SER421: Web Applications and Mobile Systems Fall 2018

Lab 2, due Thursday, 9/27/18 at 11:59pm via online submission to Blackboard

The goal of Lab 2 is to get you working in NodeJS, understanding the event loop, the file APIs, and the low-level HTTP module.

Activity 1 (15 points): HTTP

Summarize the following (meaning: define, indicate where it came from, and state its significance; think 2-3 sentences each):

1. HTTP/2
2. HTTPS
3. HTTP Strict Transport Security

- Use W3Techs to look up the percentage of Internet traffic that uses each of these 3 (for https use Default protocol https).
- Finally, determine a) the percentage of sites using compression, and b) the percentage of sites using cookies. Explain how you arrived at your answer.
- I am pretty insistent on writing in complete sentences. Please write like you are graduating from college!

Activity 2 (30 points): File I/O

For this activity you will implement a news service in NodeJS Javascript code using the "fs" and "xmldom" packages.

NewsService

The NewsService is an object representing a content management system for a hypothetical news organization. The NewsService should provide these features: *R1-R4 are worth 5 points, R5 10 points*

R1. The ability to write a new news story to the persistent store
R2. The ability to update the headline of an existing news story
R3. The ability to change the content of an existing news story
R4. The ability to delete an existing news story
R5. The ability to return a collection of NewsStory objects based on a filter, where the filter checks for one or more of:
  a. Substring of the title
  b. Date Range
  c. Author

Constraints/NFRs on the NewsService:

C1. The service should prevent concurrent read/write problems (for example, lost updates).
C2. A NewsStory is a type of object that represents a news article and is described by the following attributes: author (a username), title, public flag, story content, and date.
C3. The persistent store for the NewsService must be an XML file named news.xml and in the same directory as your top-level executable code. We will provide an example for you, and you are free to generate and share additional test cases

Overall Constraints:

C4. Put your code in 1 file, NewsService.js
C5. You do not have to write a user interface, but you do have to specify the API for your object types in your README.txt, and provide example starting files and a sample test case of each service. That is, you must provide examples of how to instantiate your service object, and how to invoke it so we can test it manually.
C6. Your code must be clear and well-written.
C7. Use the synchronous file I/O features in NodeJS (it makes C1 MUCH easier). If you are interested in making an asynchronous version, you may try and do so for extra credit (see the EC section).

Activity 3 (55 points): Implement a simple News delivery service

For this activity you will implement a web application leveraging your solution from Activity 2.

Background:

There are 3 roles in the system: "Author", "Guest", and "Subscriber". Here is what each can do:
  a. Guests can read any public news story
  b. Subscribers can read any news story, public or private
  c. Authors can read any public story, and private stories that s/he has authored

Login/Logout (use HTTP POST): **(12 points)**

R1. On the landing (home) page, provide an HTML form asking for a username, a password, and a radio button that selects the Role (default to Guest). Your login logic: *4 points*
  a. We will fudge authentication by simple testing that the username and passwords match (obviously not how we really want to do it!). If the login fails, display a page indicating failure with a link back to the landing page.
  b. If login is successful take the user to the "View News" page (R5)
R2. All screens for a logged in user should indicate the username and role, and provide a "Logout" link that logs the user out and returns the user to the landing page. *4 points*
R3. When the user returns to the landing page of the application in their browser, the application should remember what username that was last used on that browser, and their role. The landing page should say "Welcome [Guest/Author/Subscriber] <username>, please enter your password", and have the username pre-filled in the textbox for entering a username. *4 points, and this should be done via a specific mechanism (you need to figure this out) discussed in the class. If done any other way we will deduct 2 points.*

Features (use HTTP GET or POST as you feel is best): **(43 points)**

R4. A View News (the landing) page will display the titles of all news stories for all users. This page should do the following:

a. If the user in her/his role can View the story, then the title should be a hyperlink to that story (R6).

b. If the user in her/his role is not allowed to view the story, then only display the title of the story (not hyperlinked).
Based on the role-based requirements above. You have to test this requirement, both parts, for all roles:

- *For a Guest, only Public stories should be hyperlinked and viewable (3 pts), and non-public only title (2 pts)*
- *For an Author all Public plus stories that Author has authored should be hyperlinked and viewable (3 pts), and non-public/non-authored only title (2 pts)*
- *For a Subscriber, all stories should be hyperlinked and viewable (2 pts)*

R5. If the Role is Author then the View News page should have a link at the top for Creating a new news story. If clicked, an HTML form should be displayed allowing the Author to create a new news story with all of the fields, with "Save" and "Cancel" options to the form. Upon "Save", persist the story via the NewsService. *(6 points)*

a. If the save fails, stay on the Create Story page and provide an informative error message

b. If the save succeeds, then return the user to an updated View News page (R5)

R6. Render news stories that are hyperlinked from the View News page in an HTML-friendly format that you design. We do not grade on aesthetics, and no Javascript is allowed. It just has to be complete (all fields of a news story) and readable. News stories are provided by the NewsService. Each story structure is defined in Activity 2 Constraint C2. *(6 pts).*

R7. If the user is the Author of the story being viewed, then provide a link on the page rendering the story that allows the Author to Edit or Delete the story

a. If Delete, then invoke the delete functionality from Activity 2 and return to the View News page *(3 pts)*

b. If Edit, see requirement R8

R8. The Edit page should display an HTML form of your design pre-populated with data about the story. It should allow the Author to edit any part of the story and provide "Save" and "Cancel" options to the form. *(6 points)*

R9. A user should not be able to go to any page directly without logging in first. That is, you should not allow "bookmarkable" URLs; if the user attempts to go directly to a page via the browser bar then redirect to the landing page. *(4 points)*

R10. Error-checking:

a. You must decide the most appropriate HTTP method to use for any link or form (GET or POST). Your code should use only the proper method for a given action. If the wrong method is used, you should return the proper HTTP response code (you should look this up!). *4 pts*

b. Any unknown URL presented to the web server should return the proper response code. *2 pts*

Constraints:

C1. No Javascript in the browser whatsoever. CSS may only be used for embedded styling (no external links) and may not be used to satisfy any functional requirements (i.e. no "hiding" content via CSS to accomplish role differentiation).

- *If Javascript or CSS is used on the front-end to satisfy any requirement then give 0 points for that requirement).*
- *If Javascript appears in the browser in any form deduct 10 points.*

C2. No 3rd party packages or libraries unless we have discussed them in class. If you have a question on it then ask before you just go use something! *Do not award credit to features implemented via such a library, and deduct an additional 10 points.*

C3. URLs: the landing page should be at the root URL (/). You must run on port 3000. *deduct 5 points*

C4. As stated above your web application must use the NewsService from Activity2 without modification. *deduct 10 points for modifications*

C5. End users should never see stacktraces or other "developer" errors. Trap all errors you can think of and present user-friendly error messages and directions for recovering to the user. *deduct up to 5 points*

C6. Name your solution file activity3.js. Our test scripts will rely on this so please follow this instruction! *deduct 5 points*

EXTRA CREDIT:

EC1. (15 pts) You may implement Activity 2's NewsService using asynchronous I/O. In this case you may use features we have not discussed yet, namely promises and/or async (if you don't know what they are then I wouldn't try this route), or you may simply deal with callbacks, your choice. If you do this you must modify Activity 3 to use your asynchronous NewsService!

EC2. (10 pts) In Activity 3, introduce an event emitter that emits an event every time a user logs in, logs out, fails a login, or there is an HTTP error. The event should capture the client IP address and information relevant to the event, like the username for login/logout, or contextual information if it is an HTTP error. Add a listener that logs these events to a file events.txt (or json). How often should a logger write to a logfile?

SUBMISSION INSTRUCTIONS (READ CAREFULLY and ASK QUESTIONS!):

1. Create a zipfile named <asurite>_421Lab2.zip where <asurite> is your ASURITE id. No RAR or 7zip files!
2. The zipfile should have a root folder with the NodeJS code and xml files. You may populate whatever test cases you like in these directories with your code.
3. Also put your Activity 1 solution in there and name it Activity1.docx (or .odt or .txt)
4. In the root folder you should include a file named README.txt with any information you want us to know. For example, if you only completed some of the parts of a multipart activity, you can indicate that and what is undone (we consider partial credit). It should also contain your explanation of the NewsService as requested in Activity 2.
5. I strongly suggest, especially on programming problems, that you get a stable solution to a part, save it, and then move on. We do give partial credit. We allow as many submissions as you want to do and only grade the latest!