Collaborative learning platform With personalized study plans

Software Requirements Specification

Version 1.0

Date - 24/08/2025
Mentor – Mr. Meelan Bandara
PID – 1

Group Members:
220008E- Abeyrathna A.H.M.R.T.
220022P - Amanethmi N.V.S.
220044J –Arunodi B.N.

Contributions

Member ID & Name	Assigned Sections
220008E – Abeyrathna A.H.M.R.T.	- Overall Description - Functional Requirements - Non-Functional Requirements - Licensing, Legal, Copyright & Notices

220022P – Amanethmi N.V.S.	- Introduction - Applicable Standards - Supporting Information - Purchased Components - Online User Documentation
220044J – Arunodi B.N.	- Constraints - Interfaces - Database Requirements - Third-Party Integrations

Revision History

Date	Version	Description	Author
24/08/2025	1.0	Initial document	220008E-Abeyrathna A.H.M.R.T. 220022P -Amanethmi N.V.S. 220044J –Arunodi B.N.

Table of Contents

Software Requirements Specification

1.Introduction

1.1 Purpose

The purpose of this Software Requirements Specification (SRS) is to define the functional and non-functional requirements for the development of the **Collaborative Learning Platform with Personalized Study Plans**. This platform aims to facilitate collaborative learning among students by enabling the creation and management of study groups, resource sharing, personalized study plan generation, progress tracking, collaborative document editig,document Quering, quiz attempting, and real-time communication.

This SRS will serve as a guide for the development team, testers, and stakeholders to ensure that the final product meets the specified requirements and aligns with the intended goals of the system. It will also act as a reference for future maintenance, upgrades, and enhancements.

1.2 Scope

The **Collaborative Learning Platform with Personalized Study Plans** is a web-based application that allows students to create or join study groups, share educational resources, collaborate on documents, take quizzes, track progress, and receive AI-assisted personalized study plans.
Key functionalities include:

- Secure user authentication and profile management.
- Study group creation and management based on subjects or interests.
- Centralized repository for resource sharing with tagging and search capabilities.
- Real-time collaborative document editing.
- Document querying option for selected documents
- Collaborative document editing
- AI-powered personalized study plan generation.
- Progress tracking with visual analytics dashboards.
- Interactive quizzes and assessments.
- Integrated communication tools (forums).

The system will be used by students, and group administrators. It will improve learning efficiency, promote peer collaboration, and help users achieve their goals more effectively. The SRS applies to the full version of the platform and its associated subsystems.

1.3 Definitions, Acronyms, and Abbreviations

- **SRS** – Software Requirements Specification
- **AI** – Artificial Intelligence
- **UI** – User Interface
- **UX** – User Experience
- **API** – Application Programming Interface

1.4 References

The following documents, standards, and online resources were referenced in the preparation of this Software Requirements Specification (SRS):

1. **IEEE Std 29148-2018: Systems and Software Engineering – Life Cycle Processes – Requirements Engineering-**Institute of Electrical and Electronics Engineers (IEEE), 2018.-Available at: https://ieeexplore.ieee.org/document/8559686-
2. **React.js Documentation-**Meta Platforms, Inc., Accessed 2025,Available at: https://react.dev/
3. **Material UI Documentation-**MUI Core Team, Accessed 2025. Available at: https://mui.com/
4. **Nest.js Documentation-**OpenJS Foundation, Accessed 2025,Available at: https://docs.nestjs.com/
5. **Supabase Documentation,**Supabase Inc., Accessed 2025.Available at: https://supabase.com/docs
6. **Firebase Documentation-**Google LLC, Accessed 2025,Available at: https://firebase.google.com/docs
7. **WebSockets API Documentation,** MDN Web Docs, Mozilla Foundation, Accessed 2025.
   Available at: https://developer.mozilla.org/en-US/docs/Web/API/WebSockets_API
   .

1.5 Overview

This Software Requirements Specification (SRS) document defines the requirements for the **Collaborative Learning Platform with Personalized Study Plans**.

The document is organized into the following sections:

- **Introduction:** Provides the purpose, scope, definitions, references, and structure of the SRS.
- **Overall Description:** Describes the product perspective, key functionalities, target users, operating environment, and design constraints.
- **System Features:** Lists and explains the functional requirements of the platform in detail.
- **External Interface Requirements:** Specifies the user interfaces, hardware interfaces, software interfaces, and communication interfaces.
- **Non-functional Requirements:** Defines performance, security, usability, and other quality attributes.
- **Other Requirements:** Includes any additional constraints, standards, or regulatory considerations.
- **Appendices:** Provides supplementary information such as glossary, diagrams, and referenced materials.

The SRS will serve as a shared reference for stakeholders, developers, and testers, ensuring that the final system aligns with the agreed objectives and specifications.

2. Overall Description

2.1 product perspective

The Collaborative Learning Platform with Personalized Study Plans (CLPPSP) is a web-based application designed to enhance student learning by combining resource sharing, collaboration, and personalized study planning into a single integrated environment.

The product is envisioned as a standalone application that leverages microservices architecture, enabling modularity, scalability, and ease of integration with external services in the future. It will provide both individual learning support through AI-powered study plans and collaborative features through group discussions, real-time document editing, document querying , and shared resources.

The platform consists of several core subsystems working together:

- Frontend Layer (React.js + Material UI + TypeScript): Provides an interactive and user-friendly interface for students and study groups.
- Backend Layer (Nest.js + Microservices+Python): Manages business logic, authentication, resource management, study plan generation, and communication tools.
- Database Layer (Supabase + Firebase): Handles persistent storage of user data, group information, resources, and real-time updates.
- Real-time Communication Layer (WebSockets): Supports collaborative document editing, live chat, and instant notifications.
- AI/ML Module (Python): Generates personalized study plans and enables document query functionality, enhancing intelligent learning support.

The product is designed to be platform-independent (accessible via modern browsers) and scalable to support a growing number of users and study groups. Initially, it targets university students but can be extended to broader audiences such as professionals and online learners.

2.2 product functions

The system provides the following high-level functionalities:

- **User Authentication and Profiles**
  - Secure sign-up and login using Supabase/Firebase authentication.
  - Individual user profiles track learning preferences, progress, and achievements.
- **Group Creation and Management**
  - Users can create study groups based on subjects, courses, or interests.
  - Group admins can manage membership, roles, and group settings.
- **Resource Sharing**
  - Centralized repository for learning materials: notes, videos, links, and documents.
  - Tagging, rating and search functionality to easily find resources.

- **Collaborative Document Editing**
    - Real-time editing of shared documents within study groups.
    - WebSocket-based synchronization ensures conflict-free updates.
- **Personalized Study Plan Generation**
    - AI-powered recommendations generate study plans tailored to individual goals and learning pace..
- **Progress Tracking and Analytics**
    - Dashboards provide visual analytics of individual and group performance.
    - Identifies areas needing improvement and patterns in study behavior.
- **Interactive Quizzes and Assessments**
    - Users can create and participate in quizzes.
    - Immediate feedback and detailed performance analytics.
- **Document Querying Functionality**
    - Students can ask questions about any document in the resource section.
    - AI-assisted responses provide accurate answers quickly.
- **Communication Tools**
    - Integrated forums for real-time discussion within groups.

2.3 user characteristics

- **Students (Primary Users):**
    - Age group: any
    - Basic computer and internet literacy
    - Interested in group learning, resource sharing, and self-improvement
- **Admins (Secondary Users):**
    - Manage user accounts and monitor group activity
    - Moderate shared content for safety and quality
- **Guests (Optional Users):**
    - May view public resources
    - Require registration to access full features

2.4 constraints

1. Platform Constraints

- The application must be web-based and responsive across devices including desktops, laptops, tablets, and smartphones.
- Browser compatibility should be ensured for major modern browsers such as Chrome, Firefox, Edge, and Safari.

2. Performance Constraints

- Real-time collaboration features (document editing, chat) require stable and moderate-speed internet connectivity.
- The system should handle simultaneous users in multiple study groups without significant delays or performance degradation.
- API response time should be within acceptable limits (< 2 seconds for common operations).

3. Security Constraints

- User authentication must use secure methods (Supabase/Firebase authentication) with password hashing and secure sessions.
- Access control must ensure that only authorized users can view or edit group resources or documents.

4. Data Constraints

- Data stored in the system (user profiles, resources, quizzes) should be structured and consistent, avoiding duplication or corruption.
- Storage capacity may be limited depending on server configuration

5. Development Constraints

- The system will be implemented using the specified tech stack: React.js, Material UI, Nest.js, Supabase, Firebase, WebSockets, Python microservices, and TypeScript.
- Microservices architecture must follow modularity principles, allowing independent deployment and scaling.
- AI-based study plan generation will rely on Python-based algorithms and may have limitations on computation time for large datasets.

6. Operational Constraints

- System availability depends on server uptime and network connectivity. Downtime should be minimized.
- Users are expected to have modern devices and updated browsers for proper platform functionality.
- Any third-party integrations (AI services) may introduce dependency limitations.

2.5 Assumptions and dependencies

Assumptions

1. **User Access:**
   a. Users will have reliable internet connectivity and devices (desktop, laptop, tablet, or smartphone) to access the platform.
   b. Users possess basic knowledge of using web applications and digital learning tools.
2. **User Engagement:**
   a. Students and educators are motivated to actively participate in collaborative learning activities.
   b. Users will regularly update their profiles and track progress to make the personalized study plans effective.
3. **System Usage:**
   a. The system will be primarily accessed during normal academic hours, though it should support 24/7 availability for asynchronous study.
   b. Users will follow platform guidelines for content sharing, collaboration, and respectful communication.
4. **Data Availability:**
   a. Required learning resources (documents, videos, links) will be uploaded and maintained by users or facilitators.
   b. AI-based study plan recommendations assume sufficient historical data and user inputs for meaningful suggestions.

Dependencies

1. **Technology Dependencies:**
   a. **Supabase/Firebase:** For authentication, database storage, and real-time updates.
   b. **WebSockets:** For real-time collaboration in document editing and chat.
   c. **Python microservices:** For AI-powered personalized study plan generation.
   d. **React.js & Material UI:** For frontend development and responsive UI.
   e. **Nest.js:** For backend API and microservices management.
2. **Third-Party Services:**
   a. cloud storage, hosting services, or AI libraries used are assumed to be reliable and available.
   b. Notifications, email, or messaging services depend on external service availability if integrated.
3. **Environmental Dependencies:**
   a. Browser compatibility for modern web browsers.
   b. Adequate server resources (CPU, memory, storage) for handling multiple users and real-time operations.
4. **Organizational Dependencies:**
   a. Students and educators will follow proper workflows for uploading resources, creating groups, and participating in quizzes or discussions.
   b. Administrative support may be required for user management and monitoring system usage.

2.6 requirements subsets

1. Functional Requirements

The system shall provide the following core functionalities:

- **User Authentication and Profiles** – Secure user registration, login, and personalized profiles to track learning progress.
- **Group Management** – Users can create, join, and manage study groups with different roles and permissions.
- **Resource Sharing** – A centralized repository for uploading, organizing, tagging, and accessing study materials.
- **Collaborative Document Editing** – Real-time editing of shared documents and notes using WebSockets.
- **Personalized Study Plan Generation** – AI-driven study plans tailored to user goals, pace, and available resources.
- **Progress Tracking and Analytics** – Dashboards for monitoring individual and group performance.
- **Quizzes and Assessments** – Ability to create, attempt, and analyze quizzes with instant feedback.
- **Document Querying** – Students can ask questions related to any document in the resource section and receive responses.
- **Communication Tools** – discussion forums, and notifications for effective collaboration.

2. Non-Functional Requirements

These describe the overall qualities and constraints of the system:

- **Performance** – The platform should handle multiple concurrent users with minimal latency (less than 2 seconds for normal operations).
- **Security** – All user data must be stored securely, using encrypted communication and role-based access.
- **Usability** – The UI must be intuitive, responsive, and accessible across different devices.
- **Reliability** – The system should maintain 99% uptime for essential services.
- **Scalability** – The microservices architecture should allow independent scaling of modules.
- **Maintainability** – Codebase should follow modular and clean design practices for easy updates.

3. Interface Requirements

The system shall support:

- **User Interface** – A responsive web-based interface developed using React.js and Material UI.
- **API Interface** – Backend services exposed via REST APIs built in Nest.js.
- **Real-Time Interface** – WebSocket connections for chat and collaborative editing.
- **External Integrations** – Supabase/Firebase for authentication, data storage, and real-time synchronization.

4.Data Requirements

The system must store and manage the following data:

- **User Data** – Personal details, goals, preferences, and progress history.
- **Resource Data** – Documents, videos, links, and associated metadata.
- **Group Data** – Study group information, memberships, and roles.
- **Analytics Data** – Quiz results, study patterns, and engagement metrics.
- **Study Plans** – AI-generated study plans with logs of updates and recommendations.

5.System Constraints

The development and operation of the system are bound by:

- **Browser Compatibility** – Must support Chrome, Firefox, Edge, and Safari.

- **Device Compatibility** – Fully responsive design for desktop, tablet, and mobile.
- **Performance Standards** – Stable performance under concurrent usage.
- **Security Compliance** – Must follow data protection and privacy regulations.
- **Technology Stack** – Restricted to React.js, Material UI, Nest.js, Supabase, Firebase, WebSockets, Python, and TypeScript.

3. Specific Requirements

3.1 Functional Requirements

3.1.1 User Authentication and Profiles

- The system should allow new users to register using email, password, and optional OAuth providers (Google).
- The system should allow existing users to log in securely using authentication tokens (JWT).
- The system should allow users to reset passwords through email verification.
- Each user should have a personalized profile containing:
  - Name
  - Email
  - Profile picture
  - Bio/short description etc.
- Users should be able to update their profile information.
- The system should track progress metrics (study hours, completed tasks, quiz performance) .

3.1.2 Group Creation and Management

- The system shall allow users to create study groups based on subjects or interests.
- A group shall have:
  - Group name
  - Subject/interest category
  - Description
  - Group admin (creator)
- The system shall allow users to invite other members to join groups.
- Group admins shall be able to:
  - Approve/reject join requests
  - Assign co-admins
  - Remove users from the group
- The system shall allow members to leave groups voluntarily.
- In each group there can be threads and according to user interest they can join that threads
- Each group shall have a dashboard displaying:
  - Member list
  - Forum
  - Study plan generation option

    - Group progress metrics
    - Threads
        - Shared resources
        - progress
        - Thread description
        - Quizzes
        - Documents editing at that time

3.1.3 Resource Sharing

- The system shall provide a centralized repository for learning materials.
- Users shall be able to upload resources (documents, links, videos).
- The system shall allow organizing resources by tags, categories, or subject.
- Users shall be able to search and filter resources by keywords, type, or uploader.
- The system shall restrict file size limits (50 MB per file).
- The system shall ensure that only authorized group members can view group-specific resources.
- In each resource there is the chance to rate, comment and tag any other resources on them.

3.1.4 Functional Requirement Four – Collaborative Document Editing

- The system shall support real-time collaborative editing of documents.
- Multiple users shall be able to edit the same document simultaneously.
- The system shall show live cursor indicators for each active user.
- The system shall maintain a revision history, allowing users to view or revert to previous versions.
- Only group members shall be allowed to access/edit documents shared within that group.
- Members should request the access to edit a specific document

3.1.5 Personalized Study Plan Generation

- The system shall provide a feature to generate study plans tailored to:
    - User's learning goals
    - Time availability
    - Subject preferences
- The system shall use an AI-powered algorithm to recommend study schedules.
- The study plan shall include:
    - List of topics to study
    - Suggested resources
    - Deadlines and milestones
- Users shall be able to manually adjust the generated study plan.
- The system shall track adherence to the plan (completed vs. pending tasks).

3.1.6 Progress Tracking and Analytics

- The system shall provide a visual dashboard for users to track progress.
- Metrics displayed shall include:
    - Study hours logged
    - Quizzes completed
    - Resources accessed
    - Group activity levels

3.1.7 Interactive Quizzes and Assessments

- The system shall allow users/admins to create quizzes with:
    - Multiple-choice questions
    - True/False questions
    - Short-answer questions
- The system shall allow members to participate in quizzes in real-time or asynchronously.
- The system shall provide instant feedback on answers.
- Quiz results shall be stored in the user's profile and contribute to progress analytics.
- The system shall allow admins to export quiz results for analysis.

3.1.8 Forum

- The system shall provide an integrated forum within every study group.
- features shall include:
    - Text messages
    - reactions
    - Image sharing
- The system shall send notifications for new messages or group activity.
- Student can reply for any massage in the forum
- Admin can keep the control of the forum and have the massage pin option.

3.9 Document Querying Functionality

The system shall allow users to query documents uploaded in the resource section.

- The system shall allow users to ask questions related to the content of a document.
- retrieve relevant responses using AI-based assistance.
- The system shall link each query to the specific document for context.
- The system shall allow users to view previously asked questions and answers.
- The system shall support to give the access to query on any selected document within that thread.

3.2 Usability

3.2.1 Training Time

- A **normal user (student)** shall be able to learn how to:
  - Register/login,
  - Join/create study groups,
  - Access resources,
  - Participate in quizzes,

  within **30 minutes** of first use without formal training.
- A **power user (group admin)** shall be able to perform advanced operations (group management, quiz creation, analytics interpretation) about 1 hours of guided practice.

3.2.2 Task Efficiency

- To :
  - upload and share a resource (PDF, link) shall not exceed 2 minutes.
  - create a study group and invite members shall not exceed 3 minutes.
  - generate a personalized study plan shall not exceed 1 minute.
  - get a response for document querying shall not exceed 30 seconds.

3.2.3 Familiarity and Standards

- The system shall follow common usability standards such as:
  - Microsoft's GUI standards for navigation (familiar icons, menu layouts).
  - WCAG 2.1 accessibility standards for readability (font size, color contrast, keyboard navigation).
  - Mobile-first responsive design (consistent experience across desktop, tablet, and mobile).
- The system's UI/UX shall be similar to widely used learning platforms reducing the learning curve for new users.

3.2.4 Help and Error Recovery

- The system shall provide contextual help (tooltips, help icons) for all major features.
- The system shall provide a searchable help center with FAQs and guides.
- The system shall provide clear error messages and recovery options ("File too large – please upload a file under 50 MB").

3.2.5 Accessibility

- The platform shall support screen readers for visually impaired users.
- All images shall have alternative text (alt tags).
- The system shall allow keyboard-only navigation for users with motor impairments.

3.3 Reliability

3.3.1Availability

- The system shall be available 99.5% of the time on a monthly basis, excluding scheduled maintenance.
- The system shall support 24/7 access for users in different time zones.
- Scheduled maintenance shall not exceed 4 hours per month, and users must be notified at least 24 hours in advance.
- In the event of a degraded mode (heavy traffic), the system shall still allow basic functionality (login, access to study materials, and messaging).

3.3.2 Mean Time Between Failures (MTBF)

- The system shall achieve an MTBF of at least 500 hours (approx. 20 days) in production.
- Failures are defined as critical outages that prevent users from logging in, accessing study plans, or using core group features.

3.3.3 Mean Time To Repair (MTTR)

- The system shall achieve an MTTR of less than 2 hours for critical issues.
- For non-critical issues (minor UI glitches), fixes shall be deployed within 72 hours.

3.3.4 Accuracy

- Personalized study plans shall have a minimum accuracy of 95% in reflecting user preferences and performance history.
- Quiz scoring shall be 100% accurate, with no tolerance for calculation errors.
- Resource uploads/downloads shall maintain file integrity with 0% corruption rate.
- Document querying responses should be 100% accurate.

3.3.5 Maximum Bug/Defect Rate

- The system shall maintain a maximum defect rate of 1 bug per 1,000 lines of code (1 bug/KLOC) in production.
- Critical bugs (system crash, data loss, or login failure) shall not exceed 1 per release cycle.
- Significant bugs (incorrect output, broken features) shall not exceed 3 per release cycle.
- Minor bugs (UI inconsistencies, typos) shall not exceed 10 per release cycle.

3.4 Performance and Security

3.4.1 Performance Requirements

3.4.1.1 Response Time

- The system shall respond to standard user transactions (login, opening a group, accessing a resource) within an average of 2 seconds and a maximum of 5 seconds under normal load.
- Real-time features such as chat and collaborative editing shall update within 1 second of user input
- Document querying responses should come maximum 30 seconds.

3.4.1.2 Throughput

- The system shall support at least 100 transactions per second across all services.
- Real-time collaboration (WebSockets) shall handle at least 500 concurrent active connections without performance degradation.

3.4.1.3 Capacity

- The system shall support 1,000 registered users and at least 100 concurrent active users.
- Each study group shall support up to 50 members with smooth operation.

3.4.1.4 Degradation Modes

- In case of partial failure, essential services (authentication, resource access, communication tools) shall remain functional.
- Non-critical services (analytics) may be temporarily disabled during system degradation.

3.4.2 Security Requirements

3.4.2.1 Authentication & Authorization

- The system shall enforce secure authentication using Supabase/Firebase with encrypted credentials.
- Role-based access control shall restrict functionality based on user type (student, admin, educator).

3.4.2.2 Security Requirement Two – Data Protection

- All sensitive data (user profiles, passwords, quiz results) shall be stored in encrypted format.
- All communication between client and server shall use HTTPS encryption.

3.4.2.3 Privacy

- User data shall not be shared with third parties without consent.
- The system shall comply with data protection regulations (such as GDPR principles).

3.4.2.4 Backup and Recovery

- The system shall maintain daily backups of all critical data.
- In case of data corruption, recovery shall be possible within 6 hours.

3.4.2.5 Audit and Monitoring

- The system shall maintain logs of all critical activities (logins, resource uploads, group creations, quiz submissions).
- Admins shall have access to audit logs for security monitoring.

3.5 Supportability

3.5.1 Coding Standards

- The system shall follow recognized coding standards for all technologies used (React.js, Nest.js, Python, TypeScript).
- Developers shall follow the Airbnb JavaScript/TypeScript style guide for frontend and backend code.
- Code shall be reviewed regularly through peer review and pull requests to maintain quality.
- Linting tools (e.g., ESLint, Prettier) shall be integrated into the build pipeline to enforce consistency.

3.5.2 Naming Conventions

- Variable names shall be written in camelCase.
- Class names shall use PascalCase.
- Constants shall be written in UPPER_SNAKE_CASE.
- Database fields shall follow snake_case for consistency across SQL/NoSQL systems.
- File and folder naming conventions shall be consistent across the project (components in React start with uppercase).

3.5.3 Modular Design

- The system shall follow a microservices architecture so that each service (authentication, quiz, analytics, chat, etc.) is independent.
- Each service shall expose well-documented REST APIs or WebSocket endpoints for interaction.
- Updates, patches, or bug fixes in one service shall not break other services.
- The system shall follow loose coupling and high cohesion principles to maximize maintainability.

3.5.4 Documentation

- The system shall provide:
  - **Developer documentation** (installation steps, environment setup, code structure).
  - **API documentation** (using Swagger/OpenAPI for REST APIs).
  - **Architecture diagrams** to explain system components and their interactions.
  - **End-user documentation** (help guides, FAQs, tutorials for students and admins).
- Documentation shall be updated whenever new features are added.

3.5.5 Maintenance Tools and Access

- The system shall integrate logging and monitoring tools (e.g., Supabase logs, Firebase analytics, third-party monitorin).
- The system shall provide error tracking tools to identify bugs quickly.
- Admins shall have access to a maintenance dashboard to view system health, user activities, and resource usage.
- Automated deployment tools (GitHub Actions, Docker) shall be used for CI/CD.
- The system shall support rollback mechanisms in case a deployment introduces errors.

3.5.6 Testing and Debugging

- The system shall include unit tests, integration tests, and end-to-end tests for all major features.
- Test automation frameworks shall be used to ensure continuous quality.
- A minimum of 80% code coverage shall be maintained for all core modules.
- Debugging utilities and logs shall be available for developers during maintenance.

3.5.7 Scalability and Extensibility

- The system shall be designed to scale horizontally, allowing new services to be added without major redesign.
- APIs shall be version-controlled to allow backward compatibility with older clients.
- New features shall be added as independent modules with minimal changes to existing code.

3.6 Constraints

3.6.1 Standards Compliance

- The system shall comply with recognized web and software development standards, including:
  - **W3C standards** for HTML, CSS, and JavaScript.

    - **Accessibility standards** such as WCAG 2.1 for users with disabilities.
    - **Security standards** such as TLS/HTTPS for data transmission and OWASP best practices for web security.
    - **UI/UX guidelines** based on Microsoft GUI and Material Design standards to ensure consistency and usability.
- All third-party libraries or frameworks used shall meet licensing and security compliance requirements.

3.6.2 Software Languages and Frameworks

- The system shall be implemented using the following mandated technology stack:
    - **Frontend:** React.js with Material UI for responsive and interactive interfaces.
    - **Backend:** Nest.js with TypeScript for REST API services.
    - **Real-time communication:** WebSockets for collaborative editing and chat.
    - **Database & Authentication:** Supabase and Firebase.
    - **AI and computation services:** Python-based microservices for personalized study plan generation and document querying.
- Developers shall not introduce alternative languages or frameworks that conflict with the approved stack.

3.6.3 Hardware Limitations

- The system shall function on standard consumer-grade devices, including:
    - Desktops, laptops, tablets, and smartphones.
    - Minimum specifications: 4GB RAM, modern web browser, and internet speed of at least 5 Mbps.
- The server infrastructure shall be scalable but must optimize CPU, memory, and storage usage to remain cost-effective.
- Large media files shall be compressed or streamed to avoid excessive local resource usage.

3.6.4 Architectural Constraints

- The system shall follow a microservices architecture to separate functionalities such as authentication, resource management, chat, analytics, and AI services.
- All services shall communicate through well-defined APIs and message queues where necessary.
- The system shall support horizontal scaling, allowing services to scale independently based on demand.

3.6.5 Development and Deployment Tools

- Version control shall be maintained using **Git**.
- Continuous Integration/Continuous Deployment (CI/CD) shall use tools like **GitHub Actions**.
- Containerization shall be used for deployment (**Docker**) to ensure environmental consistency.
- Testing frameworks such as **Jest, Cypress, and PyTest** shall be used for unit, integration, and end-to-end testing.

3.6.6 Third-Party Components

- Only approved third-party libraries or APIs shall be used.

- Integration with cloud services (Supabase, Firebase) must follow their service limitations and pricing constraints.
- All external components must be compatible with the main technology stack and meet security standards.

3.7 On-line User Documentation and Help System Requirements

3.7.1 User Manuals

- The system shall provide comprehensive online user manuals for all types of users (students, group admins).Manuals shall include step-by-step instructions for core functionalities such as user registration and login Creating, joining, and managing study groups, uploading and accessing resources, participating in quizzes and assessments ,using collaborative document editing and chat and Document querying.
- Manuals shall include screenshots and examples where necessary for clarity.

3.7.2 Context-Sensitive Help

- The system shall provide context-sensitive help that allows users to access guidance directly from the page they are on.
- Help pop-ups or tooltips shall explain specific fields, buttons, or functionalities.
- Users shall be able to search help topics via a search bar in the documentation/help interface.

3.7.3 FAQs and Troubleshooting Guides

- The system shall maintain an **online FAQ section** addressing common questions and issues.
- A troubleshooting guide shall be provided for resolving common problems, such as login issues, file upload errors, and connectivity problems.
- FAQs and guides shall be updated regularly as new features are added.

3.7.4 Help About Notices

- The system shall provide an About/Help section that includes:
  - Software version and update information
  - Contact information for technical support
  - Links to detailed documentation and tutorials
- Users shall be able to submit support tickets or feedback via the Help section.

3.8 Purchased Components

1. Supabase

Provides authentication, database management, and real-time synchronization. Used for CRUD operations, user authentication, and real-time collaborative features. Must comply with Supabase's free-tier or paid plan limitations. Fully compatible with React.js frontend and Nest.js backend; supports WebSockets for collaborative editing and chat.

2. Firebase

Supports authentication, hosting, notifications, and real-time database features.Utilizes Firebase SDKs for login, session management, and notifications. Must comply with Firebase service limits and licensing agreements. Works across supported devices, browsers, and integrates with the system's main technology stack.

3.Third-Party Libraries

Used for UI components, routing, charting, and real-time communication (Material UI, React Router, Chart.js/Recharts, WebSocket libraries).Must use open-source libraries with permissive licenses (e.g., MIT). No proprietary or incompatible libraries shall be used. Must integrate seamlessly with React.js frontend, Nest.js backend, and Python microservices.

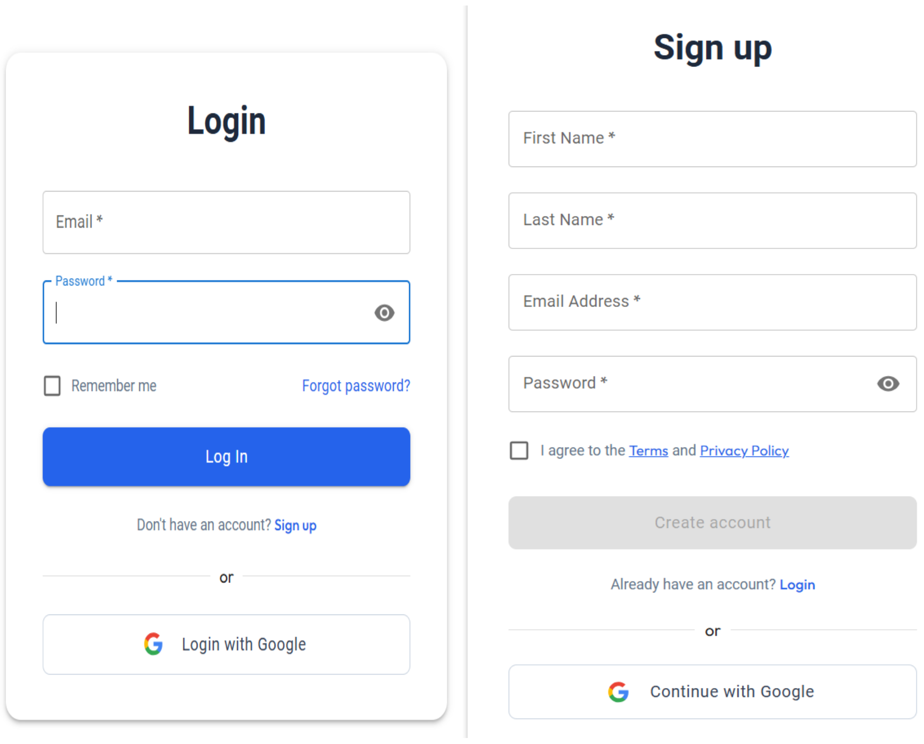
3.9 Interfaces

3.9.1 User Interfaces

The system shall provide intuitive and responsive interfaces for all users (students, group admins, educators). Key interface considerations include:

Login & signUp Page

- Input fields: Email/Username, Password, Confirm Password
- Buttons: Login, Forgot Password
- Google login options



Home Page

Hero Section

- Serves as the entry point of the application.
- Provides functionality to **search for study groups** based on subject, course, or keywords.
- Allows users to **create a new group** or **join an existing group** directly from the homepage.
- Contains a prominent call-to-action to encourage immediate engagement.

Key Features Section

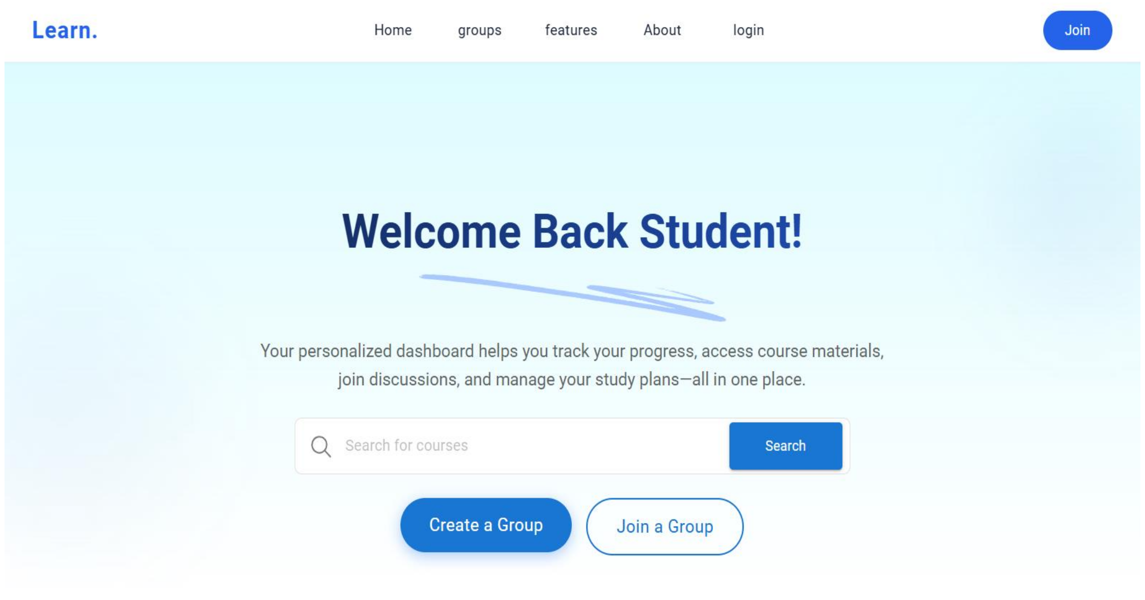
- Highlights the core features of the system such as collaborative document sharing, group chat, AI-powered assistance, and personalized study plans.
- Each feature is visually represented with icons and concise descriptions to improve user understanding.

Popular Groups Section

- Displays a selection of trending or most active groups.
- Helps new users quickly discover relevant communities without needing to search extensively.
- Encourages participation by showcasing member activity and popularity metrics.

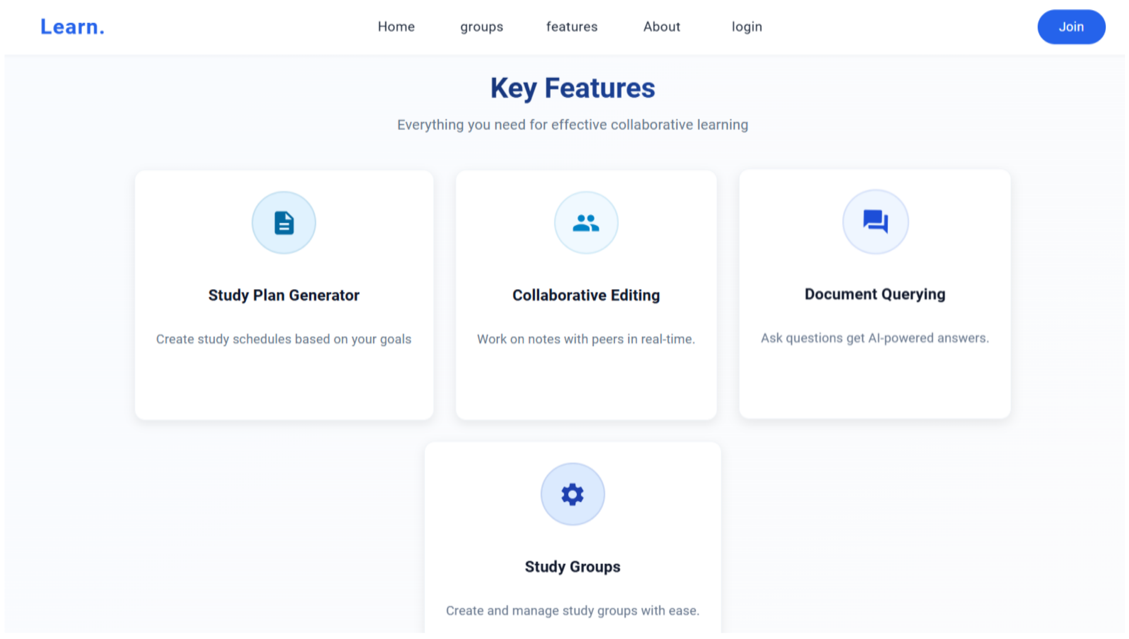
Footer Section

- Provides navigation to supporting pages (About, Contact, Privacy Policy, Terms of Service).
- Contains quick links to external resources and social media channels.
- Offers a consistent ending point for every page with branding and copyright details.



Welcome Back Student!

Your personalized dashboard helps you track your progress, access course materials, join discussions, and manage your study plans—all in one place.



Key Features

Everything you need for effective collaborative learning

Study Plan Generator

Create study schedules based on your goals

Collaborative Editing

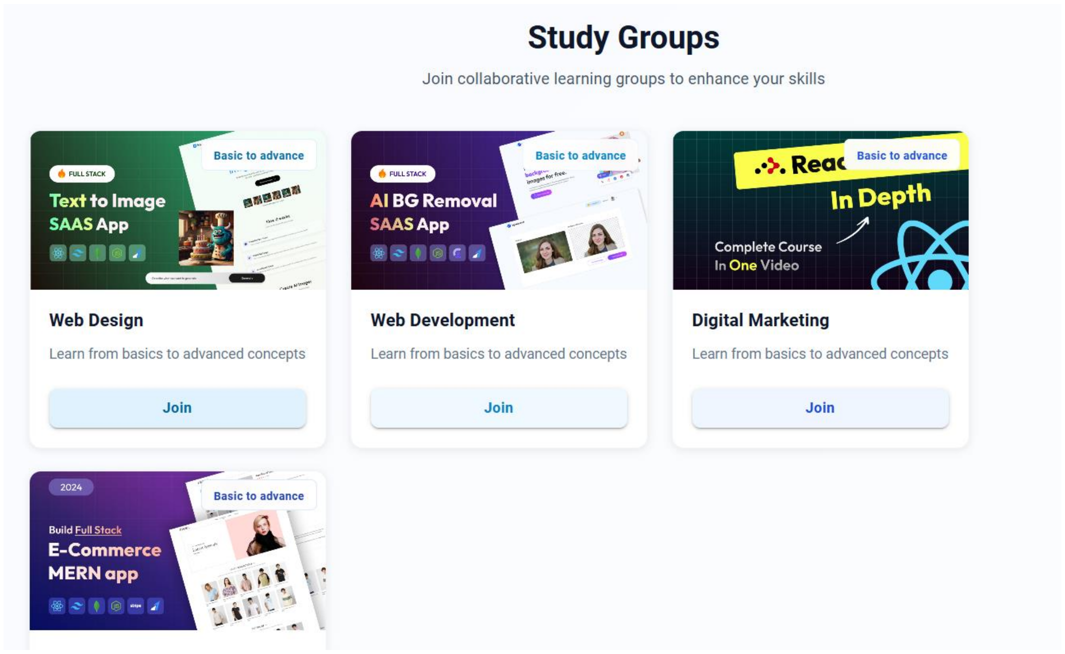
Work on notes with peers in real-time.

Document Querying

Ask questions get AI-powered answers.

Study Groups

Create and manage study groups with ease.



Study Groups

Join collaborative learning groups to enhance your skills

Basic to advance

Web Design

Learn from basics to advanced concepts

Join

Basic to advance

Web Development

Learn from basics to advanced concepts

Join

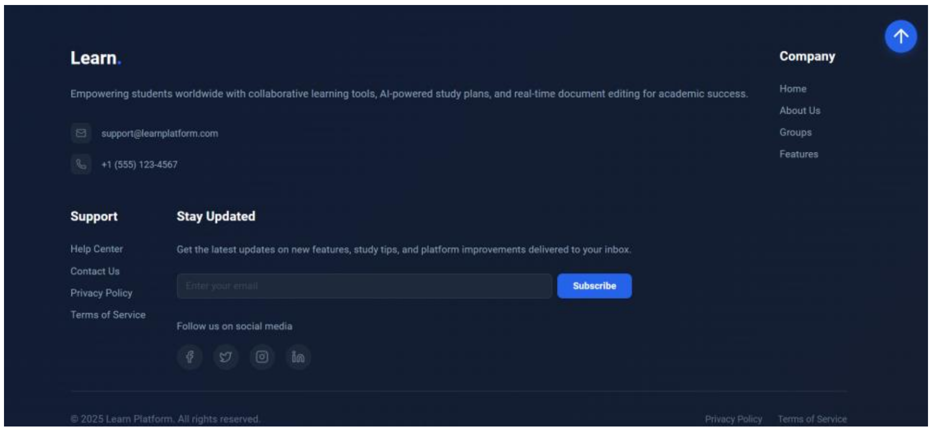
Basic to advance

Digital Marketing

Learn from basics to advanced concepts

Join

Basic to advance



Learn.

Empowering students worldwide with collaborative learning tools, AI-powered study plans, and real-time document editing for academic success.

support@learnplatform.com

+1 (555) 123-4567

Company

Home

About Us

Groups

Features

Support

Help Center

Contact Us

Privacy Policy

Terms of Service

Stay Updated

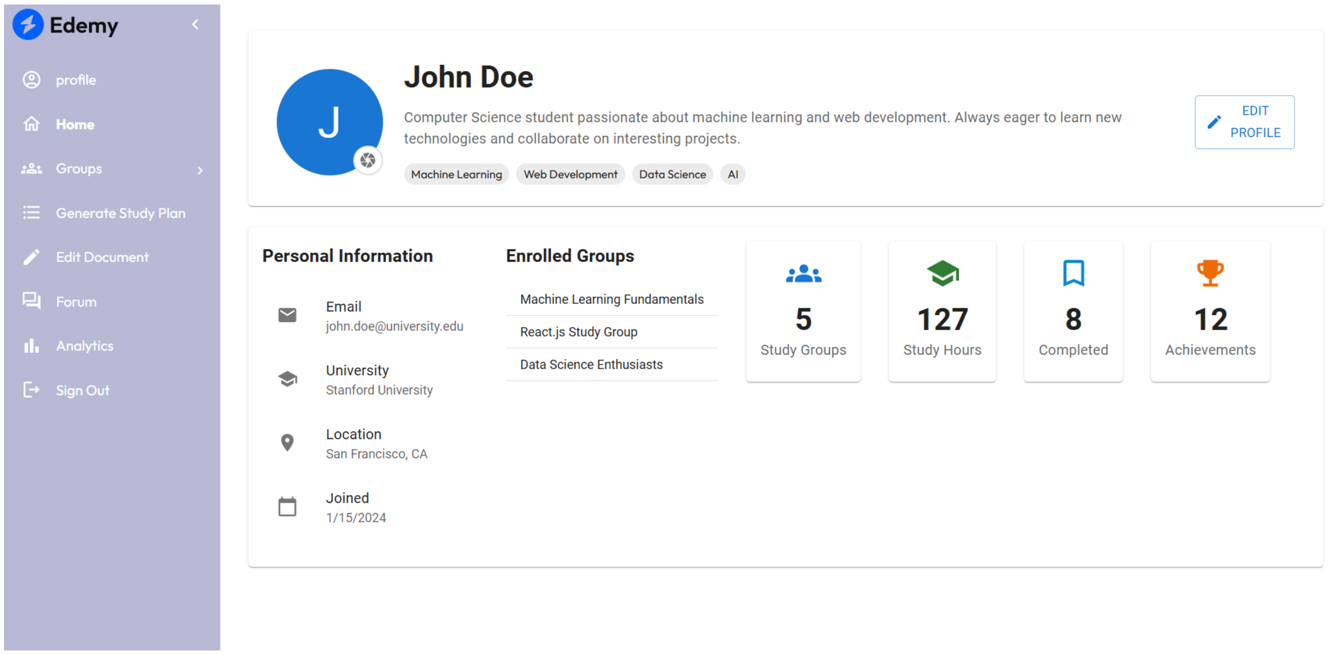
Get the latest updates on new features, study tips, and platform improvements delivered to your inbox.

Follow us on social media

© 2025 Learn Platform. All rights reserved.

Privacy Policy Terms of Service

Profile Page



John Doe

Computer Science student passionate about machine learning and web development. Always eager to learn new technologies and collaborate on interesting projects.

Machine Learning · Web Development · Data Science · AI

Personal Information

Email
john.doe@university.edu

University
Stanford University

Location
San Francisco, CA

Joined
1/15/2024

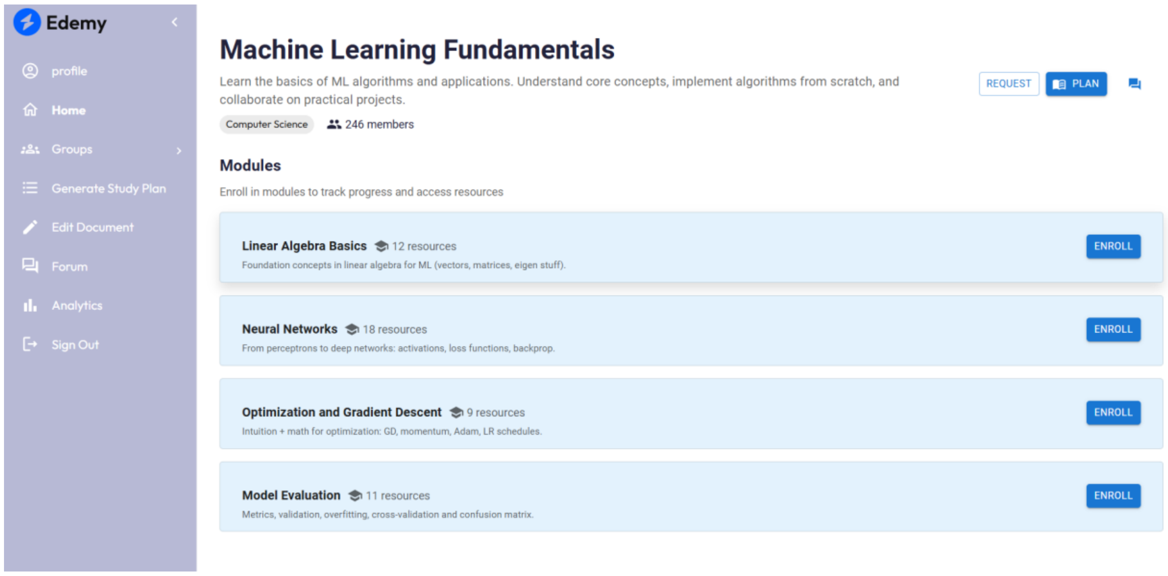
Enrolled Groups

Machine Learning Fundamentals
React.js Study Group
Data Science Enthusiasts

5 Study Groups
127 Study Hours
8 Completed
12 Achievements

Group Pages

- Panels showing group members, forum and study plan generation option
- Buttons: request for the group,enroll for threads



Machine Learning Fundamentals

Learn the basics of ML algorithms and applications. Understand core concepts, implement algorithms from scratch, and collaborate on practical projects.

Computer Science · 246 members

Modules

Enroll in modules to track progress and access resources

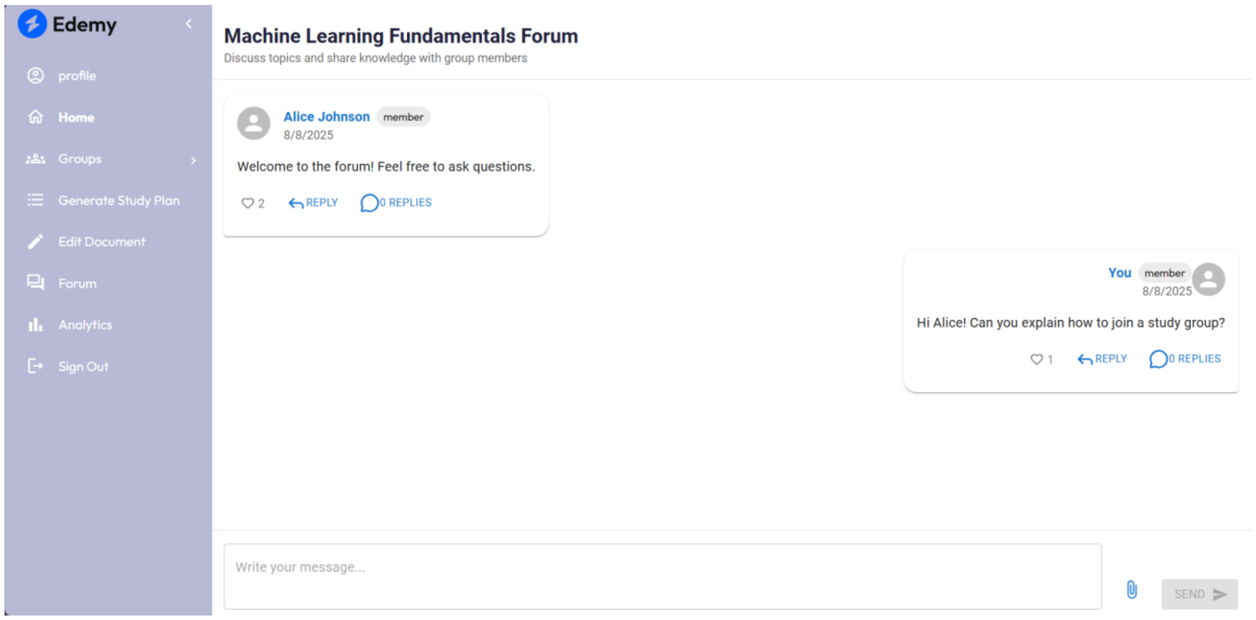
Linear Algebra Basics 12 resources
Foundation concepts in linear algebra for ML (vectors, matrices, eigen stuff).

Neural Networks 18 resources
From perceptrons to deep networks: activations, loss functions, backprop.

Optimization and Gradient Descent 9 resources
Intuition + math for optimization: GD, momentum, Adam, LR schedules.

Model Evaluation 11 resources
Metrics, validation, overfitting, cross-validation and confusion matrix.

Forum



Machine Learning Fundamentals Forum

Discuss topics and share knowledge with group members

Alice Johnson member
8/8/2025

Welcome to the forum! Feel free to ask questions.

♡ 2 REPLY 0 REPLIES

You member
8/8/2025

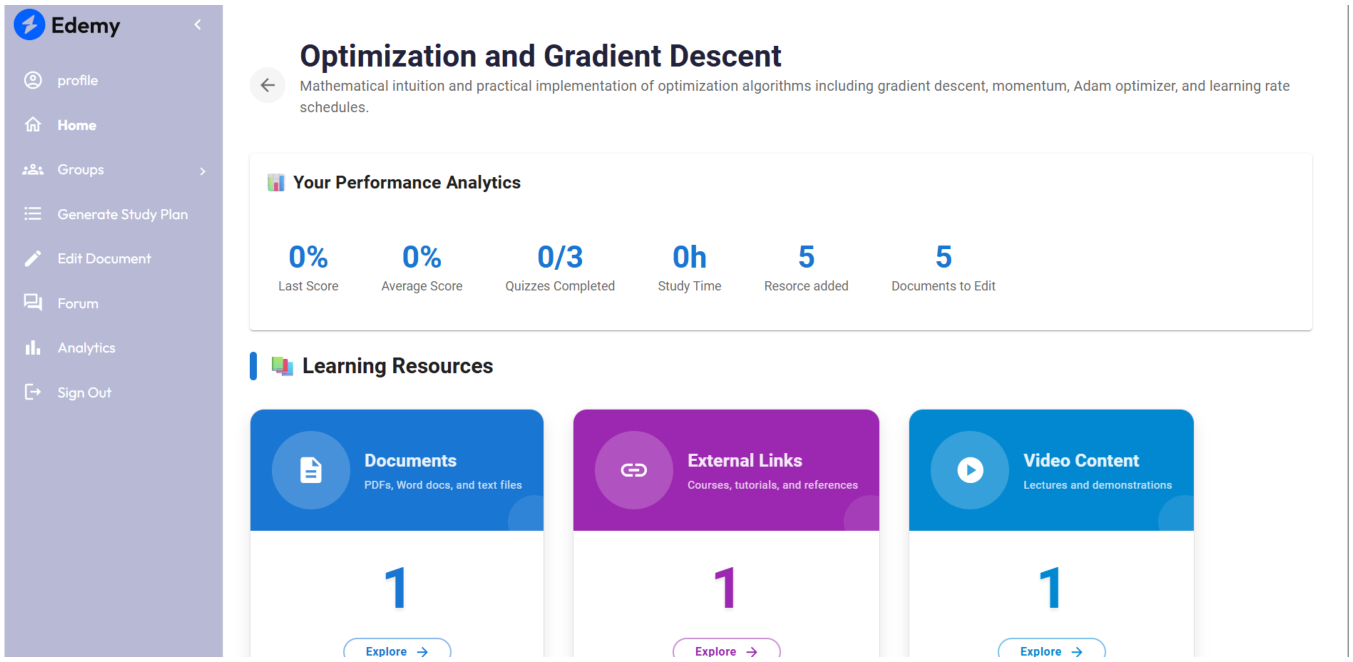
Hi Alice! Can you explain how to join a study group?

♡ 1 REPLY 0 REPLIES

Write your message...

SEND

Module/Thread Pages



Optimization and Gradient Descent

Mathematical intuition and practical implementation of optimization algorithms including gradient descent, momentum, Adam optimizer, and learning rate schedules.

Your Performance Analytics

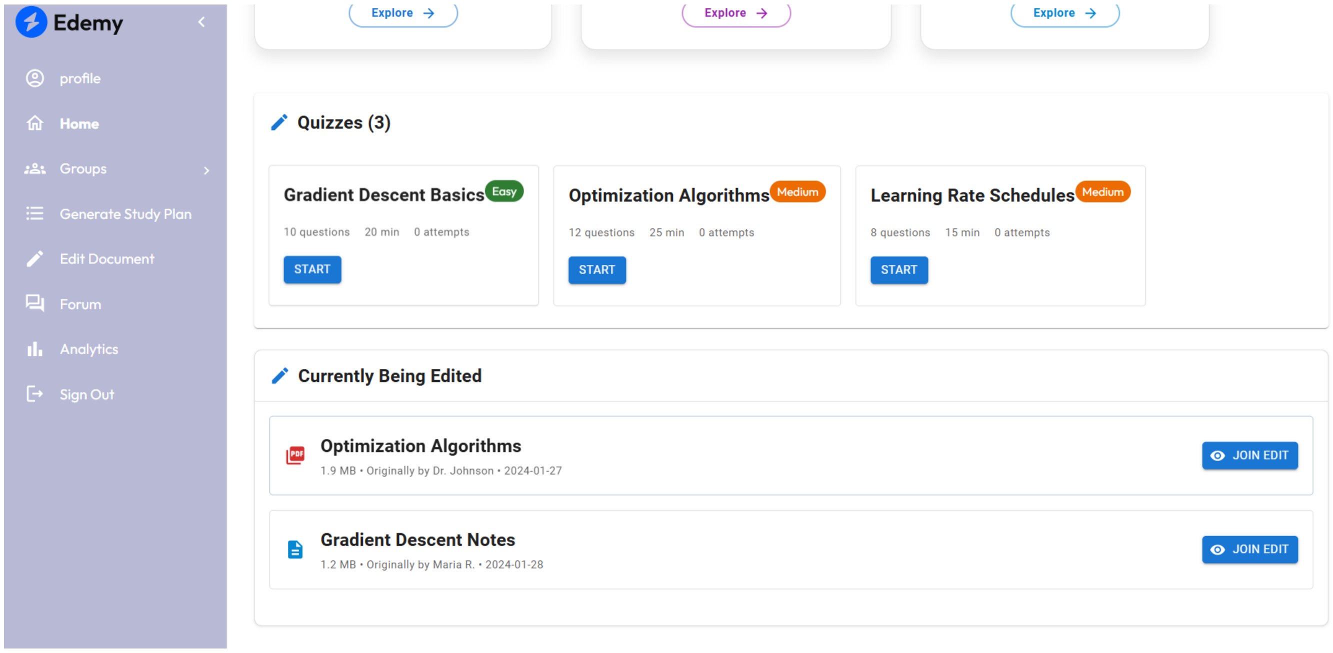
0%	0%	0/3	0h	5	5
Last Score	Average Score	Quizzes Completed	Study Time	Resorce added	Documents to Edit

Learning Resources

Documents — PDFs, Word docs, and text files — 1 — Explore

External Links — Courses, tutorials, and references — 1 — Explore

Video Content — Lectures and demonstrations — 1 — Explore



Quizzes (3)

Gradient Descent Basics Easy
10 questions 20 min 0 attempts
START

Optimization Algorithms Medium
12 questions 25 min 0 attempts
START

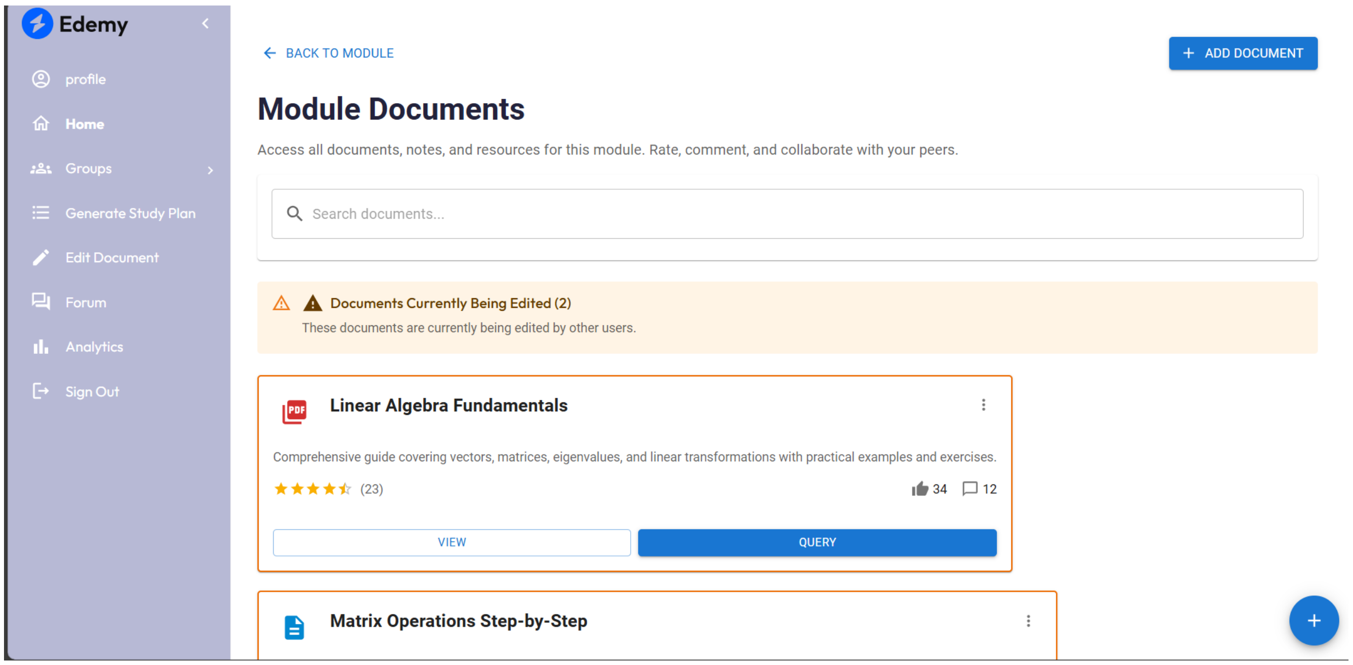
Learning Rate Schedules Medium
8 questions 15 min 0 attempts
START

Currently Being Edited

Optimization Algorithms
1.9 MB • Originally by Dr. Johnson • 2024-01-27
JOIN EDIT

Gradient Descent Notes
1.2 MB • Originally by Maria R. • 2024-01-28
JOIN EDIT

Resources Pages (Document)



Module Documents

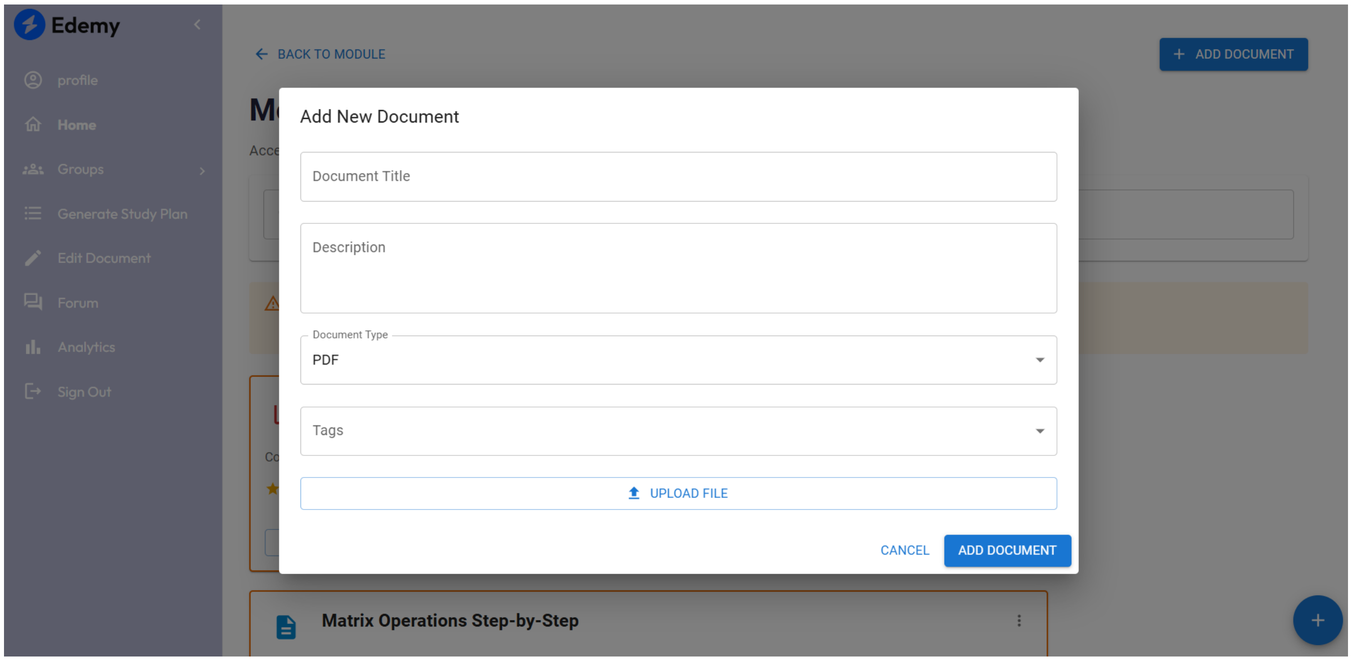
Access all documents, notes, and resources for this module. Rate, comment, and collaborate with your peers.

Search documents...

Documents Currently Being Edited (2)
These documents are currently being edited by other users.

Linear Algebra Fundamentals
Comprehensive guide covering vectors, matrices, eigenvalues, and linear transformations with practical examples and exercises.
★★★★☆ (23) 34 12
VIEW QUERY

Matrix Operations Step-by-Step


Edemy

profile
Home
Groups
Generate Study Plan
Edit Document
Forum
Analytics
Sign Out

BACK TO MODULE

ADD DOCUMENT

Add New Document

Document Title

Description

Document Type: PDF

Tags

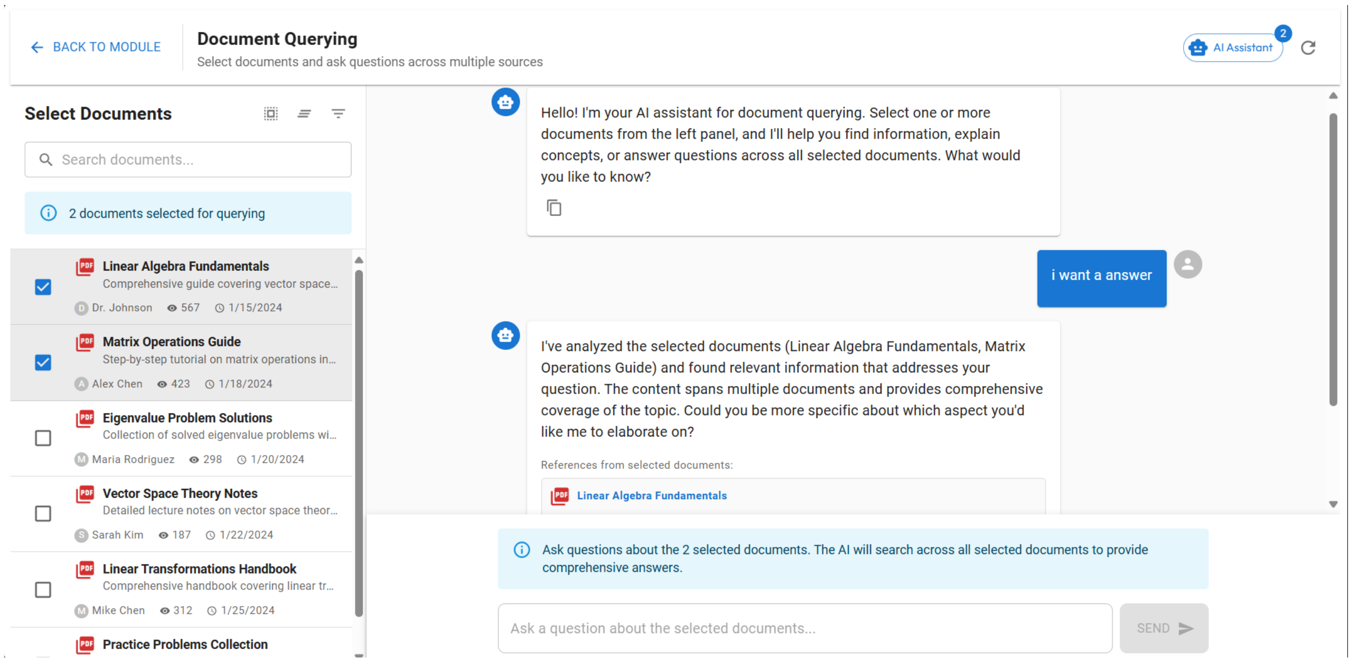
UPLOAD FILE

CANCEL
ADD DOCUMENT

Matrix Operations Step-by-Step

Document Querying Feature:

- Users can select any uploaded resource and ask questions about its content.
- The interface displays AI-generated answers directly linked to the document.
- Users can view past queries and answers for reference.
- Supports submitting new questions in real-time or asynchronously.


BACK TO MODULE

Document Querying

Select documents and ask questions across multiple sources

AI Assistant 2

Select Documents

Search documents...

2 documents selected for querying

[✓] Linear Algebra Fundamentals
Comprehensive guide covering vector space...
Dr. Johnson 567 1/15/2024

[✓] Matrix Operations Guide
Step-by-step tutorial on matrix operations in...
Alex Chen 423 1/18/2024

[ ] Eigenvalue Problem Solutions
Collection of solved eigenvalue problems wi...
Maria Rodriguez 298 1/20/2024

[ ] Vector Space Theory Notes
Detailed lecture notes on vector space theor...
Sarah Kim 187 1/22/2024

[ ] Linear Transformations Handbook
Comprehensive handbook covering linear tr...
Mike Chen 312 1/25/2024

Practice Problems Collection

Hello! I'm your AI assistant for document querying. Select one or more documents from the left panel, and I'll help you find information, explain concepts, or answer questions across all selected documents. What would you like to know?

i want a answer

I've analyzed the selected documents (Linear Algebra Fundamentals, Matrix Operations Guide) and found relevant information that addresses your question. The content spans multiple documents and provides comprehensive coverage of the topic. Could you be more specific about which aspect you'd like me to elaborate on?

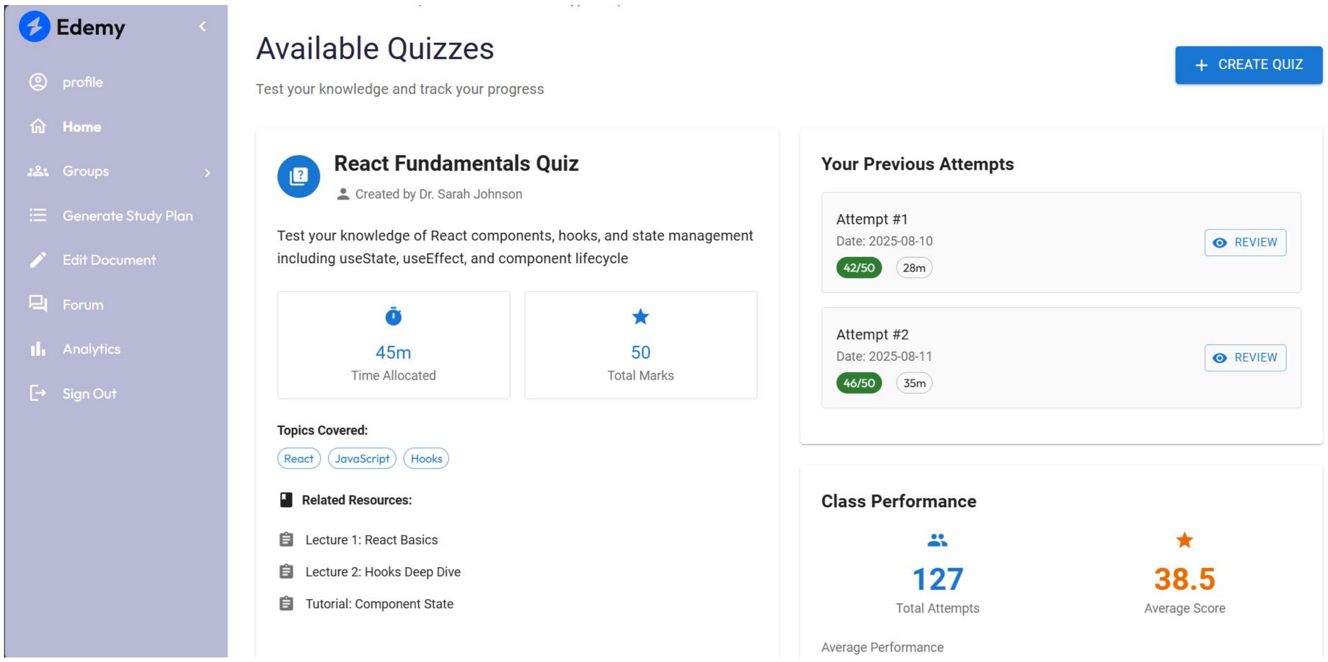
References from selected documents:
Linear Algebra Fundamentals

Ask questions about the 2 selected documents. The AI will search across all selected documents to provide comprehensive answers.

Ask a question about the selected documents...

SEND

Quizzes & Assessments Pages



Available Quizzes

Test your knowledge and track your progress

React Fundamentals Quiz

Created by Dr. Sarah Johnson

Test your knowledge of React components, hooks, and state management including useState, useEffect, and component lifecycle

45m
Time Allocated

50
Total Marks

Topics Covered:

React JavaScript Hooks

Related Resources:

Lecture 1: React Basics

Lecture 2: Hooks Deep Dive

Tutorial: Component State

Your Previous Attempts

Attempt #1
Date: 2025-08-10
42/50 28m

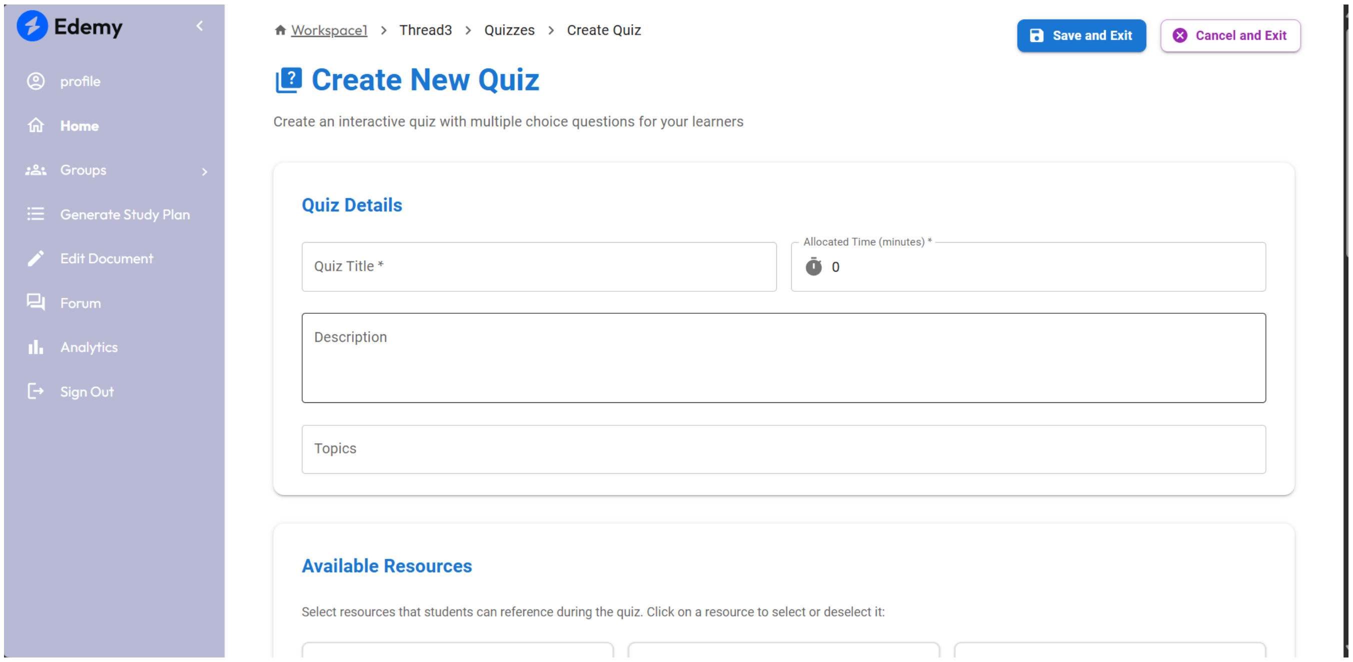
Attempt #2
Date: 2025-08-11
46/50 35m

Class Performance

127
Total Attempts

38.5
Average Score

Average Performance



Workspace1 > Thread3 > Quizzes > Create Quiz

Create New Quiz

Create an interactive quiz with multiple choice questions for your learners

Quiz Details

Quiz Title *

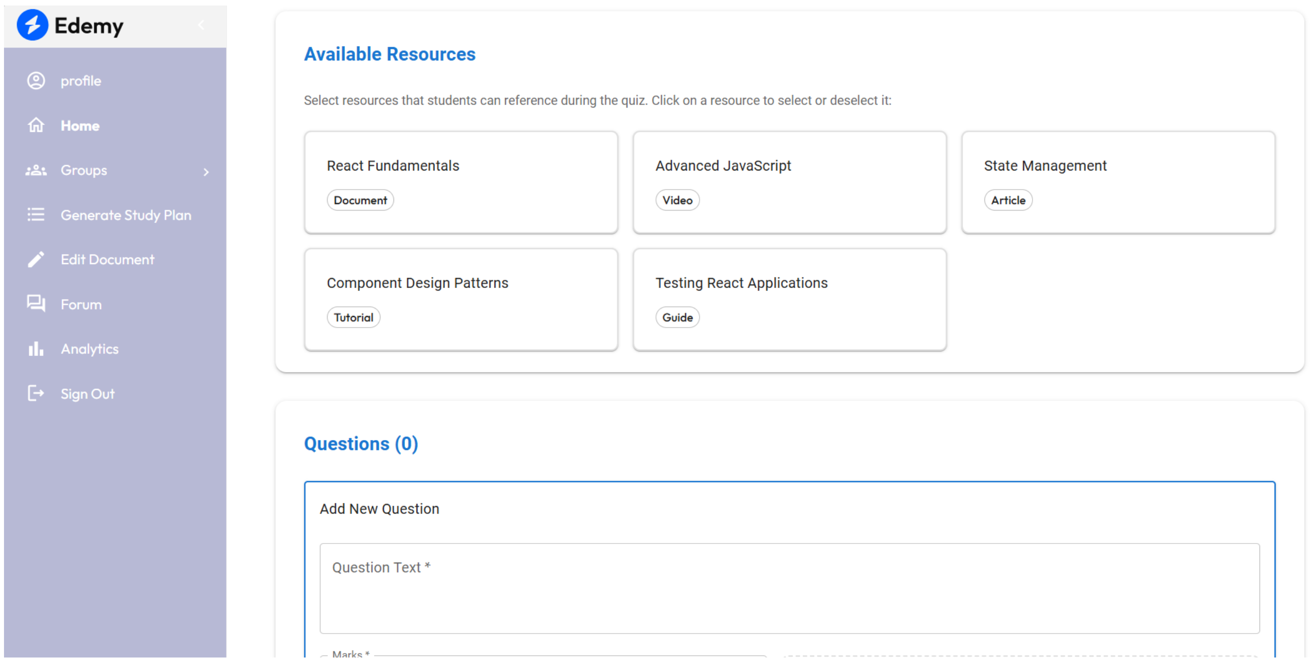
Allocated Time (minutes) * 0

Description

Topics

Available Resources

Select resources that students can reference during the quiz. Click on a resource to select or deselect it:



Available Resources

Select resources that students can reference during the quiz. Click on a resource to select or deselect it:

React Fundamentals — Document

Advanced JavaScript — Video

State Management — Article

Component Design Patterns — Tutorial

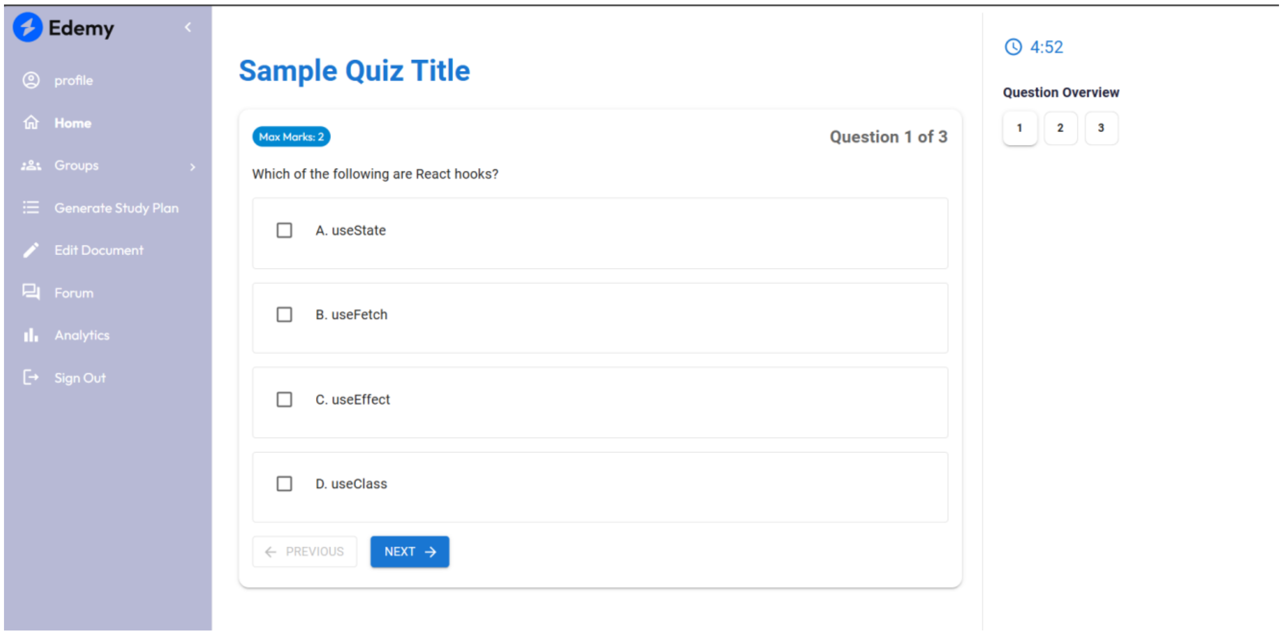
Testing React Applications — Guide

Questions (0)

Add New Question

Question Text *

Marks *



Sample Quiz Title

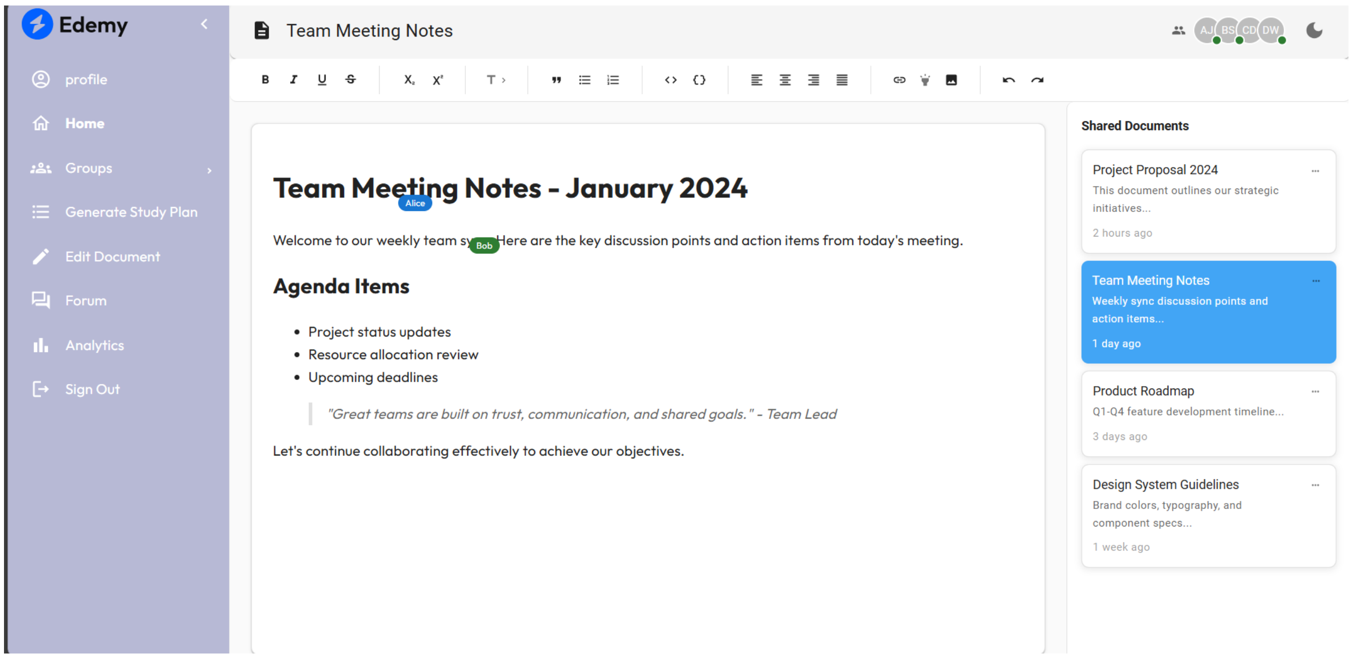
Max Marks: 2

Question 1 of 3

Which of the following are React hooks?

- [ ] A. useState
- [ ] B. useFetch
- [ ] C. useEffect
- [ ] D. useClass

Collaborative Document Editing Feature



Team Meeting Notes - January 2024

Welcome to our weekly team sync. Here are the key discussion points and action items from today's meeting.

Agenda Items

- Project status updates
- Resource allocation review
- Upcoming deadlines

> "Great teams are built on trust, communication, and shared goals." - Team Lead

Let's continue collaborating effectively to achieve our objectives.

Shared Documents

Project Proposal 2024
This document outlines our strategic initiatives...
2 hours ago

Team Meeting Notes
Weekly sync discussion points and action items...
1 day ago

Product Roadmap
Q1-Q4 feature development timeline...
3 days ago

Design System Guidelines
Brand colors, typography, and component specs...
1 week ago

3.9.2 Hardware Interfaces

3.9.2.1Client-Side Hardware

Minimum Requirements:

- **RAM:** 4 GB or higher
- **Processor:** Intel i3 or equivalent
- **Storage:** At least 2 GB free for caching and temporary storage
- **Display:** Minimum resolution 1366x768 for proper UI rendering
- **Network:** Internet connection with minimum 5 Mbps for real-time collaboration and document querying

Expected Behavior:

- The software shall run smoothly on standard desktops, laptops, and tablets meeting the above specifications.
- Real-time collaboration (chat, document editing) and AI-powered document querying must function without noticeable lag.

3.9.2.2 Server-Side Hardware

Minimum Requirements:

- **CPU:** 4 cores, 2.5 GHz or higher
- **RAM:** 16 GB or higher
- **Storage:** SSD 500 GB minimum, scalable with cloud storage
- **Network:** High-speed internet for API calls, WebSockets, and real-time data synchronization

Expected Behavior:

- The backend must handle multiple concurrent users, including real-time interactions, resource queries, and document querying AI processing.
- The server must maintain high availability and reliability under peak loads.

3.9.2.3 Peripheral Requirements

- No specialized peripherals are required.
- standard input devices (keyboard, mouse, touchscreen) and output devices (monitor, speakers) are sufficient.
- Users accessing document querying or multimedia resources should have audio support for any video or tutorial content.

3.9.3 Software Interfaces

3.9.3.1 Server Interfaces

Connection Requirements:

- The frontend (React.js) connects with the Node.js/Nest.js backend via RESTful API endpoints.
- Real-time features (chat, collaborative document editing) use WebSocket connections.

API Endpoints:

- **User Management:** /api/users, /api/login, /api/register, /api/profile
- **Group Management:** /api/groups, /api/groups/create, /api/groups/join
- **Resource Management:** /api/resources, /api/resources/upload, /api/resources/query
- **Quiz & Assessment Management:** /api/quizzes, /api/quizzes/submit, /api/quizzes/results
- **Document Querying:** /api/documents/query – processes user questions about resources and returns AI-generated answers.

Data Handling:

- All data transmitted securely using **HTTPS protocols**.

- Request and response data formatted in **JSON**.
- Input validation and error handling implemented on the server-side to ensure data integrity.

3.9.2.2 Third-Party Integrations

Supabase & Firebase:

- Interfaces for authentication, database operations, and real-time updates.
- **Endpoints:** Supabase REST API endpoints for CRUD operations; Firebase SDKs for authentication and notifications.
- **Security:** Secure access using API keys and proper authentication mechanisms.

Document Querying AI Service:

- Interfaces with backend via REST API.
- **Endpoints:** /api/documents/query
- **Functionality:** Processes user queries on uploaded documents and returns relevant answers.
- **Security & Data Handling:** Ensures no sensitive user data is exposed; uses encrypted transmission.

Study Plan Generation

- **Interfaces**: Connects to OpenAI's API for intelligent content generation.
- **Endpoints**: /api/studyplan/generate invokes OpenAI API with user-specific parameters such as subject, topics, and deadlines.
- **Functionality**: Generates **personalized study plans** including schedules, topic breakdowns, and recommendations.
- **Security & Data Handling**: API keys stored securely; requests and responses transmitted over HTTPS; sensitive academic data anonymized before sending to OpenAI.

3.9.2.3 Authentication and Authorization

JWT Tokens:

- Usage of access and refresh tokens for secure session management.

Role-Based Access Control (RBAC):

- Ensures only authorized users can access specific features:
    - Students: view/join groups, query documents, take quizzes
    - Group Admins: manage groups, resources, approve requests
    - Educators: manage quizzes, generate study plans

3.9.4 Communications Interfaces

3.9.4.1 Protocols

The application supports communication protocols for data exchange:

HTTP/HTTPS:

- Asynchronous HTTP requests using REST APIs over the internet for frontend-backend communication.
- Used for user authentication, resource management, quizzes, and document querying.

WebSocket:

- Used for real-time updates, such as:
  - Chat messages
  - Collaborative document editing
  - Live notifications for group activity
  - Instant feedback for quizzes

3.9.4.2 Ports

Frontend:

- Typically served over port **80 (HTTP)** or **443 (HTTPS)**.

Backend API:

- Accessible over a specified port (e.g., **3000 for Nest.js**) for internal and external communication.

WebSocket Connections:

- Real-time features (chat, collaborative editing, document querying) may use a dedicated port (**3001**) for persistent connections.

3.10 Database Requirements

3.10.1 Database Technology

- The system shall use **Supabase (PostgreSQL-based)** as the primary database for structured data storage.
- **Firebase Realtime Database/Firestore** may be used for real-time synchronization, notifications, and caching where required.

3.10.2 Data Storage Requirements

- **User Data:** Profiles, authentication credentials (encrypted), learning preferences, progress analytics.
- **Group Data:** Study groups, membership lists, roles, permissions, group activity logs.
- **Resource Data:** Uploaded files (links to cloud storage), metadata, tags, categories, and document querying indexes.

- **Quiz & Assessment Data:** Quiz definitions, questions, answers, results, and performance analytics.
- **Forum:** Messages, reactions, images.

3.10.3 Data Security and Privacy

All sensitive data (e.g., passwords, tokens) shall be encrypted using industry-standard algorithms (e.g., AES-256, bcrypt).Database access shall be controlled via Role-Based Access Control (RBAC).Strict separation between user data and resource data complies with privacy and compliance standards.Backups must be automated daily and securely stored.

3.10.4 Performance and Scalability

The database shall support concurrent access by up to 1,000 users during peak times.Indexing and query optimization shall be implemented to support fast search and document querying.Horizontal scaling shall be supported for increased storage and workload.

3.10.5 Backup and Recovery

The database management system is configured to perform automated backups at regular intervals, with full backups scheduled daily during off-peak hours to minimize performance impact. Incremental backups are also taken periodically throughout the day to capture changes since the last full backup, reducing potential data loss in case of a failure. Backup files are encrypted before being securely transferred to off-site cloud storage solutions such as AWS S3, which provides high durability and availability. A comprehensive backup retention policy is implemented, ensuring that backups are stored for an appropriate duration to comply with data retention regulations while optimizing storage costs.

To mitigate the risk of prolonged downtime due to catastrophic failures such as hardware malfunctions, cyber-attacks, or natural disasters, a robust disaster recovery plan is in place. This plan includes the maintenance of a secondary,geographically distributed database server that acts as a hot standby. This secondary server is continuously synchronized with the primary database to ensure real-time replication of data. In the event of a primary server failure, automated failover mechanisms are triggered, redirecting traffic to the secondary server with minimal disruption. Regular disaster recovery drills are conducted to test and refine the recovery procedures, ensuring that the system can recover within the defined Recovery Time Objective (RTO) and Recovery Point Objective (RPO). These detailed database requirements ensure that the theatre reservation system not only operates efficiently and securely but also adheres to best practices in data management, offering reliability, scalability, and compliance with industry standards.

3.10.6 Data Integrity

Strong **referential integrity** enforced via foreign keys in Supabase/PostgreSQL. Validation rules applied at both application and database levels.Duplicate prevention mechanisms (e.g., unique constraints on usernames, group IDs).

3.11 Licensing, Legal, Copyright, and Other Notices

3.11.1 Licensing Enforcement

- **Software Licenses**: All third-party components and libraries used in the platform (React.js, Material UI, Nest.js, Supabase, Firebase, WebSockets, Python, TypeScript, OpenCV, OpenAI, etc.) must comply with their respective open-source or commercial licenses (MIT, Apache 2.0, GPL). Redistribution or modification must respect the original license terms.

3.11.2 Legal Disclaimers

- **Liability Disclaimer**: The platform shall include a liability disclaimer to protect developers/owners from legal responsibility in cases of system downtime, data loss, unauthorized access, or incorrect AI-generated study plan/document query results.
- **Data Protection Compliance**: The system must comply with GDPR, CCPA, and other applicable data protection regulations, ensuring user consent for data collection, secure storage, and transparent handling of personal data and learning resources.

3.11.3 Copyright Notices

- **Original Code**: All original code and intellectual property developed for the Collaborative Learning Platform are copyrighted by the project team/organization. Proper copyright notices must be included in the source code and documentation.
- **User-Generated Content**: Students and instructors retain ownership of resources (documents, notes, videos) they upload but grant the platform a non-exclusive license to store, display, and share content within study groups.
- **Third-Party Components**: Copyright notices for third-party dependencies (React.js, Firebase, Supabase, Stripe, OpenCV, OpenAI, etc.) must be preserved in their original form as per their respective licenses.

3.11.4 Trademark and Logo Compliance

- **Branding**: Proper use of third-party branding (e.g., “Powered by Stripe” for payments, “Supabase” for database services, “OpenCV” for AI/ML functionalities) must follow official trademark usage guidelines. Unauthorized use of third-party logos without permission is strictly prohibited.
- **Platform Identity**: The Collaborative Learning Platform’s own name, logo, and branding elements are proprietary and must not be used or modified without explicit permission from the project owners.
- **Compliance Monitoring**: Regular internal reviews must be conducted to ensure compliance with all licensing, copyright, and trademark requirements.

3.12 Applicable Standards

3.12.1 Legal and Regulatory Standards

- **3.12.1.1 General Data Protection Regulation (GDPR):** The system must comply with GDPR requirements for protecting the personal data of students and instructors in the EU. This includes obtaining consent for data collection, allowing data portability, and providing the right to deletion upon request.
- **3.12.1.2 California Consumer Privacy Act (CCPA):** The platform must comply with CCPA requirements for users in the U.S., ensuring transparency in data collection, providing opt-out options, and respecting user privacy rights.
- **3.12.1.3 Children’s Online Privacy Protection Act (COPPA):** If the platform is used by minors under 13, it must include parental consent mechanisms for account creation and data usage.

3.12.2 Industry Standards

- **3.12.2.1 ISO/IEC 27001:** The platform should follow ISO/IEC 27001 standards for information security management, ensuring secure handling of learning resources, personal profiles, and communication data.
- **3.12.2.2 ISO/IEC 25010:** The platform must comply with the ISO/IEC 25010 software quality model to ensure quality attributes such as functionality, reliability, usability, efficiency, maintainability, and portability.
- **3.12.2.3 OWASP Top 10:** The system must follow OWASP Top 10 security standards to mitigate risks such as SQL injection, XSS, broken authentication, and insecure deserialization.

3.12.3 Usability Standards

- **3.12.3.1 ISO 9241-210:** The platform's user interface must align with ISO 9241-210 standards for human-centered design, ensuring intuitive navigation, collaborative learning ease, and efficiency in task completion.
- **3.12.3.2 Web Content Accessibility Guidelines (WCAG 2.1):** The platform must comply with WCAG 2.1 Level AA to ensure accessibility for students with disabilities, including text alternatives for non-text content, full keyboard navigation, and sufficient color contrast.

3.12.4 Interoperability Standards

- **3.12.4.1 RESTful API Standards:** APIs must comply with RESTful standards, including standard HTTP methods (GET, POST, PUT, DELETE), proper error handling, and clear documentation for integration with third-party tools such as document querying AI or external learning management systems.
- **3.12.4.2 WebSocket Protocol:** The platform must support WebSocket standards for real-time features such as group chats, collaborative document editing, and live quiz participation.
- **3.12.4.3 OAuth 2.0 and JWT:** The system must implement OAuth 2.0 and JWT (JSON Web Tokens) for secure authentication and authorization, ensuring only authorized users can access sensitive features like study group management or payment services.

3.12.5 Internationalization Standards

- **3.12.5.1 Unicode Standard (UTF-8):** The platform must use UTF-8 encoding to support multiple languages in study plans, group chats, and uploaded resources.
- **3.12.5.2 ISO 8601:** All date and time formats must follow ISO 8601 (YYYY-MM-DD format) to ensure consistency across international users.
- **3.12.5.3 Multi-Language Support:** The system should be designed for easy localization (L10n) and internationalization (i18n), enabling support for future translation of the interface and study content.

3.12.6 Operating System Compliance

- **3.12.6.1 POSIX Compliance:** The backend services should adhere to POSIX standards for compatibility across different UNIX-based systems such as Linux and macOS.
- **3.12.6.2 Cross-Platform Support:** The application must be accessible across Windows, macOS, Linux, and mobile operating systems (Android and iOS), ensuring wide usability among students.
- **3.12.6.3 Browser Compatibility:** The platform must support all major browsers (Chrome, Edge, Firefox, Safari) with consistent UI rendering and functionality.

4.Supporting Information

The following supporting information is provided to enhance the usability of this SRS. These materials include references to standards, tools, algorithms, and similar works in collaborative learning and personalized education systems.

4.1 Table of Contents

The full Table of Contents is provided at the beginning of this document for quick navigation across sections such as Functional Requirements, Interfaces, Performance, Security, and Standards Compliance.

4.2 Index

The Index section offers a quick reference guide to key terms, concepts, and components mentioned in the SRS. It includes cross-references to where these terms are discussed in detail within the document, aiding users in efficiently navigating the content.

4.3 Reference

This section lists all the references cited throughout the SRS, ensuring that all standards, tools, and technologies mentioned are properly documented for further reading. The references are provided in a standard format to maintain consistency and accuracy.

The following references were used to prepare this SRS document:

[1] ISO/IEC 25010:2011, *"Systems and Software Engineering — Systems and Software Quality Requirements and Evaluation (SQuaRE) — System and Software Quality Models,"* ISO, 2011.

[2] Web Content Accessibility Guidelines (WCAG) 2.1, W3C, 2018. [Online]. Available: https://www.w3.org/TR/WCAG21/. (Accessed: Aug. 23, 2025).

[3] General Data Protection Regulation (GDPR), European Union, 2018. [Online]. Available: https://gdpr-info.eu/. (Accessed: Aug. 23, 2025).

[4] React, *"A JavaScript Library for Building User Interfaces."* [Online]. Available: https://reactjs.org/.

[5] NestJS, *"A Progressive Node.js Framework for Building Efficient Server-Side Applications."* [Online]. Available: https://nestjs.com/.

[6] FastAPI, *"A Modern, Fast Web Framework for Building APIs with Python 3.7+."* [Online]. Available: https://fastapi.tiangolo.com/.

[7] Tailwind CSS, *"A Utility-First CSS Framework for Rapidly Building Custom Designs."* [Online]. Available: https://tailwindcss.com/.

[8] Supabase, *"The Open-Source Firebase Alternative."* [Online]. Available: https://supabase.com/.

[9] Firebase, *"Firebase Storage — Store and Serve User-Generated Content."* [Online]. Available: https://firebase.google.com/products/storage.

[10] I. Fette and A. Melnikov, *"The WebSocket Protocol,"* RFC 6455, Dec. 2011. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc6455.

[11] M. Jones, J. Bradley, and N. Sakimura, *"JSON Web Token (JWT),"* RFC 7519, May 2015. [Online]. Available: https://datatracker.ietf.org/doc/html/rfc7519.

[12] OpenCV, *"Open Source Computer Vision Library."* [Online]. Available: https://opencv.org/.

[13] OpenAI, *"OpenAI API Documentation."* [Online]. Available: https://platform.openai.com/docs.