Evan Tilley

elt2141

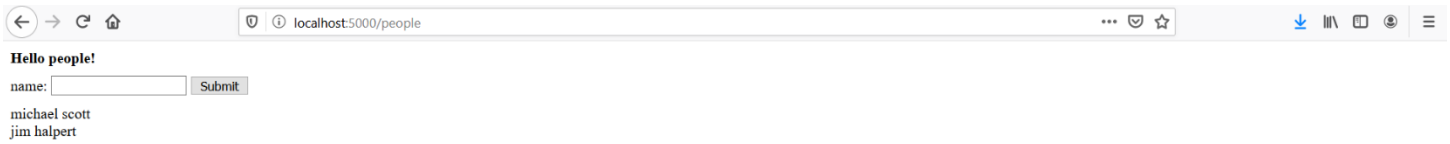
UI PSET #5 Problem 1

1. This webpage serves 3 basic webpages.

a. Webpage served by '/' route:

Example webpage served by '/hello/<name>' route (<name> can be replaced with any text):

Webpage served by '/people':



(following questions on next page)

2. In hello.html, the curly braces around “name”: {{name}}, make the <name> show up in HTML, as the double curly braces are signifying scripting code, and thus the string {{name}} will be replaced by the name parameter passed into the page route.

3. In people.html, the line “<script type="text/javascript" src="{{ url_for('static', filename = 'people.js') }}"></script>” imports people.js. The two parameters are in the function call to the function url_for: url_for('static', filename = 'people.js'). The parameters are ‘static’ and filename = ‘people.js’. ‘static’ is the name of the folder with the file we are trying to generate a URL to is located in and ‘people.js’ is the name of the file in the folder.

4. In people.html, the line: “var data = {{data|tojson}}” gets the data sent from the server, puts it into JavaScript and parses the string into JSON.

5. In people.js, the function saveName(name) contains the ajax call.

6. The saveName function sends data to the ‘/add_name’ route.

7. The function sends a JSON string with the format {“name”: name}. So, for instance, if the entered name was “Herbert”, the function would send {“name”: “Herbert”}.

8. I would describe the data it sends as a JSON string, similar to a dictionary, which contains the “name” attribute and the value held in the name variable.

9. If everything goes perfectly on the server side, the following lines of code will execute on the front end:

```
success: function(result){
    var all_data = result["data"]
    data = all_data
    displayNames(data)
}
```

10. If something goes wrong on the server side, the following lines of code will execute on the front end:

```
error: function(request, status, error){
    console.log("Error");
    console.log(request)
    console.log(status)
    console.log(error)
}
```

11. On the server, the add_name function, retrieves the JSON data that is sent to it, in the line: json_data = request.get_json(). The function then extracts the value stored in the "name" attribute of the JSON data, and stores this in the variable "name". It then creates a new entry to be added to the globally stored data, consisting of a unique id and the name value. It then appends this entry to the globally stored data array and then sends the entire updated array back to the client (front-end) in JSON format, so the webpage can reload with the newest data.

12. If all goes well, the add_name function return an array of JSON data, with each entry in the array consisting of a "name" and "id" parameter. For instance, it could return:

```
[
  {
    "id": 1,
    "name": "michael scott"
  },
  {
    "id": 2,
    "name": "jim halpert"
  },
  {
    "id": 3,
    "name": "john cena"
  }
]
```

13. Yes, if you refresh the web page, the changes you made to the data will still be there because the webpage will send a request to the server, which still has all the data stored, and the page will reload with all the previously stored data.

14. No, if you restart the server the changes you made to the data will not still be there because the global data array has been reset back to the default array.