Experiment 1

Student Name: Shubham **UID: 22BCS15853**
Branch: CSE **Section/Group: KRG 2 B**
Semester: 6th **Date of Performance: 08/01/25**
Subject Name: Advanced Programming LAB-II **Subject Code: 22CSP-351**

1. **Aim:**

Full Stack Development (MERN). The primary aim of this experiment is to provide students or developers with an understanding of full-stack development involving MongoDB, Node.js, React, and Express.

1. Problem 1.1.1: Give understanding of MongoDB, Nodejs, React, Express.
2. Problem 1.1.2: Create a Frontend design of Login/Signup pages and create a backend of it.
3. Problem 1.1.3: Test the Backend API Using Postman

2. **Objective:**

- Understand the fundamentals of MongoDB, Node.js, React, and Express
- Create a functional frontend for Login/Signup pages
- Develop a backend using Express and MongoDB
- Test the backend API using Postman

3. **Implementation/Code:**

Backend:

- Open vs code
- New terminal
- Cd backend
- Nodemon server.js

Server.js

```
const express = require("express");
const mongoose = require("mongoose");
const bodyParser = require("body-parser");
const cors = require("cors");
const app = express();
app.use(bodyParser.json());
```

```
app.use(cors());
mongoose.connect(
"mongodb+srv://nidhimalik901:2Ck1FLS5njJhKoCZ@cluster0.h4jig.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0",
{
useNewUrlParser: true,
useUnifiedTopology: true,
} )
.then(() => console.log("MongoDB connected"))
.catch((err) => console.error("MongoDB connection failed:", err));
const userSchema = new mongoose.Schema({
email: String,
password: String,
});
const User = mongoose.model("User", userSchema);
app.post("/register", async (req, res) => {
const { email, password } = req.body;
const userExists = await User.findOne({ email });
if (userExists) {
return res.status(400).json({ message: "User already exists" });
}
const newUser = new User({ email, password });
await newUser.save();
res.status(201).json({ message: "User registered successfully" });
});
app.post("/login", async (req, res) => {
const { email, password } = req.body;
const user = await User.findOne({ email, password });
if (!user) {
return res.status(400).json({ message: "Invalid credentials" });
}
res.status(200).json({ message: "Login successful" });
});
const PORT = 5000;
app.listen(PORT, () => console.log(`Server running on http://localhost:${PORT}`));
```

Frontend :

- New terminal
- Cd frontend
- Cd login
- Npm run dev

➢ **Signup page:- (SignupPage.jsx)**

```
import React, { useState } from "react";

import axios from "axios";

const SignupPage = () => {

const [email, setEmail] = useState("");

const [password, setPassword] = useState("");

const [message, setMessage] = useState("");

const handleSignup = async (e) => {

e.preventDefault();

try {

const response = await axios.post("http://localhost:5000/register", { email, password });

setMessage(response.data.message);

} catch (error) {

setMessage("Signup failed. Please try again.");

} };

return (

<div className="h-screen flex items-center justify-center bg-gradient-to-r from-green-400 via- blue-

500 to-purple-600">

<div className="bg-white p-8 rounded-xl shadow-xl w-96 transform transition-all hover:scale-105

duration-500">
```

```
<h1 className="text-3xl font-semibold text-center text-gray-800 mb-6">Signup</h1>
<form onSubmit={handleSignup}>
<input
type="email"
placeholder="Email"
className="w-full p-3 mb-4 border-2 border-gray-300 rounded-lg focus:outline-none focus:ring-2
focus:ring-green-500 transition-all duration-300"
value={email}
onChange={(e) => setEmail(e.target.value)}
required />
<input
type="password"
placeholder="Password"
className="w-full p-3 mb-6 border-2 border-gray-300 rounded-lg focus:outline-none focus:ring-2
focus:ring-green-500 transition-all duration-300"
value={password}
onChange={(e) => setPassword(e.target.value)}
required />
<button className="w-full p-3 bg-green-500 text-white rounded-lg hover:bg-green-600 transition-all
duration-300">
Signup
</button>
</form>
{message && <p className="text-red-500 mt-4 text-center">{message}</p>}
</div>
```

```
</div>
  );
 };
export default SignupPage;
```

- **login page:- (LoginPage.jsx)**

```
import React, { useState } from "react";
import axios from "axios";
const LoginPage = () => {
const [email, setEmail] = useState("");
const [password, setPassword] = useState("");
const [message, setMessage] = useState("");
const handleLogin = async (e) => {
e.preventDefault();
try {
const response = await axios.post("http://localhost:5000/login", { email, password });
setMessage(response.data.message);
} catch (error) {
setMessage("Login failed. Please try again.");
}   };
return (
<div className="h-screen flex items-center justify-center bg-gradient-to-r from-purple-400 via-pink-
500 to-red-500">
 <div className="bg-white p-8 rounded-xl shadow-xl w-96 transform transition-all hover:scale-105
duration-500">
```

```
<h1 className="text-3xl font-semibold text-center text-gray-800 mb-6">Login</h1>
<form onSubmit={handleLogin}>
<input
type="email"
placeholder="Email"
className="w-full p-3 mb-4 border-2 border-gray-300 rounded-lg focus:outline-none focus:ring-2
focus:ring-blue-500 transition-all duration-300"
value={email}
onChange={(e) => setEmail(e.target.value)}
required />
<input
type="password"
placeholder="Password"
className="w-full p-3 mb-6 border-2 border-gray-300 rounded-lg focus:outline-none focus:ring-2
focus:ring-blue-500 transition-all duration-300"
value={password}
onChange={(e) => setPassword(e.target.value)}
required />
<button className="w-full p-3 bg-blue-500 text-white rounded-lg hover:bg-blue-600 transition-all
duration-300">
Login
</button>
</form>
{message && <p className="text-red-500 mt-4 text-center">{message}</p>}
```

```
 </div>
</div>
  );
  };
export default LoginPage;
```

- **App.jsk :-**

```
import React from "react";
import { BrowserRouter as Router, Route, Routes } from "react-router-dom";
import LoginPage from "./LoginPage";
import SignupPage from "./SignupPage";
const App = () => {
return (
<Router>
<Routes>
<Route path="/login" element={<LoginPage />} />
<Route path="/signup" element={<SignupPage />} />
</Routes>
</Router>
);
};
export default App;
```

4. Output:

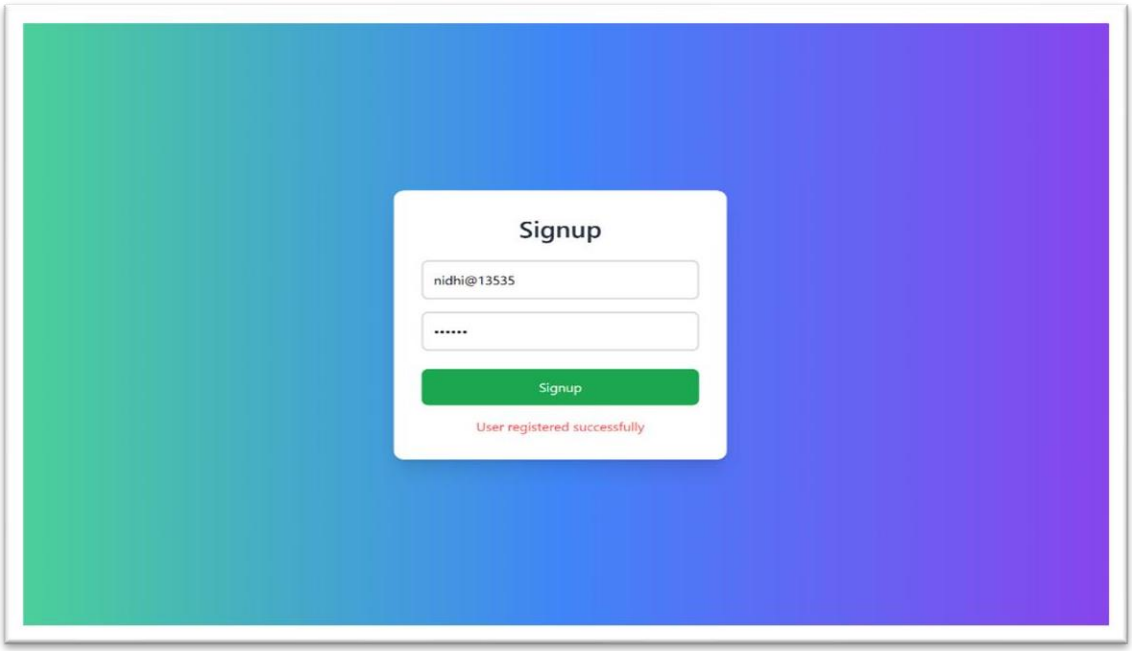

Figure 1:- Signup Page

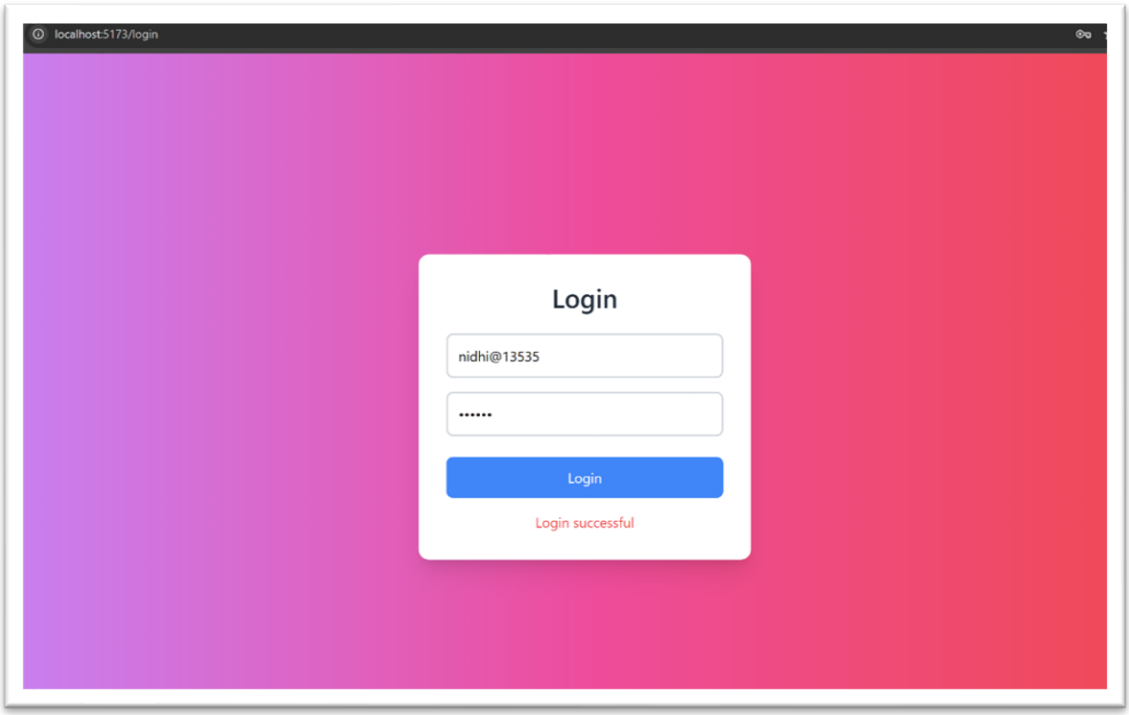

Figure 2:- Login Page

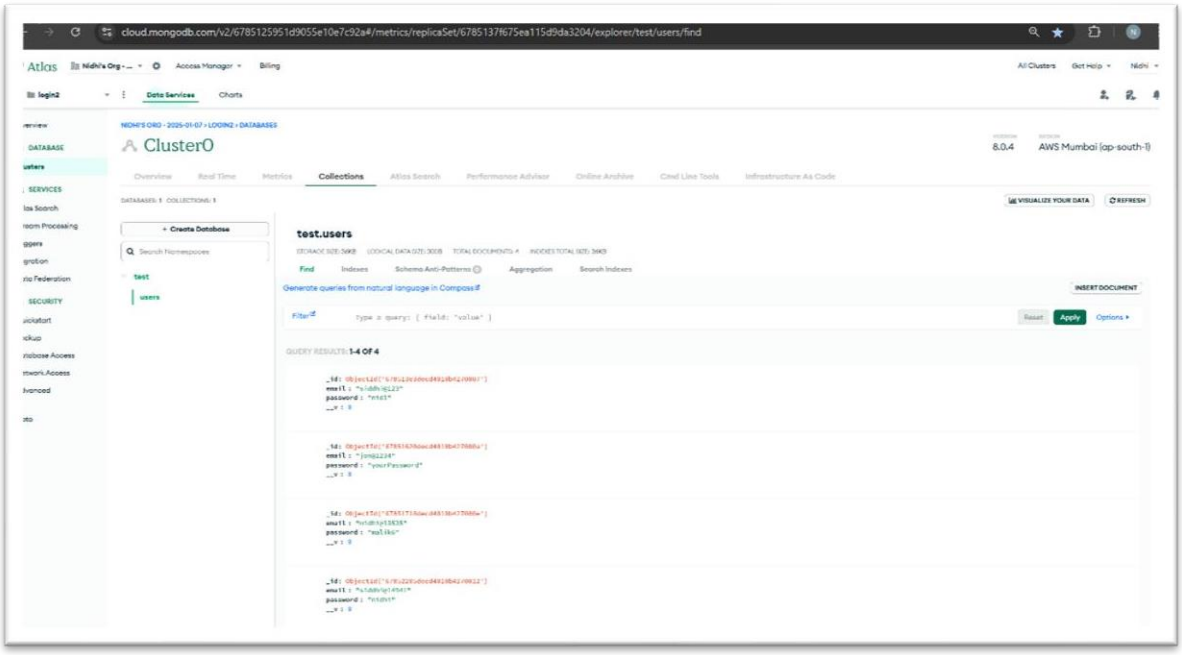

Figure 3:- Mongodb

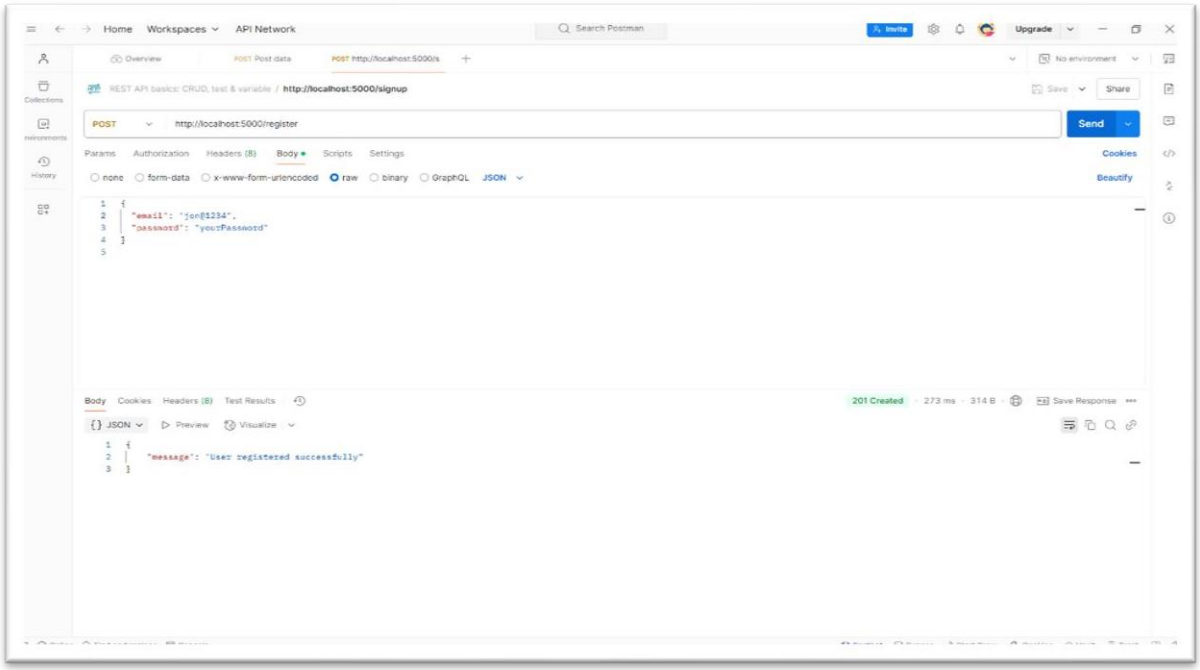

Figure 4:-Testing using Postman

4. **Learning Outcome:**

- **Understanding MongoDB:** Gain knowledge of how to use MongoDB as a NoSQL database for storing and retrieving user information.
- **Node.js Basics:** Learn how to set up and use Node.js as a backend server to process and respond to API requests.
- **Exploring Express.js:** Acquire skills to use Express.js for creating routes and managing HTTP requests on the Node.js server.
- **Frontend with React:** Learn to build a simple user interface with React for handling interactions such as login and signup.
- **API Testing:** Use tools like Postman to test backend APIs and verify that the server responds correctly.
- **Full-Stack Integration:** Combine the React frontend with the backend API to build a complete authentication system.