **API Documentation:**

- Detailed technical documentation for administrators and developers, outlining how the system's APIs can be used to integrate with other college systems, such as the SIS.

## 6. Delivery Formats and Standards

- **Digital Manuals:**

- Manuals and quick start guides will be provided in PDF format for easy distribution, printing, and offline access by users.

- **Online Accessible Content:**

- All help sections and tutorials will be accessible directly within the platform, ensuring the content adheres to WCAG accessibility standards.

- **Interactive Tutorials:**

- Step-by-step guided tutorials for onboarding new users, accessible directly from the dashboard or during first-time login.
- Includes short videos or animations for complex features such as adaptive practice exams or exam management.

- **Video Content:**

- Video tutorials will be hosted both within the system and potentially on an external platform, such as YouTube or the college's online resources, making them easily accessible to students and administrators.

- **Chatbot Integration:**

- **Student-Focused Study Assistant:** The chatbot study buddy, powered by OpenAI's language model, provides students with on-demand support for

studying, navigating the platform, and understanding key concepts related to their learning pathway.

- **Role-Specific Access (Planned):** While the current chatbot interface is available only to students, future development may include separate interfaces or functionalities tailored for administrators and support staff.
- **Context-Aware Responses:** The chatbot delivers helpful responses, which may include study tips, explanations of exam topics, and links to relevant materials or internal platform resources. Integration with structured FAQs and documentation is planned for future implementation.

### 3. Use Cases

The following is a collection of Use Cases modeling the functional requirements for the WAV2T platform, designed to serve prospective, current, and graduated students, as well as administrators.

#### 3.1 Register for an Account

##### 1. Actors:

- **User:** A prospective, current, or graduated student who wants to register for an account on the WAV2T platform.

##### 2. Preconditions:

- The user has access to a web browser with internet connectivity.
- The WAV2T platform URL is accessible to the user.
- The user has not previously registered for an account with WAV2T.

##### 3. Basic Flow:

- The user navigates to the WAV2T platform URL.
- The landing page is displayed with options to log in or register.
- The user clicks on the "Register" button.
- The platform prompts the user to fill in required details, including name, pathway (SCA, CAD, Cyber), semester of enrollment, student ID or application ID (if applicable), email, and password.

- The platform validates the details and checks if the username or email is already in use.
- Upon successful registration, the user is redirected to the login page and can log in with their credentials.

#### 4. **Alternate Flows:**

- If the email format is invalid, the system displays an error message and prompts the user to re-enter a valid email.
- If the chosen username is already taken, the system requests the user to select a different one.
- If the password is weak, the system requests a stronger password.

#### 5. **Postconditions:**

- The user successfully registers for an account and can log in to access the system.
- The user's information is securely stored in the database.

### 3.2 Login/Logout

#### 1. **Actors:**

- **User:** A prospective, current, or graduated student who wants to log in or out of the WAV2T platform.

#### 2. **Preconditions:**

- The WAV2T platform is accessible via a web browser.
- The user has registered an account and possesses valid login credentials.

#### 3. **Basic Flow (Login):**

- The user navigates to the WAV2T platform URL.
- The platform displays the login page with fields for email and password.
- The user enters their credentials, and the platform validates them against the stored database.
- If the credentials are valid, the user is redirected to the dashboard.

#### 4. **Alternate Flows:**

- If incorrect credentials are entered, the system displays an error message.
- If the user forgets their password, the system provides a "Forgot Password" option to reset it via email.

**5. Postconditions (Login):**

- The user successfully logs into the WAV2T platform and gains access to features such as exams and study materials.

**6. Logout Flow:**

- The user navigates to their profile or settings and clicks on "Log out."
- The system clears the user session and redirects to the login page.

**7. Postconditions (Logout):**

- The user successfully logs out, and session data is invalidated.

### **3.3 Create Administrator Account**

**1. Actors:**

- **Admin:** A system administrator responsible for managing user roles and permissions.

**2. Preconditions:**

- The administrator is logged into the system and has elevated privileges.

**3. Basic Flow:**

- The administrator accesses the admin dashboard.
- The system provides options to create new administrator accounts.
- The administrator fills out the form with details such as email, username, and password for the new admin.
- The system validates the information and creates the account.

**4. Alternate Flows:**

- If invalid details are entered, the system displays error messages for correction.

**5. Postconditions:**

- A new administrator account is successfully created.

### 3.4 Create Exam

#### 1. Actors:

- **Admin:** A system administrator responsible for managing entrance exams for the platform.

#### 2. Preconditions:

- The administrator is logged into their account.

#### 3. Basic Flow:

- The admin navigates to the exam management section.
- The system displays options to create new exams for the pathways (SCA, CAD, Cyber).
- The admin enters details for the exam, including the question bank, number of questions, and time limit.
- The system validates the exam details and saves the exam.

#### 4. Alternate Flows:

- If the details are invalid, the system requests corrections.

#### 5. Postconditions:

- A new exam is successfully created and made available to prospective students.

### 3.5 Take Entrance Exam

#### 1. Actors:

- **Prospective Student:** A student applying to the program and taking the entrance exam.

#### 2. Preconditions:

- The user is registered and logged into the WAV2T platform.
- The entrance exam is available for the student's chosen pathway.

#### 3. Basic Flow:

- The student navigates to the exam section and selects their entrance exam.
- The system displays instructions and a disclaimer regarding the use of AI or external resources.
- The student confirms and begins the exam, answering multiple-choice and short-answer questions within the time limit.
- The student submits the exam, and the system confirms that results will be available in 3-5 business days.

#### 4. **Alternate Flows:**

- If the student does not complete the exam in time, the system automatically submits their answers.
- If the student exits the exam, the system notifies them of the incomplete submission.

#### 5. **Postconditions:**

- The student completes the exam, and the system stores the responses for review.

### 3.6 Access Study Materials

#### 1. **Actors:**

- **Student:** Current or graduated students accessing study materials for exams or certifications.

#### 2. **Preconditions:**

- The user is logged into the system and has access to their respective study pathway.

#### 3. **Basic Flow:**

- The student navigates to the study materials section.
- The system displays videos, texts, and practice exams for the chosen pathway.
- The student selects and interacts with the materials, such as watching videos or taking practice tests.

#### 4. **Alternate Flows:**

- If the student selects a practice exam, they can choose between timed or untimed modes.

**5. Postconditions:**

- The student successfully accesses study materials and completes practice exams.

### **3.7 View Exam Results**

**1. Actors:**

- **Prospective/Current/Graduated Student:** The student awaiting entrance or certification exam results.

**2. Preconditions:**

- The student has completed an exam and results are available.

**3. Basic Flow:**

- The student receives a notification that their exam results are ready.
- The student logs into their account and navigates to the results section.
- The system displays the exam score and any additional feedback.

**4. Alternate Flows:**

- If the exam results are delayed, the system notifies the student of the delay.

**5. Postconditions:**

- The student views their exam results and can proceed based on the outcome.

### **3.8 Chatbot Interaction**

**1. Actors:**

- **User:** Current Student or administrator of the WAV2T program.

**2. Preconditions:**

- The user is logged in and accesses the chatbot feature.

**3. Basic Flow:**

- The user types a query into the chatbot interface.



- The chatbot provides relevant answers or links to documentation.

**4. Alternate Flows:**

- The chatbot escalates complex issues to admin or support staff.

**5. Postconditions:**

- The user receives relevant assistance or is directed to further support.

### **3.9 Receive Notifications**

**1. Actors:**

- **Admin:** Receives exam grading alert.
- **Student:** Receives update on exam results, study material releases, and deadlines.

**2. Preconditions:**

- Notifications are configured and enabled.

**3. Basic Flow:**

- The system sends a notification to the correct user.

**4. Alternate Flows:**

- Notification fails the system retries or logs the failure.

**5. Postconditions:**

- The user correct user receives the notification.

### **3.10 Generate Reports**

**1. Actors:**

- **Admin:** Generates analytics reports for user performance and system activity.

**2. Preconditions:**

- The admin is logged in with reporting privileges.

**3. Basic Flow:**

- The admin selects parameters and generates a report

**4. Alternate Flows:**

- Invalid parameters are input, the system requests corrections.

#### 5. Postconditions:

- A report is generated and available for analysis.

## 4. Non-Functional Requirements

### 4.1 User Interface Design

This section defines the design principles, key interface components, and layout guidelines for the WAV2T platform, ensuring a user-friendly experience tailored to the needs of prospective, current, and graduated students, as well as administrators. The design prioritizes accessibility, consistency, and scalability across all interfaces.

#### 4.1.1 Mockups and Description of Key Interfaces

##### 1. Login Page:

- **Purpose:** Secure entry point for all user classes (prospective students, current students, graduated students, and administrators).
- **Features:**
  - Fields for email and password input.
  - Login button.
  - Password recovery link for users who forget their credentials.
  - Two-factor authentication (2FA) prompt for enhanced security.
- **Standards:**
  - Simple, intuitive design focused on user ease-of-use.
  - Clear layout to ensure accessibility for all users, including keyboard navigation and screen reader compatibility.

##### 2. Admin Dashboard:

- **Purpose:** Central hub for administrators to manage exams, study materials, and user accounts.
- **Features:**
  - Navigation menu with links to system settings, user management, exam creation, and analytics dashboards.

- Activity logs for tracking user actions, exam submissions, and notifications.
- Quick links for adding study materials, creating exams, and reviewing results.
- **Layout:** Modular, widget-based dashboard with collapsible sections for different administrative functions, allowing flexibility and customization.

### 3. Student Dashboard (Prospective and Current Students):

- **Purpose:** Primary interface for students to access exams, study materials, and track academic progress.
- **Features:**
  - List of available entrance exams, study materials, and practice tests.
  - Navigation to practice exams, certification study materials, and exam results.
  - Progress tracker displaying the student's status in their pathway.
- **Layout:** Organized layout with intuitive navigation and prominent sections for upcoming deadlines and study material recommendations.
- **Purpose:** Primary interface for students to access exams, study materials, and track academic progress.

### 4. Graduated Student Interface:

- **Purpose:** Interface for graduated students to access study materials for ongoing certifications.
- **Features:**
  - Quick access to certification-related resources (e.g., CompTIA, AWS).
  - Study tools, such as practice tests and study games.
  - Certification progress overview and reminders for upcoming certification exams.
- **Layout:** Simplified interface prioritizing ease of navigation, with a sidebar for category-based study material browsing.

### 5. Chatbot Integration Interface:

- **Purpose:** Provide on-demand assistance via a chatbot powered by OpenAI 3.5.
- **Features:**
  - Chat window accessible from all pages, offering contextual help based on the user's current activity.
  - Links to documentation, FAQs, and key platform features.
  - Ability to escalate queries to administrators or support staff.

## 6. GUI Standards and Style Guides

### 1. Color Scheme:

- Uses a clean, modern palette aligned with the college's branding.
- Prioritizes readability and minimizes eye strain with light and dark modes for different user preferences.

### 2. Typography:

- Standardized fonts (e.g., Arial) for clear, easy-to-read text.
- Adequate spacing, font sizes, and contrast to meet accessibility standards.

### 3. Icons and Buttons:

- Consistent icons across the platform for common actions (e.g., edit, save, submit exam).
- Buttons are color-coded to indicate actions: green for confirmation, red for cancel, blue for navigation.

### 4. Common UI Elements:

- **Navigation Bar:**
  - Present on all key interfaces, providing quick links to the most commonly accessed areas such as exams, study materials, user profile, and help.
- **Help Button:**
  - Contextual help button available on every screen, offering guidance and explanations relevant to the current page or activity.
- **Error Messages and Notifications:**

- Consistent error and feedback messages throughout the platform.
- Inline form validation for user inputs, and modal dialogs for more critical issues (e.g., failed exam submission).

### 5. Keyboard Shortcuts:

- **Purpose:** Designed to improve usability, particularly for frequent users like administrators.
- **Examples:**
  - Save: Ctrl+S
  - Refresh F5
  - Navigate back Alt+Left Arrow

### 6. Cross-Platform Compatibility:

- **Purpose:** Ensure the platform is accessible and functional across desktops, laptops, tablets, and mobile devices.
- **Implementation:**
  - Responsive web design using HTML, CSS, and JavaScript frameworks.
  - Testing across multiple browsers and screen sizes.
- **Requirements:**
  - Consistent UI/UX experience regardless of device type or operating system.

## 4.1.3 Accessibility Guidelines

### 1. Compliance with WCAG 2.1:

- **Purpose:** Ensures keyboard navigation, screen reader compatibility, and sufficient color contrast for visually impaired users.

### 2. Accessible Forms and Buttons:

- **Purpose:** Clear labels for form fields, including error handling and inline validation. Large, touch-friendly buttons for mobile users.

### 3. Dynamic Content Accessibility:

- **Purpose:** ARIA roles and properties applied to dynamic elements (e.g., modals, dropdowns) for compatibility with assistive technologies.

## 4.1.4 Documentation

## 1. User Interface Specification Document:

- **Purpose:** A comprehensive guide detailing the design and implementation of the platform's UI.
- **Examples:**
  - **Wireframes and Mockups:** High-fidelity visuals of all key interfaces (Appendix E).
  - **CSS Style Definitions:** Color codes, font sizes, and spacing guidelines.
  - **Accessibility Details:** Specific measures to support WCAG compliance.

## 2. Interactive Prototypes:

- Developed to collect user feedback during early design phases.
- Shared with stakeholders for iterative refinement.

## 4.2 Software Interfaces

The WAV2T platform is engineered to interact with various software components to ensure efficient operation, data integrity, and seamless functionality. This section outlines the key software interfaces, database design, and integration points with external systems.

### 4.2.1. Database Design:

The WAV2T system utilizes SQLite as the primary database, providing lightweight, file-based storage for managing platform data. The database schema is designed to support relational data management with the following key tables:

- **Users Table:** Stores user details, including names, roles (e.g., prospective, current, graduated students, administrators), login credentials, and profile information.
- **Exams Table:** Contains information about entrance exams, including exam IDs, pathways, question sets, time limits, and statuses.
- **Study Materials Table:** Manages resources such as videos, practice exams, certification guides, and other materials categorized by pathway (SCA, CAD, Cyber).
- **Results Table:** Logs exam results, including scores, completion dates, feedback, and adaptive performance metrics.

- **Notifications Table:** Tracks system notifications for events such as exam results, study material updates, grading alerts for administrators, and reminders.

#### 4.2.2 Integration with External Systems:

- **Email Systems:** Email integration is not yet implemented. In the future, the platform will support outbound email notifications using SMTP or third-party services such as SendGrid or AWS Simple Email Service (SES) for alerts related to exam results, account actions, and study material updates.
- **Authentication Services:** The platform currently uses Django Allauth for secure user authentication, including login, registration, and role-based access. Advanced features like two-factor authentication (2FA) and integration with external identity providers (e.g., AWS Cognito, Auth0) are planned for future phases.

#### 4.2.3 OpenAI-Powered Chatbot Integration:

- The chatbot study buddy is powered by the OpenAI API and offers on-demand assistance for students by answering study-related questions, guiding users to resources, and helping them navigate the platform. Future enhancements may include role-specific chatbot functionality for administrators and deeper integration with documentation and FAQs.

#### 4.2.4 API and Data Exchange:

- **RESTful API:** The WAV2T platform is being developed with future REST API capabilities in mind. These APIs will enable secure access to user data, quiz results, and study materials. Standard CRUD operations (GET, POST, PUT, DELETE) will be supported once implemented.
- **Data Sharing and Formats:** Data is exchanged in JSON format to ensure compatibility with other systems and ease of integration for future expansions.
- **Webhooks:** Webhook support is not currently active but is planned for production deployment. This will allow real-time notifications to external systems when key events occur, such as exam submission, material updates, or account changes.

#### 4.2.5 Software Components and Libraries:

- **Django (Python Framework):** Serves as the primary backend framework, offering vast support for user authentication, URL routing, database interaction, and security out of the box.

- **Django ORM:** Handles object-relational mapping between Python-based models and the SQLite database. Simplifies CRUD operations and enforces consistency between application logic and data structure.
- **HTML/CSS/JavaScript:** Powers the frontend UI alongside Django's templating engine, providing responsive layout, dynamic interactions, and cross-browser compatibility.
- **Bootstrap:** A frontend library used to ensure responsive design, consistent styling, and accessibility across devices and screen sizes.
- **OpenAI API:** Powers the chatbot study buddy, providing context-aware, AI-generated responses to student questions using natural language processing.
- **Python Logging Library:** Handles application-level logging for debugging, auditing, and tracking user/system events.
- **Machine Learning Logic (Custom):** Implements logic for adaptive testing, adjusting quiz difficulty based on individual student performance tracked through the UserQuizProgress model.

#### 4.2.6 Communication Protocols:

- **HTTPS:** All communication between users and the platform is secured using HTTPS to ensure data confidentiality and integrity during transmission.
- **Internal Push Notifications:** All current notifications (e.g., exam grading alerts, study material updates) are sent directly through the platform UI. Email delivery via SMTP is planned for future implementation.

#### 4.2.7 Implementation Constraints:

- **Data Consistency:** Ensures consistent data across all components using Entity Framework Core's transaction management and SQLite's support for atomic operations.
- **Concurrency Management:** Designed to handle multiple simultaneous users accessing exams or study materials, employing efficient concurrency control mechanisms to prevent data conflicts or loss.
- **Local Hosting (PoC):** The platform currently runs on a local development server. Production deployment will require migration to scalable infrastructure like AWS or institutional hosting.

#### 4.2.8 Adaptive Testing System:



- **Purpose:** Integrates machine learning models to dynamically adjust question difficulty based on the user's performance.
- **Implementation:**
  - The adaptive algorithm is implemented as a backend service, interacting with the Exams and Results tables in the database.
  - Adaptive algorithm is implemented as a backend service, interacting with the Exams and Results tables in the database.
- **Integration:**
  - The adaptive system interfaces with the main application through the business logic layer and utilizes machine learning libraries.
  - Results of the adaptive model are stored in the database for review and analysis.

#### 4.2.9 Hosting Environment:

- **Local Hosting:**
  - WAV2T is currently hosted locally using Django's built-in development server for testing and proof of concept.
- **Scalability:**
  - The application's modular architecture supports future scaling via containerization (e.g., Docker) or cloud orchestration (e.g., AWS EC2, Kubernetes).
- **Constraints:**
  - Budget limitations require local hosting until a server can be built for the project.

### 4.3 Performance Requirements

The WAV2T platform's performance requirements ensure reliable operation, scalability, and responsiveness as the user base grows. The platform is hosted on scalable AWS cloud infrastructure to meet these performance standards efficiently.

#### 1. Response Time:

- The system should respond to user inputs and load pages within 500 milliseconds under normal operating conditions.

## 2. Concurrency Handling:

- The system can support at least 500 concurrent users taking exams, viewing study materials, or accessing their profiles without significant performance degradation.

## 3. Real-Time Data Processing:

- The platform should process real-time actions, such as exam submissions and notifications, within 2 seconds from the user action.

## 4. Bulk Data Operations:

- Batch processes, such as importing exam results or generating reports, must be completed within 5 minutes under peak load conditions.

## 5. Database Transactions:

- Database read and write operations must be completed within 100 milliseconds under typical user loads.

## 6. System Scalability:

- The system is designed to scale horizontally, allowing additional server resources to be added as the number of users increases without major changes to the architecture.

## 4.4 Safety Requirements

The WAV2T platform prioritizes reliability, error handling, and user safety, ensuring that the system operates securely and complies with safety standards.

### 1. System Redundancy and Failover:

- **Redundant Infrastructure:** The current proof of concept is hosted locally without redundancy. When deployed in production, the system will adopt cloud-based redundancy (e.g., institutional servers or AWS) to ensure service continuity.
- **Failover Mechanisms (Planned):** Backup and rerouting strategies will be implemented upon cloud deployment to prevent downtime in the event of system failure.

### 2. Error Handling:

- **Advanced Error Logging:** Django's logging framework is configured to capture system-level and application-level errors for debugging and audit trails.
- **User Feedback:** Displays user-friendly error messages to inform users without exposing technical details or compromising security.
- **Data Integrity:** The system uses Django's transaction management to ensure data rollback in the event of an error during save operations, maintaining data consistency.

### 3. Safety Monitoring:

- **Continuous Manual Monitoring:** During PoC, error logs and usage data are monitored locally by developers.
- **Automated Monitoring (Planned):** When migrated to a cloud provider or local server, tools like Prometheus, Sentry, or AWS CloudWatch may be integrated for real-time alerts and anomaly detection.

### 4. Safety Training for Users:

- **Training Resources:** Provides comprehensive safety resources, including written guides, interactive tutorials, and video instructions on secure usage practices.
- **Focus Areas:** Topics include:
  - Best practices for password management.
  - Recognizing and reporting phishing attempts.
  - Responding to alerts for unusual activity, such as multiple failed login attempts.

### 5. Data Backup and Recovery:

- **Backup Policies:** Backups of the SQLite database are created manually during local development.
- **Future Backup Strategy:** Upon deployment, automated backups and recovery protocols will be implemented. Options may include scheduled snapshots and version-controlled recovery via institutional servers or cloud storage.
- **Recovery Time Objective (RTO):** The goal is to restore functionality within 15–30 minutes of critical failure once cloud hosting is in place.

## 4.5 Security Requirements

The security requirements for the WAV2T platform are designed to uphold the principles of confidentiality, integrity, and availability of sensitive data. These measures protect against unauthorized access, ensure secure data handling, and maintain compliance with privacy laws and industry standards.

### 1. Data Encryption

- **Encryption in Transit:** All data transmitted between clients and servers must be encrypted using Transport Layer Security (TLS 1.3). This includes login credentials, exam submissions, and communications with external systems like APIs and email servers.
- **Encryption at Rest:** Sensitive fields (e.g., passwords) are hashed using Django's secure hashing (PBKDF2 by default). Additional encryption for stored data (AES-256) will be added when hosting is migrated to a production environment.

### 2. Authentication and Authorization

- **Multi-Factor Authentication (MFA):** MFA is not yet implemented but is planned for future versions. Django Allauth allows future integration with external identity providers or MFA tools.
- **Role-Based Access Control (RBAC):** Enforced using Django's built-in permissions and groups, ensuring only authorized users can access certain views or actions.
- **Password Policies:** Strong password enforcement is implemented, including minimum length, complexity, and built-in Django validation mechanisms.

### 3. Security Audits and Penetration Testing

- **Annual Security Audits:** WAV2T will undergo regular annual security audits conducted by third-party experts to evaluate system adherence to best security practices.
- **Bi-Annual Penetration Testing:** Penetration testing will be conducted twice a year to identify and address vulnerabilities before they can be exploited.

### 4. Data Privacy Compliance

- **FERPA Compliance:** The platform will comply with U.S. privacy laws, including the Family Educational Rights and Privacy Act (FERPA), and other relevant

federal and state regulations. Policies will be updated regularly to ensure ongoing compliance.

- **Data Minimization:** Collect only the necessary data required for platform functionality and ensure regular purging of outdated or unnecessary records.

## 5. User Access Control

- **RBAC Management:** User roles and permissions are managed via Django's admin panel, following the principle of least privilege. Admins have the authority to assign or adjust roles.

## 6. Incident Response Plan

- A detailed Incident Response Plan (outlined in Appendix D) includes procedures for identifying, containing, investigating, and resolving security incidents.
- Key steps include notifying affected users and stakeholders, securing compromised systems, and implementing preventive measures to avoid recurrence.

## 7. Continuous Monitoring and Threat Detection

- **Real-Time Monitoring:** While the PoC does not include real-time monitoring, cloud-hosted deployments will include logging tools (e.g., Sentry, AWS CloudWatch, or ELK Stack).
- **Automated Threat Detection:** Brute-force protection and anomaly detection (e.g., using Django-Axes or security plugins) will be deployed to detect unauthorized access attempts.

## 8. Secure Development Practices

- **Code Security:** Implement secure coding practices, such as input validation, output encoding, and avoiding hardcoded secrets.
- **Dependency Management:** Regularly update third-party libraries and frameworks to patch known vulnerabilities.

## 4.6 Software Quality Attributes

The WAV2T platform is designed to meet high software quality standards, ensuring reliability, usability, and flexibility to meet the diverse needs of students and administrators.

### 1. Reliability

- The system is designed to achieve 99.99% uptime, excluding scheduled maintenance, ensuring availability for students and administrators at all times.

## 2. Usability

- At least 90% of new users (students and administrators) should be able to complete basic tasks, such as taking exams and accessing study materials, without assistance.

## 3. Maintainability

- The platform's design prioritizes maintainability, with all system issues being addressed within 48 hours to minimize disruption to users.

## 4. Performance

- The system must maintain a response time of under 2 seconds for user interactions (e.g., loading study materials, submitting exams) under typical load conditions.

## 5. Scalability

- The current version is locally hosted. However, the backend is architected to support **future horizontal scaling** using containerization (e.g., Docker) or migration to cloud environments like AWS. Django's built-in scalability and caching strategies (e.g., per-view or database-level caching) will support increased traffic with minimal degradation.

## 6. Portability

- The system will operate on major browsers across desktops, tablets, and mobile devices, ensuring cross-platform compatibility for Windows, macOS, Linux, Android, and iOS users. Responsive web design ensures consistent functionality and usability across various screen sizes and resolutions.

## 7. Testability

- The Django testing framework is being integrated to support automated unit and integration testing. While CI/CD pipelines are not active in the proof of concept phase, the system is structured to adopt them in production to reduce deployment risk and ensure feature stability.

## 8. Security

- Security is embedded throughout the system via Django’s built-in protections (CSRF, authentication, session management). The platform is prepared for regular vulnerability assessments and will implement automated security scans in production to protect user data and platform integrity.

#### 4.7 Standards (IEEE/ACM/ADA)

The WAV2T platform aligns with industry standards from the Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM), and Americans with Disabilities Act (ADA) to ensure compliance with ethical, engineering, and accessibility requirements.

##### 1. IEEE Standards Implementation:

- **Software Development Protocols:** The development process adheres to IEEE software engineering standards, ensuring that documentation, testing, and quality assurance protocols are followed throughout the project lifecycle.
- **Ethical Guidelines Compliance:** All operational and development practices align with IEEE’s ethical standards, promoting integrity, transparency, and respect for user privacy.

##### 2. ACM Ethical Compliance:

- **Code of Ethics Integration:** The ACM Code of Ethics is integrated into the project’s culture and decision-making processes, ensuring responsible practices and user-centric design.
- **Best Practices Implementation:** The software development lifecycle incorporates ACM-recommended best practices to enhance professionalism, security, and ethical conduct.

##### 3. ADA Compliance for Accessibility:

- **Accessible Design Principles:** From the initial design phase, the WAV2T platform follows ADA guidelines to ensure it is navigable and usable by people with various disabilities, including compatibility with screen readers and keyboard navigation.
- **Continuous Accessibility Evaluation:** Regular audits and specialized tools are used to monitor and improve the platform’s accessibility, ensuring that all users can efficiently access the system’s features.

## Appendix A: Glossary

This glossary provides definitions for terms and acronyms used throughout the Software Requirements Specification (SRS) for the WAV2T platform.

1. **API (Application Programming Interface):** A set of protocols and tools for building software applications that specify how software components should interact.
2. **CCPA (California Consumer Privacy Act):** A state statute intended to enhance privacy rights and consumer protection for residents of California, USA.
3. **FERPA (Family Educational Rights and Privacy Act):** A U.S. federal law that protects the privacy of student education records.
4. **GUI (Graphical User Interface):** A form of user interface that allows users to interact with electronic devices through graphical icons and visual indicators.
5. **HTTPS (Hypertext Transfer Protocol Secure):** An extension of HTTP used for secure communication over a computer network, widely used to secure websites.
6. **MFA (Multi-Factor Authentication):** A security mechanism that grants access only after a user provides two or more pieces of evidence (e.g., password, phone verification) for authentication.
7. **OAuth2:** An authorization framework that enables third-party applications to grant limited access to a user's resources without exposing login credentials.
8. **RBAC (Role-Based Access Control):** A method of restricting system access to authorized users based on their roles within the organization (e.g., administrator, student).
9. **RESTful API:** An API that adheres to REST (Representational State Transfer) principles, using HTTP requests to perform operations such as GET, PUT, POST, and DELETE.
10. **SIS (Student Information System):** A software application for educational institutions to manage student data, including enrollment, grading, and attendance.
11. **SMTP (Simple Mail Transfer Protocol):** An Internet standard for email transmission across IP networks.
12. **SQL (Structured Query Language):** A programming language designed for managing data in a relational database system.



13. **TLS (Transport Layer Security):** A cryptographic protocol designed to provide communications security over a computer network, commonly used in web browsers to ensure data privacy.
14. **WCAG (Web Content Accessibility Guidelines):** A set of guidelines aimed at improving the accessibility of web content for people with disabilities, published by the World Wide Web Consortium (W3C).
15. **Uptime:** The amount of time a system is available and operational, usually expressed as a percentage of total time.
16. **WAV2T:** The platform designed to provide students in the college program with resources, including entrance exams, study materials, and notifications.

## Appendix B: Issues List

### 1. User Manual

- **Issue:** No user manual has been developed yet for the WAV2T platform.
- **Action Required:** Create a detailed user manual outlining system functionality, usage instructions, and troubleshooting steps for prospective, current, and graduated students, as well as administrators.
- **Responsible Party:** Documentation Team
- **Deadline:** TBD

### 2. Training Materials

- **Issue:** Training materials, such as videos and quick-start guides, have not yet been created.
- **Action Required:** Develop comprehensive training materials, including video tutorials, quick-start guides, and detailed training documents covering key system features like exam management and study materials access.
- **Responsible Party:** Training and Development Team
- **Deadline:** TBD

### 3. System Development Completion

- **Issue:** The platform is still under development, with core features such as exam functionality, study materials, and notification systems not yet fully completed.
- **Action Required:** Finalize development of all planned features and modules as outlined in the technical specifications, ensuring full functionality for the WAV2T platform.
- **Responsible Party:** Development Team
- **Deadline:** TBD

### 4. Security Measures

- **Issue:** Security features, including data encryption, multi-factor authentication, and compliance with privacy regulations, are not yet fully implemented.

- **Action Required:** Implement all required security measures, such as role-based access control (RBAC), encryption, and compliance with FERPA and other data protection regulations.
- **Responsible Party:** Security Team
- **Deadline:** TBD

## 5. System Testing and Validation

- **Current Status:** Development is still in progress. The backend and frontend components are being developed and tested independently in a localized environment. The system has not yet undergone full end-to-end or formal testing.
- **Local Testing Capabilities:** Current efforts focus on backend unit testing (e.g., model logic, data validation, adaptive quiz behavior) and frontend rendering in isolation. Integration testing will begin once the UI and backend are connected.
- **Action Required:** Conduct phased testing including:
  1. **Unit Testing:** Validate individual components such as models, views, and forms.
  2. **Integration Testing:** Begin once backend and frontend are linked.
  3. **System Testing:** Verify full workflows across the platform.
  4. **User Acceptance Testing (UAT):** Collect user feedback from stakeholders and students once the functional system is available.
- **Responsible Party:** Quality Assurance with support of development team during initial testing.
- **Deadline:** TBD

## 6. Compliance and Regulatory Approval

- **Issue:** The system's compliance with legal and regulatory requirements, such as FERPA and ADA, has not yet been confirmed.
- **Action Required:** Review and ensure compliance with all relevant laws and regulations related to student data privacy, accessibility, and security standards.

- **Responsible Party:** Compliance Officer
- **Deadline:** TBD

## 7. Project Timeline and Milestones

- **Issue:** A detailed project timeline with milestones and deadlines for system completion has not yet been fully established.
- **Action Required:** Develop a comprehensive project timeline, outlining milestones, deliverables, and deadlines to track progress and ensure timely completion of the WAV2T platform.
- **Responsible Party:** Project Management Team
- **Deadline:** TBD

## **Appendix C: Incident Response Plan (IRP)**

### **WAV2T Platform**

**Purpose:** The purpose of this Incident Response Plan (IRP) is to provide a predefined set of procedures and guidelines to effectively identify, respond to, and recover from security incidents affecting the WAV2T platform.

**Scope:** This plan applies to all security incidents that impact the integrity, availability, or confidentiality of the platform and the data it handles, including student records, exam results, and study materials.

**Objectives:**

- To contain and mitigate incidents to minimize damage and impact on operations quickly and effectively.
- To ensure timely communication with all stakeholders during an incident.
- To document and learn from each incident to improve future response efforts and system resilience.

### **Consolidated Roles for a Smaller Team:**

Due to the reduced number of personnel, responsibilities will be combined to ensure all aspects of the incident response are covered.

- **Incident Manager & Security Lead:**
  - Leads the response effort and handles both the management of the incident and technical analysis.
  - Responsible for decision-making, containment, eradication, and overseeing recovery.
- **Communications & Legal Lead:**
  - Manages all internal and external communications during and after the incident.
  - Ensures legal compliance with privacy regulations such as FERPA.
- **IT Support & Recovery Specialist:**
  - Handles system restoration, backup management, and ensuring normal operations are resumed post-incident.

- Provides technical support to affected users.

**Incident Classification:**

Incidents are classified into the following categories to ensure the response is scaled appropriately:

- **Low:** Minimal impact or risk, such as minor system errors.
- **Medium:** Moderate risk, requiring a timely response to prevent escalation.
- **High:** Major incident with significant impact on operations or data, requiring immediate response.

**Incident Response Phases:****1. Preparation:**

- Regular training and incident response drills for the small team.
- Ensure incident response tools and the IRP document are kept up to date.
- Ensure all team members understand their combined roles.

**2. Identification:**

- Detect potential security incidents through monitoring tools or reports.
- Assess and classify the severity of the incident.
- Notify the Incident Manager and document the initial findings.

**3. Containment:**

- Implement short-term containment measures to limit the spread of the incident.
- Plan for long-term containment while maintaining system availability for unaffected users.

**4. Eradication:**

- Identify and remove the root cause of the incident.
- Ensure any related vulnerabilities are identified and addressed.

**5. Recovery:**

- Restore affected systems to normal operations.
- Monitor systems for signs of recurrence or vulnerabilities.

- Validate system functionality and security.

#### **6. Post-Incident Analysis:**

- Conduct a detailed review of the incident, response effectiveness, and involved systems.
- Update the IRP based on lessons learned.
- Prepare a report summarizing the incident, response actions, and recommendations for future prevention.

#### **Communication Protocol:**

- **Immediate internal notification:** Upon identification of a medium or high-severity incident, all team members will be notified immediately.
- **Stakeholder updates:** The Communications Lead will provide regular updates to internal stakeholders, including students and administrators. External communications with regulatory bodies will also be managed if required.
- **Post-incident report:** After resolving the incident, a detailed report will be prepared by the Communications Lead and shared with management and other relevant stakeholders.

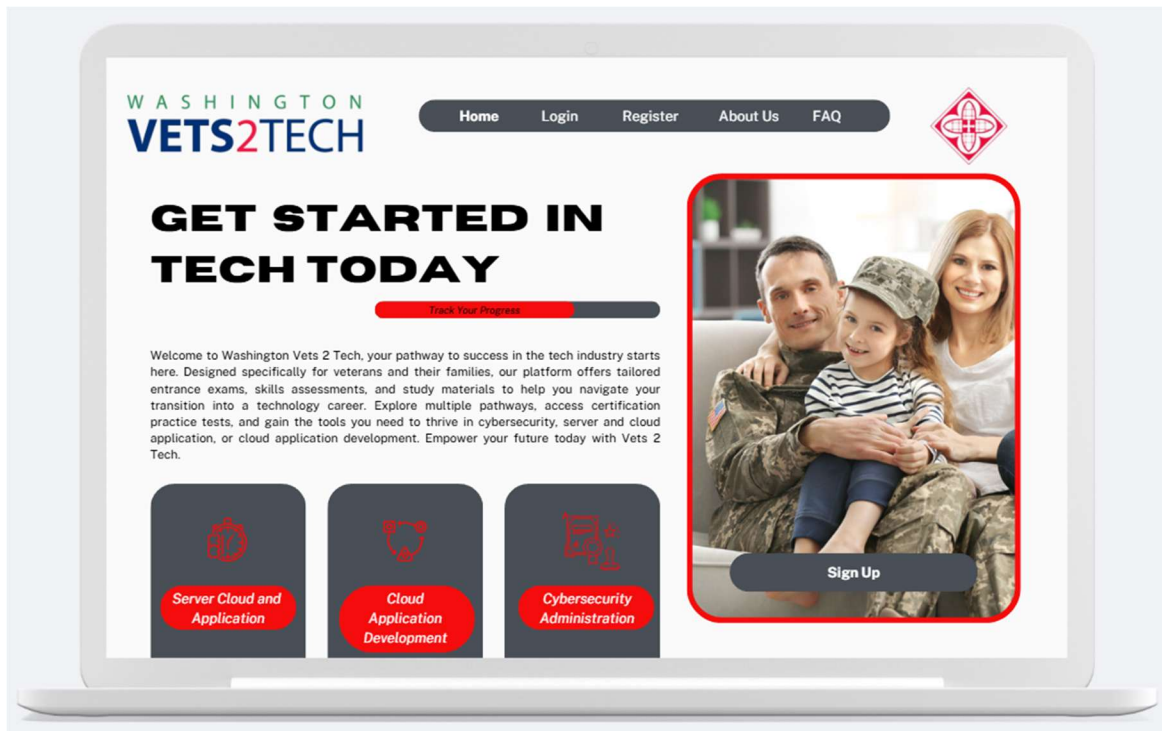
#### **Documentation:**

- All actions taken during an incident response must be documented by the Security Lead and reviewed by the Incident Manager.
- The documentation should include the description of the incident, how it was discovered, the response actions taken, individuals involved, and the outcomes.

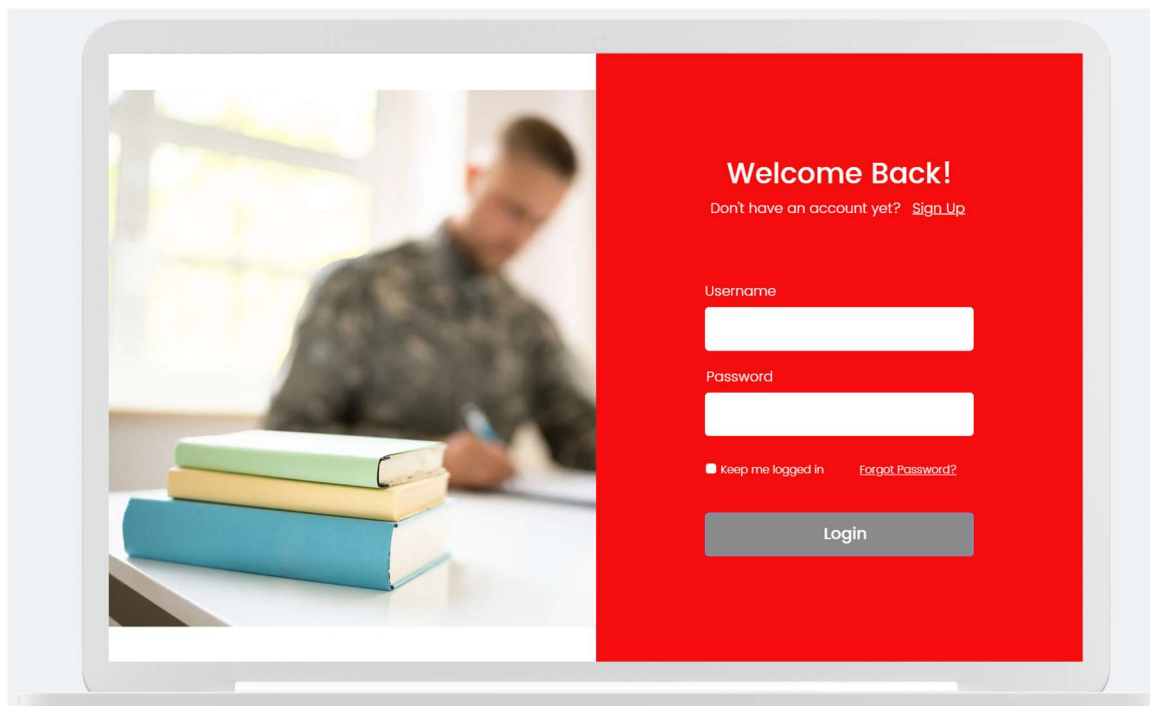
This documentation will be securely stored and used for post-incident analysis and compliance purposes.

## Appendix D: Wireframes

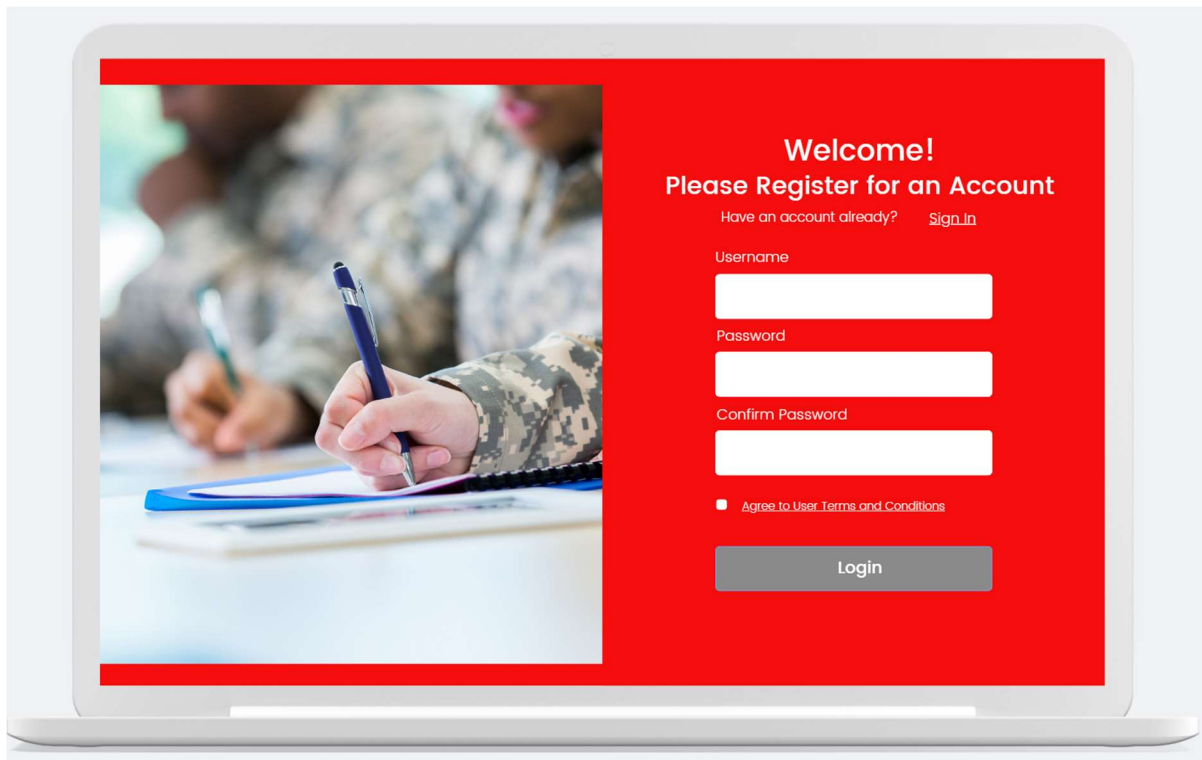
### Home Page:



### Login:





**Register:**

**Welcome!**  
**Please Register for an Account**

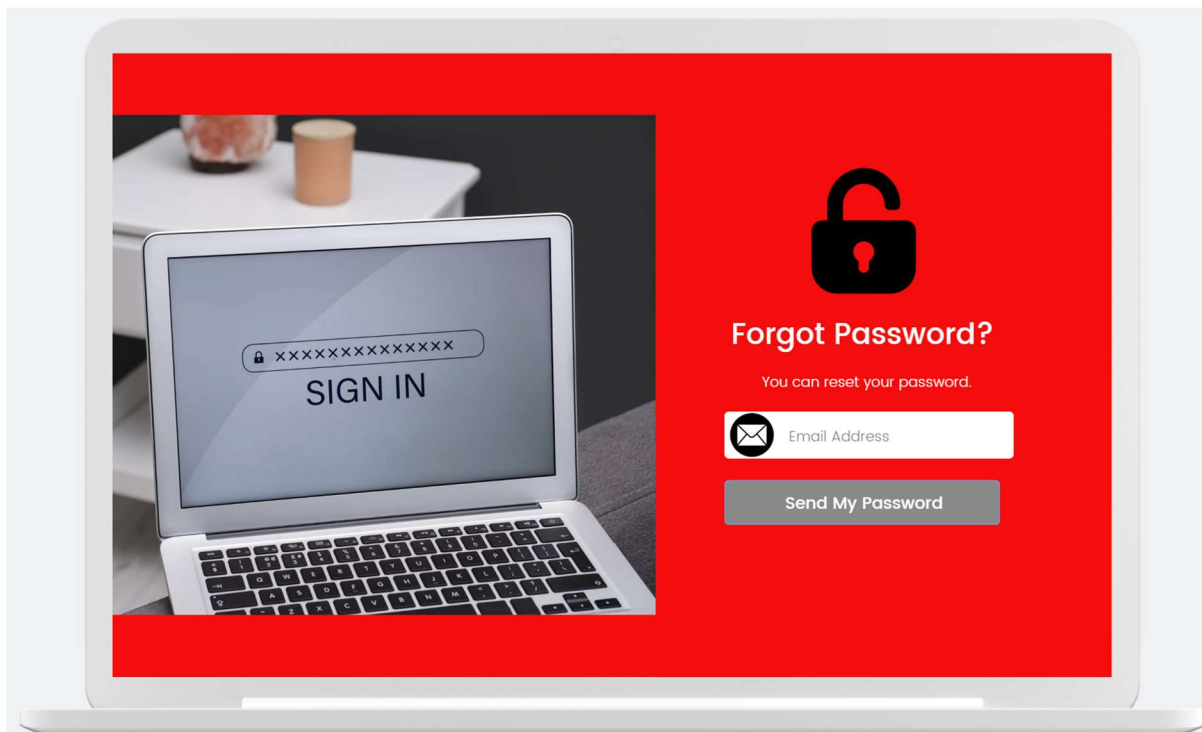
Have an account already? [Sign In](#)

Username

Password

Confirm Password

☐ [Agree to User Terms and Conditions](#)

**Forgot Password:**

**Forgot Password?**

You can reset your password.