MERN Stack Training
Company: Sensation Software Solutions
Student Name: Tajinder Kaur
Training Duration: 6 Months
Days: 86

Objective of the Day

The objective of **Day 86** was to thoroughly understand the requirements of the **Student Task Tracker application**, analyze real-world use cases, and design a scalable backend architecture. The focus of the day was on **requirement analysis**, **database schema planning**, and **initial backend setup** using Node.js, Express.js, and MongoDB.

Another important goal was to gain conceptual clarity on how **task management systems** are designed in professional environments and how file uploads (task-related images) are handled securely in backend systems.

Requirement Analysis and Project Understanding

The day began with a detailed discussion led by the trainer about the **purpose of the Student Task Tracker system**. This project was designed to simulate a real-world academic or organizational task management tool where students can manage daily tasks, track deadlines, and monitor task progress visually.

Key functional requirements discussed were:

- Students should be able to **create new tasks**
- Each task should have a **status** (Pending, In-Progress, Completed)
- Tasks may include an **optional image** as proof or reference
- Tasks should be **editable and deletable**
- Users should be able to **filter tasks by status**
- All tasks should be stored securely in a database

The trainer emphasized that this project closely reflects industry-level CRUD systems used in productivity tools.

Backend Technology Stack Planning

After understanding the requirements, the backend technology stack was finalized:

- **Node.js** – JavaScript runtime for backend execution
- **Express.js** – Lightweight framework for building REST APIs
- **MongoDB** – NoSQL database for storing task data
- **Mongoose** – ODM for schema modeling and validation
- **Multer** – Middleware for handling file uploads
- **Cloudinary (Conceptual)** – For cloud image storage

The trainer explained why MongoDB is suitable for flexible data models and how Express simplifies API routing and middleware handling.

Backend Folder Structure Design

A professional backend folder structure was created to follow **industry standards**:

```
backend/
├── models/
│    └── Task.js
├── routes/
│    └── taskRoutes.js
├── middleware/
│    └── upload.js
├── config/
│    └── db.js
├── server.js
```

Each folder was explained in detail:

- **models/** – Database schemas
- **routes/** – API endpoints
- **middleware/** – Custom logic like file uploads
- **config/** – Database configuration
- **server.js** – Application entry point

This structure improves scalability and maintainability.

Task Schema Design

The **Task model** was designed with real-world considerations:

- `title` – Required task name
- `description` – Optional detailed notes
- `status` – Enum: pending, in-progress, completed
- `dueDate` – Optional deadline
- `image` – Image URL
- `createdAt` – Auto timestamp

Schema validation and default values were discussed to ensure data integrity.

Learning Outcome

By the end of Day 86:

- Strong understanding of full project requirements
- Knowledge of backend architecture planning
- Clear idea of task-based system design
- Understanding of file upload workflows

Conclusion

Day 86 successfully laid a **strong technical foundation** for the Student Task Tracker project. Proper planning and backend architecture ensured that future development would be smooth, scalable, and aligned with real-world industry practices.