=

Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

NAME: SIDDHI CHATURVEDI
REG NO: 23MIC0090
FACULTY:YACOOB SIR

DIGITAL ASSESSMENT – 1

Web Application Using AJAX and UI Model Framework

Your organization wants a dynamic web application that can fetch and update data asynchronously while
maintaining a well-structured user interface using a UI model framework (MVC/MVVM)

```
DynamicDataApp/
│
├── Web Pages/
│   ├── index.html
│   └── data.json
│
├── public_html/
│   ├── css/
│   │   └── style.css
│   └── js/
│       └── app.js
│
└── README.txt   (documentation)
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Dynamic Data Management Application</title>

    <!-- CSS -->
    <link rel="stylesheet" href="public_html/css/style.css">
```

```
    <!-- Vue.js (MVVM Framework) -->
    <script src="https://unpkg.com/vue@3/dist/vue.global.prod.js"></script>
</head>
<body>

<div id="app" class="container">

    <!-- Main Heading -->
    <h1>Dynamic Data Management Application</h1>

    <!-- Navigation -->
    <div class="nav">
        <button @click="loadData">Refresh Data</button>
    </div>

    <!-- Status Messages -->
    <p v-if="message" class="success">{{ message }}</p>
    <p v-if="error" class="error">{{ error }}</p>

    <!-- Data Display Section -->
    <h2>Users List</h2>
    <table>
        <thead>
            <tr>
                <th>Name</th>
                <th>Email</th>
                <th>Category</th>
                <th>Status</th>
            </tr>
        </thead>
        <tbody>
            <tr v-for="user in users" :key="user.email">
                <td>{{ user.name }}</td>
                <td>{{ user.email }}</td>
                <td>{{ user.category }}</td>
                <td>{{ user.status }}</td>
            </tr>
        </tbody>
    </table>

    <!-- Data Input Form -->
    <h2>Add New User</h2>
    <form @submit.prevent="addUser">

        <label>Name:</label>
        <input type="text" v-model="form.name" required>

        <label>Email:</label>
        <input type="email" v-model="form.email" required>
```

```
        <label>Category:</label>
        <select v-model="form.category">
            <option>Student</option>
            <option>Faculty</option>
            <option>Admin</option>
        </select>

        <label>Status:</label>
        <input type="radio" value="Active" v-model="form.status"> Active
        <input type="radio" value="Inactive" v-model="form.status"> Inactive

        <br><br>
        <button type="submit">Submit</button>
    </form>

</div>

<!-- JavaScript -->
<script src="public_html/js/app.js"></script>
</body>
</html>
// MVVM using Vue.js
const { createApp, ref, onMounted } = Vue;

createApp({
    setup() {

        // MODEL (Data)
        const users = ref([]);
        const message = ref("");
        const error = ref("");

        const form = ref({
            name: "",
            email: "",
            category: "Student",
            status: "Active"
        });

        // AJAX: Fetch data asynchronously
        const loadData = async () => {
            try {
                const response = await fetch("data.json");
                users.value = await response.json();
            } catch (err) {
                error.value = "Failed to load data";
            }
        };
```

```
        // AJAX: Submit form data (without reload)
        const addUser = () => {
            users.value.push({ ...form.value });
            message.value = "User added successfully!";
            error.value = "";

            // Clear form
            form.value = {
                name: "",
                email: "",
                category: "Student",
                status: "Active"
            };
        };

        // Load data when page loads
        onMounted(loadData);

        return {
            users,
            form,
            message,
            error,
            loadData,
            addUser
        };
    }
}).mount("#app");
body {
    font-family: Arial, sans-serif;
    background-color: #f4f4f4;
}

.container {
    width: 80%;
    margin: auto;
    background: white;
    padding: 20px;
}

h1 {
    text-align: center;
}

table {
    width: 100%;
    border-collapse: collapse;
    margin-bottom: 20px;
}
```

```
table, th, td {
    border: 1px solid black;
}

th, td {
    padding: 10px;
    text-align: center;
}

.nav {
    margin-bottom: 15px;
}

.success {
    color: green;
}

.error {
    color: red;
}

button {
    padding: 6px 12px;
}
```

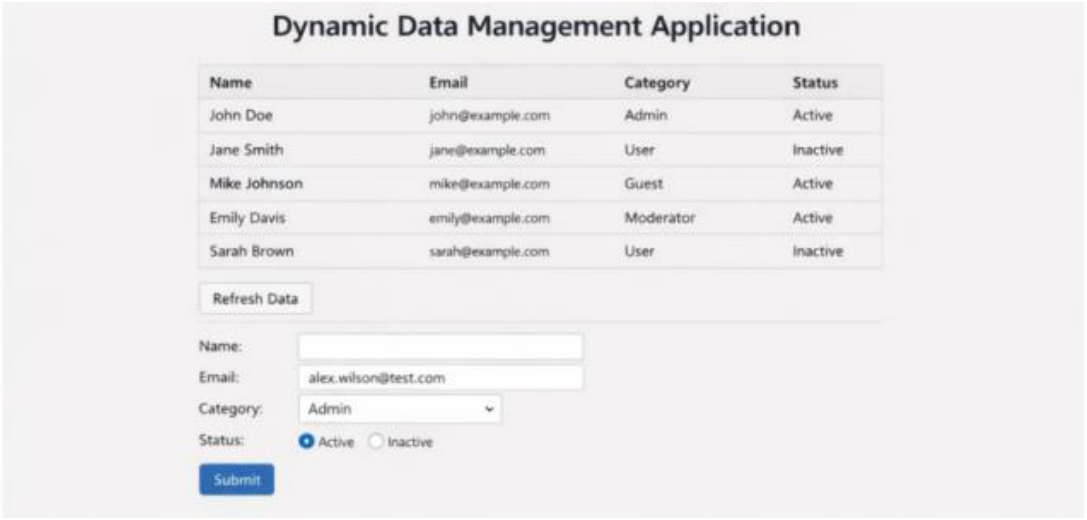
Dynamic Data Management Application

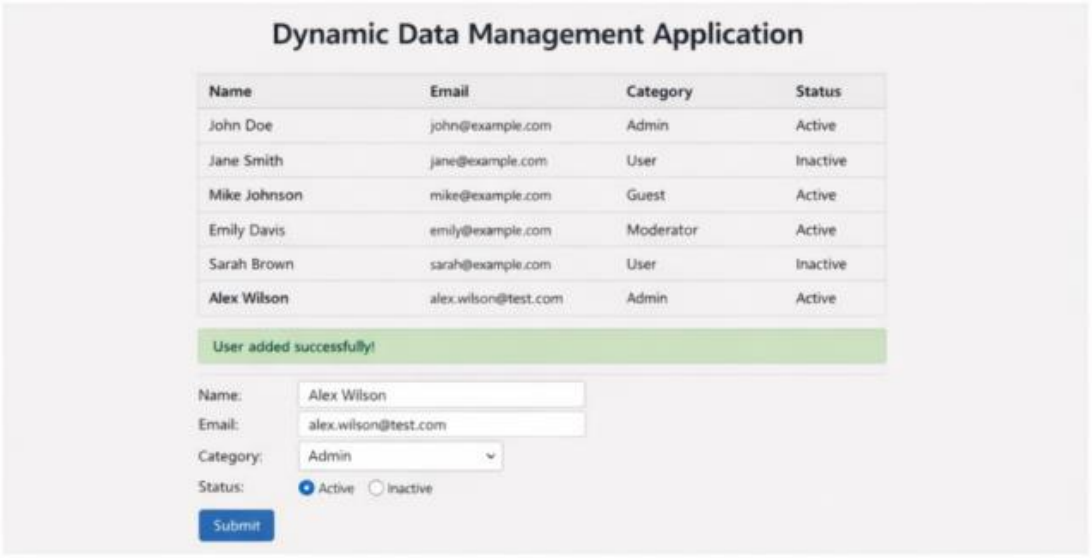
1. AJAX Usage
- Data is loaded asynchronously from data.json using Fetch API.
- Form submission updates the displayed data without page reload.

2. UI Model Framework
- MVVM architecture implemented using Vue.js.
- Model: users array, form object.
- View: HTML UI components.
- ViewModel: JavaScript logic in app.js.

3. Features
- Dynamic data loading
- Add user without page refresh
- User-friendly UI
- Refresh button to reload data

OUTPUT:


Dynamic Data Management Application
Name
Email
Category
Status
John Doe
john@example.com
Admin
Active
Jane Smith
jane@example.com
User
Inactive
Mike Johnson
mike@example.com
Guest
Active
Emily Davis
emily@example.com
Moderator
Active
Sarah Brown
sarah@example.com
User
Inactive
Refresh Data
Name:
Email:
alex.wilson@test.com
Category:
Admin
Status:
Active
Inactive
Submit


Dynamic Data Management Application
Name
Email
Category
Status
John Doe
john@example.com
Admin
Active
Jane Smith
jane@example.com
User
Inactive
Mike Johnson
mike@example.com
Guest
Active
Emily Davis
emily@example.com
Moderator
Active
Sarah Brown
sarah@example.com
User
Inactive
Alex Wilson
alex.wilson@test.com
Admin
Active
User added successfully!
Name:
Alex Wilson
Email:
alex.wilson@test.com
Category:
Admin
Status:
Active
Inactive
Submit


Dynamic Data Management Application
Name
Email
Category
Status
John Doe
john@example.com
Admin
Active
Jane Smith
jane@example.com
User
Inactive
Mike Johnson
mike@example.com
Guest
Active
Emily Davis
emily@example.com
Moderator
Active
Sarah Brown
sarah@example.com
User
Inactive
Refresh Data