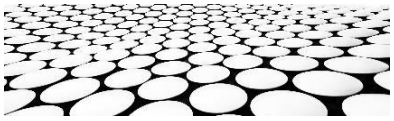
SEG3125 Analysis and Design of User Interfaces



Lab 6: Survey Analysis

GOALS

In Lab 1, you did a small survey related to the evaluation of a website's user interface of your choice. That was just the customer part. In order to understand client/server interactions, this lab asks you to develop an application in Node.js to compile the results of your survey.

During this laboratory, you will need to:

- Learn a JavaScript framework, node.js, that allows you to develop the server-side application, in JavaScript.

This is a UI design course but it's important to understand the client/server chain, and also to discover Node.js, because several new JavaScript frameworks on the client side (such as React that we'll soon discover) are often used with Node.js on the server side to develop complete applications.

SUBMISSION DEADLINE

- Wednesday, March 17th, 2021, 11:30 p.m.

SUBMISSION METHOD

- In Brightspace, the Lab 6 module contains a link to your submission.
- As this is a web application running on your local server (localhost), we cannot use GitHub pages to view the rendering of your pages, as in previous labs. But we can continue to use GitHub to keep source files of your app.
- Don't submit a file, submit two links:
    1. A link to your github folder containing your app files.
       Don't include dependencies. Instead, use a package.json file in your app to contain dependencies.
    2. A link to a video (use a shared link in your Google Drive) that shows:
        - Compiling the survey
        - The form to complete.

WARNING: Any code or even "little piece of code" you take from such a stack overflow or other website *should be accompanied by a comment that recognizes the source. You must, inyour submission text, say "Code for X* inspired *by (link html)"*

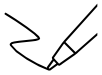
INSTRUCTIONS / TUTORIALS TO FOLLOW

Node.js

Go first to the https://nodejs.org/en/ website and download Node. During installation, keep the "default settings" you are offered. No need to download the dependencies, we don't need them.

Learning Node.js is not trivial and will take you a little time. I point you to several videos of "The Net Ninja" that will help you. Each video is very short and well made. These videos are really great, and the author explains well. I highly recommend them (the whole series!). Obviously, as they date from 2016, there will be some outdated aspects, but still, they give good explanations about Node.js.

1. What is Node.js? Node JS Tutorial for Beginners - Introduction
2. How to call Node from the command line. Node JS Tutorial for Beginners #2 - Installing Node JS.
3. The notion of modules. Node JS Tutorial for Beginners #6 - Modules and requires()
4. Writing/Reading files. Node JS Tutorial for Beginners #9 - Reading and Writing Files (fs)
5. An important utility when working with Node, the Node Package Manager. Node JS Tutorial for Beginners #20 – The Node package Manager. It will install modules, such as Express which will help for client/server exchanges.
6. Package.json file for dependency management. Node JS Tutorial for Beginners #21-The package.json File
7. The Express module that will facilitate routing and client/server exchanges. Beware, from this video the author uses the "nodemon" command instead of "node" to launch his application, but you can continue to use "node."
    a. Node JS Tutorial for Beginners #23 - Introduction to Express
    b. Node JS Tutorial for Beginners #24 - Express Route Params
8. The EJS (Embedded JavaScript templating) module to display HTML pages with dynamic Content Node JS Tutorial for Beginners #25 - Template Engines
9. POST request - Node JS Tutorial for Beginners #30 - Handling POST Requests



Design

In this lab, we explore the server side to better understand client/server communication. So we won't do much design. However, as you will see in the original code, The view of the "survey participant" and the "analyst" view are seperated. These two users of the same application have completely different needs.

The only design requirement for the UI is to minimally use Bootstrap to make your survey prettier on the "participant" side. I'm not asking you to improve the "analyst" side because it's HTML with dynamic content, and it's more complicated. I'd rather you focus on the participant's view.

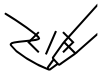
STARTING POINT

Before you look at the starting code, I recommend you watch 4 more videos of the Net Ninja, so that it is easier for you to understand the structure used, and what to install to run the application. The starting code follows the structure set in the Net Ninja videos which shows how to make a To-do list app. It uses the same packages (express, ejs, body- parser). So, it would be a very good exercise to follow the tutorials for the TO DO list, and then you will be better prepared to understand the code.

- Project Structure: Node JS Tutorial for Beginners #31 - Making a To-do App (part 1)
- Controller Explanation: Node JS Tutorial for Beginners #32 - Making a To-do App (part 2)
- Using EJS: Node JS Tutorial for Beginners #33 - Making a To-do App (part 3)
- POST request : Node JS Tutorial for Beginners #34 - Making a To-do App (part 4)

Here's the folder with the code in Node.js to do the survey analysis. Here's the structure:

- app.js (entry point)
- surveyController.js (controller in the MVC pattern)
- data (data directory – .json files including response statistics)
    - fruit.json
    - animal.json
    - color.json
- public/assets (in order to be compatible with the instructions in the video, I used this same folder)
    - action.js (jQuery code that I will do the POST when the survey is submitted)
- Views
    - niceSurvey.html (survey, view of the person doing the survey)
    - showResults.ejs (results, view of the person analyzing the results)
- package.json (file that handles dependencies)



Programming

Basic requirements of your page

Data

Normally, the data would be in a database on the server side, but in this lab, we are not going to go to the server/database exploration. We will limit ourselves to the server/json file.

1. Create the data files that match your questions.

The starting point code has 3 data files. These files are fruit.json, animal.json and color.json. These are files that contain the accumulated survey results. You can keep the information in one file. This is just to show you that it is possible to access several. This could be useful for a good data organization.

Customer side / Survey view

You can access this view from http://localhost:3000/niceSurvey

2. Improve your survey's user interface.
    a. Now that you know Bootstrap, you can use Bootstrap to improve your survey done in Lab 1.
3. Make sure your form structure is adequate.
    b. You must have had six questions of various types. In the starting code, I show you 3 types (text, list, checkbox).
    c. You had to use the <form> tag in Lab 1, but perhaps not sure why. This tag makes sense when the form is now submitted to the server that can analyze it. The “submit" button is used to send the information.
4. Make sure the survey is clearly visible (accessible on http://localhost:3000/niceSurvey)
   It is the surveyController.js who is responsible for putting the form online by responding to a GET query.

Transfer and backup of information (server side)

5. Make sure the form is properly sent to the server.
    a. The action file.js of the starting code takes care of doing the POST.
    b. The surveyController.js file takes care of the data from the POST.
6. Update data files based on the survey’s responses.
    c. Change the surveyController.js based on your data

Customer side

This view would be accessible (in the code) from http://localhost:3000/analysis.

7. Show the results in a different view of the survey.
    a. Adapt the starting code to show the data in your data files.

Optional

You already know Node.js? You can add:

- Make viewing results much prettier with graphics (pie chart or other)
- Change the survey to make it "multi-form," with the user responding to only 1 or 2 questions per page.
- Put a database in the backend rather than files.

EVALUATION

- This lab is worth 3.5%.
- Any student who has met the 7 requirements (points 1 to 7 above) will be awarded 10/10.
- Failure to provide code-inspired sources of online code will result in a score of zero.
- Any delay, beyond the deadline, will have a penalty of 10% per day of delay, up to 2 days.

Questions

- You can ask your questions during the lab session on Wednesday, March 10th.
- You can also send your questions to your TA Farouk (bfarouk@uottawa.ca)