Lab 4

Project

Spring 2019

TDDD97 Web Programming

http://www.ida.liu.se/~TDDD97/

Department of Computer and Information Science (IDA)

Linköping University

Sweden

Objective

In this lab, you will learn additional web programming concepts, techniques and technologies. In addition, you will independently search, assess, apprehend, and apply information about new technologies and third-party resources as well as downloading, installing, configuring and troubleshooting relevant libraries and frameworks.

Task

Your task is to extend the Twidder application, which you developed in the first three labs, with additional functionality. You can choose to implement different functionality from a list of alternatives described below, where each criteria give a certain number of points. Your final grade for the course will be determined by the total number of points your implementation collects (see Table 1).

Table 1. The number of points required for achieving each grade

Total number of criteria points	**Course grade**
3	**3**
6	**4**
9 + well-documented code	**5**

Below you will find a description of each alternative criteria, including suggestions for third-party libraries or frameworks, and the number of points they give. You may use other libraries and frameworks, but in this case you must discuss them with you lab assistant before starting the work.

Alternatives

1. **Providing Live Data Presentation [3 points]**

   This criterion requires your application to present dynamic data on the fly. The data may change over time while being presented as diagrams or charts, using a third-party library, such as D3JS and ChartJS. The data must be application-related and be

composed of at least three different fields, which are updated by the backend. For example, the user could be able to see a live presentation of the number of posts on his/her wall, number of views of his/her page, and total number of users currently online. You need to use websockets or other available technologies to push the data to the clients on the fly.

D3 official website
http://d3js.org/

ChartJS official website
http://www.chartjs.org/

2. **Use of HTML5 for Drag and Drop [1 point]**

This criterion requires your application to provide drag-and-drop functionality for performing a specific task. Please note that the drag-and-drop functionality itself matters in this assignment even though you can also add a new requirement, which can be used by using the drag-and-drop functionality. For example, the user can copy the posts on his/her wall, posted by himself/herself or other users, into a text area to be edited and reused.

HTML5 Drag and Drop
http://www.w3schools.com/htmL/html5_draganddrop.asp

3. **Performing Client-side Routing + Overriding Back/Forward buttons using the History API [2 point]**

This criterion requires your application to provide a valid URL associated to each view or resource. The difference from the traditional form of URL handling, is that it shall happen at the client-side and shall mainly be done by using Javascript and third-party libraries. This approach requires having at least three separately-assigned URLs. Note that your implementation must tolerate any unwanted page refreshes and maintain the correct URL in the address bar upon any refresh event. You are also required to use the History API to make back and forward buttons usable in your application. At the current stage of the Twidder application, these buttons are used for moving through pages, clicking on the back button will cause the browser to leave your application.

PageJS a small client-side routing library
http://smalljs.org/client-side-routing/page/

Manipulating the browser history
https://developer.mozilla.org/en-US/docs/Web/Guide/API/DOM/Manipulating_the_browser_history

4. **Third-Party Authentication Using OpenID/OAuth 2.0 [2 point]**

> **This criterion requires Criterion 3 to be satisfied first. This criterion may require your application to accessible publicly via the internet.**

Give the user the possibility to login via a third-party service, such as Facebook or Google. Please note, since your application is based on SPA architecture, you need to avoid page refreshes as much as you can.

What is OpenID
http://openid.net/get-an-openid/what-is-openid/

Google OAuth 2.0
https://developers.google.com/accounts/docs/OAuth2

5. **Applying Further Security Measures [3 points]**

This criterion requires the following two security measures to be applied:

1. Protecting passwords from being stolen directly from the server, in case of the database being compromised, by hashing them before storing in the database. In other words instead of the password itself, its hashed value shall be stored in the database once the user signs up.

   Password Hashing
   https://crackstation.net/hashing-security.htm

   Flask BCrypt
   http://flask-bcrypt.readthedocs.org/en/latest/

2. Protecting token while being transmitted. Since data, including tokens, are transmitted unencrypted over the network, they are subject to be retrieved and abused by a third party tapping the network. In this case, a villain can send requests to the server on behalf of the victim until the token becomes invalid by logout. You are required to follow the provided instructions and avoid such situation. Please note that your task is to protect only the token and only after the authentication is done. Another advantage by applying the following measure is to make sure that the request is rejected upon any interceptions and changes to any of data fields/important data fields on the way to the server.

   Guideline
   https://www.ida.liu.se/~TDDD97/labs/hmacarticle.pdf

6. Testing Using Selenium [2 points]

This criterion requires you to test your application by using the Selenium testing framework. You need to test at least five (5) different requirements of your application. Please note that it is not allowed to use the Selenium IDE and instead you need to use another language like python for writing the tests.

Selenium official website
http://www.seleniumhq.org/

Selenium documentation
http://docs.seleniumhq.org/docs/index.jsp

7. Client-side Templating Using a Third-Party API [1 point]

By using templates, you can define an HTML structure that always remains the same. Inside the template, there are variable parts that change each time the template is rendered depending on the data provided. Templates can be used to implement different views inside of an application, for example. Here, you are required to use a third-party library, such as Mustache and HandlebarsJS, to perform client-side templating to reimplement the existing views inside of Twidder application.

Mustache official website
https://mustache.github.io/

HandlebarsJS official website
http://handlebarsjs.com/

8. Media Streaming [3 points]

This criterion requires you to have both Audio/Video and Image files stored at the server-side and streamed to the client when required. Users should be able to upload media files to the server, which stores them in a database or in a directory on the disc. For example, in your Twidder application, the user could be able to share media on his/her wall to be seen by other users. The user could also upload and set a profile picture for himself/herself to be seen by other users.

HTML5 Video
http://www.w3schools.com/html/html5_video.asp

Note: In case of storing media files on disc, make sure they are not directly accessible by using URL pointing to them. You need to add a service(s) that sends media files to the client based on the received parameters.

9. Styling and Responsive Design [2 points]

This criterion requires you to use a third-party framework, such as Bootstrap and Pure, for handling layout and styling. Also, you may need to to use your own CSS code for certain cases. You should follow the responsive design principles to make your application adaptable to different display resolutions from mobile to desktop for at least three different display-resolution ranges, mobile-view/tablet-view/pc-view. As a guideline, there are two areas in which your application needs to adapt to different screen sizes: 1. The *layout* and 2. The *resources* such as images which are displayed to the user. For example, in the PC view all media and images shall be displayed in multiple columns while in the mobile view no or a few images are displayed and all text is displayed in one column. The tablet view stands among the two.

Bootstrap official website
http://getbootstrap.com/

Pure official website
http://purecss.io/

Guide to Responsive Web Design
http://blog.teamtreehouse.com/modern-field-guide-responsive-web-design

Responsive Web Design Basics
https://developers.google.com/web/fundamentals/layouts/rwd-fundamentals/

HTML Responsive Web Design
http://www.w3schools.com/html/html_responsive.asp