

Module 4

Exercise 8(NextJs API Route)

Q1.) Create a json file with 5 records of teachers and 30 records of students.

A1.) students.json

```
Module4 > Exercise8 > api-proj > public > data >  students.json > ...
```

```
1  {
2    "students": [
3      {"id":1,"name":"Aarav","standard":"10th","mentor":"Sapna"},  
4      {"id":2,"name":"Vivaan","standard":"9th","mentor":"Ashish"},  
5      {"id":3,"name":"Aditya","standard":"8th","mentor":"Tanya"},  
6      {"id":4,"name":"Krishna","standard":"10th","mentor":"Samar"},  
7      {"id":5,"name":"Ishaan","standard":"9th","mentor":"Mayank"},  
8      {"id":6,"name":"Ananya","standard":"8th","mentor":"Sapna"},  
9      {"id":7,"name":"Diya","standard":"7th","mentor":"Ashish"},  
10     {"id":8,"name":"Saanvi","standard":"6th","mentor":"Tanya"},  
11     {"id":9,"name":"Aanya","standard":"10th","mentor":"Samar"},  
12     {"id":10,"name":"Riya","standard":"9th","mentor":"Mayank"},  
13     {"id":11,"name":"Arjun","standard":"8th","mentor":"Sapna"},  
14     {"id":12,"name":"Kabir","standard":"7th","mentor":"Ashish"},  
15     {"id":13,"name":"Reyansh","standard":"6th","mentor":"Tanya"},  
16     {"id":14,"name":"Atharv","standard":"10th","mentor":"Samar"},  
17     {"id":15,"name":"Shaurya","standard":"9th","mentor":"Mayank"},  
18     {"id":16,"name":"Myra","standard":"8th","mentor":"Sapna"},  
19     {"id":17,"name":"Kiara","standard":"7th","mentor":"Ashish"},  
20     {"id":18,"name":"Navya","standard":"6th","mentor":"Tanya"},  
21     {"id":19,"name":"Meera","standard":"10th","mentor":"Samar"},  
22     {"id":20,"name":"Ira","standard":"9th","mentor":"Mayank"},  
23     {"id":21,"name":"Yash","standard":"8th","mentor":"Sapna"},  
24     {"id":22,"name":"Rohan","standard":"7th","mentor":"Ashish"},  
25     {"id":23,"name":"Aryan","standard":"6th","mentor":"Tanya"},  
26     {"id":24,"name":"Dev","standard":"10th","mentor":"Samar"},  
27     {"id":25,"name":"Harsh","standard":"9th","mentor":"Mayank"},  
28     {"id":26,"name":"Nitya","standard":"8th","mentor":"Sapna"},  
29     {"id":27,"name":"Tanvi","standard":"7th","mentor":"Ashish"},  
30     {"id":28,"name":"Pari","standard":"6th","mentor":"Tanya"},  
31     {"id":29,"name":"Siddharth","standard":"10th","mentor":"Samar"},  
32     {"id":30,"name":"Laksh","standard":"9th","mentor":"Mayank"}  
33   ]
```

teachers.json

```
Module4 > Exercise8 > api-proj > public > data > teachers.json > [ ] teachers > { } 4 > abc subject
1  {
2    "teachers": [
3      {
4        "id": "1",
5        "name": "Sapna",
6        "subject": "OOPS"
7      },
8      {
9        "id": "2",
10       "name": "Ashish",
11       "subject": "C++"
12     },
13     {
14       "id": "3",
15       "name": "Tanya",
16       "subject": "Java"
17     },
18     {
19       "id": "4",
20       "name": "Samar",
21       "subject": "Javascript"
22     },
23     {
24       "id": "5",
25       "name": "Mayank",
26       "subject": "ReactJs"
27     }
28   ]
29 }
```

Q2.) Create an API endpoint to fetch the list of all the teachers.

A2.) api/teacher

```
e4 > Exercise8 > api-proj > app > api > teacher > route.js > POST
import fs from 'fs'
import path from 'path';
export async function GET() {
    try{
        const res=await fetch('http://localhost:3000/data/teachers.json');
        const data= await res.json();

        return Response.json(data)
    }catch(error){
        return Response.json(
            {error:error.message},
            {status:500}
        )
    }
}
```

/teachers

```
e4 > Exercise8 > api-proj > app > teachers > page.jsx > TeachersPage
"use client";

import { useEffect, useState } from "react";
import './page.css'

export default function TeachersPage() {
    const [teachers, setTeachers] = useState([]);
    const [form, setForm] = useState({
        id:0,
        name: "",
        subject: ""
    });

    const getTeachers = async () => {
        const res = await fetch("/api/teacher");
        const data = await res.json();
        setTeachers(data.teachers || []);
    };

    useEffect(() => {
        getTeachers();
    }, []);
}
```

Output

The image shows a mobile application interface with a dark blue background. At the top, the title "Teachers List" is displayed in a large, bold, white font. Below the title, there is a vertical list of five items, each enclosed in a rounded rectangular box with a light gray background and a thin gray border. Each item consists of a teacher's name followed by a dash and their subject. The items are: "Sapna - OOPS", "Ashish - C++", "Tanya - Java", "Samar - Javascript", and "Mayank - ReactJs".

Teacher	Subject
Sapna	OOPS
Ashish	C++
Tanya	Java
Samar	Javascript
Mayank	ReactJs

Q3.) Create an API endpoint to fetch the list of all the students.

A3.) api/student

```
/ Exercices / API / app / api / students / GET.js
export async function GET(request) {
  try {
    const { searchParams } = new URL(request.url);
    const teacherName = searchParams.get('teacherName');
    const res=await fetch('http://localhost:3000/data/students.json');
    const {students}=await res.json();

    if (!teacherName) {
      return Response.json(
        students
      );
    }
  }
}
```

/students

```
/ Exercices / API / app / students / StudentsPage.js
"use client";

import { useState } from "react";
import './page.css'

export default function StudentsPage() {
  const [teacherName, setTeacherName] = useState("");
  const [students, setStudents] = useState([]);

  const fetchStudents = async () => {
    if(teacherName===""){
      const res=await fetch('/api/student');
      const data=await res.json();

      setStudents(data);
    }
  }
}
```

Output

Aarav - Mentor: Sapna

Vivaan - Mentor: Ashish

Aditya - Mentor: Tanya

Krishna - Mentor: Samar

Ishaan - Mentor: Mayank

Ananya - Mentor: Sapna

Diya - Mentor: Ashish

Saanvi - Mentor: Tanya

Aanya - Mentor: Samar

Riya - Mentor: Mayank

Arjun - Mentor: Sapna

Kabir - Mentor: Ashish

Reyansh - Mentor: Tanya

Atharv - Mentor: Samar

Shaurya - Mentor: Mayank

Myra - Mentor: Sapna

Kiara - Mentor: Ashish

Q4.) Create an API endpoint to fetch the list of students who are under a specific teacher.

A4.) api/student

```
const filteredStudents = students.filter(
  (student) =>
    student.mentor.toLowerCase() === teacherName.toLowerCase()
);

return Response.json(filteredStudents);
} catch (error) {
  return Response.json(
    { error: error.message },
    { status: 500 }
  );
}
}
```

/students

```
else{
  const res = await fetch(
    `/api/student?teacherName=${teacherName}`
  );

  const data = await res.json();

  setStudents(data);
}

};

return (
  <div className="students-page">
    <h2>Find Students by Teacher</h2>

    <input
      type="text"
      placeholder="Teacher Name"
      value={teacherName}
      onChange={(e) => setTeacherName(e.target.value)}
    </>
  </div>
);
```

Output

The screenshot shows a user interface for a search application. At the top left is a search bar containing the text "Sapna". To the right of the search bar is a blue button labeled "Search". Below the search bar, there is a list of six items, each consisting of a name followed by a dash and the text "Mentor: Sapna". The items are listed vertically: "Aarav - Mentor: Sapna", "Ananya - Mentor: Sapna", "Arjun - Mentor: Sapna", "Myra - Mentor: Sapna", "Yash - Mentor: Sapna", and "Nitya - Mentor: Sapna". The background of the interface is dark.

Sapna

Search

Aarav - Mentor: Sapna

Ananya - Mentor: Sapna

Arjun - Mentor: Sapna

Myra - Mentor: Sapna

Yash - Mentor: Sapna

Nitya - Mentor: Sapna

Q5.) Create an endpoint to add a teacher in the list. (Optional)

A5.) /api/teacher

```
const filePath=path.join(process.cwd(),'public/data/teachers.json')
export async function POST(req) {
    try{
        const body=await req.json();
        const fileData=fs.readFileSync(filePath,'utf-8');
        const {teachers}=JSON.parse(fileData);
        teachers.push(body);

        fs.writeFileSync(filePath,JSON.stringify({teachers},null,2));

        return Response.json({
            message:"Teachers are added successfully"
        },{
            status:200
        });
    }
    catch(error){
        return Response.json({
            message:error.message,
            {status:error.status}
        })
    }
}
```

/teachers

```
const handleSubmit = async (e) => {
  e.preventDefault();

  const res = await fetch("/api/teacher", {
    method: "POST",
    headers: {
      "Content-Type": "application/json",
    },
    body: JSON.stringify(form),
  });

  if (res.ok) {
    setForm({ id:0, name: "", subject: "" });
    getTeachers();
  }
};

return (
  <div className="teachers-page">
    <h2>Add Teacher</h2>

    <form onSubmit={handleSubmit}>
      <input
        type="number"
        placeholder="Id"
        value={form.id}
        onChange={(e) =>
          setForm({ ...form, id: e.target.value })
        }
        required
      />
    </form>
  </div>
);
```

Output

Add Teacher

6 Naman Web Dev Add

Teachers List

Sapna - OOPS
Ashish - C++
Tanya - Java
Samar - Javascript
Mayank - ReactJs

Teachers List	
Sapna - OOPS	
Ashish - C++	
Tanya - Java	
Samar - Javascript	
Mayank - ReactJs	
Naman - Web Dev	