**Experiment - 1**

**Student Name: Ajay**

**Branch: BE-CSE Semester: 6th**

**Subject Name: AP Lab - 2**

**UID: 22BCS12696**

**Section/Group: NTPP - 603/B Date of Performance: 07/01/24 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.
2. **Objective:**
3. Learn about MongoDB: Understand how to use MongoDB as a NoSQL database for storing and retrieving user data.
4. Learn about Node.js: Understand how to set up and use Node.js as a backend server and handle API requests.
5. Learn about Express.js: Understand how to use Express.js to create routes and handle HTTP requests in the Node.js server.
6. Learn about React: Learn how to create a simple frontend interface with React to handle user interactions (login/signup).
7. Backend API Testing: Use tools like Postman to test backend APIs and ensure the server is responding correctly.
8. Integration: Integrate the frontend (React) with the backend API to create a full-stack authentication system.
9. **Implementation/Code:**

**//**REACT CODE:-

import React, { useState } from 'react'; import axios from 'axios';

const Login = () => {

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

const [password, setPassword] = useState(''); const [message, setMessage] = useState('');

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

try {

const response = await axios.post(['http://localhost:5000/api/auth/login',](http://localhost:5000/api/auth/login%27) {

email, password,

});

setMessage(response.data.message);

localStorage.setItem('token', response.data.token); // Save JWT to local storage

} catch (error) {

setMessage(error.response?.data?.message || 'Login failed');

}

};

return (

<div style={{ maxWidth: '400px', margin: 'auto', padding: '20px' }}>

<h2>Login</h2>

<form onSubmit={handleSubmit}>

<div>

<label>Email</label>

<input type="email" value={email}

onChange={(e) => setEmail(e.target.value)} required

/>

</div>

<div>

<label>Password</label>

<input type="password" value={password}

onChange={(e) => setPassword(e.target.value)} required

/>

</div>

<button type="submit">Login</button>

</form>

{message && <p>{message}</p>}

</div>

);

};

export default Login;

//EXPRESS

// Login Route

router.post('/login', async (req, res) => { const { email, password } = req.body;

try {

const user = await User.findOne({ email }); if (!user) {

return res.status(400).json({ message: 'User not found' });

}

const isMatch = await bcrypt.compare(password, user.password); if (!isMatch) {

return res.status(400).json({ message: 'Invalid credentials' });

}

const token = jwt.sign({ id: user.\_id }, process.env.JWT\_SECRET, { expiresIn: '1h',

});

res.status(200).json({ message: 'Login successful', token });

} catch (error) {

res.status(500).json({ message: 'Server error' });

}

});

module.exports = router;

//MONGODB

const mongoose = require('mongoose');

const connectDB = async () => { try {

await mongoose.connect(process.env.MONGO\_URI, { useNewUrlParser: true,

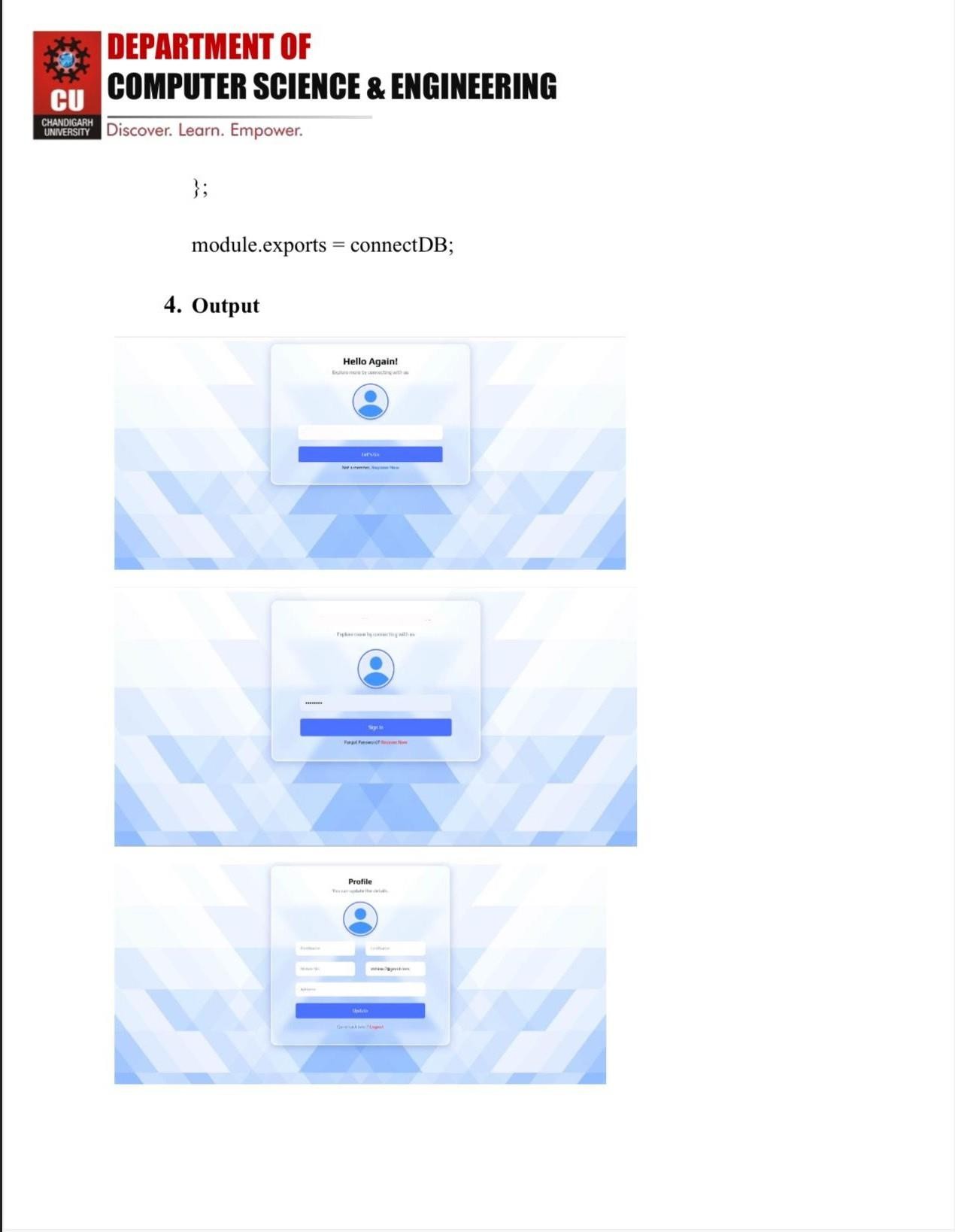
useUnifiedTopology: true,

});

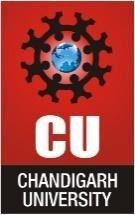
console.log('MongoDB connected...');

} catch (error) { console.error(error.message); process.exit(1); // Exit process with failure

}



# DEPARTMENT OF



COMPUTER SCIENCE & ENGINEERING

1. **Learning Outcome:**
   1. We Learn About the use of React.
   2. We Learn About the use of Express.
   3. We Learn About the use of MongoDB.
   4. We learn About the Connection.
   5. We Learn About the Calling For the Username and password.