High Level Design Document

Introduction

This High Level Design (HLD) document outlines the architecture and core components for **EduNotes - Student Notes Management**. EduNotes is a backend API enabling students to upload, organize, and retrieve study notes, supporting file uploads and user authentication. The system is built using Node.js, Express, and MongoDB.

---

1. System Architecture Overview

Architecture Description:
EduNotes follows a modular, RESTful backend architecture. Clients interact with the API via HTTP(S). The backend handles authentication, file storage, and data management, interfacing with a MongoDB database.

Component	Role/Responsibility
API Gateway	Receives and routes HTTP requests
Authentication	Manages user registration, login, and JWT issuance
Notes Service	Handles CRUD operations for notes and file uploads
File Storage	Stores uploaded note files (local or cloud)
Database (MongoDB)	Persists user and notes metadata

---

2. Component Interactions

Sequence Step	Interaction Description
1	Client sends HTTP request to API Gateway (Express)
2	Authentication module validates JWT or processes login/signup
3	Notes Service processes note-related requests (CRUD, file upload)
4	File Storage module saves/retrieves files as needed
5	Database module reads/writes user and note metadata
6	API Gateway returns response to client

---

3. Data Flow Overview

Data Flow	Source	Destination	Description
User Registration/Login	Client	Auth Module	User credentials sent, JWT returned

Note Upload	Client	Notes Service	Note data and file uploaded
File Storage	Notes Service	File Storage	File saved and reference stored in DB
Metadata Persistence	Notes Service	MongoDB	Note/user metadata created/updated
Note Retrieval	Client	Notes Service	Notes fetched, files streamed if requested

---

4. Technology Stack

Layer/Component	Technology/Framework
Backend API	Node.js, Express
Database	MongoDB
Authentication	JWT (JSON Web Tokens)
File Storage	Local filesystem or S3 (extensible)
API Security	HTTPS, JWT-based auth

---

5. Scalability & Reliability

- **Scalability:**
  - Stateless API enables horizontal scaling.
  - MongoDB supports sharding for large datasets.
  - File storage can be migrated to scalable cloud storage (e.g., AWS S3).
- **Reliability & Security:**
  - JWT-based authentication secures endpoints.
  - Input validation and error handling at API layer.
  - HTTPS recommended for all client-server communication.
  - Regular database backups and monitoring.

End of Document