TASK 1

Hello there! 🌟
 Ready to dive into the exciting world of **APIs**??

Your mission, should you choose to accept it, is to explore and interact with our very own **College Student Information API**. This isn't just about reading documentation — it's about getting **hands-on** and understanding how data flows in a real-world application.

🧩 First Things First: What is CRUD?

Before we start, let's break down the magic word you'll hear all the time in web dev: **CRUD**.

Think of CRUD like the **four powers of a superhero app** 💪:

- **C → Create (POST)**: Add new data (e.g., register a new student).
- **R → Read (GET)**: Fetch and view data (e.g., display all students).
- **U → Update (PUT)**: Modify existing data (e.g., fix a student's CGPA).
- **D → Delete (DELETE)**: Remove data (e.g., delete a student record).

That's it! Every app you use — Instagram, Amazon, Netflix — is just CRUD with different clothes on 👀.

And now, you'll build your own CRUD app with students. 🎓

🎯 Goal

- By default, all the students will be displayed.
- There will be an **“Add”** button which, when clicked, should take input from the user and add a new student’s data.
- For every student, there should be an **Update** ✏️ and **Delete** ❌ button to edit or remove that student’s details.

🗓️ **Deadline**: 1st September (Next Monday)

This task will help you get familiar with the four core HTTP requests used in web development:
👉 **GET, POST, PUT, DELETE**

📚 API Documentation

Base URL: `https://studentapi-m7a7.onrender.com`

1. Add a Student

- **API**: `/student/add`
- **Method**: POST
- Adds a new student entry to the database.
- ⚠️ Each **sapid must be unique**.

Sample JSON:

```
{
  "studentName": "John Doe",
  "college": "DJ Sanghvi",
  "cgpa": "9.1",
  "phone": "9876543210",
  "sapid": "6000425999",
  "batch": "C2-3",
  "year": "Second Year",
  "address": "Andheri"
}
```

Errors you might see:

- `400 Bad Request` → Missing required fields
- `400 Bad Request` → Duplicate sapid not allowed
- `500 Internal Server Error` → Database/server issue

---

2. Get All Students

- **API**: `/student/all`
- **Method**: GET
- Returns a list of all students

Sample response:

```
[

  {
    "_id": "66f15e1234abcd56780",
    "studentName": "Smit Pawar",
    "college": "DJ Sanghvi",
    "cgpa": "9.7",
    "phone": "9090858510",
    "sapid": "6000425001",
    "batch": "C3-2",
    "year": "Third Year",
    "address": "Ghatkopar",
    "isDefault": true
  },
  {
    "_id": "66f15e1234abcd56781",
    "studentName": "Raj Mehta",
    "college": "DJ Sanghvi",
    "cgpa": "8.9",
    "phone": "9876543210",
    "sapid": "6000425002",
    "batch": "C3-2",
    "year": "Third Year",
    "address": "Andheri",
    "isDefault": true
  }
]
```

---

3. Update a Student

- **API**: `/student/update/:id`
- **Method**: PUT
- Updates details of a specific student by their MongoDB `_id`
- ⚠️ **Note**: Default 20 students cannot be updated

---

4. Delete a Student

- **API**: `/student/delete/:id`
- **Method**: DELETE
- Deletes a specific student by their MongoDB `_id`
- ⚠️ **Note**: Default 20 students cannot be deleted

Successful deletion response:

```
{ "message": "Student deleted successfully" }
```

⚠️ Important Notes

- 20 default students are already stored → you cannot **update** or **delete** them.
- You can only edit/remove students **you** add.
- While adding a new student, **all fields are required** (no empty fields).
- Each student must have a **unique SapId**.
- API data is visible to everyone → so be respectful and don't misuse others' entries.
- Please do not spam the server — delete extra test students once you're done.
- UI should be simple and intuitive. Remember:

A good UI is like a good joke — if you have to explain it, it’s bad. 😅

✨ Bonus Tasks (Optional)

- Add a **search bar** to find students by name, college, etc.
- Add **sorting by CGPA**.
- Implement **pagination** (e.g., show 10 students per page).
- Use **Tailwind CSS** for styling.
- Use **axios** for fetching.

🎥 Learning Resources

- Intro to APIs
- CRUD Operations Explained
- Building a Student App
- Axios

(You are free to use any other resources too.)