Ex. No. 10	Arrays and Routing using Angular JS
Date of Exercise	24/10/2025

Aim:

To develop a **Node.js server-side web application** that handles **user registration**, processes HTML form data, stores the submitted information in a **MongoDB database**, and retrieves and displays all user details dynamically in an HTML table using **Mongoose**.

Description:

This project demonstrates how to build a **server-side web application** using **Node.js** and **MongoDB** for managing user registration data.

The application performs the following major functions:

1. **Display a Registration Form:**
    - When the user accesses the URL http://localhost:6000, the server serves an HTML form (home.html) that collects user details such as **Name**, **Email**, **Phone**, **Address**, etc.
    - All fields are mandatory and validated using HTML5 form validation attributes (like required, type="email", and pattern).
2. **Process Form Submission:**
    - Upon clicking the "Submit" button, the form data is sent to the server via POST method to the URL http://localhost:6000/save.
    - The Node.js server receives and parses the form data.
3. **Store Data in MongoDB:**
    - The application connects to a **MongoDB database** named **"ecommerce"** using the **Mongoose ODM (Object Data Modeling)** library.
    - The data is then stored in a **collection** named **"customers"** with fields such as name, email, phone, and address.

4. **View All User Details:**
    - The home page also contains a hyperlink **"View All User Details"**.
    - When clicked, the server retrieves all records from the **"customers"** collection and displays them in an HTML table format using Node.js response streaming.

Technologies and Modules Used

- **Node.js:** For creating the backend server.
- **Express (optional enhancement):** For handling routing (or http module can be used).
- **Mongoose:** For connecting to and managing MongoDB.
- **MongoDB:** For storing customer registration data.
- **HTML5 & CSS:** For creating and validating the frontend form.

Application Workflow

1. The server starts at port **6000**.
2. When the home page is loaded, a **registration form** is displayed.
3. After submission, the form data is processed and stored in **MongoDB**.
4. The user can then click **"View All User Details"** to see all stored customer data in an HTML table.

Program:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>User Registration Form</title>
```

```
    <link rel="stylesheet" href="/style.css">
</head>
<body>
    <div class="container">
        <!-- Registration Form Section -->
        <div id="formSection" class="form-section">
            <div class="form-container">
                <h1>User Registration Form</h1>
                <!-- Display validation errors -->
                <div id="errorMessages" class="error-messages" style="display:none;"></div>
                <form id="registrationForm" class="registration-form">
                    <div class="form-row">
                        <label for="name">Name<span class="required">*</span></label>
                        <input type="text" id="name" name="name" placeholder="John David"
required>
                    </div>
                    <div class="form-row">
                        <label for="password">Password<span class="required">*</span></label>
                        <input type="password" id="password" name="password" placeholder="••••••••"
required>
                    </div>

                    <div class="form-row">
                        <label for="age">Age<span class="required">*</span></label>
```

```
            <input type="number" id="age" name="age" placeholder="30" min="1"
max="120" required>
        </div>

        <div class="form-row">
            <label for="mobileNumber">Mobile Number<span
class="required">*</span></label>
            <input type="tel" id="mobileNumber" name="mobileNumber"
placeholder="9923457891" maxlength="10" required>
        </div>

        <div class="form-row">
            <label for="email">Email<span class="required">*</span></label>
            <input type="email" id="email" name="email"
placeholder="johndavid@gmail.com" required>
        </div>

        <div class="form-row gender-row">
            <label>Gender<span class="required">*</span></label>
            <div class="gender-options">
                <label class="radio-label">
                    <input type="radio" name="gender" value="Male" required> Male
                </label>
                <label class="radio-label">
                    <input type="radio" name="gender" value="Female"> Female
```

```
        </label>
    </div>
</div>

<div class="form-row">
    <label for="state">State<span class="required">*</span></label>
    <select id="state" name="state" required>
        <option value="">Select State</option>
        <option value="Andhra Pradesh">Andhra Pradesh</option>
        <option value="Arunachal Pradesh">Arunachal Pradesh</option>
        <option value="Assam">Assam</option>
        <option value="Bihar">Bihar</option>
        <option value="Chhattisgarh">Chhattisgarh</option>
        <option value="Goa">Goa</option>
        <option value="Gujarat">Gujarat</option>
        <option value="Haryana">Haryana</option>
        <option value="Himachal Pradesh">Himachal Pradesh</option>
        <option value="Jharkhand">Jharkhand</option>
        <option value="Karnataka">Karnataka</option>
        <option value="Kerala">Kerala</option>
        <option value="Madhya Pradesh">Madhya Pradesh</option>
        <option value="Maharashtra">Maharashtra</option>
        <option value="Manipur">Manipur</option>
```

```
                <option value="Meghalaya">Meghalaya</option>
                <option value="Mizoram">Mizoram</option>
                <option value="Nagaland">Nagaland</option>
                <option value="Odisha">Odisha</option>
                <option value="Punjab">Punjab</option>
                <option value="Rajasthan">Rajasthan</option>
                <option value="Sikkim">Sikkim</option>
                <option value="Tamil Nadu">Tamil Nadu</option>
                <option value="Telangana">Telangana</option>
                <option value="Tripura">Tripura</option>
                <option value="Uttar Pradesh">Uttar Pradesh</option>
                <option value="Uttarakhand">Uttarakhand</option>
                <option value="West Bengal">West Bengal</option>
            </select>
        </div>
        <div class="form-actions">
            <button type="submit" class="btn btn-submit">Submit</button>
        </div>
    </form>

    <div class="view-link">
        <a href="javascript:void(0);" onclick="showTableView()">View All User Details</a>
    </div>
```

```
        </div>
    </div>
    <!-- Customer Details Table Section -->
    <div id="tableSection" class="table-section" style="display:none;">
        <div class="table-container">
            <h2>List Customer Details</h2>
            <div id="tableContent"></div>

            <div class="view-link">
                <a href="javascript:void(0);" onclick="showFormView()">Back to Registration Form</a>
            </div>
        </div>
    </div>
</div>

<script>
    // Form submission handler
    document.getElementById('registrationForm').addEventListener('submit', async function(e) {
        e.preventDefault();
        const formData = {
            name: document.getElementById('name').value,
            password: document.getElementById('password').value,
```

```
        age: parseInt(document.getElementById('age').value),
        mobileNumber: document.getElementById('mobileNumber').value,
        email: document.getElementById('email').value,
        gender: document.querySelector('input[name="gender"]:checked').value,
        state: document.getElementById('state').value
    };
    try {
        const response = await fetch('/save', {
            method: 'POST',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(formData)
        });

        const result = await response.json();

        if (result.success) {
            // Reset form
            document.getElementById('registrationForm').reset();
            document.getElementById('errorMessages').style.display = 'none';

            // Show table view
```

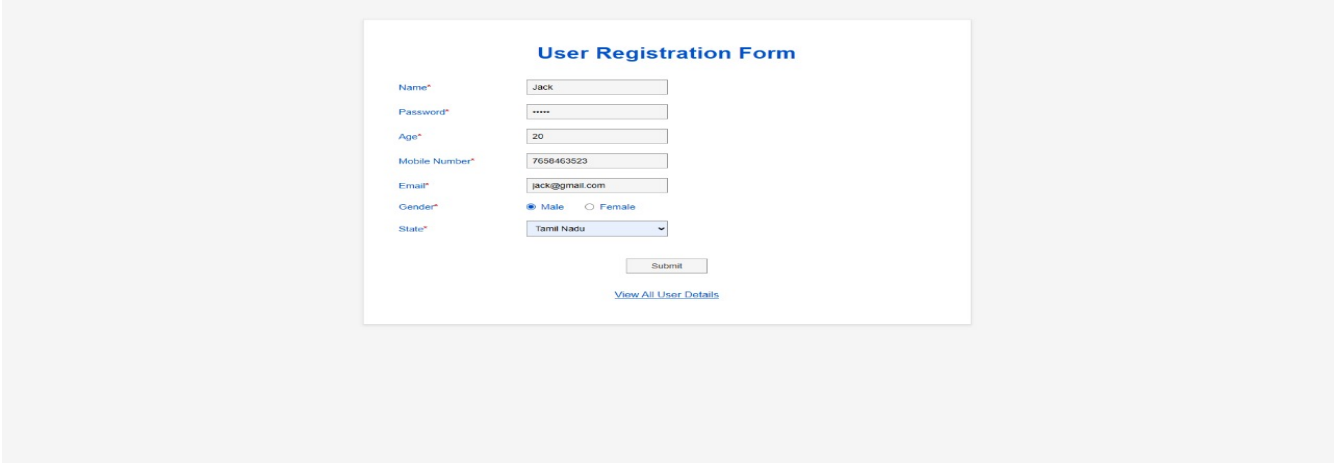
```
            showTableView();
        } else {
            // Show errors
            displayErrors(result.errors);
        }
    } catch (error) {
        console.error('Error:', error);
        displayErrors([{ msg: 'An error occurred. Please try again.' }]);
    }
});

function displayErrors(errors) {
    const errorDiv = document.getElementById('errorMessages');
    errorDiv.innerHTML = '';
    errors.forEach(error => {
        const p = document.createElement('p');
        p.className = 'error';
        p.textContent = error.msg;
        errorDiv.appendChild(p);
    });
    errorDiv.style.display = 'block';
}
async function showTableView() {
```

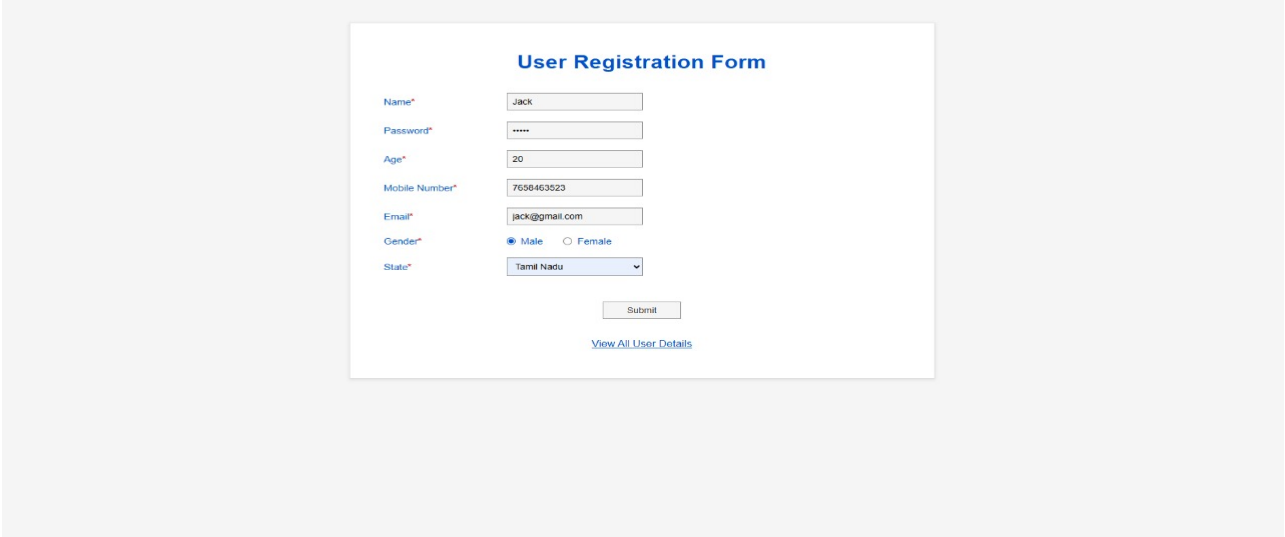
```
        try {
            const response = await fetch('/api/customers');
            const result = await response.json();

            if (result.success) {
                displayCustomersTable(result.data);
                document.getElementById('formSection').style.display = 'none';
                document.getElementById('tableSection').style.display = 'block';
            }
        } catch (error) {
            console.error('Error fetching customers:', error);
        }
    }
    function displayCustomersTable(customers) {
        const tableContent = document.getElementById('tableContent');

        if (customers.length === 0) {
            tableContent.innerHTML = '<p class="no-customers">No customers registered yet.</p>';
            return;
        }
        let table = '<table class="customer-table"><thead><tr>';
        table += '<th>Name</th><th>Password</th><th>Age</th><th>Mobile Number</th>';
        table += '<th>Email</th><th>Gender</th><th>State</th>';
```

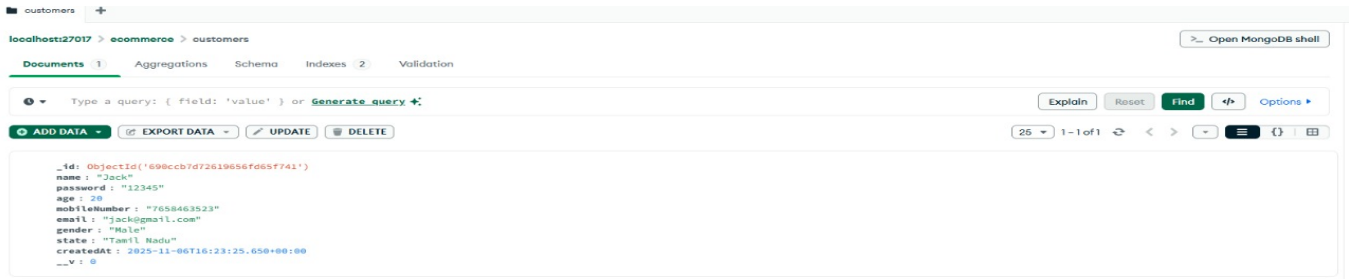
```
        table += '</tr></thead><tbody>';
        customers.forEach(customer => {
            const maskedPassword = '•'.repeat(customer.password.length);
            table += '<tr>';
            table += `<td>${customer.name}</td>`;
            table += `<td>${maskedPassword}</td>`;
            table += `<td>${customer.age}</td>`;
            table += `<td>${customer.mobileNumber}</td>`;
            table += `<td>${customer.email}</td>`;
            table += `<td>${customer.gender}</td>`;
            table += `<td>${customer.state}</td>`;
            table += '</tr>';
        });

        table += '</tbody></table>';
        tableContent.innerHTML = table;
    }

    function showFormView() {
        document.getElementById('tableSection').style.display = 'none';
        document.getElementById('formSection').style.display = 'block';
    }
</script>
```

```
</body>

</html>
```

Output:







Result:

Successfully designed and implemented a Node.js server-side web application integrated with a MongoDB database for user registration, data storage, and retrieval, and executed the application to dynamically display all user details in a tabular format.