NM-IBM INTERNSHIP

Day 3 Knowledge Transfer Task Documentation

College Name: Meenakshi college of Engineering College

College Code : 3114

Name : Afreen Naz A

Department : B.Tech(Information Technology)

Naan Mudhalvan (NM) ID : autit1002

Phone Number : 6380566957

Email ID: salmabegum54566@gmail.com

STUDENT MANAGEMENT APP

Node.js & Express.js Based Backend Application

---

1. Introduction

The Student Management App is a backend web application developed using Node.js and Express.js.
The main purpose of this project is to manage student records by performing basic CRUD operations (Create, Read, Update, Delete).

This application provides RESTful APIs that allow users to:

- **Add new student details**
- **View all students**
- **View a student by ID**
- **Update student details (single & multiple updates)**
- **Delete student records**

The project follows a modular folder structure and uses Express Router to maintain clean and scalable code.

2. Tools & Technologies Used

Tool / Technology	Purpose
Node.js	JavaScript runtime environment
Express.js	Backend web framework

Tool / Technology	Purpose
npm	Package manager
Visual Studio Code	Code editor
Postman / Thunder Client	API testing
Command Prompt	Project execution
JSON	Data exchange format

3. Project Setup & Implementation Process

Step 1: Install Node.js

- **Download Node.js from the official website**
- **Install Node.js (includes npm)**
- **Verify installation using:**

```
node -v
npm -v
```

Step 2: Create Project Folder

```
mkdir STUDENT-MANAGEMENT-APP
cd STUDENT-MANAGEMENT-APP
```

Step 3: Open Project in VS Code

code .

Step 4: Initialize Node Project

npm init -y

This creates:

- **package.json**

Step 5: Install Express

npm install express

This creates:

- **node_modules**
- **package-lock.json**

Step 6: Create Main Server File

- **Create index.js**
- **Configure Express server**
- **Enable JSON middleware**
- **Connect routes**

Step 7: Create Routes Folder

mkdir routes

Inside routes, create:

- **studentRoutes.js**

Step 8: Implement CRUD Operations

Student Fields

- **ID**
- **Name**
- **Department**
- **Age**

CRUD Operations Implemented

1. **CREATE – Add a new student**
2. **READ – Get all students**
3. **READ BY ID – Get student by ID**
4. **UPDATE – Update student details**
5. **UPDATE MULTIPLE – Update multiple students in one request**
6. **DELETE – Remove a student**

Temporary data storage is handled using an in-memory array, simulating a database.

Step 9: API Testing

- **APIs are tested using Postman or Thunder Client**
- **HTTP methods used:**
  - **GET**
  - **POST**
  - **PUT**
  - **DELETE**

Step 10: Run the Application

node index.js

Server runs on:

http://localhost:3000

---

4. API Endpoints

Method	Endpoint	Description
POST	/students/add	Add new student
GET	/students	Get all students
GET	/students/:id	Get student by ID
PUT	/students/update/:id	Update student

Method	Endpoint	Description
PUT	/students/update-multiple	Update multiple students
DELETE	/students/delete/:id	Delete student

---

5. Project Structure

STUDENT-MANAGEMENT-APP

```
|
├── node_modules
|
├── routes
|   └── studentRoutes.js
|
├── index.js
├── package.json
├── package-lock.json
```

Description

- **index.js → Main server file**
- **routes/studentRoutes.js → Handles all student APIs**
- **node_modules → Installed dependencies**

- **package.json → Project configuration**

---

6. Features of the Application

- **RESTful API design**
- **Modular routing**
- **CRUD operations**
- **Bulk update functionality**
- **Simple and readable code**
- **Easy to extend with database (MongoDB/MySQL)**

---

7. Limitations

- **Uses in-memory data (data resets on server restart)**
- **No authentication**
- **No frontend UI**

---

8. Future Enhancements

- **Database integration (MongoDB / MySQL)**
- **Authentication & authorization**
- **Input validation**
- **Frontend integration (React)**

- **Deployment on cloud**

---

9. Conclusion

The Student Management App demonstrates the practical use of Node.js and Express.js for backend development.
It provides a solid foundation for building scalable REST APIs and can be enhanced further by integrating databases and frontend technologies.

10.Github Repository Link:

https://github.com/afreennaz/Day-3-TASK