# Step 2: Connecting React with Spring Boot

Software Engineering Project (Sprint 2)

Tutor: Haythem Ghazouani

## 1 Introduction

In this sprint, we bridge the gap between the user interface and the server. We will use **Axios** to consume the REST endpoints we created in the previous step.

## 2 Frontend Setup

Our React application is powered by **Vite**. Ensure you have installed the necessary dependencies:

```
npm install axios lucide-react
```

## 3 State Management & API Calls

We use the `useState` and `useEffect` hooks to manage the project data.

**Sprint 2 Task: Fetching Data**

```
const fetchProjects = async () => {
    try {
        const response = await axios.get('http://localhost
   :8080/api/projects');
        setProjects(response.data);
    } catch (error) {
        console.error("API Call failed", error);
    }
};
```

## 4 Handling Component State

The dashboard updates in real-time. When a new project is added via the form, we trigger a re-fetch or update the local state to reflect the change immediately.

Architecture Tip

**CORS Warning:** If the backend doesn't have `@CrossOrigin`, the browser will block the request. Our Spring Boot controller is already configured to allow all origins for this prototype.

# 5 Exercise

1. Add a loading spinner while the projects are being fetched.
2. Implement a "Status" toggle button that updates the project status from "To Do" to "Done" via a PUT request (you will need to add the backend endpoint first!).