CS 142 Fall 2017 Final Exam Solutions

Problem 1 (8 pts) (Mendel)

GraphQL contains two advantages over REST APIs that address the listed challenges. GraphQL allows the requestor to specify the properties that should be retrieved allowing the smaller screens with less information display to fetch only the information needed for the view as opposed to what is specified as the resource of an REST endpoint. This also lowers the amount of bandwidth required addressing the second challenge. GraphQL also allows multiple requests to be bundled together in the same GraphQL query allowing a single round-trip to the server to fetch all of a view's model data that might take multiple round trips for REST resources. Reducing the number of round-trips is a big win on high latency networks.

Problem 2 (8 pts) (Ellen)

A

2 points for reasonable explanation of scale out advantages (1 if on the right track)
Reasonable choices:

- Fault tolerance
- Cost efficiency
- Won't have to adjust to big expansion later (less short-sighted solution?)
- The individual machines are likely to be cheaper

2 points for reasonable explanation of scale up advantages (1 if on the right track)

- Ease of configuration
- OK if not many users

B

Person gets 4 points if they adequately synthesized the above into a real app context (most should get full points), saying something about the growing number of users, etc.

2 points if explanation is vague or something said invalidates a correct answer

Other points people mentioned:

- Our photo app is stateless, making it a good candidate for scale-out

Problem 3 (8 pts) (Mendel)

Assigning to innerHTML is an easy way to inject user generated content into the DOM but it also leaves the app open to XSS attack. In the user generated content is adding to the DOM by creating and modifying DOM nodes the app will need to look at the content explicitly create/modify DOM nodes. Any XSS attack would need to have this code generate SCRIPT tags in the DOM. Since there is little reason for including code to generate SCRIPT tags the security hole would need to be explicitly coded into the system. Whereas the innerHTML approach requires the developer to explicitly detect and code around XSS attacks, the DOM approach would stop the XSS attack unless the developer when out of their way to support it. Hence the DOM modification approach wouldn't be considered risky.

Problem 4 (8 pts) (Hana)

- Mongoose maps ORM concepts to MongoDB
- Mongoose allows us to make schemas (can also refer to these as “models”) for more structured queries to the DB
- Mongoose takes care of type casting and validation
- Any other correct explanation of something Mongoose improves, adds, or makes easier

+4 points for each piece of information, up to a max of 8
-2 points for each piece that is vague or inadequately explained

Problem 5 (8 pts) (Hana)

5a

+3 points for mentioning a secondary index on usernames
+2 points for explaining why a secondary index would speed up queries (no need to scan row by row)

5b

+3 points for explaining that the DB must update the index every time a new user account is created, which takes extra time

Problem 6 (8 pts) (Nina)

6a

Use public-key cryptography. A server will get a certificate from certificate authority, this certificate proves the server's identity. When establishing connection with the server, the user will verify the server's identity given the certificate. Once verified, the user will do key exchange with the server by signing it using the server's public key. Only the server will be able to decrypt it using server's private key, so they can share a secret key.
Rubric:
+1 if mentioned public, private key pair
+1 if mentioned actual key exchange process
+2 if mentioned certificate authority to get server's public key

6b
If the attacker gets the session cookie it can just impersonate the current user using that session cookie directly. However, the attacker won't be able to modify the session cookie to impersonate a different user because MAC is used to detect data tampering.
Rubric:
1.5 marks for each part of the question

Problem 7 (8 pts) (Jason)

2 points for each attack and 2 points for each mitigation

attacks:

/register: have the bot create many fake users to take up server processing time and database space
OR
/login: using the fake users above, login to take up valuable session storage space
OR
/userPhoto: have the bot login and retrieve server intensive resources like the user photo list
OR
/newPhoto: continuously add large photos to fill up the database
OR
other reasonable answer

mitigations:

Front end:
include a captcha before allowing the site to send resource-intensive API calls like fetching a user's photos
OR
other reasonable answer

Back end:
add more resources (?)
OR
keep track of users and limit them if they make too many requests within a given timeframe
OR
include caches for commonly requested resources like userList
OR
other reasonable answer

Problem 8 (8 pts) (Jeff)

$http is more general. $resource follows REST api guidelines. They both use xmlhttprequest under the hood. $resource only does CRUD operations (assumes its a RESTful api dealing with models), while $http can perform any type of HTTP request, to any type of server. (Note: $http could also be used to communicate with a REST api server, because any web server uses http to communicate).

$resource could be built on top of $http. This is because $http is a lower level API and is more flexible/powerful. $resource is too restricting (it assumes RESTful principles), so could not be used to implement $http.

Problem 9 (8 pts) (Jeff)

Using caching reduces server load for resources (images, html, css, etc.) that do not change very often.
Disadvantage: If the resource changes, the updated version is not sent until the cache expires.

Problem 10 (6 pts) (Jeff)

Pause parsing and rendering of HTML, send HTTP or HTTPS GET for the url in the src attribute. Once it downloads, the browser runs the javascript code to completion. Finally, html parsing and rendering resumes.

Problem 11 (8 pts) (Ellen)

A (6 points)

1) 1 point for GET; 2 points for components and explanation (there are multiple correct answers)

    a) Answer: Expected students to answer “all” (but in a single request may not need DB)
2) 1 point for POST; 2 points for components and explanation
    a) Answer: Expected the students to answer “all systems besides memcache”
    b) Note: if the person included the memcache but had a reasonable explanation (e.g. storing session), then that’s fine

B (2 points)

1 point for saying (1); 1 point for saying something about either memcache or writes potentially being slower if some explanation is provided for why they think writes are slow

Problem 12 (6 pts) (Mendel)

Computing is done on servers regardless how cloud computing is marketing. The serverless refers to the abstraction exported by the cloud computing vendor. Early cloud computing vendors exports an abstraction of a virtual machine (i.e. a server) you could run your software in. The developer would configure one or more virtual servers to implement the backend of a web application. In the serverless approach the developer doesn't view the cloud as a collection of servers but instead provides code that is run by a service being run by the cloud computing provider. Rather than viewing the backend as a collection of servers, the develop thinks in terms of events and functions to run when the events happen.

Problem 13 (10 pts) (Ellen)

A (4 points)

Socket connection stays alive longer than a single request, which doesn’t work with load balancing.
Alternate answer (that only got credit if very precisely explained): WebSockets allow TCP connections, which violates the GAE API, which is supposed to be a clean HTTP request handler.
Give 2 points if explanation is close or hits at some other issue that isn’t the main reason (e.g. “user can write network related activity, which GAE doesn’t like” **(was quite strict about what actually got the 2 points--few students got partial credit)**

B (6 points)

3 points for reasonable advantage(s); 3 points for reasonable disadvantage(s)
Advantage examples:

- Scale easily

- Optimally configured security, etc.

Disadvantage examples:

- Inflexible
- Hard to switch backend to something else later

May lose a point or 2 if the reason is correct, but the person invalidates that reason with something incorrect.

Problem 14 (10 pts) (Marcella)

2 points for each part.
-1 for no/wrong/insufficient explanation.
-1 for wrong property

A. Body
   - Security purposes. We don't want to expose user's password (sensitive information)

B. Query Params
   - In this request, we are doing a search/filter. Given the many possible combinations of the input string, query params would be the best place to put the search string.

C. Body
   - This is a POST request to the server and we are posting user's information, so we don't want to expose it in the URL.
   - Occupation info is not unique, so not relevant to put it in the URL.

D. Hierarchical part
   - We are fetching the details for a **dedicated** page in the app (for one photo), and the photo ID is **unique**.

E. Query Params
   - Similar to (B)

F. Body
   - Given the input is the pixel bounding box (4 corner points of the rectangle), the body is the best place to populate.

G. Hierarchical part
   - Similar to (D). The input photo ID is unique and we are not doing any search/filtering.

H. Body
   - Photo image can only be appended/encoded to the request body.

Problem 15 (12 pts) (Hana)

15a

Solution:

a) Incorrect count (will return 0)
b) Uses Express incorrectly (will try to send multiple responses)
c) Correct count
d) Incorrect count (will likely return 0)

+2 points for each correct classification
-1 point for each incorrect value for number of users returned

15b

+4 points for a correct explanation of what happens (request eventually times out and an error is displayed in the browser)
-2 points for each incorrect event (e.g. server crashes, browser hangs forever)

Problem 16 (8 pts) (Jason)

source: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Promise/then

```
p.then(onFulfilled[, onRejected]);
p.then(function(value) {
  // fulfillment
}, function(reason) {
  // rejection
});
```

```
+  4 for correct .then syntax
+  2 for correct function for fulfilled promise
+  2 for correct function for rejected promise
```

Problem 17 (12 pts) (Jason)

PART A

1. YES [1 pt]
   read only content that doesn't change often [2 pt]
2. NO [1 pt]
   model data for a view changes often with interactions to the web application, thus putting it in a CDN is disadvantageous. [2 pt]
3. NO [1 pt]
   controller code is often changed to make updates to features of the web app. [2pt]

If the YES/NO answer is incorrect, the student can still earn up to 1.5 pts/part if the short justification is reasonable.

PART B

DNS servers will resolve the DNS query to IP addresses of different servers depending on the physical location of the browser running the web app. [3 pt]

Partial answers [1.5 pt]

Problem 18 (12 pts) (Marcella)

3 points for each part: 1 point for correct Yes/No, 2 points for correct explanation.
-3 for wrong answers

For A-B:

-2 Wrong explanation

For C-D: If correct Yes/No:

-2 No explanation
-1 Wrong explanation

A. Yes
   - The middleware can be used to block all non "read-only" checkpoints.

   OR

   No
   - The "read-only" can be encoded in the user schema.

B. Yes
   - When the user request is received by the server. The middleware can check the quota (fetch the quota info and check) before forwarding it to the other endpoints.

   OR

   No

- Checking in the middleware might require the backend to do a lot of query to the database to get the quota information. So the each endpoint should handle this check.

C. Yes
- Middleware can parse the body from the request, to filter for the obscene/offensive words before passing it to the next endpoints.

D. Yes
- Middleware can be attached to all endpoint and can be used to print information about incoming REST call before procedding to the next endpoints.

Problem 19 (8 pts) (Nina)

Javascript runs a single-threaded event loop that will handle events in event queue, which means that if busy waiting is used in an event, it will halt the execution of all other events.
Rubric:
Give 4 marks if didn't mention single-threaded, but give a reasonable explanation of why events will be blocked.

Problem 20 (8 pts) (Marcella)

-1 Correct outputs, incorrect ordering.
-1 for each incorrect output (max: -7)

Console Output:
Start
A
A
Async done
D
E
End

Problem 21 (8 pts) (Nina)

Using the prototype makes for faster object creation (also save space), since that function does not have to be re-created each time a new object is created.
Rubric:
Give 4 marks if give the following answer:
Using prototype allows all classes that inherit Rectangle to have access to the area function as well.