Lab assignment 5

TI1506 2016/17; ti1506-students.slack.com

Introduction

In this assignment, you will continue working on your Web application. Every one of the tasks (apart from the last one) is relatively small-scale. The descriptions of each task are kept rather general on purpose, as every group by now has built quite a different application. The goal for you is to get hands-on experience with every one of the node.js aspects introduced in recent lectures as well as one aspect we have not covered explicitly in class (task 4).

Deadline: when this assignment is due depends on the cluster your team is in. Every team is assessed biweekly. Make sure you know your cluster and what this means for your lab deadlines.

In order to pass the assessment, you have to make a valid attempt at working through the lab assignment.

1. Modularization

We finally covered the topic of node.js modules. Based on your understanding of node.js modules, modularize your code and organize the different functionalities into separate modules (e.g. one module to keep track of all configuration settings, one module for routes, one module to interact with the database).

2. Routing

Allow your users to make small mistakes in the URL paths (as an example, if you have a route 'listtodos' a user typing http://localhost:3000/listtodoss should be served http://localhost:3000/listtodos). Every route you have in your application should allow such small deviations to some degree (it is up to you to find sensible regular expressions).

Introduce routing parameters into at least one of your routes (it is up to you to find out where it makes most sense).

3. Templating

So far, when you access your application's URL for the first time, an empty todo list is returned (unless you already added code to mitigate this effect).
In a subsequent Ajax request, the existing todos are sent from the server to the client.

Templates allow us to change this setup and immediately send the existing todos. In this exercise you are asked to use EJS for templating. To get started, follow the EJS procedure outlined in lecture 9.

4. package.json

In class we have not covered the `package.json` file, it forms however an important part of the node.js – npm infrastructure.

Read about it and create a basic `package.json` file for your own application. It must be complete enough so that

- calling `npm install` in your application's directory will install all required dependencies, and
- calling `npm start` actually starts your application backend.

You should also be able to explain what the difference between `npm install --save passport` and `npm install --save-dev passport` is.

There are many resources on the Web available, pick the one or two you feel most comfortable with.

---

5. Client-side cookies

Cookies can be created not just by the server (and sent to the client) but also on the client (i.e. the browser) itself. Those client-side generated cookies are usually used to store a user's preferences with respect to font size, background color, etc.

Use cookies to allow users to set their preferences with respect to some visual characteristics of your application (e.g. the text size and/or the background and font color and more). The cookie or cookies should be persistent, once the user closes the browser and reopens it, your application should be shown in the user's preferred way.

Implement at least one possible change in visual characteristic. To give you one concrete example, head to the American Foundation for the Blind website and check the top elements. The text size can be changed directly with +/-, while the change in colors leads to a new page where the user can make detailed choices. When you view the cookies on that page (the lecture slides show you how to do that), you will see how this information is stored.

For this exercise, you can make use of the jQuery cookie plugin introduced in class.

---

6. Third-party authentication

We finally make use of the application's splash screen. Implement a third-party authentication of your own choice (Twitter was covered in class, Facebook is covered in one of the recommended readings). Use the `passport` middleware for this task: http://passportjs.org/

The todos of each user using your application should only be visible to him/her.
Note that you will have to install `passport` via `npm` (don't forget to update your `package.json`, i.e. use the `–save` option).