Lab assignment 5

TI1506 2017/18; 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 (e.g., if you have a route 'listhabits' a user typing https://localhost:3000/listhabitts should be served https://localhost:3000/listhabits). 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 habit list is returned (unless you already added code to mitigate this effect). In a subsequent Ajax request, the existing habits are sent from the server to the client.

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

Note: this exercise requires you (once again) to rewrite parts of the code you have already written. This is not a mistake; it is intended and one of the goals of the lab: we want you to realise the different manners in which the same result can be achieved.

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 eps` and `npm install —-save-dev pjs` is.

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

5. Optional: Third-party authentication

We finally make use of the application's splash screen. Implement a third-party authentication of your own choice (e.g. Twitter [as covered in class] or Facebook). Use the `passport` middleware for this task: http://passportjs.org/.

The habits 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).

Note: we will cover this topic in a lecture in week 2.7. Although this is an optional exercise we strongly encourage you to work through it as it will lead to a fully functioning Web application in the end.