CS 142 Final Examination

Winter Quarter 2022

You have 3 hours (180 minutes) for this examination; the number of points for each question indicates roughly how many minutes you should spend on that question. Make sure you print your name and sign the Honor Code below. During the examination you may consult two double-sided pages of notes; all other sources of information, including laptops, cell phones, etc. are prohibited.

I acknowledge and accept the Stanford University Honor Code. I have neither given nor received aid in answering the questions on this examination.

____
(Signature)

____
(Print your name, legibly!)

____@stanford.edu
(SUID - Stanford email account for grading database key)

Problem	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10
Points	12	12	12	12	10	16	10	12	16	10
Problem	#11	#12	#13	#14	#15					Total
Points	10	15	12	11	10					180

Problem #1 (12 points)

The “same origin property” allows browsers to isolate cookies from different websites. Websites at different locations (i.e. hostname and port number) on the Internet can be assured that their cookies aren't accessible to the other locations. In addition to protecting cookie access at different locations, the “same origin property” also includes the scheme in the origin definition. Thus, a single website supporting different access protocols like HTTP and HTTPS might require multiple cookies, with one per location/protocol pair.

A. A website could choose to hand out the same cookie for both HTTP and HTTPS and not have to worry about different cookie values. Explain the advantage for the backend engineer of having the same cookie for both protocols.

B. Explain why having the same cookie value won't be a good idea and the website should use different values.

Problem #2 (12 points)

Our photoShare React.js program uses HTTP GET requests to fetch model data from the Node.js web server. For example, HTTP GET to the URL `/photosOfUser/:id` where `:id` is the MongoDB id of a user object will return model data of all the photos of a user.

Express.js allows our web server to fill in the HTTP response to the HTTP GET request so the body of the response has a JSON-encoded array of the photos. Express.js also allows us to set properties in the HTTP response header. For example:

```
response.set("Cache-control", "max-age=300");
```

will set the HTTP Cache-control property in the response header.

A. Explain the advantages and disadvantages (from the perspective of the web app developer) of setting this `Cache-control` value when fetching model data from `/photosOfUser/:id`?

B. What changes in app behavior would the end user of the photo app see from this change?

Problem #3 (12 points)

A. When using the MVC (model, view, controller) decomposition for view construction, all three MVC components must be present for the view to be rendered. When the rendering is done in the browser as we did for our React.js photoApp, we end up fetching the components from the web server. What can you say about the order that the components are fetched?

B. If we consider the most optimal solution for fetching the model data of MVC components, it would be to launch a single request that specified all the model data needed for the currently rendered MVC components. Explain why GraphQL is superior to REST APIs in achieving this optimal approach.

Problem #4 (12 points)

The Domain Name System (DNS) is the system used by browsers that allow URLs to contain hostnames (e.g. `www.stanford.edu`) rather than the actual IP address of the web server (e.g `146.75.94.133`). Although a Content Distribution Network (CDN) is not a browser, it also utilizes the DNS system. Describe how a CDN uses DNS.

Problem #5 (10 points)

REST APIs have web servers export the abstraction of resources that the client can access and manipulate. For example, the client can send an HTTP GET request to read a particular resource or an HTTP POST request to create a resource.

In systems, we describe an operation as being **atomic** if the operation as a whole will either happen or not. An atomic operation can't only partially update the state. The term gets its name from the notion that an atom is indivisible.

Describe the problem of using a REST API to implement an atomic operation that would create two resources.

Problem #6 (16 points)

We saw in class that code injection attacks can happen both in the browser and in the web server. Although these are very different environments potentially located on different continents, an attack in one location can be used to set up an attack in the other location. For each of the scenarios below, describe how an attack would work.

A. A code injection attack in a browser leads to a code injection attack on the server.

B. A code injection attack on the server leads to a code injection attack in the browser.

Problem #7 (10 points)

Message authentication codes (MACs) are normally generated on the server and sent to the client's browser. Consider modifying our Photo App to compute a MAC of each of the images stored in our system. We would then transfer an image's MAC along with the image to our web app frontend. What use, if anything, could our JavaScript code in the front end make with these image MACs? Explain your answer?

Problem #8 (12 points)

A. Describe what **extended validation certificates** have in addition to normal **certificates** when issued by a certificate authority.

B. Describe the problem **extended validation certificates** are trying to address.

Problem #9 (16 points)

In language environments with threads like Java and C++ there is usually a "sleep" function call that will pause the thread's execution for some amount of time. For example, the function:

```
function test(x) {
    console.log("A");
    let p = sleep(x);
    console.log("B");
}
test(10);
```

would output A followed by B 10 seconds later.

A. In JavaScript, you could write a "sleep" function that simply looped reading the time until time had advanced "x" seconds. The above function would have the same A followed by B 10 seconds later functionality. Explain why this wouldn't be considered an acceptable way to implement sleep in JavaScript.

B. JavaScript has promises to deal with this problem. Assume the "sleep" in the above code is a JavaScript function that returns a promise that is resolved in "x" seconds. Explain why the promise version would no longer exhibit the expected behavior (i.e. A followed by B in 10 seconds) and show what changes would need to be made to have the JavaScript work. Note that test is not declared to be an async function so await is not available.

Problem #10 (10 points)

The Express.js session module we used in our web server generated a cookie that contains a pointer to the session state stored in the web server's memory. The session state of the photo app was quite small in size so we could replace the pointer with the state itself. Assume we keep security used to protect the pointer and have it protect the session state.

Would the approach of keeping the session state in the cookie better scale to a large number of users, compared to the original pointer-based approach? Justify your answer.

Problem #11 (10 points)

An eavesdropper looking at the packets flowing between the browser and a web server would see very different content depending if HTTP or HTTPS protocol is being used. Nevertheless, the Express.js handlers in the web server can be identical for the two protocols. Explain how protocols with different packet contents can use the same Express.js handlers?

Problem #12 (15 points)

Cryptography has been helpful for addressing some of the attacks that web applications face. For each of the following attack types, state if cryptography could be helpful and if so, how.

A. Network Attacks

B. Session Attacks

C. Code Injection Attacks

Problem #13 (12 points)

The concept of a “**done callback**” function is widely used in JavaScript library interfaces. Rather than returning a value directly, the library routine will call the provided callback function at some later time with the requested value. For example:

```
      fs.readFile(fileName, doneCallback);
```

On the other hand, if the library routine has multiple different kinds of things it can return at different times, it can accept multiple callbacks. For example:

```
      routine(args, doneCallback1, doneCallback2, doneCallback3, …);
```

where the different callback functions are used to return the different values.

Things get complicated if there are many different return values that the caller may or may not be interested in. Explain the mechanism in Node.js that allows this kind of library interface to be more cleanly implemented. Describe how it works better than the multiple callback method.

Problem #14 (11 points)

HTTP uses TCP/IP protocol for communication. TCP/IP is what is known as a connection-oriented protocol where the protocol starts by establishing a connection between two machines much like a telephone call.

Explain how a scale-out architecture for web servers can be accomplished if the browser thinks it is calling up some machine by IP address for each request it sends.

Problem #15 (10 points)

The single-threaded nature of the JavaScript runtime in Node.js means that no two HTTP requests can be executing JavaScript at the same time. In spite of this limitation, Node.js can maintain multiple requests being processed at the same time by the MongoDB database. Explain how this is possible without concurrent JavaScript execution?