**Software Requirements Specification**

**for**

Align

**Version 3.0 approved**

**Prepared by: Liv Anderson**

**Zo Carroll**

**Siaka Diarra**

**Meghann Manson**

**Kursten Massey**

**February 24th, 2025**

**Table of Contents**

**Table of Contents ii**

**Revision History ii**

**1. Introduction 1**

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 2

**2. Overall Description 3**

2.1 Product Perspective 3

2.2 Product Functions 3

2.3 User Classes and Characteristics 3

2.4 Operating Environment 4

2.5 Design and Implementation Constraints 4

2.6 User Documentation 4

2.7 Assumptions and Dependencies 4

**3. External Interface Requirements 5**

3.1 User Interfaces 5-8

3.2 Hardware Interfaces 8

3.3 Software Interfaces 8

**4. System Features 10**

4.1 Syllabus Date Extraction 10

4.2 Calendar Integration 11

**5. Other Nonfunctional Requirements 13**

5.1 Performance Requirements 13

5.2 Safety Requirements 13

5.3 Security Requirements 13

5.4 Software Quality Attributes 13

5.5 Business Rules 13

**Appendix A: Glossary 14**

**Revision History**

| **Name** | **Date** | **Reason For Changes** | **Version** |
| --- | --- | --- | --- |
| Initial Version | 2/28/2025 | Initial | 0.1 |
| Requirements | 3/4/2025 | Added functional and nonfunctional requirements | 0.2 |
| Fixes | 3/15/2025 | Changes reflecting feedback from Deliverable 1 | 1.0 |
| Section 2 | 3/24/2025 | Added Section 2: Overall Description | 1.1 |
| Section 3 | 3/26/2025 | Finished Section 3: External Interface Requirements | 1.2 |
| Appendix | 3/26/2025 | Added Glossary | 1.3 |
| ToC, Section 3 | 4/19/2025 | Update ToC, updated UI images | 2.0 |
| NF requirements | 4/23/25 | Gave unique ID # for NF requirements | 2.1 |
| Section 3 | 4/27/25 | Added more UI screenshots | 2.2 |
| Section 4 | 5/9/2025 | Updated functional requirements | 3.0 |

# **Introduction**

## **Purpose**

This SRS document specifies the requirements for Align, a web application designed to help students manage their academic schedules by automatically extracting important dates and deadlines from course syllabi. This document covers the complete system including the syllabus upload functionality, calendar integration, and dashboard features.

## Document Conventions

This document follows these conventions:

* Priority levels are (High, Medium, Low) are explicitly stated for each requirement

## **Intended Audience and Reading Suggestions**

This document is intended for:

* **Development Team**: To understand what to build and implementation details
* **Project Managers**: To plan development phases and allocate resources
* **QA Testers**: To develop test plans and validation criteria
* **Stakeholders**: To confirm the application meets the intended needs

Reading suggestions by role:

* **Developers**: Read the entire document with special attention to sections 3 (Functional Requirements) and 4 (System Features)
* **Project Managers**: Focus on sections 1 (Introduction), 2 (Overall Description), and 6 (Constraints)
* **Testers**: Focus on sections 3, 4, and 5 (Non-functional Requirements)
* **Stakeholders**: Read sections 1 and 2 for a high-level understanding of the system

## **Product Scope**

Align is a web-based application that aims to simplify academic planning for students. The system allows users to upload course syllabi, from which it automatically extracts assignment due dates, exam schedules, and other important deadlines. These dates are then organized in a calendar view and dashboard that helps students visualize their workload across courses.

Key goals include:

* Reducing the time students spend manually organizing their academic schedules
* Minimizing the risk of missed deadlines through automatic extraction and notification
* Providing intelligent scheduling suggestions based on workload assessment
* Facilitating calendar synchronization with popular platforms like Google Calendar and Outlook
* Helping students align their academic tasks with their long-term educational goals

## **References**

1. React Documentation, Meta Inc., Version 19.0.0, March 2025,<https://react.dev/>
2. React Bootstrap Documentation, React Bootstrap Team, Version 2.10.9, March 2025,<https://react-bootstrap.github.io/>
3. FullCalendar Documentation, FullCalendar LLC, Version 6.1.15, March 2025,<https://fullcalendar.io/docs>
4. PDF-Parse Documentation, Version 1.1.1, March 2025,<https://www.npmjs.com/package/pdf-parse>
5. Align Project Repository README, Team Atomic Thunder, Version 0.1.0, March 2025,<https://github.com/team-atomic-thunder/align-app>

# **Overall Description**

## **Product Perspective**

Align is a new, self-contained web application designed to address the specific needs of students in managing their academic schedules. While there are existing calendar and task management tools on the market, Align differentiates itself through its specialized syllabus processing capabilities and academic-focused planning tools.

The system operates as a web application with the following key components:

* **Frontend Web Application**: A React-based user interface accessible through modern web browsers.
* **Syllabus Processing Engine**: A component that extracts dates and deadlines from uploaded PDF syllabi.
* **Calendar System**: A component that organizes and displays extracted dates.
* **External Calendar Integration**: Interfaces with third-party calendar services.

## **Product Functions**

The major functions of the Align system include:

* Syllabus Upload and Processing
* Calendar Management
* Dashboard Overview
* Smart Scheduling
* External Calendar Integration

## **User Classes and Characteristics**

Primary User Class - Students

* *Characteristics*: Typically have multiple courses with different instructors, varying levels of organizational skills, moderate to high technical proficiency
* *Usage Pattern*: Regular use at the beginning of each term and weekly check-ins throughout
* *Priority*: High

Secondary User Class - Faculty

* *Characteristics*: Need to track deadlines for courses they teach, high organizational skills, varying technical proficiency
* *Usage Pattern*: Primarily at the beginning of terms to set up courses
* *Priority*: Low

## **Operating Environment**

Align is designed to work on phone and web browsers. It will be functional on any operating system that also supports up to date versions of Chrome, Firefox, and other browsers that support javascript.

## **Design and Implementation Constraints**

Align has security constraints in regards to a user’s data. The user’s data must be protected following standard web security practices. Additionally, uploaded syllabi should not be accessible to users other than the uploader.

## **User Documentation**

To support users, Align will provide a FAQ page accessible from the application that will answer questions such as how to log on, upload a PDF and other common questions.

## **Assumptions and Dependencies**

External Libraries and Services:

* React and associated component libraries (React Bootstrap, FullCalendar)
* PDF processing capabilities (pdf-parse library)
* External calendar APIs (Google Calendar, Apple Calendar, Microsoft Outlook)

Development Tools:

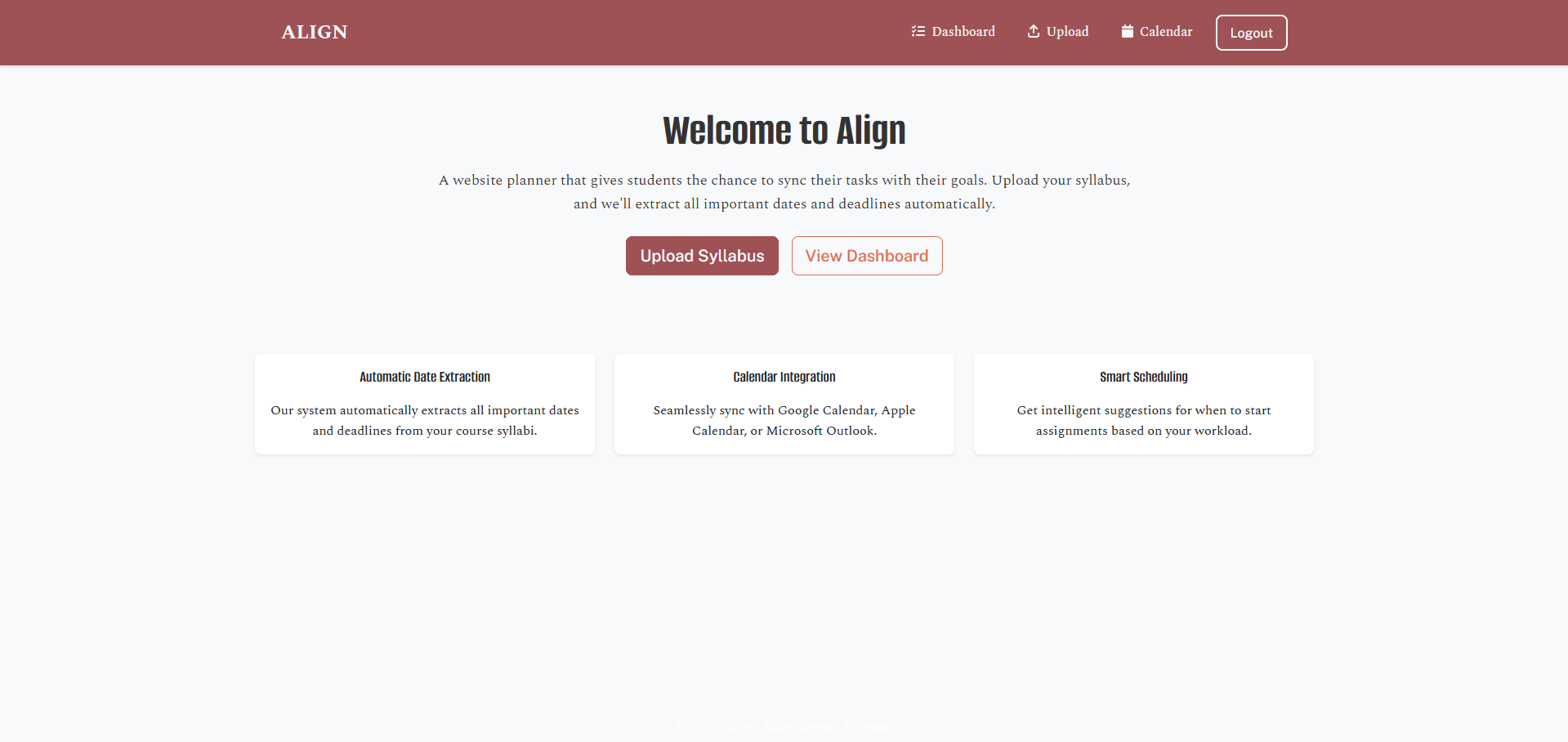
* Node.js environment for development and testing
* Jest testing framework for quality assurance

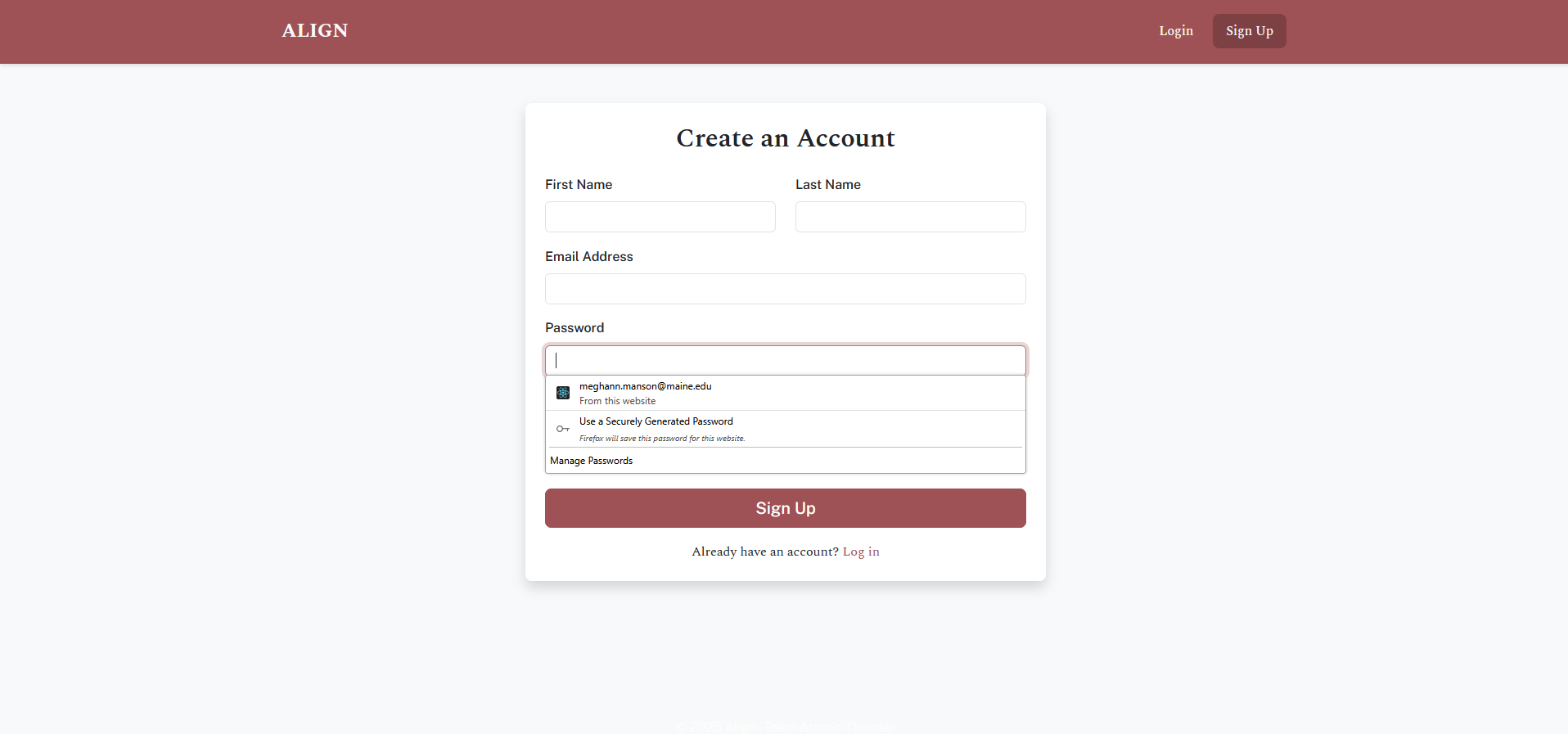
Infrastructure:

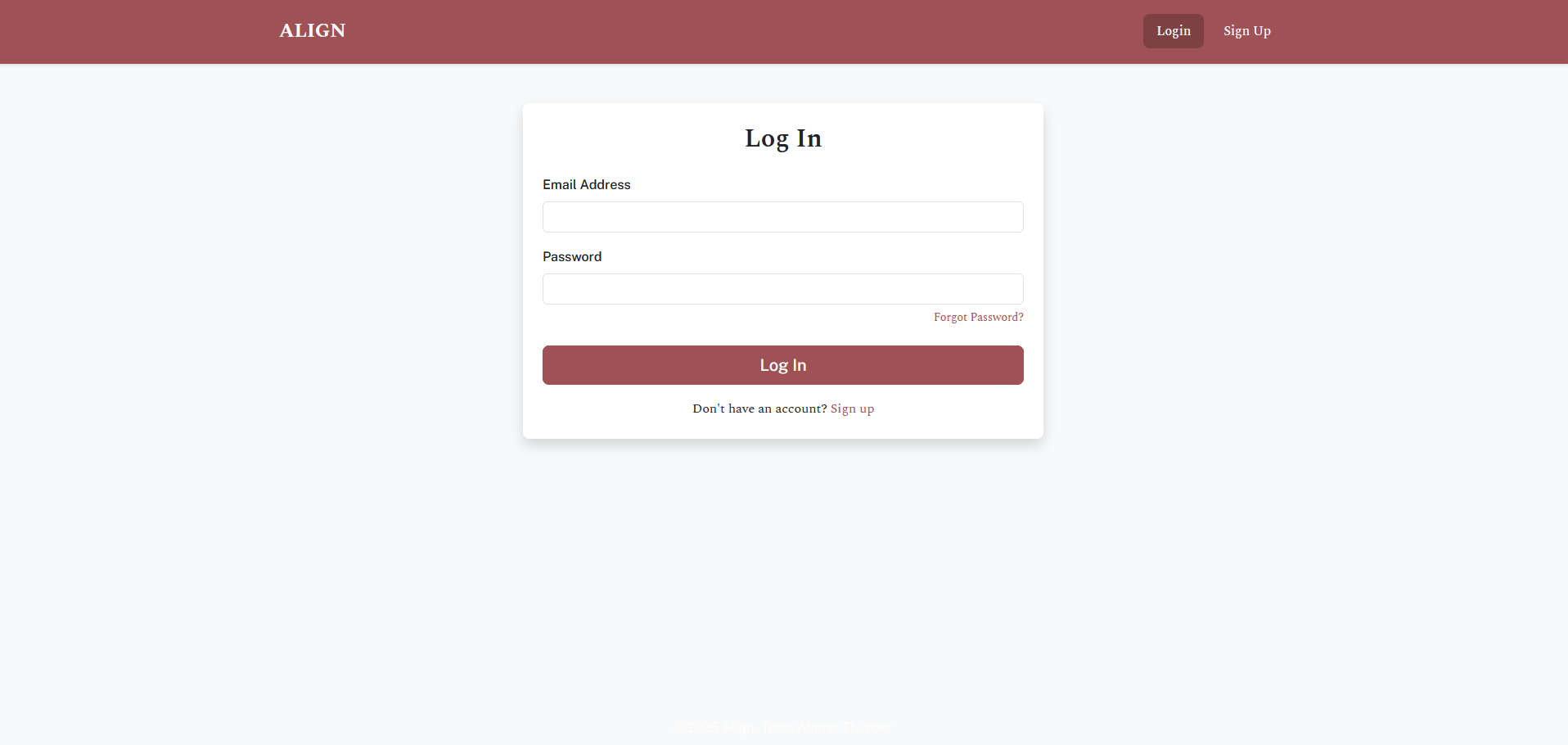
* Web hosting environment for the application
* Database services for user and calendar data storage
* Authentication services for user identity verification

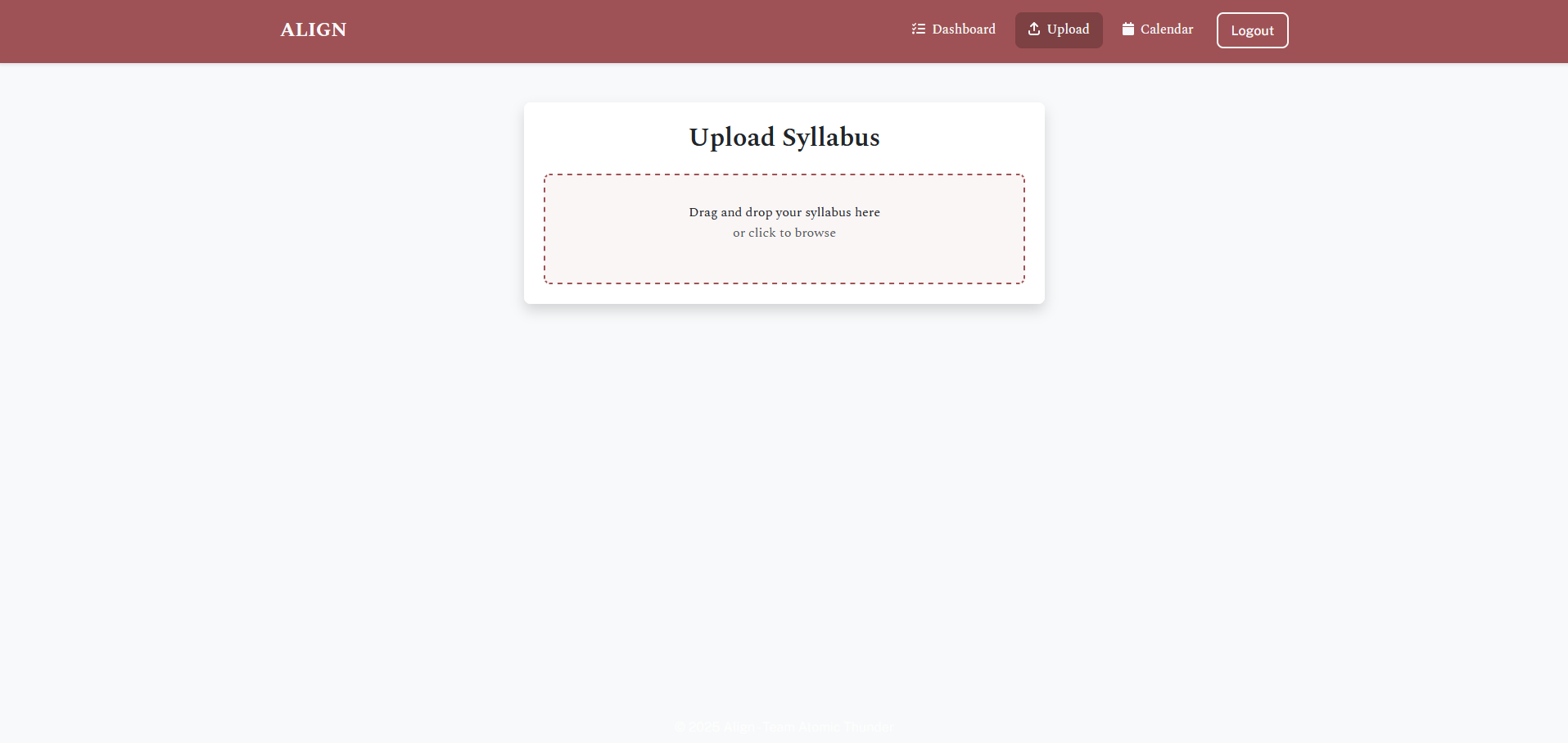
# **External Interface Requirements**

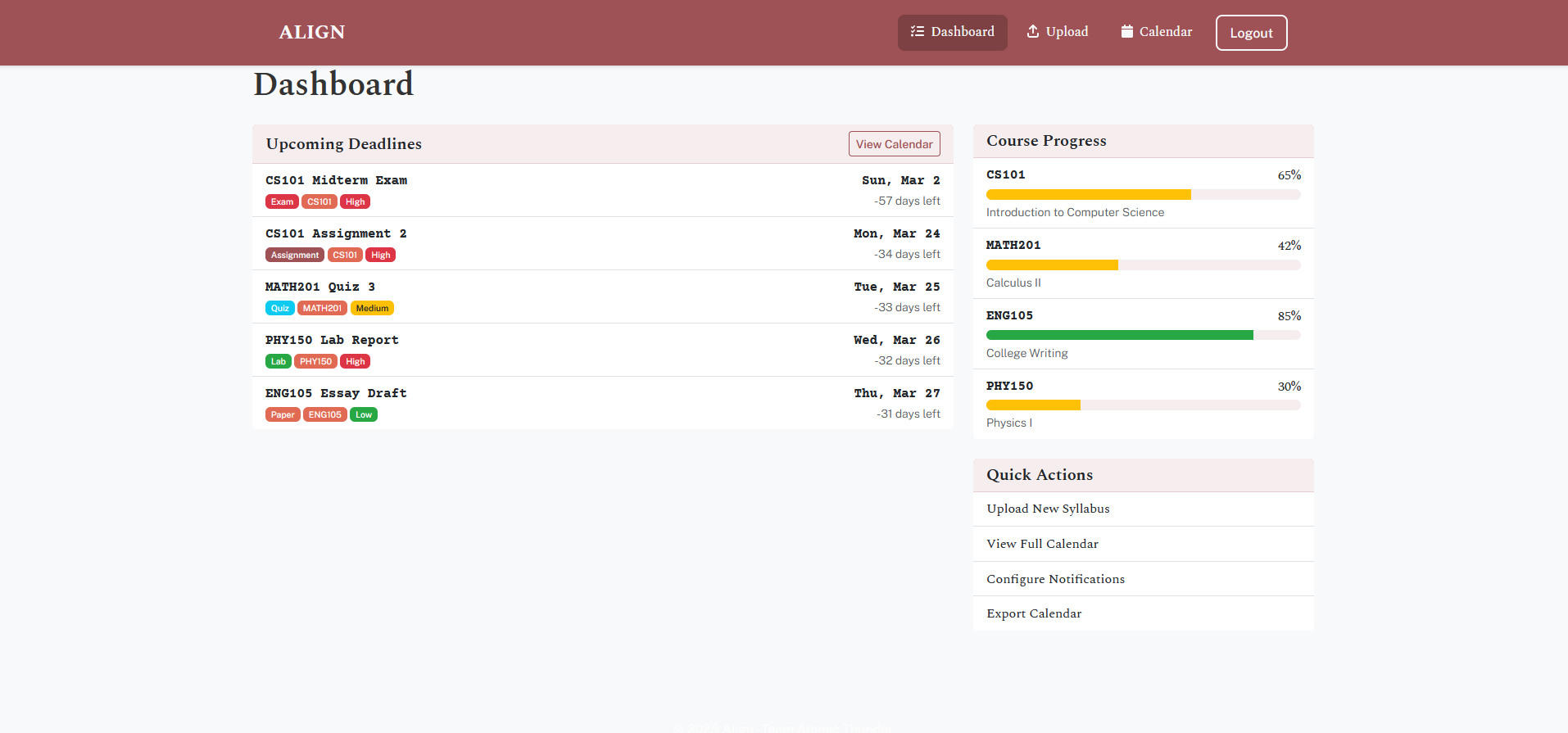
## **User Interfaces**

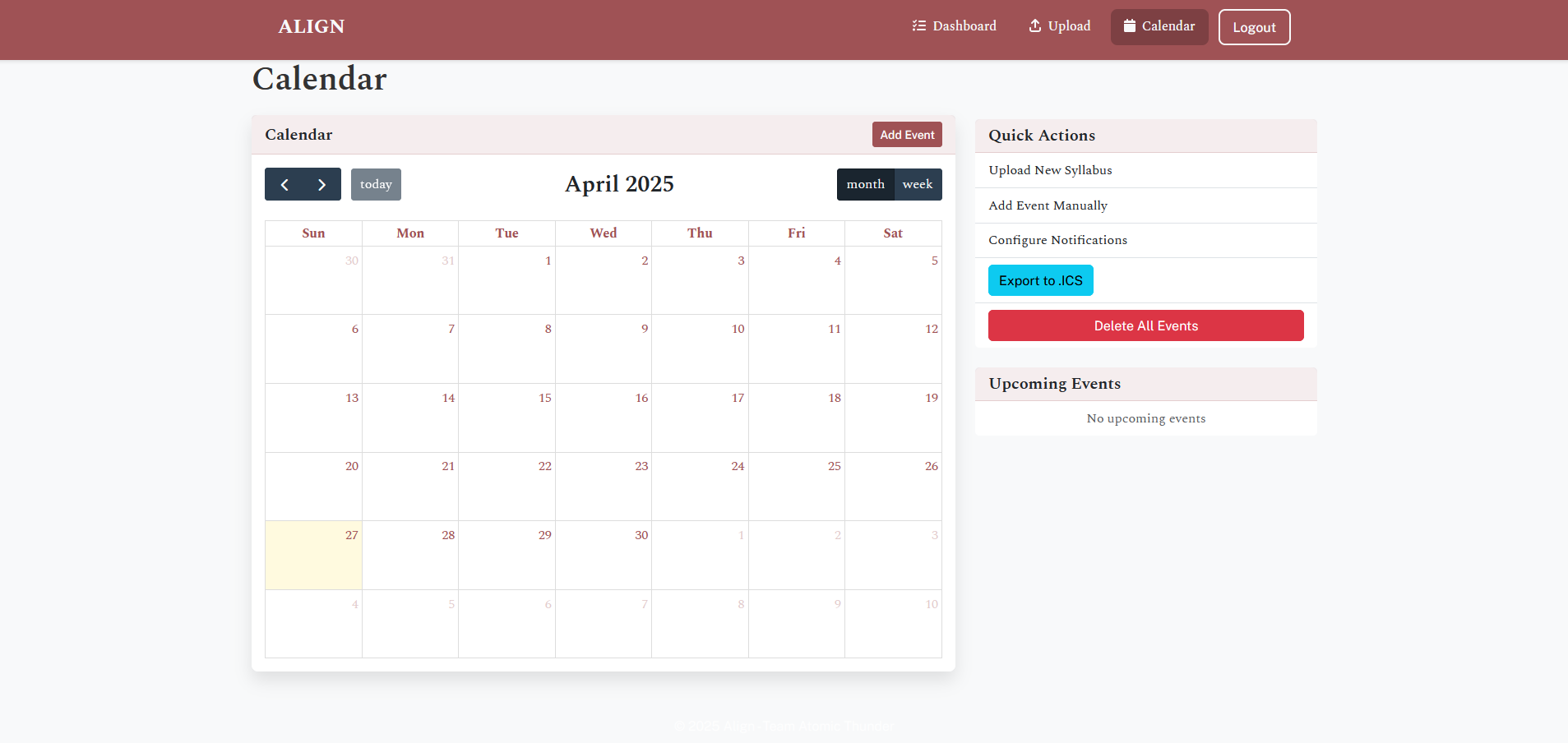
**

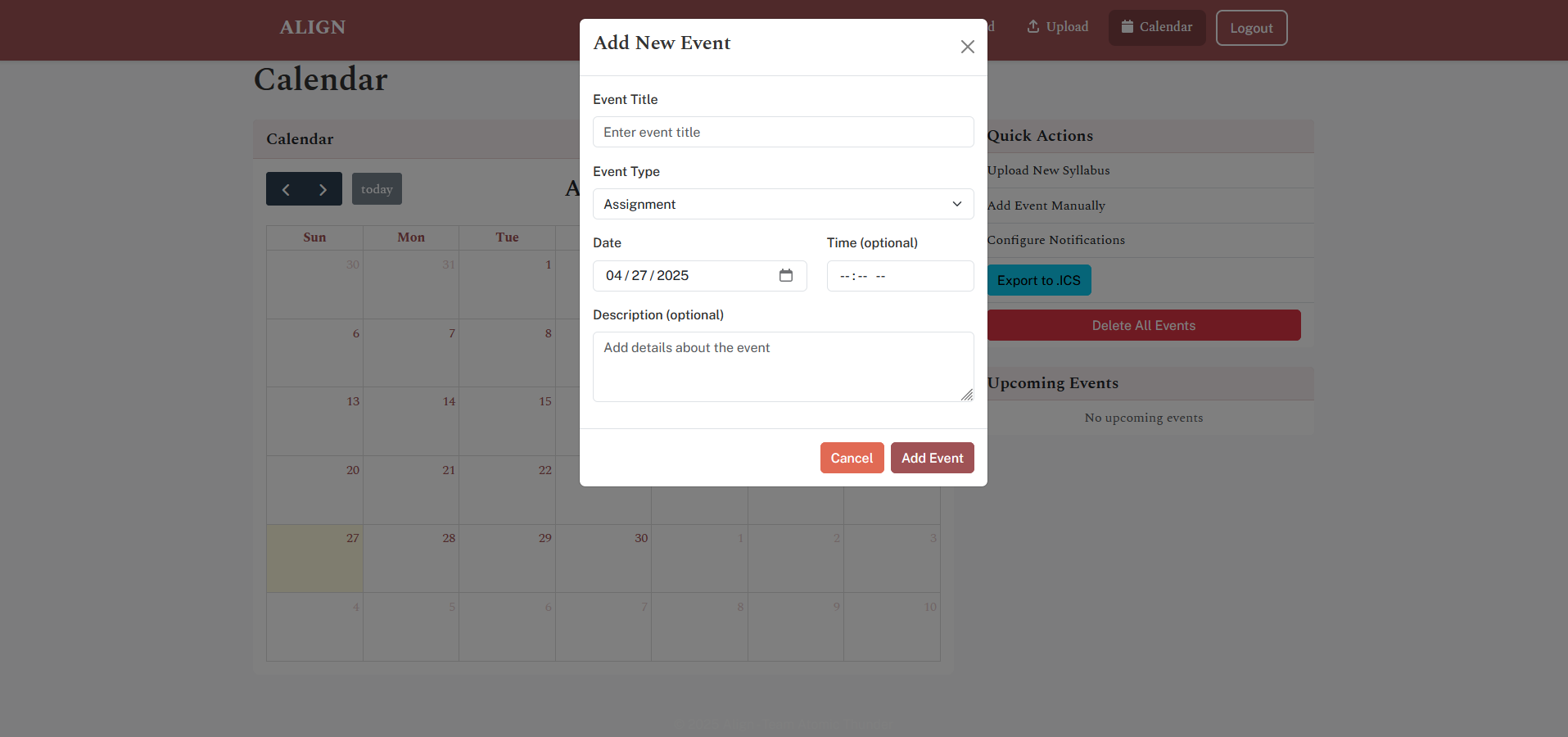












## **Hardware Interfaces**

Supported devices on the users side will include phones and computers capable of browsing the internet with a JavaScript-supported browser. The device must have a visual interface to display the calendar and syllabus information, as well as input methods such as a touchscreen, keyboard, or mouse for uploading files and interacting with the application. The user must maintain an internet connection for the app to function properly, including uploading syllabi, processing dates, and synchronizing with external calendar services.

## **Software Interfaces**

The server must be able to support:

* Node.js v19.0 or higher
* React v19.0.0 and dependencies
* Firebase for data storage
* PDF-Parse v1.1.1 for syllabus parsing

The client browser must support:

* Modern JavaScript (ES6+)
* Local file access for PDF uploads
* Web notifications API (for reminders)

The server may have any operating system installed provided that it can support the already listed packages and software. Gmail or similar email service accounts will be necessary for sending reminder notifications and account verification.

# **System Features**

## Syllabus Date Extraction

4.1.1 Description and Priority

The system shall accurately extract important dates, deadlines and associated contextual information from syllabi. This feature is **High priority** with the following ratings:

* Benefit: 9 (direct core functionality)
* Penalty: 9 (application cannot function without this capability)

4.1.2 Stimulus/Response Sequences

1. User uploads a syllabus
   1. System validates the format
   2. System processes the document for text extraction
2. System identifies dates and associated events
   1. System parses text for date formats and deadline language
   2. System categorizes events (assignments, exams, projects)
   3. System extracts contextual information (course codes, descriptions)
3. System presents extracted data to user for verification
   1. User confirms extraction accuracy
   2. User makes correction if needed

4.1.3 Functional Requirements

REQ-1: The system shall support uploading PDF documents up to 50MB in size

REQ-2: The system shall extract dates and events from uploaded syllabus document

REQ-3: The system shall identify and categorize at least four types of academic events (exams, assignments, readings, lectures)

REQ-4: The system shall recognize multiple date formats, including MM/DD/YYYY, Month DD, YYYY, and relative dates (e.g., “due next Monday”)

REQ-5: The system shall extract associated contextual information including course code, event description, and importance indicators

REQ-6: The system shall provide visual feedback during the file upload and processing stages (loading indicators, progress status)

REQ-7: The system shall validate uploaded files, ensuring they are in PDF format before processing

REQ-8: The system shall automatically associate extracted events with the user who uploaded the syllabus

REQ-9: The system shall provide error handling and recovery mechanisms during file processing and data extraction

REQ-10: The system shall support drag-and-drop file upload functionality in addition to standard file selection

REQ-11: The system shall provide confirmation messages upon successful processing and extraction

REQ-12: The system shall process text from scanned Pdf documents using text recognition technology

REQ-13: The system shall maintain an organized record of all uploaded syllabi for future reference

REQ-14: The system shall display a summary view of events extracted from each syllabus

REQ-15: THe system shall implement security measures to ensure users can only access their own uploaded syllabi and extracted events

## Calendar Integration

4.2.1 Description and Priority

The system shall create and integrate calendar events from extracted syllabus data into users’ preferred calendar platforms. This feature is **High priority** with the following ratings:

* Benefit: 8 (direct core functionality)
* Penalty: 8 (application would lose value without this)

4.2.2 Stimulus/Response Sequences

1. System generates calendar entries from extracted data
   1. System formats entries with appropriate details and context
   2. System prepares batch of events for calendar addition
2. User reviews generated calendar events
   1. User approves or modifies entries
   2. User selects target calendar
3. System adds approved events to user’s calendar
   1. System authenticates with calendar service
   2. System creates events via API
   3. System confirms successful addition

4.2.3 Functional Requirements

REQ-16: The system shall provide a verification interface allowing users to delete events

REQ-17: The system shall integrate with calendar systems to allow export of extracted events to standard calendar format (.ICS)

REQ-18: The system shall persist extracted events to a database for future reference and modification

REQ-19: The system shall allow users to manually add events that may not have been automatically extracted from syllabi

REQ-20: The system shall generate calendar events with title, date, time and description fields populated from extracted syllabus data

REQ-21: The system shall integrate with three major calendar platforms (Google, Apple, Outlook)

REQ-22: The system shall maintain desired academic context from the syllabus in the created calendar events

# **Other Nonfunctional Requirements**

## **Performance Requirements**

REQ-23: The system shall process a standard syllabus document (up to 10MB) within 30 seconds from upload completion to data extraction.

REQ-24: The calendar integration feature shall complete the addition of events to user calendars within 15 seconds of user confirmation.

REQ-25: The system shall maintain a maximum response time of 2 seconds for all user interface interactions under normal operating conditions.

REQ-26: The system shall process a standard syllabus document in under 30 seconds

## **Safety Requirements**

REQ-27: The system shall implement warning mechanisms to prevent accidental deletion of calendar events.

## **Security Requirements**

REQ-28: The system shall encrypt all stored syllabus data and extracted information using AES-256 encryption.

## **Software Quality Attributes**

REQ-29: The system shall achieve 99.5% uptime during academic semesters (excluding scheduled maintenance).

REQ:30: The system shall be compatible with all major web browsers (Chrome, Safari, Firefox, Edge).

## Business Rules

REQ-31: Only users who uploaded a syllabus shall have access to view, modify, or delete the extracted data and associated calendar events

**Appendix A: Glossary**

| **Term** | **Definition** |
| --- | --- |
| API | Application Programming Interface; allows different software systems to communicate |
| CalDAV | Calendar Access Protocol that allows client access to scheduling information |
| Firebase | A platform for creating mobile and web application used as a database (in this instance) |
| Nodejs | Open source JavaScript runtime environment that executes JavaScript code outside a web browser |
| OAuth 2.0 | Industry standard protocol for authorization that provides secure access without sharing passwords |
| PDF-Parse | JavaScript library that extracts text content from PDF documents |
| React | JavaScript library for building user interfaces |