Assignment 7.3

Integrating a frontend application with a backend server

Introduction

Integrating a frontend application with a backend server is a fundamental process in web development, allowing the frontend to communicate with the backend to fetch and manipulate data. This involves understanding RESTful APIs, making API calls from the frontend, and handling responses. Below is a detailed explanation that you can transcribe into a handwritten report.

understanding RESTful APIs

concept: REST (Representational State Transfer) is an architectural style for designing

networked applications. RESTful APIs use HTTP requests to perform CRUD (Create, Read, Update, Delete) operations on resources.

Key characteristics:

Stateless: Each API call is independent, with no stored context on the server between requests.

Resource-Based: Each piece of data (resource) is identified by a URL.

Methods: Common HTTP methods include GET, POST, PUT, DELETE.

Example of RESTful API Endpoints:

GET Users: Retrieve a list of users.

POST Users: Create a new User.

GET Users/:id: Retrieve a specific User by ID.

PUT Users/:id: Update a specific User by ID.

DELETE Users/:id: Delete a specific User by ID.

Making API calls from the Frontend

Steps to make API calls:

Setting up the Frontend:

Use JavaScript, along with frameworks/libraries like React, Angular, or Vue.js, to set up the frontend application.

Using Fetch API or Axios:

Fetch API is a built-in JavaScript function for making API requests.

AXIOS is a POPULAR Promise-based HTTP client for JavaScript.

Handling Responses:

Process the response from the server.

Handle errors appropriately.

Example using React and AXIOS

Step-by-Step Implementation:

Set up React Application:

Initialize a React application using create react APP.

Install AXIOS using npm install axios.

Create a Backend Server (e.g., Node.js with EXPRESS):

Set up a simple RESTful API with endpoints for users.

Make API calls from React:

Use AXIOS to make HTTP requests to the backend server.

Example Code:

Backend (Node.js with EXPRESS):

javascript

```
const express = require('express') const app =
express() const port = 3000
let users = [ { id: 1, name: 'Alice' },
  { id: 2, name: 'Bob' } ] app.use(express.json())
app.get('/users', (req, res) => { res.json(users)
})
app.post('/users', (req, res) => { const newUser
= req.body
newUser.id = users.length + 1
users.push(newUser) res.status(201).json(newUser)
}) app.get('/users/:id', (req, res) =>
{ const user = users.find(u => u.id ===
parseInt(req.params.id))
if (user) res.json(user)

else
```

```
res.status(404).send('user not found')
});

app.put('/users/:id', (req, res) => {
  const user = users.find(u => u.id ===
parseInt(req.params.id));
  if (user) {
    Object.assign(user, req.body);
    res.json(user);
  } else {
    res.status(404).send('user not found')
  }
});

app.delete('/users/:id', (req, res) => {
  users = users.filter(u => u.id !==
parseInt(req.params.id));
  res.status(204).send()
});

app.listen(PORT, () => {
  console.log(`server running at
http://localhost:${PORT}`)
});
```

FRONTEND (REACT WITH AXIOS):

```
import React, useState, useEffect from
react import axios from axios function APP()
 const users, setusers = useState() useEffect()
=>      Fetch users from the backend
axios.get('http://localhost:3000/users')
.then(response =>
 setusers(response.data)
 ) .catch(error =>
 console.error('There was an error fetching the
users!', error) )
 , )
 const adduser = () =>   const newuser =
 name: New User
axios.post('http://localhost:3000/users', newuser)
 .then(response =>   setusers(...users,
 response.data) ) .catch(error =>
 console.error('There was an error adding the
```