# CS 142 Final Examination

Winter Quarter 2019

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 |
|---|---|---|---|---|---|---|---|---|---|
| Score | | | | | | | | | |
| Max | 15 | 8 | 16 | 10 | 15 | 12 | 8 | 8 | 10 |
| | | | | | | | | | |
| Problem | #10 | #11 | #12 | #13 | #14 | #15 | #16 | Total | |
| Score | | | | | | | | | |
| Max | 10 | 15 | 15 | 10 | 12 | 8 | 8 | 180 | |

## Problem #1 (15 points)

Content distribution networks and Memcache servers are both storage systems commonly used increase performance and scale in large scale web applications. Although it would require changing some things, it would be relatively straightforward to remove one or both of the systems and have web app continue to function.

(a) Show your understanding of Memcache by describing what the problems or disadvantages you would see if you removed Memcache from a large scale web application. Your answer should specifically indicate what functioning would be affected.

(b) Show your understanding of Content distribution networks by describing what the problems or disadvantages you would see if you removed Content distribution networks from a large scale web application. Your answer should specifically indicate what functioning would be affected.

problem #1 continued...

(c) Both Content distribution networks and Memcache place limits on the kind of data it is appropriate to store in them. Describe the limitations for each of the systems.

## Problem #2 (8 points)

In class, it was recommended that a web application validate user input both in the front-end and back-end. A manager of the web app you are working on decided in a rush to market to only validate input in one of the two places. Which place (front-end or back-end) would you suggest doing the input validation if your web application could only have one? Justify your answer and include a discussion of the disadvantage of losing the input validation you choose to omit.

## Problem #3 (20 points)

In JavaScript, functions that block (e.g. access the database or wait for the user) need to be treated differently from ones that don't block. For example, if you had non-blocking functions (`op1` and `op2`) and wanted to pass 10 to `op1`, process the result with `op2`, and the log the output you could write:

```
var arg = 10;
var result = op2(op1(arg));
console.log(result);
```

### Callbacks

If `op1` and `op2` are blocking functions the programmer needs to change to use callback functions to return the result. Assume now that `op1` and `op2` now use callbacks to return results (e.g., `op1` function signature is now `op1(arg, callback)` where `callback` is a function that is called with the result of apply `op1` to `arg`).

(a) Write code that has the same functionality as above (i.e. console.log result) but uses the callback versions of `op1` and `op2`.

(b) Assume that the original `op1` was a simple increment by one function (i.e. `function op1(arg) { return arg+1; }`). Show what the corresponding code would be for the callback version.

```
function op1(arg, callback) {



}
```

problem #3 continued….

## Promise

Advocates of using Promises to handle blocking functions in JavaScript point to the less disruptive coding patterns required if all variables are Promises. Assume now that `op1` and `op2` now return promises as results (e.g., `op1` function signature is back to `op1(arg)`). The Promise advocates rejoice in the main part of the code:

```
var result = op2(op1(arg));
```

being identical when `arg` and `result` are promises.

Write code that has the same functionality as the original (i.e. console.log result) but uses the Promises everywhere. Hint: In order to get the main part described above working you need to create a promise that returns 10 (a promise version of `var arg = 10`) and console.log a promise.

(c) Using the same functionality assumption as part (b), show what the promise version of op1 would look like. Recall that `arg` is now a promise

```
function op1(arg) {




}
```

## Problem #4 (10 points)

(a) The word **state** was used several times during the course. We talked about (a) **state** management systems like Redux, (b) session ***state***, and (c) ***state***less web servers. How are the values that make up these states similar and/or different? Be sure to include examples of state value in your answer.

(b) In class we talked about (a) server**less** computing, (b) statel**ess** web server, is there any relationship between these two words with **less** in them? Justify your answer.

## Problem #5 (15 points)

(a) Web app-based discussion forms frequently define their own markup language (like the Markdown example given in class) rather than simply using HTML and having the browser's rendering engine do the markup. Besides syntax complexity disadvantage of HTML over something like Markdown, describe the key security problem with using HTML as a markup language in a web application.

(b) While doing a security audit on a web application, you see an HTML tag that looks like:

```
<img src="http://www.yourcompany.com/public/logo.jpg" />
```

The web application itself is served with https but the HTML coder felt that the company logo was public and didn't need encryption protection. Describe the problems with this reasoning.

Problem #5 continued...

(c) Most browsers will happily load HTML, CSS, JavaScript, images, or any other type of file from the local file system of the machine running the browser. This means JavaScript frameworks like Angular, ReactJS, and VueJS can run from the local machine. Explain why this setup doesn't work for programs that load content from a content distribution network.

## Problem #6 (12 points)

You are evaluating using a REST API or GraphQL for your web app backend. For each of the following metrics, state which of the two would do better on the metric. Be sure to include justification in all of your answers.

(a) Which of the two would result in fewer round-trip communications between the browsers and the web server? Justify your answer.

(b) Which of the options might have unnecessary values communicated from the web server to the browsers? Justify your answer.

Problem #6 continued...

(c) Which would be easier to load balance? Justify your answer.

## Problem #7 (8 points)

In Project 7, we had you add session management functionality to your photo sharing app. In the starter code we provided, we had you include the line:

```
app.use(session({secret: 'secretKey', resave: false, saveUninitialized: false}));
```

Explain what secret: ‘secretKey’ does, and why it is important we use it.  Describe what could go wrong if it was known your web app secret was 'secretKey'.

## Problem #8 (8 points)

In the ReactJS version of the projects, we had components communicate by explicitly passing callback functions between components. If component A needed to know something happened in component B, A would define a function and pass it to B which called the function when the something happened. Describe how the Listen/Emitter pattern we used in AngularJS handles this kind of communication more cleanly.

## Problem #9 (10 points)

(a) Give a brief description of the role of Middleware in ExpressJS.

(b) The webServer.js program used in the class projects utilized several ExpressJS middleware packages. The initial version of webServer.js given with Project #4 uses a middleware package and several other ones were added in later assignments. Describe the functionality provided by middleware modules in the class webServer.js including the middleware used in Project #4 and at least one of the other middleware modules added in later projects.

## Problem #10 (10 points)

The domain name system (DNS) is used in web applications.

(a) Describe how DNS can be used to do load balancing across a web application's servers.

(b) Describe the advantage that a load balancing network switch would have over using DNS is this way.

## Problem #11 (15 points)

(a) Describe an attack on a web application backend that can be done using the web application session cookie even though the attacker has no way of reading the session cookie.

(b) Explain why it is much easier for a man-in-the-middle attacker to switch a web app using HTTP to use HTTPS than it is to switch a web app using HTTPS to use HTTP.

problem #11 continued...

(c) Does HTTPS protect the URL of the HTTP request from the browser to the web server from a man-in-the-middle attack? Explain your answer.

## Problem #12 (15 points)

(a) MongoDB objects are stored in BSON (a binary version of JavaScript JSON format). We used Mongoose to enforce a schema on the objects. Explain the advantages of a database with a schema compared to JSON for returning model data.

(b) Explain why databases support multiple secondary indexes but only one primary index.

(c) Although Mongoose's syntax and functionality look like old ORM systems, explain why it is incorrect to call our use of Mongoose with MongoDB an ORM.

## Problem #13 (12 points)

In Node.js we could process all the elements of an array using the JavaScript array's `forEach` method like so:

```
['A', 'B', 'C'].forEach(processItemFunc);
```

We also introduced the `async.each` interface for processing all the elements of an array like so:

```
async.each(['A', 'B', 'C'], processItemWithCallbackFunc, doneFunc);
function doneFunc() {
    console.log('Done');
}
function processItemWithCallbackFunc(letter, callback) {
    …
    callback();
}
```

(a) What was the advantage of using a callback interface over the simple function call of `forEach`?

(b) Describe what is executed when the line `callback()` is executed. Include a description of what the code does.

## Problem #14 (12 points)

The HTTP protocol allows communication between the browser and the web server and this communication can be used to inform the opposite side to start doing something different or inform the other side of something.

(a) Explain what `Connection: keep-alive` in an HTTP request header tells the web server to do differently?

(b) Explain what `Cache-Control: max-age=120` in an HTTP response header tells the browser to do differently?

(c) Explain what an `X-XSRF-TOKEN` line in an HTTP request header tells the web server?

## Problem #15 (8 points)

The DOM in modern web browsers allows communication to the web application backend using with XMLHttpRequest or the new WebSockets API. Describe the advantage that using WebSockets would have over XMLHttpRequest when being used by a web application to talk to its backend?

## Problem #16 (8 points)

Early web application frameworks like Ruby on Rails did all view rendering on in the web server with the view rendered into HTML that is shipped to and displayed in the browser. JavaScript web app frameworks like AngularJS render the view directly into the browser's DOM from JavaScript running the browsers. The current generation of JavaScript web application frameworks like ReactJS/Angular/VueJS decouples the rendering from the browser's DOM which allows the rendering to be done either in the browser or in the web-server (ie. serverside rendering). Given all the advantages of having JavaScript directly render into the DOM, why would it ever make sense to go back to rendering on the server?