COLLEGE NAME: MEENAKSHI COLLEGE OF ENGINEERING

COLLEGE CODE:3114

YOUR NAME: ARIVANANTHAM.J

NAAN MUDHALVAN (NM) ID: C243912FBA9215698764EA53EB29B62B

PHONE NO:8807194452

EMAIL ID: nandhamarivu06@gmail.com

GIT HUB LINK: https://github.com/Arivu611/crud.git

The Student Attendance System

1. Introduction:

- Brief description of the project:

The Student Attendance System is a web-based application designed to help teachers and educational institutions efficiently record, manage, and track student attendance.

- Problem statement:

Manual attendance systems are time-consuming and prone to errors. This system automates attendance recording, reducing errors and saving time.

- Purpose of the system:
    - Easy attendance management
    - Quick access to attendance records
    - Reports and updates on student attendance

---

2. Objectives:

- To create an easy-to-use system for teachers and administrators.
- To ensure attendance is accurately recorded daily.
- To provide real-time tracking of student attendance.
- To reduce paper-based errors and save administrative time.

3. Features of the System:

- **User Interface**: Simple dashboard for teachers and admins
- **Add Student Attendance**: Mark attendance as Present/Absent
- **Update Attendance**: Ability to correct mistakes using PUT requests
- **Prevent Duplicates**: Ensures a student's attendance for a day cannot be entered twice
- **Reports**: View student attendance history
- **Database**: MongoDB stores all attendance records securely

4. Technologies Used:

- **Frontend:** HTML, CSS, JavaScript (optional framework if used)
- **Backend:** Node.js, Express.js
- **Database:** MongoDB (for storing student and attendance data)
- **Tools:** VS Code, Postman for API testing, Node Package Manager (npm)

5. System Design:

5.1 Architecture Diagram (optional: you can draw and paste)

- Client → Frontend → Backend → Database (MongoDB)

5.2 Database Design:

- **Collection:** Students
  - rollNumber (String, unique per day)
  - date (Date)
  - status (String: Present/Absent)

5.3 Routes:

Route	Method	Purpose
/attendance	POST	Add new attendance
/attendance	PUT	Update existing attendance
/attendance	GET	Fetch all attendance records

---

6. Implementation Steps:

6.1 Setup Environment:

1. Install Node.js and npm
2. Install MongoDB
3. Install project dependencies: express, mongoose

6.2 Create Server:

- Create index.js
- Connect to MongoDB
- Add middleware: express.json()

6.3 Create Routes:

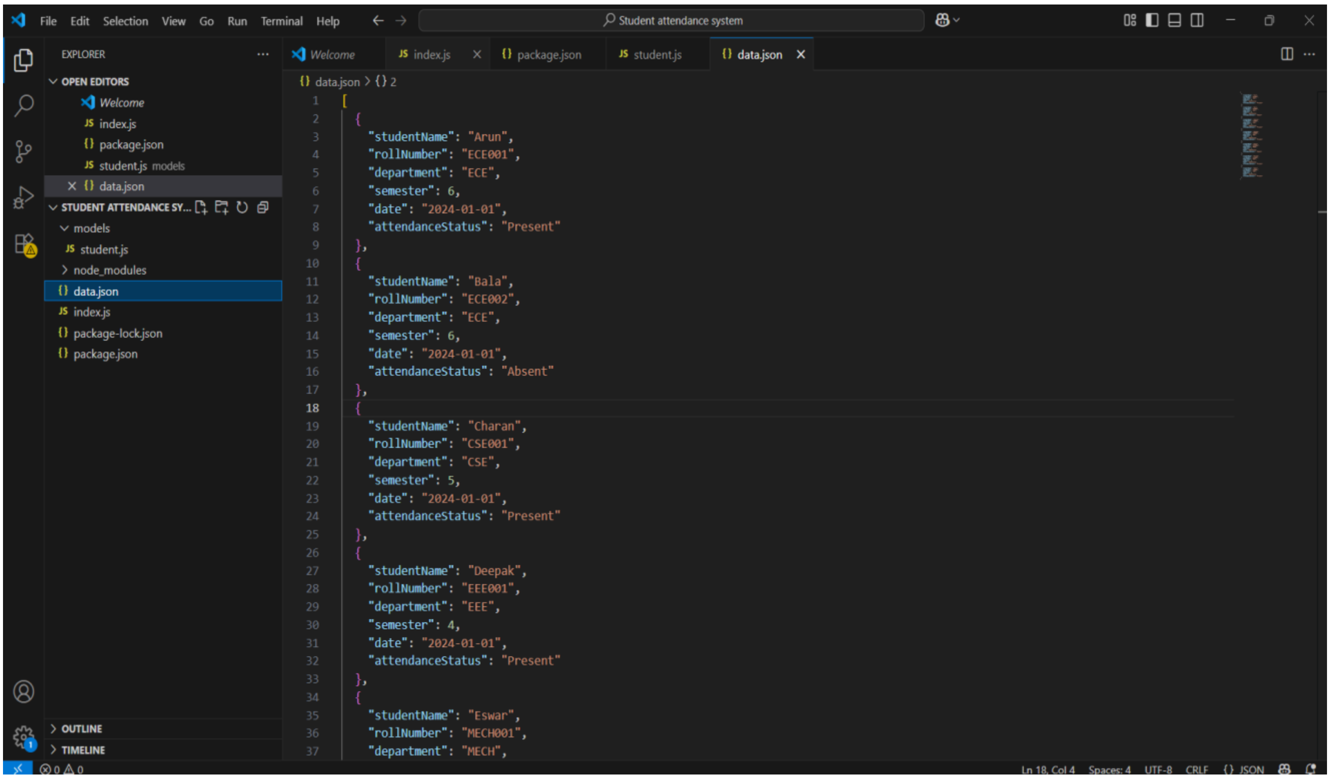
- POST: Add attendance
- PUT: Update attendance
- GET: View attendance

6.4 Create Student Schema:

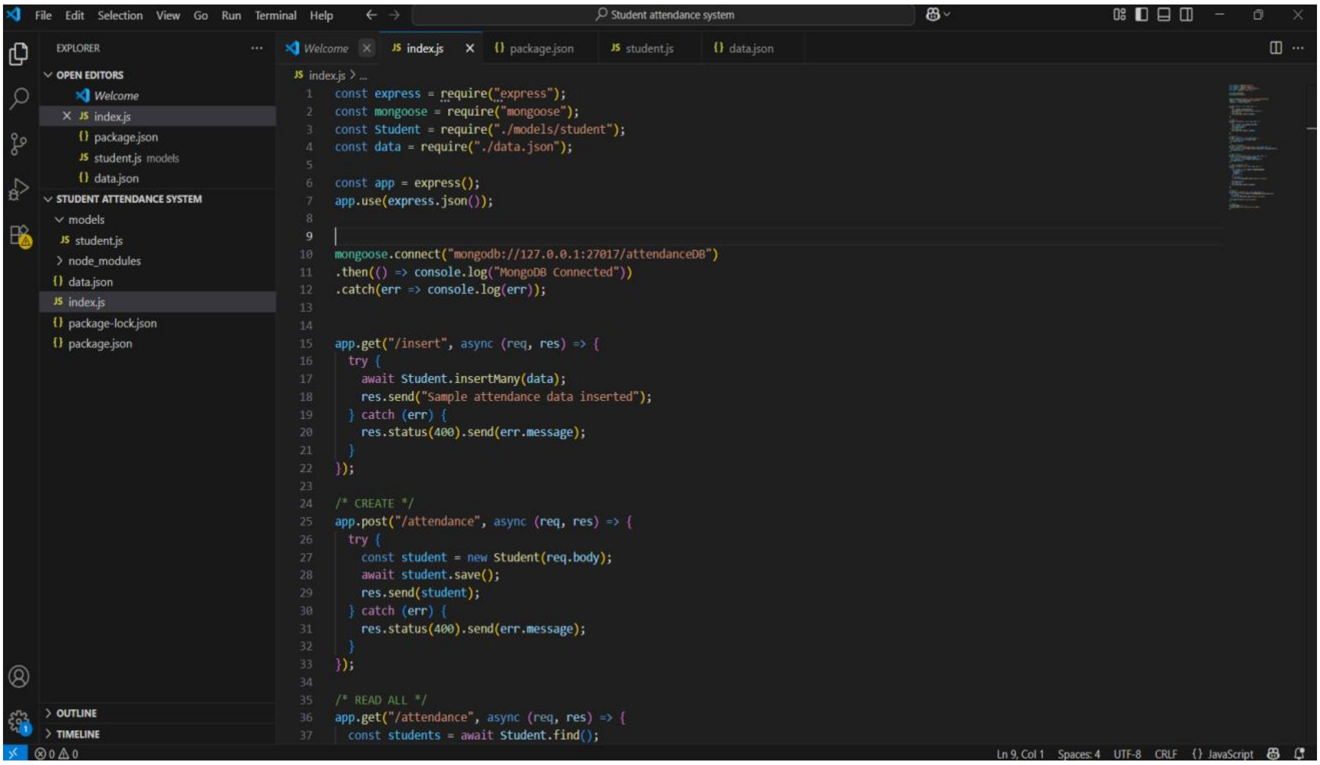
- Fields: rollNumber, date, status
- Unique index: rollNumber + date

6.5 Test API:

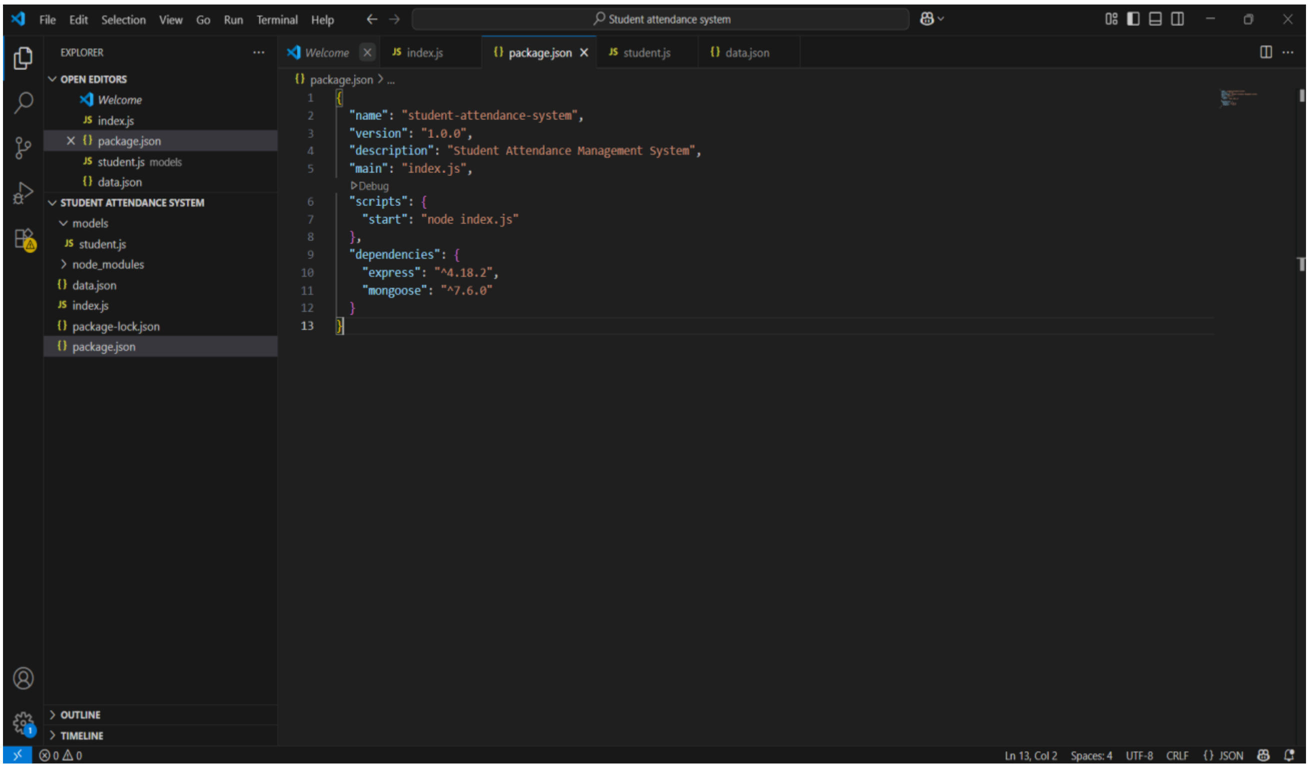
- Using Postman or browser fetch requests
- Ensure duplicate entries are handled
- Validate JSON data



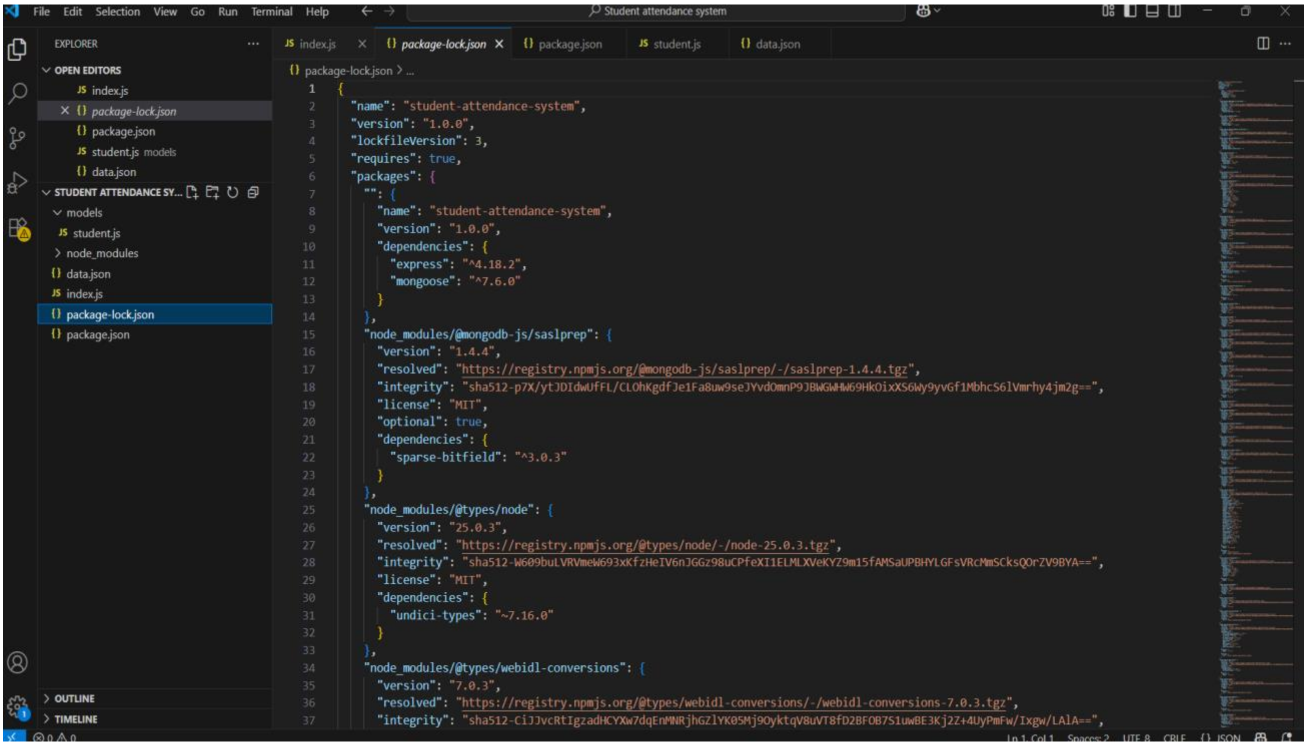
```json
[
  {
    "studentName": "Arun",
    "rollNumber": "ECE001",
    "department": "ECE",
    "semester": 6,
    "date": "2024-01-01",
    "attendanceStatus": "Present"
  },
  {
    "studentName": "Bala",
    "rollNumber": "ECE002",
    "department": "ECE",
    "semester": 6,
    "date": "2024-01-01",
    "attendanceStatus": "Absent"
  },
  {
    "studentName": "Charan",
    "rollNumber": "CSE001",
    "department": "CSE",
    "semester": 5,
    "date": "2024-01-01",
    "attendanceStatus": "Present"
  },
  {
    "studentName": "Deepak",
    "rollNumber": "EEE001",
    "department": "EEE",
    "semester": 4,
    "date": "2024-01-01",
    "attendanceStatus": "Present"
  },
  {
    "studentName": "Eswar",
    "rollNumber": "MECH001",
    "department": "MECH",
```



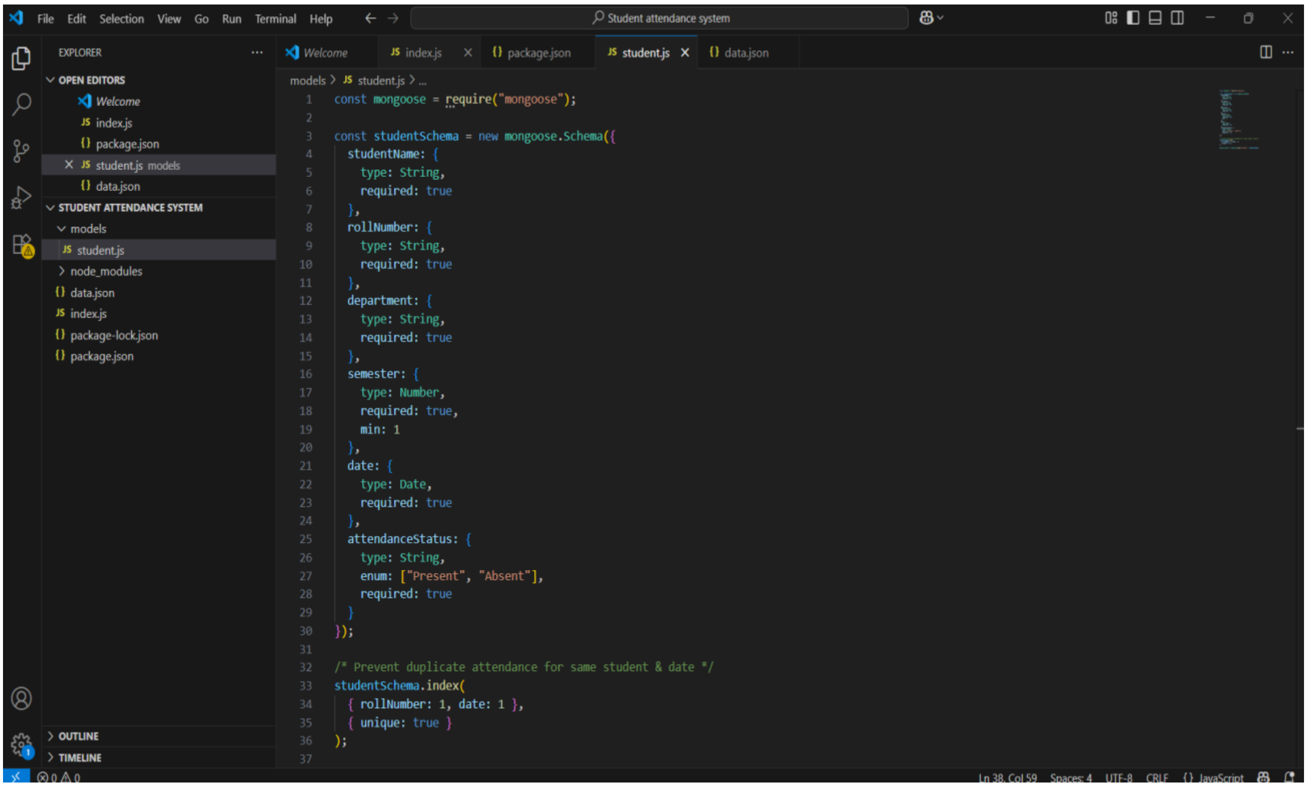
```javascript
const express = require("express");
const mongoose = require("mongoose");
const Student = require("./models/student");
const data = require("./data.json");

const app = express();
app.use(express.json());


mongoose.connect("mongodb://127.0.0.1:27017/attendanceDB")
.then(() => console.log("MongoDB Connected"))
.catch(err => console.log(err));


app.get("/insert", async (req, res) => {
  try {
    await Student.insertMany(data);
    res.send("Sample attendance data inserted");
  } catch (err) {
    res.status(400).send(err.message);
  }
});

/* CREATE */
app.post("/attendance", async (req, res) => {
  try {
    const student = new Student(req.body);
    await student.save();
    res.send(student);
  } catch (err) {
    res.status(400).send(err.message);
  }
});

/* READ ALL */
app.get("/attendance", async (req, res) => {
  const students = await Student.find();
```



```json
{
  "name": "student-attendance-system",
  "version": "1.0.0",
  "description": "Student Attendance Management System",
  "main": "index.js",
  "scripts": {
    "start": "node index.js"
  },
  "dependencies": {
    "express": "^4.18.2",
    "mongoose": "^7.6.0"
  }
}
```



```json
{
  "name": "student-attendance-system",
  "version": "1.0.0",
  "lockfileVersion": 3,
  "requires": true,
  "packages": {
    "": {
      "name": "student-attendance-system",
      "version": "1.0.0",
      "dependencies": {
        "express": "^4.18.2",
        "mongoose": "^7.6.0"
      }
    },
    "node_modules/@mongodb-js/saslprep": {
      "version": "1.4.4",
      "resolved": "https://registry.npmjs.org/@mongodb-js/saslprep/-/saslprep-1.4.4.tgz",
      "integrity": "sha512-p7X/ytJDIdwUfFL/CLOhKgdfJe1Fa8uw9seJYvdOmnP9JBWGWHW69HkOixXS6Wy9yvGf1MbhcS6lVmrhy4jm2g==",
      "license": "MIT",
      "optional": true,
      "dependencies": {
        "sparse-bitfield": "^3.0.3"
      }
    },
    "node_modules/@types/node": {
      "version": "25.0.3",
      "resolved": "https://registry.npmjs.org/@types/node/-/node-25.0.3.tgz",
      "integrity": "sha512-W609buLVRVmeW693xKfzHeIV6nJGGz98uCPfeXI1ELMLXVeKYZ9m15fAMSaUPBHYLGFsVRcMmSCksQOrZV9BYA==",
      "license": "MIT",
      "dependencies": {
        "undici-types": "~7.16.0"
      }
    },
    "node_modules/@types/webidl-conversions": {
      "version": "7.0.3",
      "resolved": "https://registry.npmjs.org/@types/webidl-conversions/-/webidl-conversions-7.0.3.tgz",
      "integrity": "sha512-CiJJvcRtIgzadHCYXw7dqEnMNRjhGZlYK05Mj9OyktqV8uVT8fD2BFOB7S1uwBE3Kj2Z+4UyPmFw/Ixgw/LAlA==",
```



```javascript
const mongoose = require("mongoose");

const studentSchema = new mongoose.Schema({
  studentName: {
    type: String,
    required: true
  },
  rollNumber: {
    type: String,
    required: true
  },
  department: {
    type: String,
    required: true
  },
  semester: {
    type: Number,
    required: true,
    min: 1
  },
  date: {
    type: Date,
    required: true
  },
  attendanceStatus: {
    type: String,
    enum: ["Present", "Absent"],
    required: true
  }
});

/* Prevent duplicate attendance for same student & date */
studentSchema.index(
  { rollNumber: 1, date: 1 },
  { unique: true }
);
```

8. Testing and Screenshots:

- Show screenshots of Postman testing POST, PUT, GET
- Show sample attendance records in MongoDB Compass (optional)
- Explain any errors and how they were fixed (like duplicate key errors, JSON syntax errors)

---

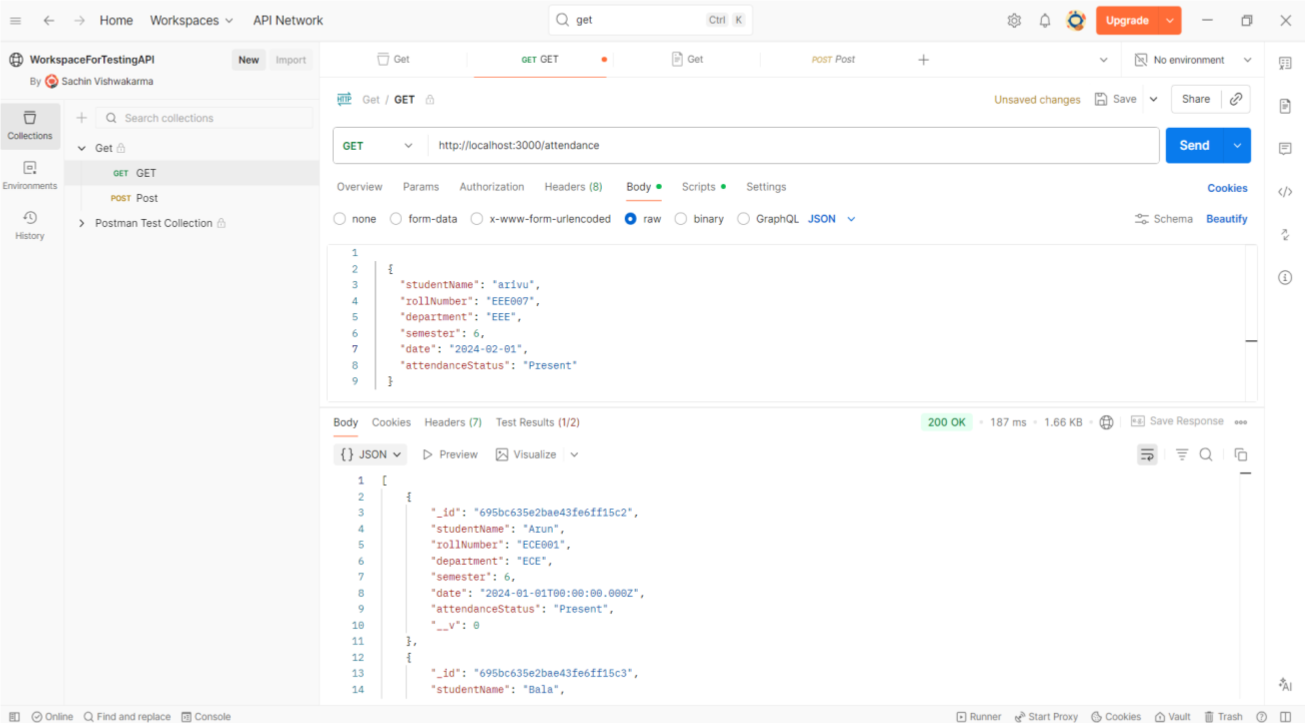
9. Conclusion:

- Summarize the benefits:
  - Accurate, fast, and automated attendance system
  - Reduced manual effort and errors
  - Easy tracking and reporting of student attendance
- Future improvements:
  - Add login system for teachers
  - Generate monthly or yearly attendance reports
  - Add notifications for absent students

---

10. References:

- Node.js Documentation
- MongoDB Documentation
- Express.js Documentation
- Tutorials or blogs you followed



GET http://localhost:3000/attendance

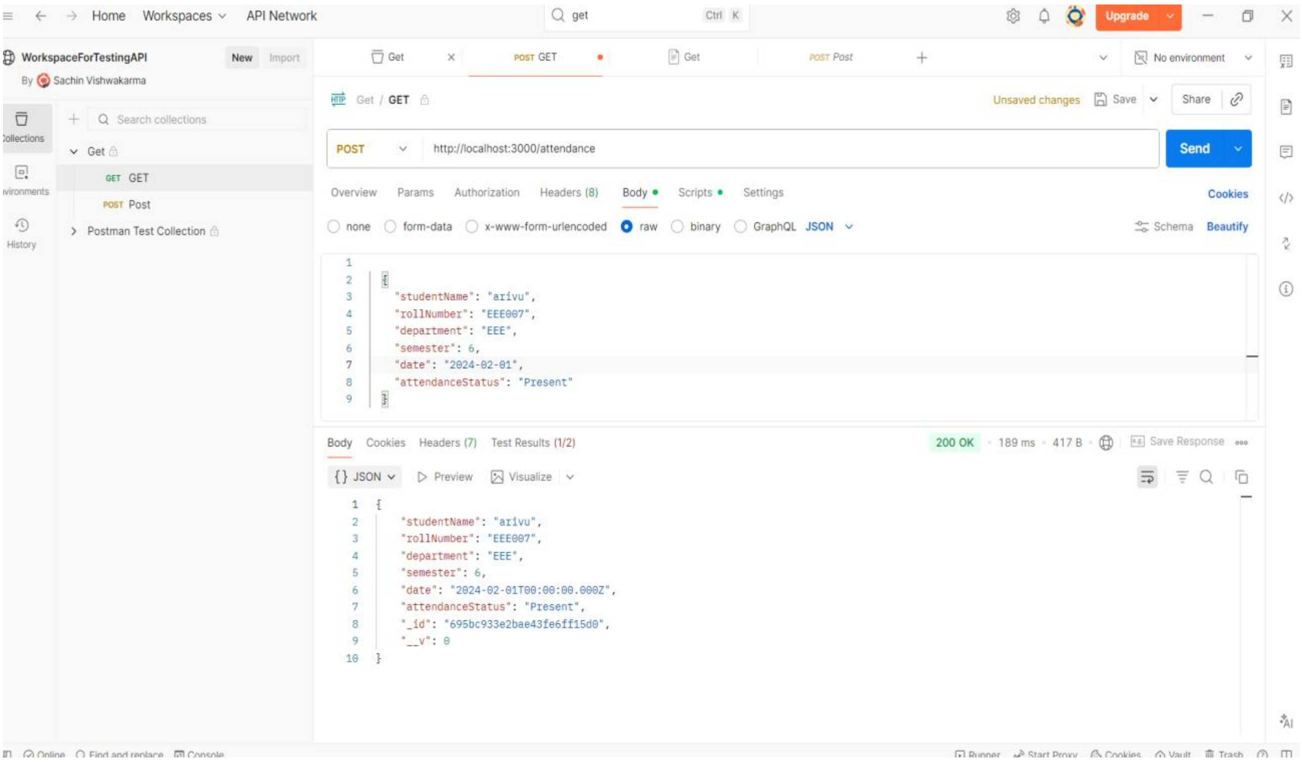
Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```

{
  "studentName": "arivu",
  "rollNumber": "EEE007",
  "department": "EEE",
  "semester": 6,
  "date": "2024-02-01",
  "attendanceStatus": "Present"
}
```

Body Cookies Headers (7) Test Results (1/2) 200 OK · 187 ms · 1.66 KB

{} JSON

```
[
    {
        "_id": "695bc635e2bae43fe6ff15c2",
        "studentName": "Arun",
        "rollNumber": "ECE001",
        "department": "ECE",
        "semester": 6,
        "date": "2024-01-01T00:00:00.000Z",
        "attendanceStatus": "Present",
        "__v": 0
    },
    {
        "_id": "695bc635e2bae43fe6ff15c3",
        "studentName": "Bala",
```



POST http://localhost:3000/attendance

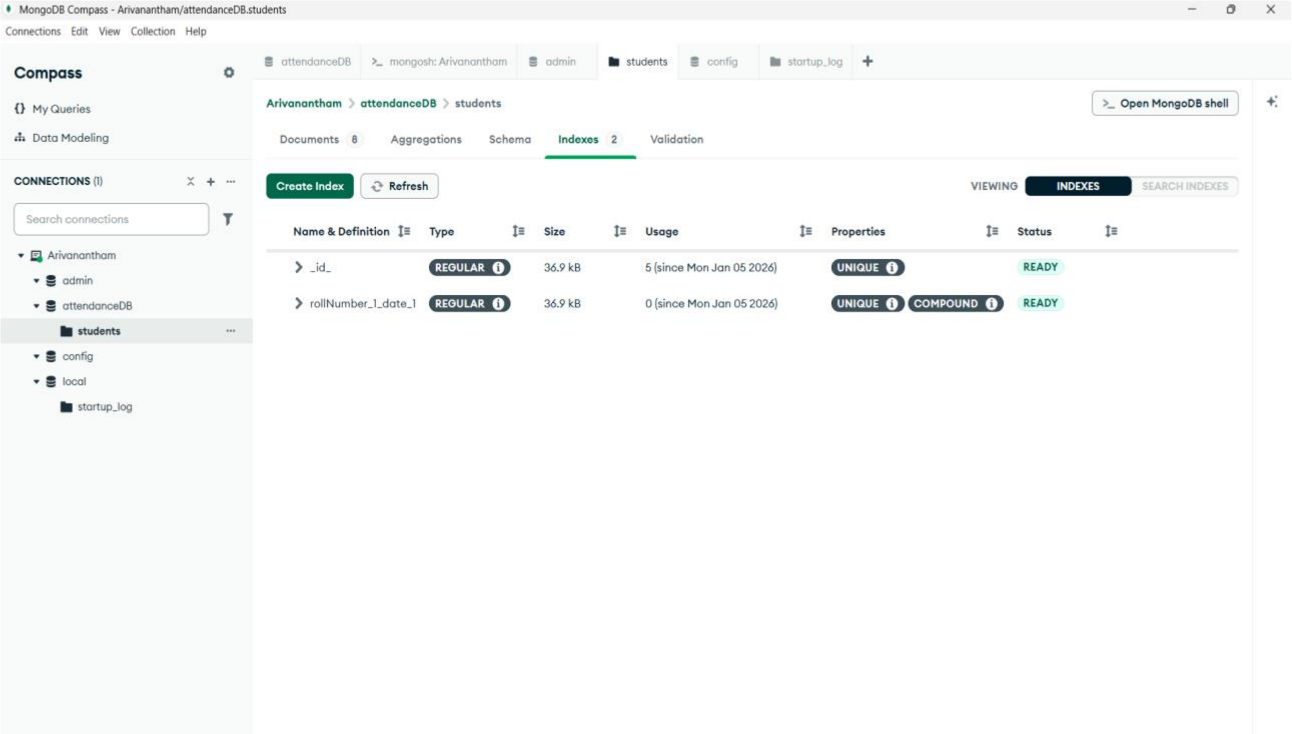
Body: ○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON

```

{
  "studentName": "arivu",
  "rollNumber": "EEE007",
  "department": "EEE",
  "semester": 6,
  "date": "2024-02-01",
  "attendanceStatus": "Present"
}
```

Body Cookies Headers (7) Test Results (1/2) 200 OK · 189 ms · 417 B

{} JSON

```
{
    "studentName": "arivu",
    "rollNumber": "EEE007",
    "department": "EEE",
    "semester": 6,
    "date": "2024-02-01T00:00:00.000Z",
    "attendanceStatus": "Present",
    "_id": "695bc933e2bae43fe6ff15d0",
    "__v": 0
}
```



Name & Definition	Type	Size	Usage	Properties	Status
_id_	REGULAR	36.9 kB	5 (since Mon Jan 05 2026)	UNIQUE	READY
rollNumber_1_date_1	REGULAR	36.9 kB	0 (since Mon Jan 05 2026)	UNIQUE COMPOUND	READY