**Software Design Specifications**

<ProjectHUB>

# 

# ProjectHub: Collaborate, Contribute, Create

Version: 1.0

Prepared by: Group-32  
Hriday Macha (SE22UARI056), Aditya Shastry (SE22UARI008), Grisha Saini (SE22UARI218), Medha Potru (SE22UARI093)

## Document Information

|  |  |
| --- | --- |
| Title | ProjectHub Design Specification |
| Project Manager | Group-32 |
| Document Version No | 1.0 |
| Document Version Date | 03-03-2025 |
| Prepared By | Group-32 |
| Preparation Date | 03-03-2025 |

## Version History

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Ver. No | Ver. Date | Revised By | Description | Filename |
| 1.0 | 03-03-2025 | Group-32 | Initial version | ProjectHub\_SDD\_v1.0.doc |

**1. INTRODUCTION**  
This Software Design Specification outlines the structural and architectural design of ProjectHub, a web-based platform for student project collaboration. It details system components, module interactions, and implementation strategies based on the SRS. The document ensures alignment between functional goals and technical solutions. Key terminologies, acronyms, and references are also defined. This serves as a blueprint for developers, stakeholders, and QA teams.  
  
**1.1 Purpose**  
This Software Design Specification describes the architecture and components of the ProjectHub web platform. It is intended for use by developers, project managers, and university stakeholders for implementation and maintenance. It supports the SRS developed for ProjectHub and maps out the detailed design components.  
  
**1.2 Scope**  
ProjectHub is a web-based collaboration platform for university students to discover, apply for, and participate in projects. It includes features like AI-based recommendations, mentor collaboration, and productivity tools.  
 **1.3 Definitions, Acronyms, and Abbreviations**  
**AI (Artificial Intelligence)**:  
Refers to the simulation of human intelligence processes by machines, especially computer systems. In ProjectHub, AI is used to analyze user data and preferences to recommend the most relevant projects to students.

**ML (Machine Learning)**:  
A subset of AI focused on enabling systems to learn and improve from experience without being explicitly programmed. ProjectHub utilizes ML algorithms to enhance the accuracy of project matching and recommendation features over time.

**SSO (Single Sign-On)**:  
An authentication method that allows a user to log in with a single set of credentials to multiple applications. ProjectHub supports SSO using university email credentials, simplifying access for students and faculty.

**OAuth (Open Authorization Protocol)**:  
A secure authorization framework that allows applications to access user information from third-party services without exposing user passwords. In ProjectHub, OAuth is used to enable login via Google or university credentials while maintaining user data security.

**SOW (Statement of Work)**:  
A formal document that outlines the specific tasks, deliverables, timelines, and responsibilities for a project. The SOW for ProjectHub was prepared by Group-32 and forms the foundation for both the SRS and this Software Design Specification.

**1.4 References**  
- Software Requirements Specification for ProjectHub (Group-32)  
- IEEE SRS and SDD Guidelines

**2 Use Case View**

This section outlines the primary use cases of the ProjectHub platform based on the core functionalities outlined in the requirements document and use case diagram. These use cases form the foundation for the system’s architectural and design decisions. Each use case corresponds to major system features involving students, mentors, administrators, and the AI-based recommendation engine.

**2.1** **Use Case**

L**og in / Log out**: Users authenticate via Google or University SSO to access the platform securely.

**Project Discovery**: Students explore and apply for projects based on academic year and skills. AI recommendations enhance matching.

**Profile & Settings**: Users manage personal details and preferences; mentors update mentor profiles.

**Project Management**: Students track tasks, deadlines, and milestones for active projects.

**Community Collaboration**: Enables project group creation, group chat, and mentor-student interactions.

**To-Do & Task Tracking**: Students organize personal tasks and maintain productivity.

**Admin Project Review**: Administrators review, approve, and manage project listings.

**AI-Based Recommendations**: The AI engine suggests suitable projects by analyzing user profiles.

**3. Design Overview**

This section outlines the high-level design structure of ProjectHub, emphasizing modularity, scalability, and maintainability. It builds upon the SRS by detailing system decomposition, dependencies, and design principles used to achieve the desired functionality.

**3.1 Design Goals and Constraints**

* Ensure high scalability to support a large number of concurrent users.
* Use modular architecture (frontend, backend, and AI engine) for independent development and testing.
* Implement secure authentication and authorization via OAuth.
* Integrate cloud infrastructure (e.g., AWS/GCP) for hosting and auto-scaling.
* Utilize modern development tools like ReactJS, NodeJS, MongoDB.
* Adhere to project timelines and team division, ensuring each module is assigned to specific team members.

**3.2 Design Assumptions**

* All users (students, mentors, admins) will authenticate using institutional credentials.
* AI-based project recommendations depend on sufficient and accurate user profile data.
* Internet connectivity is assumed to be available for accessing the cloud-hosted system.
* External APIs (Google OAuth, LinkedIn) are operational and accessible.
* Mentors will actively participate in community collaboration and feedback processes.

**3.3 Significant Design Packages**

* **Authentication Package**: Handles login, OAuth flow, and user session management.
* **Project Matching Engine**: AI module that recommends suitable projects using user profiles and historical data.
* **User Profile Management**: Manages user data, roles, preferences, and skills.
* **Team and Collaboration Tools**: Enables group chat, task assignment, and community discussions.
* **Mentorship and Feedback Module**: Facilitates mentor-student interactions and feedback exchange.
* **Admin Management**: For monitoring system usage, managing flagged content, and adding projects.

## 3.4 Dependent External Interfaces

The following table lists the external systems and interfaces that **ProjectHub** depends on for core functionalities like authentication, data analysis, and user profile enhancement.

| **External Application and Interface Name** | **Module Using the Interface** | **Functionality / Description** |
| --- | --- | --- |
| Google OAuth API | Authentication Module | Used for Single Sign-On (SSO) to authenticate users securely using institutional Google accounts. |
| LinkedIn API | User Profile Management | Pulls user academic and project-related data to enhance profile accuracy and recommendations. |
| AI-Based Recommendation Engine (ML API) | Project Matching Engine | Queries user profile data and historical records to suggest suitable projects. |
| Cloud Database (MongoDB Atlas) | Backend Modules | Stores user profiles, projects, tasks, and discussion threads in a scalable document database. |
| SMTP Mail Server (e.g., Gmail API) | Notification System | Sends project updates, collaboration invites, and task reminders |

## 3.5 Implemented Application External Interfaces (and SOA Web Services)

The following interfaces are made available by **ProjectHub** for integration with external applications or for public/internal API exposure.

| **Interface Name** | **Module Implementing the Interface** | **Functionality / Description** |
| --- | --- | --- |
| /api/login | Authentication Module | Provides login/logout functionality using OAuth2.0. |
| /api/user/profile | User Profile Management Module | Allows retrieval and update of user profile and settings. |
| /api/projects/recommend | Project Matching Engine | Accepts user profile input and returns a list of recommended projects. |
| /api/collab/chat | Collaboration Module | Enables group chat and project discussion access. |
| /api/admin/review | Admin Module | Allows administrators to review, verify, and add project entries. |

**4. Logical View**

This section outlines the system's layered design, showing how key modules interact to implement major functionalities.

**4.1 Design Model**

ProjectHub follows a modular MVC structure with the following layers:

* **Frontend (ReactJS)**:  
  Handles user interfaces and interactions. Key components include LoginPage, Dashboard, ProjectList, ProfileEditor, and ChatBox.
* **Backend (Node.js + Express)**:  
  Manages routing, business logic, and API integration. Core services include AuthService, UserController, ProjectController, and MentorController.
* **AI Engine (Flask)**:  
  Provides project recommendations using user profiles. Includes modules like RecommendationService and FeatureExtractor.
* **Database (MongoDB)**:  
  Stores persistent data such as users, projects, applications, and chats.

**4.2 Use Case Realization**

* **Login**:  
  Frontend collects credentials → AuthService validates via OAuth → dashboard is loaded on success.
* **Project Discovery**:  
  ProjectList fetches projects → backend returns results → AI engine provides personalized recommendations.
* **Profile Management**:  
  User edits profile → UserController updates database → confirmation shown.
* **Mentor Collaboration**:  
  Chat interface connects users → messages stored and retrieved from the chats collection.
* **Task Tracking**:  
  Users manage to-dos via UI → updates handled by TaskManager backend module.

**5. Data View**

This section presents the persistent data structure of the ProjectHub system, outlining core entities and their relationships, along with the data model used for storage and retrieval.

**5.1 Domain Model**

ProjectHub uses a document-based domain model with the following primary entities:

* **User**: Represents students, mentors, or admins. Includes profile info, roles, and skills.
* **Project**: Academic or industry projects with details like title, description, required skills, and mentor.
* **Team**: Group of users working on a project, with roles and tasks assigned.
* **Application**: Tracks student applications to projects and their statuses.
* **Task**: Project-specific or personal tasks with deadlines and ownership.
* **Chat**: Messages exchanged within teams or mentor-student discussions.

**Relationships:**

* A **User** can be linked to multiple **Projects** through **Applications** or **Teams**.
* A **Project** is associated with one **Mentor** and may have a **Team**.
* Each **Task** is assigned to a **User** and linked to a **Project**.
* **Chat** entries are tied to **Users** and **Projects** or **Teams**.

**5.2 Data Model (Persistent Data View)**

The data is stored in MongoDB as collections, using JSON-like documents:

* **users**:  
  { id, name, email, role, skills, profileDetails }
* **projects**:  
  { id, title, description, requiredSkills, mentorId, yearLevel }
* **applications**:  
  { userId, projectId, status, appliedOn }
* **teams**:  
  { projectId, members: [userIds], createdOn }
* **tasks**:  
  { projectId, assignedTo, description, status, deadline }
* **chats**:  
  { projectId/teamId, senderId, message, timestamp }

**6. Exception Handling**

ProjectHub handles both system and application-level exceptions to ensure reliability and user guidance. All exceptions are logged using a centralized logging system (e.g., Winston for Node.js), and users are notified via UI alerts

| **Exception** | **Description** | **Trigger** | **Handling & Follow-up** |
| --- | --- | --- | --- |
| **InvalidCredentialsError** | Thrown when user authentication fails | Incorrect login details | Display error message; allow retry |
| **DatabaseConnectionError** | Failure to connect to MongoDB | Server downtime or wrong URI | Retry connection; log issue for admin |
| **ProjectNotFoundError** | Queried project ID doesn’t exist | Invalid project link or deletion | Redirect to home with message |
| **UnauthorizedAccessError** | Role-based permission error | User accessing restricted features | Redirect to login or error page |
| **APIFetchError** | External API call failure (e.g., recommendation engine) | AI or OAuth service unavailable | Retry logic and admin alert |
| **FileUploadError** | Invalid or failed document upload | Unsupported format or size | Show alert; allow re-upload |
| **7. Configurable Parameters**  ProjectHub allows the following parameters to be customized to adapt to various usage environments. These settings can be adjusted via environment variables or a config file.   | Configuration Parameter Name | Definition and Usage | Dynamic? | | --- | --- | --- | | RECOMMENDER\_THRESHOLD | Minimum match confidence for AI-based suggestions | Yes | | SESSION\_TIMEOUT | Duration of inactivity before user is auto-logged out (default: 30 mins) | Yes | | MAX\_TEAM\_SIZE | Maximum number of students allowed in a project team | Yes | | ENABLE\_PUBLIC\_CHAT | Toggle for enabling/disabling public community chat features | No | | DEFAULT\_USER\_ROLE | Default role assigned during new user registration (e.g., student) | Yes | | MAX\_FILE\_UPLOAD\_SIZE | Limit for uploaded document sizes in MB | No | |  |  |  |

**8. QUALITY OF SERVICE**  
  
8.1 Availability  
The system is designed for high availability, leveraging modern cloud infrastructure such as AWS or Google Cloud Platform. By utilizing auto-scaling groups and load balancers, ProjectHub ensures uninterrupted access and seamless performance during traffic spikes. The architecture guarantees a minimum uptime of 99.9%, supported by failover mechanisms and distributed services across multiple availability zones.  
  
8.2 Security and Authorization  
ProjectHub incorporates robust security measures to protect user data and ensure authorized access. The platform uses OAuth-based authentication through Google and University SSO, and enforces Role-Based Access Control (RBAC) to differentiate access levels for students, mentors, and administrators. All sensitive data is encrypted using AES-256, and regular security audits are performed to detect vulnerabilities.  
  
8.3 Load and Performance Implications  
The platform is optimized for scalability and performance under high load conditions. It can accommodate over 500 concurrent users while maintaining sub-2-second response times for AI-based project recommendations. Backend processes are distributed to handle large volumes of requests efficiently, and database queries are optimized for minimal latency.  
  
8.4 Monitoring and Control  
ProjectHub includes a comprehensive monitoring and control framework. All critical services and user activities are logged and tracked via a centralized dashboard accessible to administrators. The system emits real-time alerts for performance issues, security breaches, and API failures. Regular analytics and reports are generated to evaluate system health, allowing proactive intervention when necessary.