Databases, Network and the Web Coursework for Midterm:

MySmartHome

Introduction

In the course, we developed a small dynamic web application including routes, forms, and database access. In this assignment, you are tasked with creating another dynamic web application that will function as a dashboard to help your customers to control devices in their smart home. Essentially the dynamic web application interacts with users to provide information related to the status of devices at home such as cooling and heating systems, curtains and blinds, lighting, security, audio-visual systems such as TV, radio, music players, kitchen appliances such as fridge-freezer, hob, oven, microwave, etc. Then the system lets the customers control the devices by changing the status such as turning on or off, opening/closing, increasing/decreasing volumes, temperature, and so on.
The user should be given the choice to add information related to their smart home and store it in the database. The web application should provide customers with the option of adding a device from a pre-organised list of devices and then to set or update the status of each device or appliance. As an example, the user should be able to select the 'heating' in the device list. Then set device status by turning the heating on and setting the temperature to 22 degrees. A full list of the functionalities of the web application is explained in the next sections of this document. Some requirements are 'base' requirements to pass the assignment and some requirements are 'stretch goals' indicated as 'going beyond' and are designed for students who would like to achieve the full mark.

Working Environment

For the purpose of this assignment, we have set up a working directory called "mid-term" inside the "topic7" folder of your labs. At the moment the folder contains a dummy "index.js" file which creates a web server on PORT 8089. Open this lab and take a look at the folder structure. You can see the root route of your application by visiting "localhost:8089" on the Browser Preview VsCode extension. Remember that you need to run the server first:

- cd topic7/mid-term
- node index.js

PLEASE NOTE WE CAN ONLY MARK SUBMISSIONS THAT ARE RUNNING IN THE COURSERA LAB ENVIRONMENT. MARKERS ARE NOT ALLOWED TO INSTALL AND RUN YOUR WEB APPLICATIONS ON THEIR LOCAL MACHINES.

Requirements

The purpose of the web application is to help people manage their smart home devices by adding and deleting devices in a database and displaying the status of each device and allowing the customer to set or update their status in the database. Although this application is supposed to be

connected to smart home devices, the hardware and the connection to the hardware are not part of this project. You are only implementing the control dashboard and the database. The web application provides several functionalities that should meet the following requirements:

- R1: Home page:
  - R1A: Display the name of the web application.
  - R1B: Display links to other pages or a navigation bar that contains links to other pages.
- R2: About page:
  - R2A: Display information about the web application including your name as the developer. Display a link to the home page or a navigation bar that contains links to other pages.
- R3: Add device page:
  - R3A: Display a form to users to add a new device to the database. The form should consist of the following items: name of the device to be selected from a pre-defined set of devices and other input fields such as on/off, open/close, temperature, volume, etc. Depending on the chosen device other input fields might be applicable or not applicable. The minimum number of devices in the pre-defined list is 20. Display a link to the home page or a navigation bar that contains links to other pages.
  - R3B: Collect form data to be passed to the back-end (database) and store the device name and its corresponding initial status in the database. Each device item consists of input fields such as name, on/off, open/close, temperature, volume, etc. Some fields might not be applicable for some devices. As an example, the open/close field is not related to a heating device but applies to a blind or a curtain, or a door. As another example volume does not apply to the heating system, but the temperature does. Display a message indicating that add device operation has been done successfully.
  - R3C: Improve R3B by automatically assigning ‘not applicable’ initial value to the fields that do not apply to the chosen device. It is even better to disable or hide those input fields from users when adding a new device. As an example, if a user adds a new device called ‘heating’ then the minimum required input fields to be displayed and initialised by the user are: on/off and temperature and all other non-related input fields must be initialised to NA (not applicable) status.
  - R3D: Form validation, make sure all required form data is filled and also valid data is entered by user. If required fields are empty or data is not valid, re-display the form to the user with appropriate message to fill it again. As an example, entering a string or a value of 400 is not valid for the temperature of a ‘heating’ device.
- R4: Show device status page
  - R4A: Display a list of already added devices and then let the user choose a device from the list. Display a link to the home page or a navigation bar that contains links to other pages.
  - R4B: Display data related to the chosen device found in the database to users including name, and related settings. As an example, if TV is selected related settings would be

on/off setting, volume, and channel. If the heating system is chosen minimum settings, are on/off and temperature. Display a message to the user if not found.

- R4C: Improve R4B by automatically hiding input fields that do not apply to the chosen device.
- R4D: Going beyond by improving R4B and R4C, by a graphical display of the device settings like a dashboard.

- R5: Update device status page
  - R5A: Display a list of already added devices and then let the user choose a device from the list. Display a link to the home page or a navigation bar that contains links to other pages.
  - R5B: Display data related to the chosen device found in the database to users including name, and related settings. As an example, if TV is selected related settings would be on/off setting, volume, and channel. If the heating system is chosen minimum settings, are on/off and temperature, so users can update each field. Collect form data to be passed to the back-end (database) and store updated device status in the database. Display a message indicating the update operation has been done.
  - R5C: Improve R5B by automatically hiding input fields that do not apply to the chosen device.
  - R5D: Going beyond by improving R5B and R5C. Improvement could be achieved by letting the user update device status through a graphical display of the device settings like a dashboard.
- R6: Delete device page
  - R6A: Display a list of already added devices and then let the user choose a device from the list to be deleted. Display a link to the home page or a navigation bar that contains links to other pages.
  - R6B: It is always a good practice to ask for the user's confirmation before the delete operation. Ask the user 'Are you sure?' then if confirmed, delete the chosen device and related data from the database by collecting form data related to the device name to be passed to the back-end (database) and performing delete operation on the database. Display a message indicating the delete operation has been done successfully.

Code style and technique

Your code should be written according to the following style and technique guidelines:

- C1: Code is organised into JavaScript (.js) files and template files (.html or .ejs or .pug). JavaScript files contain web server code (index.js) and middleware (main.js in routes folder). Template files (.html and .ejs and .pug) are stored in views folder.
- C2: Each route in main.js has comments describing purpose, inputs, and outputs
- C3: Code is laid out clearly with consistent indenting
- C4: Each database interaction has comments describing the purpose, inputs, and outputs
- C5: Functions and variables have meaningful names, with a consistent naming style

Documentation

You should write a documentation report and include the link to the application environment. The submission should contain the following items and information:

- D1: URL Link to the application environment
- D2: documentation report is in PDF format
- D3: List of requirements: for each sub-requirement (R1A → R6B) state how this was achieved or if it was not achieved. Explain where it can be found in the code. Use focused, short code extracts if they make your explanation clearer.
- D4: Database structure: tables including purpose, field names, and data types for each table.

Marking criteria

We will mark your work if your code is running in the Coursera lab environment according to the set of criteria shown below, which consider the requirements, your programming technique and style, and the documentation you have provided:

a) Code style and technique:

Category	Criterion	Not addressed [0]	Attempted but did not meet requirements completely [1]	Met requirements completely. [2]	Weight
Code style and technique	C1: Code is organised into JavaScript (.js) files and template files (.html or .ejs or .pug). JavaScript files contain web server code (index.js) and middleware (main.js in routes folder), template files (.html and .ejs and .pug) are stored in views folder.				2
Code style and technique	C2: Each route in main.js has comments describing purpose, inputs, and outputs				2
Code style and technique	C3: Code is laid out clearly with consistent indenting				2

Code style and technique	C4: Each database interaction has comments describing the purpose, inputs, and outputs				2
Code style and technique	C5: Functions and variables have meaningful names, with a consistent naming style				2
					Total:10

b) Documentation report [32 marks in total]

- D1: URL Link to the application environment [2]
- D2: Report is in PDF format [2]
- D3: List of requirements: for each sub-requirement (R1A → R6B) state how this was achieved or if it was not achieved. Explain where it can be found in the code. Use focused, short code extracts if they make your explanation clearer. [each requirement 1 mark, 17 marks in total for D3]
- D4: Database structure: tables including purpose [2 marks], field names [2 marks], and data types for each table [7 marks]. [11 marks in total for D4]

c) Requirements [58 marks in total]

Requirement	Parts of the requirement	Marking
R1: Home page	R1A, R1B	• Not addressed [0] • One part of the requirement is addressed [1] • Two parts of the requirement are addressed [2]
R2: About page	R2A	• Not addressed [0] • The requirement is addressed [1]
R3: Add device page	R3A, R3B, R3C, R3D	• Not addressed [0] • One part of the requirement is addressed [2] • Two parts of the requirement are addressed [4] • Three parts of the requirement (A, B, and C) are addressed [6] • All four parts of the requirement (A, B, C and D) are addressed [12]
R4: Show device status page	R4A, R4B, R4C, R4D (going beyond)	• Not addressed [0] • One part of the requirement is addressed [2] • Two parts of the requirement are addressed [4] • Three parts of the requirement are addressed [6] • All four parts of the requirement (A, B, C and D) are addressed [16]
R5: Update device status	R5A, R5B, R5C, R5D	• Not addressed [0]

page	(going beyond)	• R5A is only addressed [3] • R5A and R5B are only addressed [8] • R5A, R5B and R5C are only addressed [10] • All four parts of the requirement (A, B, C and D) are addressed [20]
R6: Delete device page	R6A, R6B	• Not addressed [0] • R6A is only addressed [2] • Two parts of the requirement (A and B) are addressed [7]