



## Experiment 1

**Student Name** Atul Kumar

**UID** 22BCS10741

**Branch:** BE-CSE

**Section/Group:** 22BCS\_IOT\_603A

**Semester:** 6th

**Date of Performance:** 08-01-25

**Subject Name:** AP Lab-2

**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.

- ☐ Problem 1.1.1: Give understanding of MongoDB, Nodejs, React, Express.
- ☐ Problem 1.1.2: Create a Frontend design of Login/Signup pages and create a backend of it.

Problem Breakdown.

- ☐ Problem 1.1.3: Test the Backend API Using Postman.

### **2. Objective:**

1. Learn about MongoDB: Understand how to use MongoDB as a NoSQL database for storing and retrieving user data.
2. Learn about Node.js: Understand how to set up and use Node.js as a backend server and handle API requests.
3. Learn about Express.js: Understand how to use Express.js to create routes and handle HTTP requests in the Node.js server.
4. Learn about React: Learn how to create a simple frontend interface with React to handle user interactions (login/signup).
5. Backend API Testing: Use tools like Postman to test backend APIs and ensure the server is responding correctly.
6. Integration: Integrate the frontend (React) with the backend API to create a full-stack authentication system.

### **3. Implementation/Code:**



# DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
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 {
      // Send login request to the server
      const response = await axios.post('http://localhost:5000/api/auth/login', {
        email,
        password,
      });

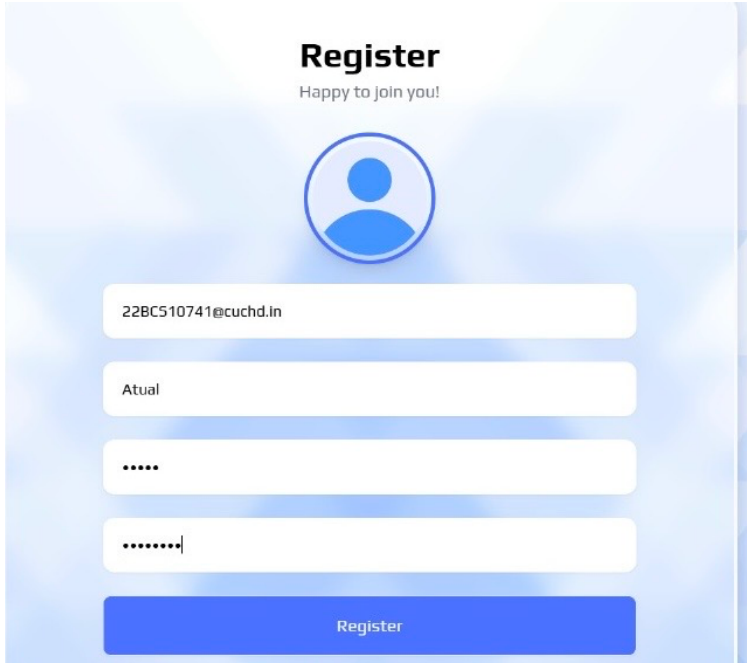
      // Update message and save the token
      setMessage(response.data.message);
      localStorage.setItem('token', response.data.token); // Store JWT token
    } catch (error) {
      // Show error message if login fails
      setMessage(error.response?.data?.message || 'Login failed');
    }
  };

  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;
```

## 4. Output



**Register**  
Happy to join you!



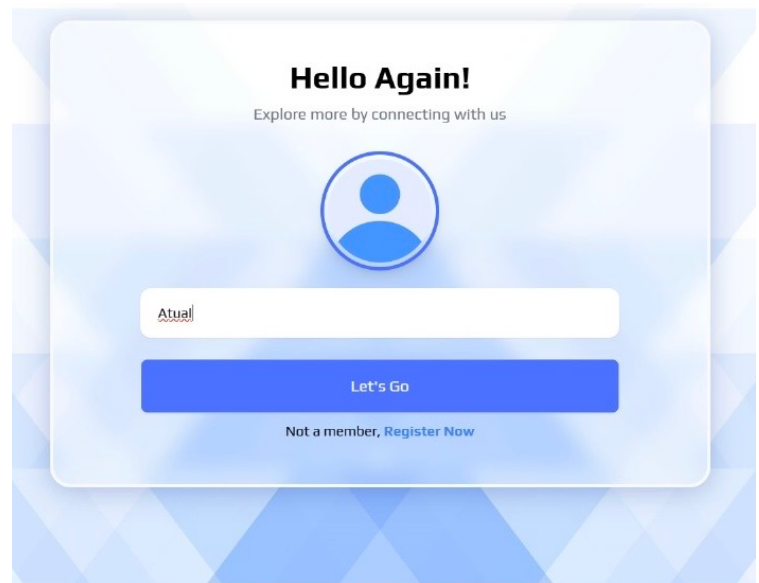
22BC510741@cuchd.in

Atual


.....

.....

Register



**Hello Again!**  
Explore more by connecting with us



Atual

Let's Go

Not a member, [Register Now](#)



# **DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

Discover. Learn. Empower.

## **5. Learning Outcome:**

- i. We Learn About the use of React.
- ii. We Learn About the use of Express.
- iii. We Learn About the use of MongoDB.
- iv. We learn About the Connection.
- v. We Learn About the Calling For the Username and password.