1. React – JSON Server

□ Theory

Q1: What is JSON Server in React?
JSON Server is a lightweight backend tool that allows you to create a mock REST API using a simple JSON file. It helps React developers test and simulate data interactions without setting up a full backend.

Q2: What is the role of JSON Server in React development?

- Acts as a **fake REST API** for testing CRUD operations.
- Helps in **frontend development** before the real backend is ready.
- Supports **GET, POST, PUT, PATCH, DELETE** operations easily.

Q3: How to install and run JSON Server?

1. Install using npm:
2. npm install -g json-server
3. Create a file named db.json with data, for example:
4. {
5. "users": [
6. { "id": 1, "name": "Aamir", "age": 25 },
7. { "id": 2, "name": "Nisha", "age": 22 }
8.]
9. }
10. Start the server:
11. json-server --watch db.json --port 5000
12. API will run at http://localhost:5000/users

Q4: How does React interact with JSON Server?
React uses fetch() or axios to make API requests to the JSON Server.

□ Practical / Lab Assignment

Task:
Create a React app that connects with a JSON Server to display and add users.

Steps:

1. Create a new React app:
2. npx create-react-app json-demo
3. cd json-demo
4. Install Axios:
5. npm install axios
6. Run JSON Server:
7. json-server --watch db.json --port 5000
8. In App.js:
9. import React, { useEffect, useState } from 'react';
10. import axios from 'axios';
11.
12. const App = () => {
13. const [users, setUsers] = useState([]);
14.
15. useEffect(() => {
16. axios.get('http://localhost:5000/users')
17. .then(res => setUsers(res.data))
18. .catch(err => console.log(err));
19. }, []);
20.
21. return (
22.
23. <h2>Users List</h2>

```
      <ul>
        {users.map(user => (
          <li key={user.id}>{user.name} - {user.age}</li>
        ))}
      </ul>
    </div>
  );
};

export default App;
```

✅ **Expected Result:**
When you run npm start, you should see the list of users fetched from JSON Server.

🔥 2. React – Firebase Realtime Database

📘 Theory

Q1: What is Firebase Realtime Database?
Firebase Realtime Database is a cloud-hosted NoSQL database that stores data in JSON format and syncs it in real time across all clients.

Q2: Why use Firebase in React?

- Provides **real-time synchronization**.
- Offers **secure, serverless backend**.
- Easy **integration** with React using Firebase SDK.
- Supports **CRUD operations** directly from frontend.

Q3: How to connect Firebase to a React project?

1. Go to Firebase Console
2. Create a project and add a **web app**.

3. Copy your Firebase config and initialize Firebase in your React project.

Example config:

```
import { initializeApp } from "firebase/app";
import { getDatabase } from "firebase/database";

const firebaseConfig = {
  apiKey: "YOUR_API_KEY",
  authDomain: "yourapp.firebaseapp.com",
  databaseURL: "https://yourapp-default-rtdb.firebaseio.com",
  projectId: "yourapp",
  storageBucket: "yourapp.appspot.com",
  messagingSenderId: "XXXXXXX",
  appId: "1:XXXX:web:XXXX"
};

const app = initializeApp(firebaseConfig);
export const db = getDatabase(app);
```

□ Practical / Lab Assignment

Task:
Create a React app that stores and retrieves user data from Firebase Realtime Database.

Steps:

1. Install Firebase:
2. npm install firebase
3. Create a new file firebaseConfig.js and paste the config code above.
4. Use this in your App.js:

```
import React, { useState, useEffect } from "react";
import { db } from "./firebaseConfig";
import { ref, set, onValue } from "firebase/database";

const App = () => {
  const [name, setName] = useState("");
  const [users, setUsers] = useState([]);

  const addUser = () => {
    set(ref(db, 'users/' + Date.now()), { name });
    setName("");
  };

  useEffect(() => {
    const userRef = ref(db, "users/");
    onValue(userRef, (snapshot) => {
      const data = snapshot.val();
      if (data) setUsers(Object.values(data));
    });
  }, []);

  return (
    <div>
      <h2>Firebase Realtime Database Example</h2>
      <input value={name} onChange={(e) => setName(e.target.value)} placeholder="Enter name"
/>
```

```
      <button onClick={addUser}>Add User</button>
      <ul>
        {users.map((u, i) => <li key={i}>{u.name}</li>)}
      </ul>
    </div>
  );
};

export default App;
```