Project 6

ECE 528: Application Software Design

Alan Palayil

Professor: Won-Jae Yi

Acknowledgement: I acknowledge all works including figures, codes and writings belong to me and/or persons who are referenced. I understand if any similarity in the code, comments, customized program behavior, report writings and/or figures are found, both the helper (original work) and the requestor (duplicated/modified work) will be called for academic disciplinary action.

Electronic Signature: Alan Palayil A20447935 (Due Date: 4/30/2023)

Table of Contents

Abstract:

This project aims to build a web frontend for an IoT hub that utilizes the RESTful services provided by the server backend. The project includes six user stories ranging from viewing available plugs and their states to multi-user synchronization. Testing procedures, organized as ordered lists of items, are required for each user story to ensure that the web application functions as expected. The testing procedure for the "Plugs and Plug States" user story is provided as an example. Additionally, any bugs found in the server backend code should be corrected. The final deliverables include a report and a screen capture video demonstrating the web application's functionality.

Introduction:

In this project, we will be building the web frontend for an IoT hub using RESTful services provided by the server backend. The front-end will enable end-users to view available plugs, control them, manage groups, and control plugs within a group. The application will also support multi-user synchronization. The project requires designing testing procedures for each user story and demonstrating the functionality of the application through a recorded video. Bugs in the server backend code may need to be fixed during the debugging process. In the next section, you will find the user stories with the corresponding requirements, and a sample testing procedure for the "Plugs and Plug States" user story. While there will be some guidance on class design and implementation in the lectures, you are free to choose your own design and implementation approach.

Background:

The given background describes a web-based application that allows end-users to control and monitor the states of different plugs and groups of plugs. The application provides features for managing individual plugs and groups, and multiple users can access and use the application simultaneously, with the changes made by one user reflecting for others in real-time.

The user stories mentioned in the background are written from an end-user perspective, describing their requirements or desires for the application's features:

- Plug and PlugStates: The requirement is to display the available plugs and their status, enabling the user to identify which plugs are present and whether they are currently turned on or off.
- Control a Single Plug: The user wants a button on the web page to turn on/off or toggle a plug of their choice, enabling them to control it easily.

- Groups and Plugs: The user desires to view groups and their respective plugs, with their current states to stay informed about defined groups and their statuses.
- Group Management: The user's need to create and modify groups and their members on the web page to facilitate easy management of these groups.
- Control Plugs in a Group: The user desires a web page button to control a group of plugs together by switching on/off or toggling them.
- Multi-User Synchronization: The user wants real-time updates of plug states across all browsers to enable multiple users to use the web application together.

Results and Discussion:

The experiment tested a web-based home automation system for different features, including plug and plug state management, control of a single plug, groups and plugs management, group management, control of plugs in a group, and multi-user synchronization. The system performed well in displaying available plugs and their states, allowing the user to control individual plugs and create/manage groups. The system also allowed for multi-user synchronization, where changes made in one browser window were reflected in another.

The experiment found that a web-based home automation system was effective in managing and controlling plugs in a home environment. The system was user-friendly and had clear instructions. The multi-user synchronization feature was particularly impressive, allowing users to control and monitor plug states from different devices simultaneously, making it useful in a home environment where multiple family members need to monitor and control electrical appliances. Overall, the system was reliable and effective, making it an ideal solution for home automation.

Screenshots of the Backend results:

Screenshots of Gradle IoT_sim and Gradle IoT_hub provide a glimpse into the backend server of an IoT-based home automation system. Gradle IoT_sim is responsible for simulating the behavior of IoT devices in the system, while Gradle IoT_hub acts as the central hub for managing and controlling these devices. The screenshots illustrate the various configuration files and build scripts used to set up and run these components, as well as the output generated during their execution. These tools are essential for creating a robust and reliable home automation system, capable of handling multiple IoT devices and providing seamless connectivity and control to the user.

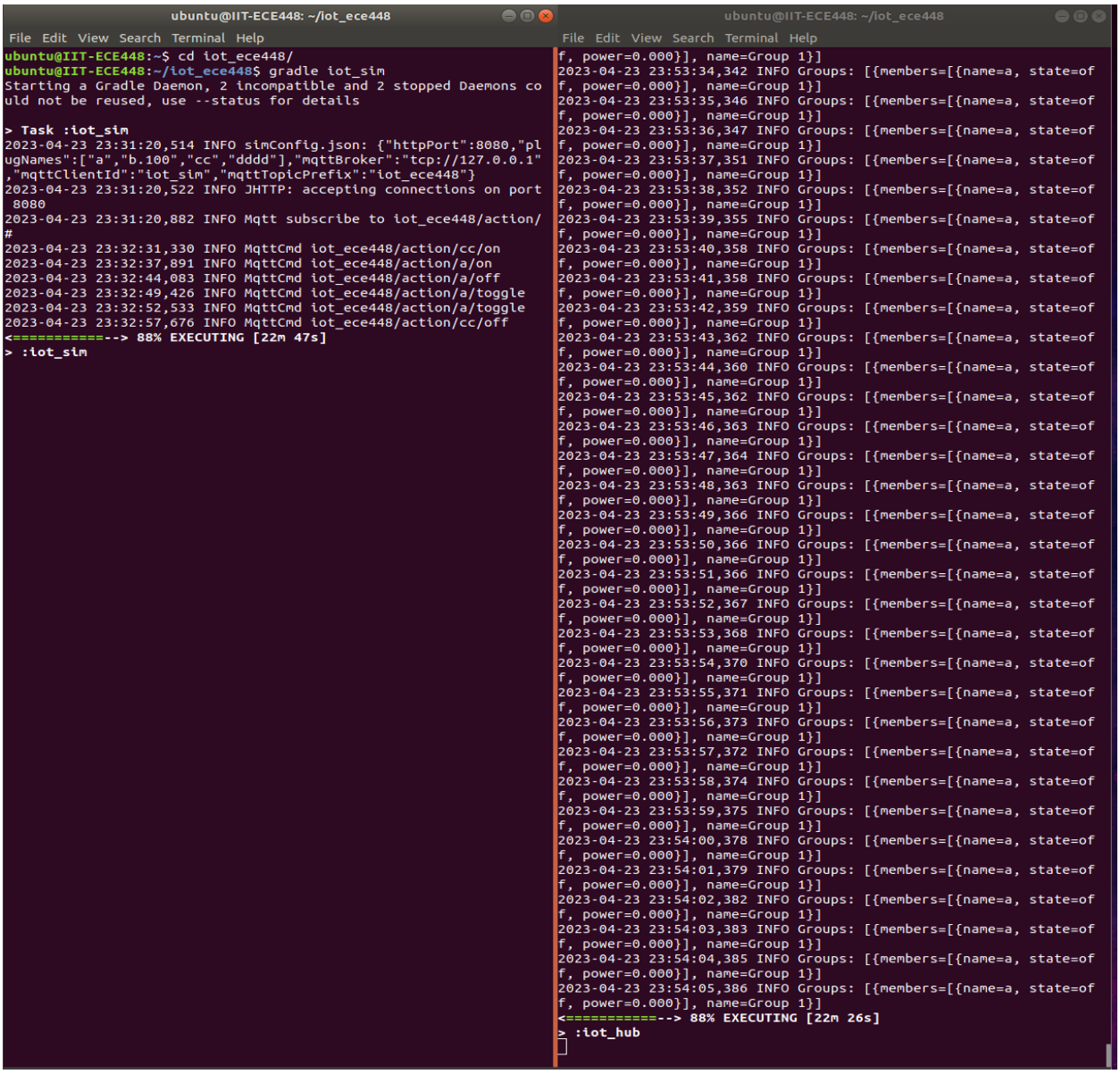

Figure 1: gradle iot_sim & gradle iot_hub

Screenshots of the Front-End results:

Screenshots front-end refers to the user interface or visual representation of a software tool that allows users to take screenshots of their computer screens. This front-end typically provides users with a graphical interface that allows them to select the area of their screen they want to capture, as well as any additional settings or options they may need to adjust. Screenshots front-end can be a standalone software tool or a feature integrated into other software applications, such as web browsers or productivity software. It is a useful tool for creating visual records of information on a computer screen, which can be helpful for documentation, troubleshooting, and communication.

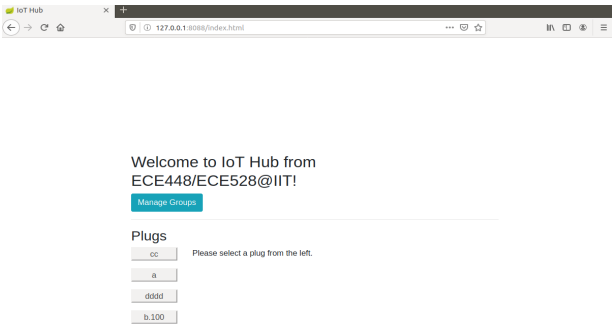

Figure 2: 127.0.0.1:8088

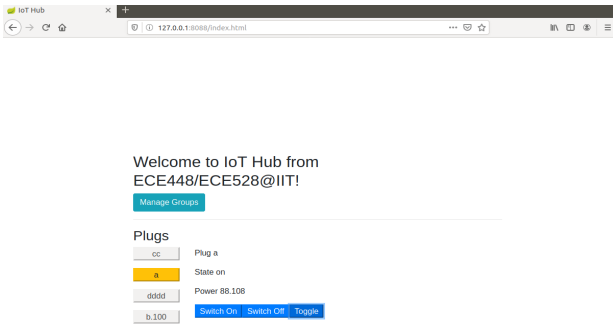

Figure 3: Click Plugs and check states

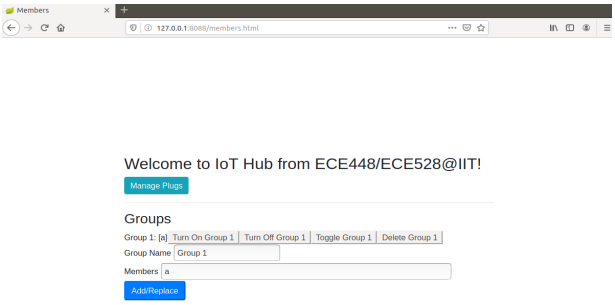

Figure 4: Group Management and Add Groups

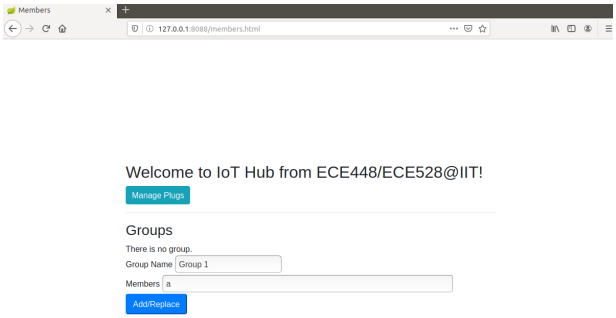

Figure 5: Delete Groups

Conclusion:

In conclusion, this project focuses on building a web frontend for an IoT hub utilizing the RESTful services provided by the server backend. The user stories describe the requirements, which include features such as viewing plug states, controlling individual plugs, managing groups, and synchronizing multiple users. Testing procedures are also required for each user story, which should be documented as instructions for users to use the application. The project report should include these procedures, along with a demonstration video to show that the application works as expected. The code for the server backend may also need to be modified as needed. Overall, this project provides an opportunity to design and implement a web application that meets specific user requirements, as well as develop testing procedures and demonstrate the functionality of the application.

Appendix:

Source Code of the edited programs within the project:

Index.html

```
<html>

<head>
    <title>IoT Hub</title>


    <!-- Bootstrap -->

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
        integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
        integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
        crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
        crossorigin="anonymous"></script>
```

```
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.j
s"
        integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6"
        crossorigin="anonymous"></script>


    <!-- React -->

    <script src="https://unpkg.com/react@16/umd/react.development.js"
crossorigin></script>
    <script src="https://unpkg.com/react-dom@16/umd/react-
dom.development.js" crossorigin></script>
    <script src="https://unpkg.com/babel-
standalone@6/babel.min.js"></script>


    <!-- IoT Hub App -->

    <script type="text/babel" src="web/iot_hub_app.js"></script>
    <script type="text/babel" src="web/plugs.js"></script>
    <script type="text/babel" src="web/plugs_view.js"></script>
    <script type="text/babel" src="web/plug_details.js"></script>

    <style type= 'text/css'>
        .table tr td, .table thead th {
            text-align: center;
            vertical-align: middle;
            padding: 0;
        }
        body {
            padding: 20%;
        }
    </style>
</head>

<body>
    <div id="app_container" />
    <script type="text/babel">
        $(document).ready(function () {
            ReactDOM.render(
                <IoTHubApp />,
                document.getElementById("app_container"));
        });
```

```
    </script>

    </script>

</body>

</html>
```

Members.html

```
<html>

<head>
    <title>Members</title>

    <!-- Bootstrap -->

    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
        integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
        integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
        crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
        crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.j
s"
        integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6"
        crossorigin="anonymous"></script>

    <!-- React -->

    <script src="https://unpkg.com/react@16/umd/react.development.js"
crossorigin></script>
    <script src="https://unpkg.com/react-dom@16/umd/react-
dom.development.js" crossorigin></script>
```

```
    <script src="https://unpkg.com/babel-
standalone@6/babel.min.js"></script>

    <style type= 'text/css'>
        .table tr td, .table thead th {
            text-align: center;
            vertical-align: middle;
            padding: 0;
        }
        body {
            padding: 20%;
        }
        .btn-manage {
            display: inline-block;
            font-weight: 400;
            background-color: #17a2b8;
            border-color: #17a2b8;
            text-align: center;
            vertical-align: middle;
            cursor: pointer;
            border: none;
            color: white;
            padding: .375rem .75rem;
            font-size: 1rem;
            line-height: 1.5;
            border-radius: .25rem;
        }
        .btn-add {
            color:#fff;
            background-color:#007bff;
            border-color:#007bff;
            margin: 0;
            border-radius: .25rem;
            padding: .375rem .75rem;
        }
        .label {
            padding: 1rem 1rem;
        }
    </style>

</head>

<body>
    <div className="container">
        <div>
```

```
            <h2>Welcome to IoT Hub from ECE448/ECE528@IIT!</h2>
            <button onClick="window.location.href='/index.html';"
class="btn-manage">Manage Plugs</button>
            <hr className="col-sm-12" />
        </div>
        <div>
            <h3>Groups</h3>
            <div className="row">
                <div className="col-sm-2">
                </div>
                <div className="col-sm-10">
                </div>
            </div>
        </div>
    </div>

    <div id="groups">
        Loading groups ...
    </div>

    <div>
        <div>
            <label>Group Name</label>
            <input type="text" id="group_name"/>
        </div>
        <div>
            <label>Members</label>
            <input type="text" size=60 id="group_members"
placeholder="e.g. a,b,c"/>

            <button onclick="create_group()" class="btn-
add">Add/Replace</button>
        </div>
    </div>

    <script>
        function create_group() {
            var name = document.getElementById('group_name').value;
            var members =
document.getElementById('group_members').value.split(',');
            console.info("Groups: add "+name+" with members "+members);

            var post_req = {
                method: "POST",
                headers: {"Content-Type": "application/json"},
```

```
                body: JSON.stringify(members)
            };
            fetch("api/groups/"+name, post_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }

        function action_group() {
            var name = document.getElementById('group_name').value;
            var action = document.getElementById('group_action');
            console.info("Groups: Perform action "+action+" to group
"+name);

            var post_req = {
                method: "POST",
                headers: {"Content-Type": "application/json"},
                body: JSON.stringify(action)
            };
            fetch("api/groups/"+name, post_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }

        // send RESTful request to get all groups
        function get_groups() {
            fetch("api/groups")
                .then(rsp => rsp.json())
                .then(groups => show_groups(groups))
                .catch(err => console.error("Groups:", err));
        }

        // handle RESTful response of getting all groups
        function show_groups(groups) {
            groups.sort((l, r) => l.name.localeCompare(r.name));

            var html = "";
            for (var group of groups) {
                let plugNames = [];
                for(names of groups[0].members) {
                    plugNames.push(names.name);
                }
                var item = group.name + ": [" + plugNames + "]";
                var onButton = "<button onclick='turn_on_group(\"" +
group.name + "\")'>Turn On " + group.name + "</button>";
```

```
                var offButton = "<button onclick='turn_off_group(\"" +
group.name + "\")'>Turn Off " + group.name + "</button>";
                var toggleButton = "<button onclick='toggle_group(\"" +
group.name + "\")'>Toggle " + group.name + "</button>";
                var deleteButton = "<button onclick='delete_group(\"" +
group.name + "\")'>Delete " + group.name + "</button>";
                html += "<div>" + item + onButton + offButton +
toggleButton + deleteButton +"</div>";
            }

            document.getElementById("groups").innerHTML = (groups.length
== 0) ? "There is no group." : html;
        }


        function delete_group(group) {
            console.info("Groups: delete "+group);

            var del_req = {
                method: "DELETE"
            };
            fetch("api/groups/"+group, del_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }
        function turn_on_group(group) {
            console.info("Groups: turn on " + group);
            var on_req = {
                method: "GET"
            };
            fetch("api/groups/" + group + "?action=on", on_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }

        function turn_off_group(group) {
            console.info("Groups: turn off " + group);
            var off_req = {
                method: "GET"
            };
            fetch("api/groups/" + group + "?action=off", off_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }
```

```
        function toggle_group(group) {
            console.info("Groups: toggle " + group);
            var off_req = {
                method: "GET"
            };
            fetch("api/groups/" + group + "?action=toggle", off_req)
                .then(rsp => get_groups())
                .catch(err => console.error("Groups:", err));
        }

        // request to get all groups when page is loading
        get_groups();
        window.setInterval(()=> this.get_groups(),1000)

    </script>

</body>

</html>
```

Members.js

```
/**
 * A model for managing members in groups.
 */
function create_members_model(groups) {
    // create the data structure
    var all_members = new Set(); // all unique member names
    var group_names = [];
    var group_members = new Map(); // group_name to set of group members
    for (var group of groups) {
        group_names.push(group.name);
        var members = new Set(group.members);
        group_members.set(group.name, members);
        members.forEach(member => all_members.add(member));
    }
    var member_names = Array.from(all_members);
    group_names.sort();
    member_names.sort();

    // create the object
    var that = {}
    that.get_group_names = () => group_names;
    that.get_member_names = () => member_names;
    that.is_member_in_group = (member_name, group_name) =>
        !group_members.has(group_name)? false:
```

```
            group_members.get(group_name).has(member_name);
    that.get_group_members = group_name => group_members.get(group_name);

    console.debug("Members Model",
        groups, group_names, member_names, group_members);

    return that;
}

/**
 * The Members controller holds the state of groups.
 * It creates its view in render().
 */
class Members extends React.Component {

    constructor(props) {
        super(props);
        console.info("Members constructor()");
        this.state = {
            members: create_members_model([]),
            inputName: "",
            inputMembers: "",
        };
    }

    componentDidMount() {
        console.info("Members componentDidMount()");
        this.getGroups();
        //setInterval(this.getGroups, 1000);
    }

    render() {
        //console.info("Members render()");
        return (<MembersTable members={this.state.members}
            inputName={this.state.inputName}
inputMembers={this.state.inputMembers}
            onMemberChange={this.onMemberChange}
            onDeleteGroup={this.onDeleteGroup}
            onInputNameChange={this.onInputNameChange}
            onInputMembersChange={this.onInputMembersChange}
            onAddGroup={this.onAddGroup}
            onAddMemberToAllGroups={this.onAddMemberToAllGroups} />);
    }

    getGroups = () => {
```

```
        console.debug("RESTful: get groups");
        fetch("api/groups")
            .then(rsp => rsp.json())
            .then(groups => this.showGroups(groups))
            .catch(err => console.error("Members: getGroups", err));
    }

    showGroups = groups => {
        this.setState({
            members: create_members_model(groups)
        });
    }

    createGroup = (groupName, groupMembers) => {
        console.info("RESTful: create group "+groupName
            +" "+JSON.stringify(groupMembers));

        var postReq = {
            method: "POST",
            headers: {"Content-Type": "application/json"},
            body: JSON.stringify(groupMembers)
        };
        fetch("api/groups/"+groupName, postReq)
            .then(rsp => this.getGroups())
            .catch(err => console.error("Members: createGroup", err));
    }

    createManyGroups = groups => {
        console.info("RESTful: create many groups
"+JSON.stringify(groups));
        var pendingReqs = groups.map(group => {
            var postReq = {
                method: "POST",
                headers: {"Content-Type": "application/json"},
                body: JSON.stringify(group.members)
            };
            return fetch("api/groups/"+group.name, postReq);
        });

        Promise.all(pendingReqs)
            .then(() => this.getGroups())
            .catch(err => console.error("Members: createManyGroup", err));
    }

    deleteGroup = groupName => {
```

```
        console.info("RESTful: delete group "+groupName);

        var delReq = {
            method: "DELETE"
        };
        fetch("api/groups/"+groupName, delReq)
            .then(rsp => this.getGroups())
            .catch(err => console.error("Members: deleteGroup", err));
    }

    onMemberChange = (memberName, groupName) => {
        var groupMembers = new
Set(this.state.members.get_group_members(groupName));
        if (groupMembers.has(memberName))
            groupMembers.delete(memberName);
        else
            groupMembers.add(memberName);

        this.createGroup(groupName, Array.from(groupMembers));
    }

    onDeleteGroup = groupName => {
        this.deleteGroup(groupName);
    }

    onInputNameChange = value => {
        console.debug("Members: onInputNameChange", value);
        this.setState({inputName: value});
    }

    onInputMembersChange = value => {
        console.debug("Members: onInputMembersChange", value);
        this.setState({inputMembers: value});
    }

    onAddGroup = () => {
        var name = this.state.inputName;
        var members = this.state.inputMembers.split(',');

        this.createGroup(name, members);
    }

    onAddMemberToAllGroups = memberName => {
        var groups = [];
        for (var groupName of this.state.members.get_group_names()) {
```

```
            var groupMembers = new
Set(this.state.members.get_group_members(groupName));
            groupMembers.add(memberName);
            groups.push({name: groupName, members:
Array.from(groupMembers)});
        }
        this.createManyGroups(groups);
    }
}

// export
window.Members = Members;
```

Members_table.js

```
const btnClassAdd = "btn btn-primary btn-block";
const btnClassDel = "btn btn-danger btn-block";

/**
 * This is a stateless view showing the table header.
 */
function Header(props) {
    var ths = [];
    for (var groupName of props.groupNames) {
        ths.push(
            <th key={groupName}>
                <button className={btnClassAdd}>{groupName}</button>
            </th>
        );
    }

    return (
        <thead>
            <tr>
                <th rowSpan="2" width="10%">Members</th>
                <th colSpan={props.groupNames.length}>Groups</th>
                <th rowSpan="2" width="10%">Remove from All Groups</th>
            </tr>
            <tr>
                {ths}
            </tr>
        </thead>
    );
}

/**
```

```
 * This is a stateless view showing one row.
 */
function Row(props) {
    var members = props.members;
    var tds = members.get_group_names().map(groupName => {
        var onChange = () => props.onMemberChange(props.memberName,
groupName);
        var checked = members.is_member_in_group(props.memberName,
groupName);
        return (<td key={groupName}>
            <input type="checkbox" onChange={onChange}
checked={checked}/></td>);
    });

    var onAddClick = () => props.onAddMemberToAllGroups(props.memberName);
    return (
        <tr>
            <td><button className={btnClassAdd}
onClick={onAddClick}>{props.memberName}</button></td>
            {tds}
            <td><button className={btnClassDel}>X</button></td>
        </tr>
    );
}

/**
 * This is a stateless view showing the row for delete groups.
 */
function DeleteGroupsRow(props) {
    var tds = props.groupNames.map(groupName => {
        var onClick = () => props.onDeleteGroup(groupName);
        return <td key={groupName}>
            <button className={btnClassDel}
onClick={onClick}>X</button></td>;
    });

    return (
        <tr>
            <td></td>
            {tds}
            <td></td>
        </tr>
    );
}
```

```
/**
 * This is a stateless view showing inputs for add/replace groups.
 */
function AddGroup(props) {
    var onChangeName = event =>
props.onInputNameChange(event.target.value);
    var onChangeMembers = event =>
props.onInputMembersChange(event.target.value);

    return (
        <div>
            <label>Group Name</label>
            <input type="text" onChange={onChangeName}
value={props.inputName}/>
            <label>Members</label>
            <input type="text" onChange={onChangeMembers}
value={props.inputMembers}
                size="60" placeholder="e.g. a,b,c"/>
            <button className="btn btn-primary"
onClick={props.onAddGroup}>
                Add/Replace</button>
        </div>
    );
}

/**
 * This is a stateless view showing the table body.
 */
function Body(props) {
    var rows = props.members.get_member_names().map(memberName =>
        <Row key={memberName} memberName={memberName}
members={props.members}
            onMemberChange={props.onMemberChange}
            onAddMemberToAllGroups={props.onAddMemberToAllGroups} />);

    return (
        <tbody>
            {rows}
            <DeleteGroupsRow groupNames={props.members.get_group_names()}
                onDeleteGroup={props.onDeleteGroup} />
            <tr><td colSpan="7">
                <AddGroup inputName={props.inputName}
inputMembers={props.inputMembers}
                    onInputNameChange={props.onInputNameChange}
                    onInputMembersChange={props.onInputMembersChange}
```

```
                    onAddGroup={props.onAddGroup} />
            </td></tr>
        </tbody>
    );
}

/**
 * This is a stateless view showing the whole members table.
 */
function MembersTable(props) {
    //console.info("MembersTable()");
    if (props.members.get_group_names().length == 0)
        return (
            <div>
                <div>There are no groups.</div>
                <AddGroup inputName={props.inputName}
inputMembers={props.inputMembers}
                    onInputNameChange={props.onInputNameChange}
                    onInputMembersChange={props.onInputMembersChange}
                    onAddGroup={props.onAddGroup} />
            </div>);

    return (
        <table className="table table-striped table-bordered">
            <Header groupNames={props.members.get_group_names()} />
            <Body members={props.members}
                inputName={props.inputName}
inputMembers={props.inputMembers}
                onMemberChange={props.onMemberChange}
                onDeleteGroup={props.onDeleteGroup}
                onInputNameChange={props.onInputNameChange}
                onInputMembersChange={props.onInputMembersChange}
                onAddGroup={props.onAddGroup}
                onAddMemberToAllGroups={props.onAddMemberToAllGroups} />
        </table>);
}

//export
window.MembersTable = MembersTable;
```

Members_app.js

```
/**
 * The App class is a controller holding the global state.
 * It creates all children controllers in render().
 */
```

```
class MembersApp extends React.Component {

    constructor(props) {
        super(props);
        console.info("MembersApp constructor()");
    }

    render() {
        console.info("MembersApp render()");
        return (
        <div className="container">
            <div className="row">
                <div className="col-sm-12">
                <h2>Manage Groups and Members</h2>
                <Members />
                </div>
            </div>
        </div>);
    }
}

// export
window.MembersApp = MembersApp;
```

Members_mockup.html

```
<html>

<head>
    <title>IoT Hub</title>

    <!-- Bootstrap -->
    <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min
.css"
        integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
    <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js"
        integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
        crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
        integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
```

```
        crossorigin="anonymous"></script>
    <script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.j
s"
        integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6"
        crossorigin="anonymous"></script>

    <!-- React -->
    <script src="https://unpkg.com/react@16/umd/react.development.js"
crossorigin></script>
    <script src="https://unpkg.com/react-dom@16/umd/react-
dom.development.js" crossorigin></script>
    <script src="https://unpkg.com/babel-
standalone@6/babel.min.js"></script>

    <!-- Members App -->
    <script type="text/babel" src="web/member_app.js"></script>
    <script type="text/babel" src="web/member.js"></script>
    <script type="text/babel" src="web/member_table.js"></script>
    <script type="text/babel" src="web/member_details.js"></script>
    <script type="text/babel" src="web/plug_details.js"></script>

    <style>
    .table tr td, .table thead th {
        text-align: center;
        vertical-align: middle;
        padding: 0;
    }
    </style>
</head>

<body>
    <div id="app_container">
    <script type="text/babel">
        $(document).ready(function () {
            ReactDOM.render(
                <IoTGroupsApp />,
                document.getElementById("app_container"));
        });
    </script>
</body>

</html>
```