House Automation – Future Is Here!

The Official Requirements

A JavaScript application simulating house automation: pressing a button on a control panel would visually turn on a light, change the temperature or close the curtains. Some constraints:

- the application must use jQuery.
- the components must have HTTP based "server" interaction (use a static file for simplicity, data persistence is not required). For example, the heating component retrieves the current temperature from the server and also sends the desired one back to the server.
- the solution has to be extensible and documented, so that we can develop our own components that react to events.

The application will be executed on a plain HTTP server with no possibility to run code server side and is being viewed in 2 major browsers of your choice.

Introduction

1. The Idea behind this home automation is creating an automation system using static assets and visual effects. The current implementation is done in jQuery and a Global JSON variable is used, which stores each and every information of the components, which can be later kept on Server. The layout for home is divided in to Living Area, Bedroom and Kitchen. When the page loads, the JSON object is assigned to a function which helps to visually represent the default values and state of the components.

I have used bootstrap theme called bootswatch/paper for getting some cool CSS. And as I stated before I have used JQuery for parsing the components when the assigned events are triggered.

Setting up the Web Page

Web page contains Header, Body → Container, and Footer section.

1. I have downloaded the JQuery CDN, rather than calling them from the webpage. And added before the end of body tags.
2. As state before for Bootstrap→ Bootswatch/paper, I have downloaded the CSS from the https://bootswatch.com/3/paper/ .
3. I have added my JSON object called **houseAutmation,** in index.js file which is also called in the home page.
4. When page loads **loadDataFromJSON()** function is called which gets all the values from the server, and assign the default state from server JSON object.
5. Also for any events called on the web page, the JSON object is updated, which will in turn update the server.

Web page components

Web page contains three main components, as of now

Living Area

Living area consist of Curtains, Lights and A/C.

Curtains:

Curtains is created with some transitions animations effect and SVG for the fill color.

Curtain consist of Left and Right division tags, out of which one start from left and other from right. When curtain is closed the ideal condition is width 100%, when the open condition is triggered the width is assigned to 0% as we have the animation assigned to the curtains, it provides the visual effects of actual curtains.

Curtain object:

```
'curtain': {

  'state': 'Open'

 }
```

The state gives the current condition for the curtains, whether it is open or closed.

Lights:

Lights is created with GIF images, which toggles between the ON/OFF images, when event triggered.

Light object:

```
'light': {

'state': 'OFF'

}
```

The state gives the current condition for the lights, whether its ON or OFF.

A/C:

A/C is designed using HTML5, CSS3. It has two attribute one is Mode, and the other is temperature.

Again the **loadDataFromJSON()** function assigns the default values from Server. The maximum and minimum temperature allowed is controlled from the JQuery click on increase and decrease functions.

A/C object:

```
'AC' : {
 'state': 'ON',
'temp' : '75'
}
```

The state gives the current state for the A/C, whether it is ON or OFF and the temperature which it should maintain.

Living Area JSON Object:

```
'livingArea': {
    'AC' : {
      'state': 'ON',
      'temp' : '80'
    },
    'light': {
      'state': 'OFF'
    },
    'curtain': {
      'state': 'Open'
    },
    'TV': {
      'state': 'ON'
    }
        }
```

Bedroom

Bedroom area consist of Curtains, Lights and A/C.

Curtains:

Curtains is created with some transitions animations effect and SVG for the fill color.

All the parameters and functions is same as **Living Area**.

Curtain object:

```
'curtain': {
  'state': 'Open'
 }
```

The state gives the current condition for the curtains, whether it is open or closed.

Lights:

All the parameters and functions is same as **Living Area**.

Light object:

```
'light': {
'state': 'OFF'
}
```

The state gives the current condition for the lights, whether its ON or OFF.

A/C:

A/C is designed using HTML5, CSS3. It has two attribute one is Mode, and the other is temperature.

All the parameters and functions is same as **Living Area**.

A/C object:

```
'AC' : {
 'state': 'ON',
'temp' : '75'
}
```

The state gives the current state for the A/C, whether it is ON or OFF and the temperature which it should maintain.

Bedroom JSON Object:

```
'bedroom': {
    'AC' : {
      'state': 'ON',
      'temp' : '75'
    },
    'light': {
      'state': 'OFF'
    },
    'curtain': {
      'state': 'Open'
    }
  }
```

Kitchen

Kitchen area consist of Curtains, Lights and A/C.

Lights:

All the parameters and functions is same as **Living Area**.

Light object:

```
'light': {
'state': 'OFF'
}
```

The state gives the current condition for the lights, whether its ON or OFF.

Kitchen JSON Object:

```
'kitchen': {
    'light': {
      'state': 'ON'
    }
  }
```

Events on Web page

There are different types of events on web page.

Curtains Open/close:

When click on Curtains Checkbox for any house area, the event will identify the type of living area and will apply the conditions and also update the JSON Object on the server, so that the server objects are also updated.

```
//function when click on curtainState
    $("#myTabContent .curtainState").on('click', function() {
        var parent = $(this).attr('area');
        $("#" + parent + " .curtain").toggleClass("open");
        if ($("#" + parent + " .curtain").hasClass("open")) {
            changeCurtainInServer(parent, "OFF")
        } else {
            changeCurtainInServer(parent, "ON")
        }
    });
```

Light On/Off:

When click on Lights Checkbox for any house area, the event will identify the type of living area and will apply the conditions and also update the JSON Object on the server, so that the server objects are also updated.

```
//Function when click on lights
    $("#myTabContent .lightState").on('click', function() {
        var parent = $(this).attr('area');
        if ($(this).prop("checked")) {
            $("#" + parent + " .lights").attr("src", "images/lighton.gif");
            changeLighstInServer(parent, "ON")
        } else {
            $("#" + parent + " .lights").attr("src", "images/lightoff.gif");
            changeLighstInServer(parent, "OFF")
        }

    });
```

A/C:

A/C has two events out of which one is ON/OFF and the other is change in temperature.

When click on A/C ON/OFF Checkbox for any house area, the event will identify the type of living area and will apply the conditions and also update the JSON Object on the server, so that the server objects are also updated.

```
//function when click on A/C
    $("#myTabContent .acState").on('click', function() {
        var parent = $(this).attr('area');
```

```
            if ($(this).prop("checked")) {
              $("#" + parent + " .ac .state").text("ON");
              changeACStateInServer(parent, "ON");
            } else {
              $("#" + parent + " .ac .state").text("OFF");
              changeACStateInServer(parent, "OFF");
            }


          });
```

When click on A/C increase or decrease button for temperature, first the current value is taken and compared with the conditions that If after increment the temperature exceeds the maximum or is less than the minimum temperature. Once the above conditions are satisfied the data is also update in the server JSON Object.

//Script for Increase and decrease Temperature

```
(function($) {
  $('.spinner .btn:first-of-type').on('click', function() {
    var input = $(this).parent().parent().children('input');
    var parent = $(this).attr('area');
    if (parseInt(input.val(), 10) + 1 < 85) {
      var newvalue = parseInt(input.val(), 10) + 1;
      input.val(newvalue);
      $("#" + parent + " .temp").text(newvalue);
      changeTempInServer(parent, newvalue);
    }

  });
  $('.spinner .btn:last-of-type').on('click', function() {
    var input = $(this).parent().parent().children('input');
    var parent = $(this).attr('area');
    var newvalue = parseInt(input.val(), 10) - 1;
    if (newvalue >= 20) {
      input.val(newvalue);
      $("#" + parent + " .temp").text(newvalue);
      changeTempInServer(parent, newvalue);
    }

  });
})(jQuery);
```

Conclusion

The above Home automation system was designed using JQuery, JS, JSON, HTML5 and CSS3 which was stated in the problem statement. It would have a dynamic working behavior If we had used AJAX and

NodeJS server, but since the requirement for this assignment was to use a static file as a replica of server, I didn’t went for it. This Home Automation system can be used to controlled Curtains, A/C and Lights at different parts of the home. It can also be used to controlled other smart electronics appliances.