Converge

System Design

SOFTWARE ENGINEERING(IT-314)

Prof. Saurabh Tiwari

Group - 23

- 202301209- Gori Faran Firozbhai (Leader)
- 202301224- Prajapati Anushka Maheshbhai
- 202301244- Atharv Shah
- 202301213- Kalriya Prayag Jagdishbhai
- 202301251- Vasani Devarsh Chintanbhai
- 202301243- Vadaviya Chaitri Hareshkumar
- 202301212- Patel Ishti Upendra
- 202301211- Rathod Harita Rajeshbhai
- 202301228- Patel Naitikkumar Dilipbhai
- 202301242- Tejani Charvik Bipinbhai

System Design

System Design Approach

- We follow a Top-Down Design Approach for Converge, ensuring that the system is broken down from High-level functionality into well-structured modules.

Key Principle followed :

1. Start with major functionalities such as Authentication,Project Handling,Task Assignment,Chat System, Calendar integration and AI Chatbot.
2. Each function is further decomposed into subsystems.
3. Keep related things together and avoid unnecessary connections so the system stays easy to manage and update.
4. Prioritize the design goals : usability,scalability,security and real-time performance.
5. Each subsystem is designed with future extensibility in mind(e.g., adding video meetings in future).

Design Goals

1. High Maintainability : Each big feature is kept in its own Django app or React module. This makes development cleaner and changes easier.
2. Scalability : The system is built using stateless APIs, Redis, AWS S3, and scalable databases like MongoDB so it can easily handle more users.
3. Security & Reliability : JWT login, all services check inputs and keep audit logs.
4. Real-Time Performance : Fast updates,chat messages,notifications, and task-status changes.

Interface Design

Objective

To understand how Converge interacts with external entities by treating the system as a black box - focusing solely on inputs,outputs, and interfaces.

External Interfaces

User Interfaces

- Web App (Team Leader, Team Members)
- Mobile App (optional future extension)

Communication Interfaces

- Real-time Chat Server (WebSockets)
- Notification System (push + email)

AI Chatbot Interface

- Context-based query engine integrated inside each project workspace.

Calendar & Scheduling Interface

- Shared Calendar View
- Personal Calendar View

Storage / Integration Interfaces

- Database for structured data
- Cloud Storage for attachments

Inputs

- Login credentials
- Project creation details
- Task details (title, description, deadline, dependency)
- Chat messages (one to one and group)
- Uploaded files and documents
- Queries to AI chatbot

Outputs

- Authentication status
- Project overview dashboards
- Task progress updates
- Chat conversations & notifications
- Daily AI-generated summaries
- Exportable reports

Key Relationships

Team Leader ↔ System

- Create/update projects
- Assign tasks
- View progress dashboards

- Use analytics

Team Member ↔ System

- View assigned tasks
- Update task status (based on team leader approvals)
- Participate in chat
- Consult chatbot for project details

AI Chatbot ↔ System

- Fetch project information
- Generate summaries
- Assist users with queries

Subsystem Decomposition

We divide Converge into major subsystems based on high cohesion & minimal coupling.

1. Authentication & Access Control Subsystem

- Handles registration, login, role-based access.
- Stores user credentials securely.
- Issues session tokens.
- Manages permission rules (Team Leader vs Team Member).

2. User & Role Management Subsystem

- Manages user profile details.

- Maintains mapping of users to projects.
- Stores personal calendar data and availability information.

3. Project Management Subsystem

- Create, update, and archive projects.
- Add or remove team members.
- Store project metadata and milestones.

4. Task & Workflow Management Subsystem

- Create tasks with deadlines, priorities, dependencies.
- Update task status (“Pending Review”, “Completed”, etc.)
- Automatic reminders before deadlines.
- Visual dependency mapping for blocked tasks.
- Task approval by team leader.

5. Chat & Collaboration Subsystem

- Real-time group and one-to-one chat via WebSockets.
- File sharing within chat.
- Chat search, message pinning, thread creation.
- Auto-create chat groups based on assigned tasks.
- Daily summaries using AI.

6. AI Chatbot Subsystem

- NLP model to understand queries.

- Provides project-related info (tasks, deadlines, progress).
- Generates chat summaries.
- Suggests action items.

7. Calendar & Scheduling Subsystem

- Shared team calendar.
- Personal calendar with professional schedules.
- Free/busy visibility matrix.
- Meeting scheduling & reminders.

8. Notification Subsystem

- Deadline reminders.
- Task updates (assignment, approval, comments).
- Calendar alerts.
- Chat mentions & pinned messages.

9. Analytics & Reporting Subsystem

- Project progress tracking.
- Task completion stats.
- Exportable reports for team leaders.

Relationships Between Subsystems

- **Data Coupling:** Shared access to users, tasks, projects.
- **Message Coupling:** Chat messages, notifications, AI summaries.
- **Closed Layering:**
    - UI Layer → Application Layer → Data Layer

Architectural Design

We adopt a **Three-Layer Architecture**:

1. Presentation Layer (Frontend)

- Web App: React.js
- Real-time chat UI
- Project dashboards
- Calendar view
- AI chatbot interface

2. Application Layer (Backend)

Built with **Django/FastAPI** (your choice).

Includes services:

- Authentication Service
- Project Service

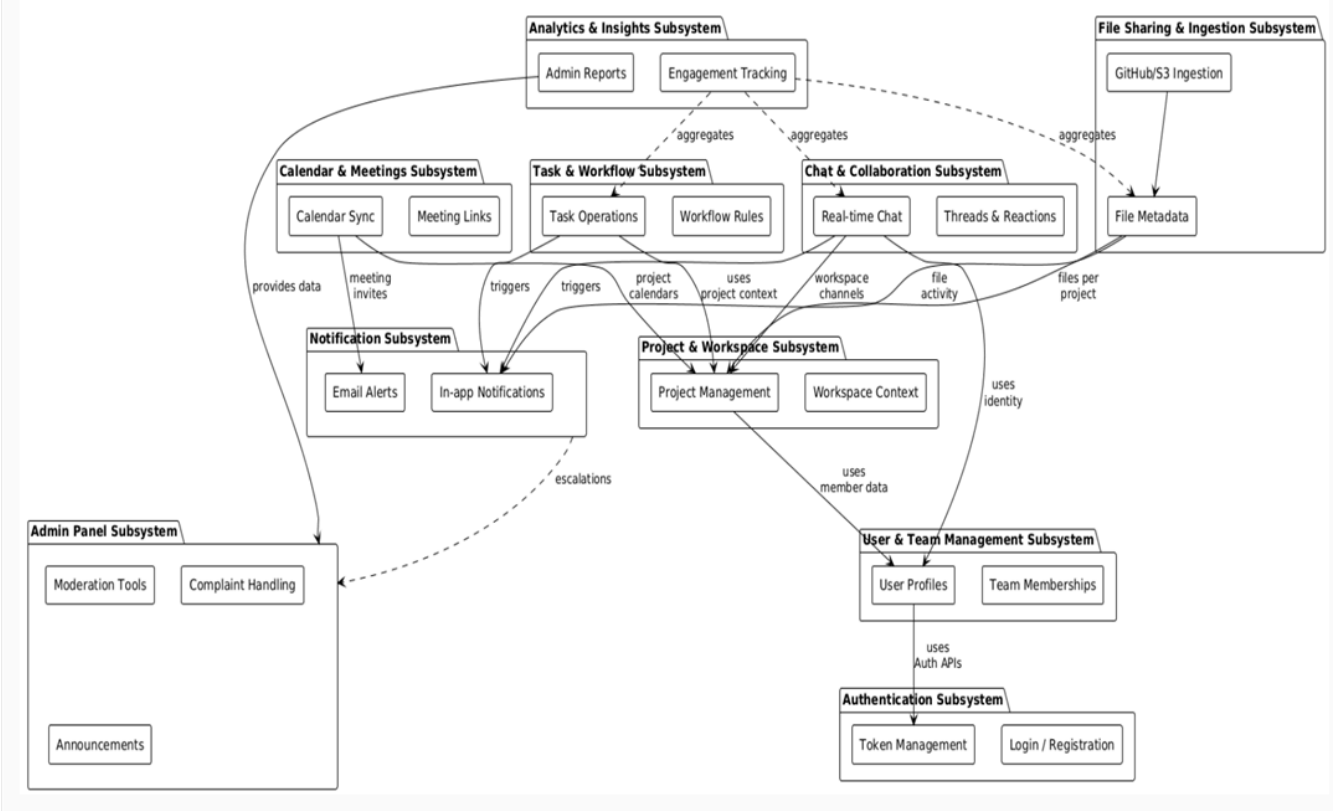
- Task Service
- Chat Service
- AI Chatbot Service
- Calendar Service
- Analytics Service
- Notification Service

Real-time interaction supported using **WebSockets**.

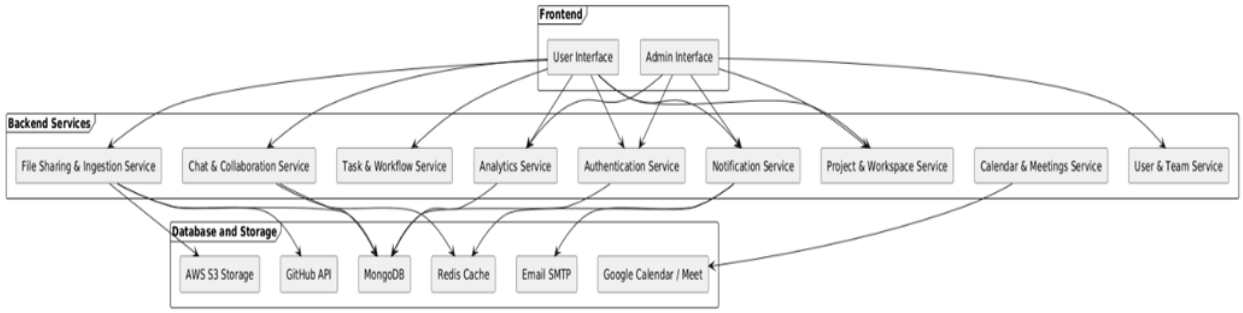
3. Data Layer

- **Primary Database:** MongoDB
- **Chat Database:** Redis (fast real-time streams)
- **Task Search:** Elasticsearch
- **File Storage:** AWS S3

Module Relation Diagram



- **Subsystem decomposition Diagram**



- **System context Diagram**