Low Level Design Document

Introduction

This Low Level Design (LLD) document outlines the backend architecture for **EduNotes - Student Notes Management**. The system provides RESTful APIs for students to upload, organize, and retrieve study notes, supporting file uploads and user authentication. The backend is built using Node.js, Express, and MongoDB.

---

1. System Components

Component	Description	Key Responsibilities
API Server	Express.js application	Route requests, handle middleware
Auth Module	JWT-based authentication	User login, registration, auth checks
Notes Module	Notes CRUD operations, file handling	Upload, retrieve, organize notes
Database Layer	MongoDB via Mongoose	Data persistence, schema enforcement
File Storage	Local or cloud storage (e.g., AWS S3)	Store/retrieve uploaded files

---

2. Class/Interface Overview

Class/Interface	Description	Key Methods/Attributes
`UserController`	Handles user endpoints	`register()` , `login()`
`NotesController`	Handles notes endpoints	`createNote()` , `getNotes()` , `deleteNote()`
`AuthMiddleware`	JWT validation	`authenticate()`
`Note` (Model)	Note schema/model	`title` , `content` , `fileUrl` , `userId` , `tags`
`User` (Model)	User schema/model	`email` , `passwordHash` , `name`

Relationships:

- Each `Note` references a `User` via `userId` .
- Controllers use Models for DB operations.

---

3. Data Structure Overview

Model	Fields
User	`_id` , `name` , `email` (unique), `passwordHash`
Note	`_id` , `title` , `content` , `fileUrl` , `tags` (array), `userId` (ref), `createdAt`

---

4. Algorithms / Logic

User Registration/Login:

```
register(email, password, name):
    if email exists in DB:
        return error
    hash password
    save user
    return JWT token

login(email, password):
    find user by email
    if password matches hash:
        return JWT token
    else:
        return error
```

Note Upload:

```
createNote(userId, title, content, file, tags):
    store file in storage
    save note with fileUrl and userId
    return note
```

Get Notes:

```
getNotes(userId, filters):
    query notes by userId and filters
    return notes list
```

---

5. Error Handling

Scenario	Handling Approach
Invalid credentials	Return 401 Unauthorized
JWT missing/invalid	Return 401 Unauthorized
File upload error	Return 400 Bad Request
DB operation fails	Return 500 Internal Server Error
Resource not found (note/user)	Return 404 Not Found
Validation error (input/schema)	Return 400 Bad Request

End of Document