University of Toronto Scarborough

CSCC09

Spring 2021

Final Exam

First Name: _____

Last Name: _____

Student Number: _____

_____ @mail.utoronto.ca

Do not turn this page until you have received the signal to start.

In the meantime, fill out the identification section above and read the instructions below carefully.

This 3 hours exam contains 22 pages (including this cover page) and 6 parts. Check to see if any pages are missing. Enter all requested information on the top of this page, and put your initials on the top of every page, in case the pages become separated.

You may **not** use notes, or any calculator on this exam.

There are several types of questions identified throughout the exam by their logos:

✓$_1$	select **the best answer** only among multiple choices.
✓$_\star$	select **all correct answers** (and those ones only) among multiple choices. At least one applies.
my answer	fill the blanks with a **short answer** (less than 5 words).
•—•	connect **each item from the left side** to one (and one only) item from the right side. A right side item might be connected to several left side items. The right side and the left side might not necessarily have the same number of items.

For each question, the following rules apply:

- **There is no partial credit (unless stated otherwise)**. This means that, for a given question, you either receive the full mark allocated if all answers are correct or 0 if there is any mistake.
- **Your answer should be clear**. Your answer to a question might be considered as incorrect if there is:
  - any part of the answer that cannot be read
  - any part of the answer that cannot be associated with its corresponding question part

Do not write anything in the table below.

Part:	1	2	3	4	5	6	Total
Points:	20	20	20	20	20	8	108
Score:							

1. **Part 1**

(1.1) 3 points - •—•

Match each technology with its concept:

HTML	•	•	Presentation
CSS	•	•	Processing
Javascript	•	•	Content

HTML •—• Content

CSS •—• Presentation

Javascript •—• Processing

(1.2) 5 points - **my answer**

Complete the HTML (by filling-in with the appropriate attributes) and CSS (by filling-in with the appropriate accessors) so that:

- The first `span` is the unique element in the entire DOM that has a blue background
- The first and second `span` elements have a `2px red solid` border around each of them
- The third `span` element has no border and no background set

HTML

```
<span _____ >first</span>
<span _____ >second</span>
<span>third</span>
```

CSS

```
_____ {
    background: blue;
}
_____ {
    border: 2px red solid;
}
```

HTML

```
<span id='header' class='row' >first</span>
<span class='row'>second</span>
<span>third</span>
```

CSS

```
#header {
    background: blue;
}
.row {
    border: 2px red solid;
}
```

(1.3) 1 point - ✓1

By mistake, a frontend developper has copy-pasted the same piece of Javascript twice in the source code (as shown below). How many times is the word `hello` printed on the console when `elmnt` is clicked:

```
elmnt.onClick = function(event) {
    console.log("hello");
};
elmnt.onClick = function(event) {
    console.log("hello");
};
```

◯ 0

√ 1

◯ 2

◯ more than 2

(1.4) 1 point - ✓1

By mistake, a frontend developper has copy-pasted the same piece of Javascript twice in the source code (as shown below). How many times is the word `hello` printed on the console when `elmnt` is clicked:

```
elmnt.addEventListener('click', function(event) {
    console.log("hello");
});
elmnt.addEventListener('click', function(event) {
    console.log("hello");
});
```

◯ 0

◯ 1

√ 2

◯ more than 2

(1.5) 1 point - ✓1

The Document Object Model is

- ◯ an API for storing structured data locally (in the browser)
- ◯ a way to structure NoSQL databases
- √ the tree structure of a webpage
- ◯ an API to access the browser configuration

(1.6) 2 points - **my answer**

An HTTP server runs behind the port 80 by default whereas an HTTPS server runs behind the port 443 by default.

(1.7) 6 points - **my answer** (or 3 points for one up to two wrong answers)

In which part of the HTTP request message are each of these piece of information stored? Each answers is either in the **headers** or in the **body**.

- the login and password (using basic authentication) are stored in the headers
- the content type (a.k.a body encoding) is stored in the headers
- he cookies are stored in the headers
- the cookie flags are stored in the headers
- the user's preferred locales (a.k.a accepted languages) are stored in the headers
- the content of uploaded files is stored in the body

(1.8) 1 point - ✓1

HTTP/2 allows

- ◯ the backend to bundle javascript and CSS files into one file sent back as an HTTP response
- ◯ the frontend to request multiple resources with a single HTTP request
- ◯ the frontend and the backend to compress HTTP requests and responses
- √ the backend to reply to a single HTTP request with multiple HTTP responses

2. **Part 2**

(2.1) 2 points - ✓★

Which HTTP methods are safe (i.e should not have any side effect):

- ☐ POST
- ☐ PUT
- √ GET
- ☐ PATCH
- ☐ DELETE

(2.2) 1 point - ✓1

When I enter the URL c09rocks.com in my browser, the browser sends an HTTP request with the method:

- ◯ POST
- ◯ PUT
- √ GET
- ◯ PATCH
- ◯ DELETE

(2.3) 2 points - ✓★

Which methods are used when retrieving images, scripts and stylesheets:

- ☐ POST
- ☐ PUT
- √ GET
- ☐ PATCH
- ☐ DELETE

(2.4) 8 points - **my answer** (or 4 points for one up to three wrong answers)

Assuming that the browser sends an HTTP request to the server to access some user's profile information. For each scenario, fill-in the blanks with the appropriate error code family that the server should returned. Each answer is either `1xx`, `2xx`, `3xx`, `4xx` and `5xx` (the last two digits are not important and will be ignored).

- ____5xx____ when the database has crashed and the user's profile cannot be retrieved
- ____2xx____ when the user's profil is returned (JSON content type)
- ____2xx____ when the user's profil is returned (HTML content type)
- ____4xx____ when the request is not authenticated to access the user's profile
- ____4xx____ when the request is authenticated but not authorized to access that user's profile
- ____3xx____ when the user's profile URL has changed and the server automatically redirects the browser to another URL
- ____4xx____ when the user's ID is not in the database
- ____4xx____ when the user's ID is ill-formed (invalid format)

(2.5) 1 point - ✓1

JSON is

- ◯ a way to structure a NoSQL database
- √ a way to represent structured data as strings
- ◯ a special HTTP request to send structured data to the server
- ◯ a specific javascript data structure

(2.6) 1 point - ✓1

JSON data can be manipulated with Javascript only:

- ◯ true
- √ false

(2.7) 1 point - ✓1

JSON is the only way to send structured data to the server:

- ◯ true
- √ false

(2.8) 1 point - ✓1

When a login and password form is submitted to the backend *without using Ajax*, the data is by default encoded as:

- ◯ application/json
- ◯ application/json-form
- √ application/x-www-form-urlencoded
- ◯ multipart/form-data

(2.9) 1 point - ✓1

When uploading a file, the content-type of the request should be set to:

- ◯ application/json
- ◯ application/x-www-form-urlencoded
- √ multipart/form-data
- ◯ the file mimetype

(2.10) 1 point - ✓1

When downloading a file, the content-type of the response should be set to:

- ◯ application/json
- ◯ application/x-www-form-urlencoded
- ◯ multipart/form-data
- √ the file mimetype

(2.11) 1 point - ✓1

Files can be uploaded using Ajax:

- √ true
- ◯ false

3. **Part 3**

(3.1) 7 points - **my answer** (or 3 points for one up to three wrong answers)

Let's assume that the database stores a collection of items. Each item has a unique id and some attributes. Following the REST design principles, provide the HTTP method and the path for each assertion below (the first one is given):

- Retrieve the item `id=42`: `GET /items/42/`
- Delete all items: DELETE /items/
- Replace all items with new ones: PUT /items/
- Create a new item given its `id=42`: PUT /items/42/
- Retrieve all items (no pagination): GET /items/
- Delete the item `id=42`: DELETE /items/42/
- Create a new item without specifying its id: POST /items/
- Replace some attributes for the item `id=42`: PATCH /items/42/

(3.2) 2 points - **my answer** Let's assume that the database stores a collection of user's profiles. We are not allowed to retrieve all users from the database but we are allowed to retrieve all users that have the same given lastname. Following the REST principle, complete the following query string to get all users named *Sans*: `GET /users/` ?lastname=Sans

(3.3) 2 points - ✓★

A piece of javascript code executed on the frontend (browser) can

√ modify a local storage key/value pair

√ modify an unprotected cookie key/value pair

□ modify a session key/value pair

(3.4) 2 points - ✓★

A piece of javascript code executed on the backend (node.js) can

□ modify a local storage key/value pair

√ modify a cookie key/value pair

√ modify a session key/value pair

(3.5) 1 point - ✓$_1$

Assuming that the client has been authenticated (stateful authentication), what happens if the client tries to access to a protected resource without sending back the session ID cookie to the server:

◯ the server recovers the session from the database, recreates the session ID cookie and grants access to the ressource

◯ the server generates a new session, recreates the session ID cookie and grants access to the ressource

√ the server denies access to the ressource

(3.6) 2 points - ✓★

Using stateful authentication

√ the same user can be authenticated on two different computers at the same time

√ the same user can be authenticated on the same computer using two different browsers at the same time

□ two users can be authenticated on the same browser at the same time

√ two users can be authenticated on the same computer using two different browsers at the same time

(3.7) 4 points - ✓★ and **my answer**

A web application **A** uses a third-party authentication (OAuth) service from **B** to authenticate a client **C**. Considering the options below, check the ones that are part the OAuth scheme and leave the others unchecked. For all options that you checked (and only those ones), fill the blanks with either: **A**, **B** or **C**.

√ C sends the login and password to B

□ the latter forwards the login/password to _____

√ the latter verifies the login/password and returns a token to C

√ the latter forwards the token to A

√ the latter ask C to verifies the token and return the user's profile

4. **Part 4**

(4.1) 1 point - ✓1

What is the purpose of in the `xhr.withCredentials` instruction in the javascript code below?

```
function request(method, url, type, content, callback){
    window.onload = function(e){
        var xhr = new XMLHttpRequest();
        xhr.onload = callback
        xhr.open(method, url, true);
        xhr.setRequestHeader('Content-Type', type);
        xhr.withCredentials = true;
        xhr.send(JSON.stringify({content: content}));
}
```

- ◯ send login and password over Ajax using the basic authentication method
- √ send the authentication cookie with cross-origin Ajax requests
- ◯ enable the `secure` cookie flag for the authentication cookie before sending it over through Ajax
- ◯ this instruction does not exist actually

(4.2) 1 point - ✓1

When visiting a specific webpage, the browser says that "the website has a non trusted certificate", it means that

- √ the certificate is not signed by a certificate authority known by my browser
- ◯ the browser does not know the private key associated with the certificate
- ◯ the certificate is not signed by the server hosting the website
- ◯ the browser does not recognized the certificate because it is the first time I am visiting this website

(4.3) 1 point - ✓1

Mixed-content occurs when

√ the client retrieves HTTP and HTTPS content from the same domain

◯ the server sets the wrong content-type header

◯ the server does no correctly sanitize users inputs

◯ the client retrieves a malicious script

(4.4) 3 points - •—•

Match each cookie flag with the type of attack it tries to mitigate (but not necessarily prevent)

`httpOnly`	•	• prevents the cookie from being read or writen by frontend javascript
`secure`	•	• prevents the cookie from being sent over HTTP
`sameOrigin`	•	• prevents the cookie from being sent sent over cross-domain requests

`httpOnly`	•—•	prevents the cookie from being read or write by frontend javascript
`secureFlag`	•—•	prevents the cookie from being sent over HTTP
`sameOrigin`	•—•	prevents the cookie from being sent sent over cross-domain requests

(4.5) 1 point - ✓1

I18N (Internationalization) is the process of developing a software that

√ is language agnostic

◯ is automatically adapted for a specific language using a translation API

◯ is specifically adapted for a specific language using a locale

(4.6) 1 point - ✓1

L10N (Localization) is the process of developing a software that

◯ is language agnostic

◯ is automatically adapted for a specific language using a translation API

√ is specifically adapted for a specific language using a locale

(4.7) 2 points - ✓★

Assuming that the browser loads a webpage from **A** that contains a piece of javascript code. Without considering any cross-sharing HTTP request, what resources from another domain **B** can be loaded by this piece of code without violating the same origin-policy:

√ a script using the `SCRIPT tag`

√ a stylesheet using the `LINK tag`

☐ a script using an Ajax request

☐ some JSON data using an Ajax request

√ an image using the `IMG tag`

√ a piece of HTML using the `IFRAME tag`

(4.8) 1 point - ✓1

Assuming that the browser loads a webpage from A that contains a piece of javascript code that performs a cross-origin Ajax request to B. What happens if the same-origin policy is violated?

◯ the client's browser blocks the HTTP request going from the client to **B**

◯ **A** blocks the HTTP request going from the client to **B**

◯ **A** blocks the HTTP response coming from **B** back to the client

√ the client's browser blocks the HTTP response coming from **B** back to the client

◯ **B** blocks the HTTP request coming from the client

(4.9) 2 points - ✓★

Assuming that the browser loads a webpage from A that contains a piece of javascript code that performs a cross-origin Ajax request to B. How to enable the Cross-Origin Ressource Sharing (CORS)?

☐ the client's browser sets the `Access-Control-Allow-Origin` header in the HTTP request going to **A**

☐ the client's browser sets the `Access-Control-Allow-Origin` header in the HTTP request going to **B**

- ☐ **A** sets the `Access-Control-Allow-Origin` header in the HTTP response going back to the client
- √ **B** sets the `Access-Control-Allow-Origin` header in the HTTP response going back to the client

(4.10) 1 point - ✓1

What is the purpose of setting up the `Access-Control-Allow-Credentials` header to `true` in the HTTP response of a cross origin request?

- ◯ to allow the browser to send the authentication cookie with the next cross-origin request
- √ to allow the browser to expose the response to frontend JavaScript code when the authentication cookie was included in the cross-origin request
- ◯ to set the `sameOrigin` cookie flags in the response to the cross-origin request to protect the authentication cookie
- ◯ this header does not exist

(4.11) 1 point - ✓1

To own the domain c09rocks.com, one should ask

- √ a domain name registrar
- ◯ an internet service provider
- ◯ a certificate authority
- ◯ the server that hosts the website

(4.12) 1 point - ✓1

Memcached is

- ◯ a technique to shadow the web server's memory in case of a crash
- ◯ a library for optimizing raw database requests
- ◯ an operation that pushes memory overflow to the database
- √ a distributed shared cache library for dynamic content

(4.13) 1 point - ✓$_1$

A load balancer

- ◯ restructures a database dynamically and improve its efficiency
- √ takes incoming HTTP requests and spread them across multiple web servers internally
- ◯ synchronizes the databases between different web servers
- ◯ routes the traffic to different web servers at the DNS level

(4.14) 1 point - ✓$_1$

A CDN is

- ◯ restructures a database dynamically and improve its efficiency
- ◯ takes incoming HTTP requests and spread them across multiple web servers internally
- ◯ synchronizes the databases between different web servers
- √ routes the traffic to different web servers at the DNS level

(4.15) 2 points - ✓$_\star$

Web analytics can be done:

- √ on the browser side
- √ on the server side

5. **Part 5**

```
function foo(){
    Array.from({length: 1000000}, () => Math.random()).sort();
}
console.log('a')
setTimeout(function(){
    console.info('b');
}, 2000);
console.log('c')
foo();
console.log('d');
```

In this program, we can assume that the function foo takes roughly 6 sec to execute.

(5.1) 2 points - ✓1 and **my answer**

Can you deterministically predict in which order the messages a, b, c, and d will be printed on the console, based on your understanding of Javascript non-blocking I/O and its event-loop execution model?

◯ no, we cannot predict the order (non deterministic execution)

√ yes, we can predict the order: 1. a, 2. c 3. d and 4. b

(5.2) 2 points - ✓1 and **my answer**

Can you deterministically estimate how long it will take for the program to terminate?

◯ no, we cannot predict the time

√ yes, we can predict that it will take 6 sec (+/-1 sec)

```
setTimeout(function(){
    console.info('a');
}, 9000);
console.log('b')
setTimeout(function(){
    console.info('c');
}, 4000);
console.log('d');
```

(5.3) 2 points - ✓1 and **my answer**

Can you deterministically predict in which order the messages `a`, `b`, `c` and `d` will be printed on the console based on your understanding of Javascript non-blocking I/O and its event-loop execution model?

◯ no, we cannot predict the order (non deterministic execution)

√ yes, we can predict the order: 1. b, 2. d 3. c and 4. a

(5.4) 2 points - ✓1 and **my answer**

Can you deterministically estimate how long it will take for the program to terminate?

◯ no, we cannot predict the time

√ yes, we can predict that it will take 9 sec (+/-1 sec)

```
function timer(time, msg){
    return new Promise(function(resolve, reject){
        setTimeout(function(){
            console.log(msg);
            resolve();
        }, time);
    });
}
timer(9000, 'a').then(function(){
    timer(4000, 'b').then(function(){
        console.log('c');
    })
});
console.log('d');
```

(5.5) 2 points - ✓1 and **my answer**

Can you deterministically predict in which order the messages `a`, `b`, `c` and `d` will be printed on the console based on your understanding of Javascript non-blocking I/O and its event-loop execution model?

◯ no, we cannot predict the order (non deterministic execution)

√ yes, we can predict the order: 1. d, 2. a 3. b and 4. c

(5.6) 2 points - ✓1 and **my answer**

Can you deterministically estimate how long it will take for the program to terminate?

◯ no, we cannot predict the time

√ yes, we can predict that it will take 13 sec (+/-1 sec)

```
function timer(time, msg){
    return new Promise(function(resolve, reject){
        setTimeout(function(){
            console.log(msg);
            resolve();
        }, time);
    });
}
Promise.all([timer(9000, 'a'), timer(4000,'b')]).then(function(){
    console.log('c');
});
console.log('d');
```

(5.7) 2 points - ✓1 and **my answer**

Can you deterministically predict in which order the messages `a`, `b`, `c` and `d` will be printed on the console based on your understanding of Javascript non-blocking I/O and its event-loop execution model?

◯ no, we cannot predict the order (non deterministic execution)

√ yes, we can predict the order: 1. d, 2. b 3. a and 4. c

(5.8) 2 points - ✓1 and **my answer**

Can you deterministically estimate how long it will take for the program to terminate?

◯ no, we cannot predict the time

√ yes, we can predict that it will take 9 sec (+/-1 sec)

```
function timer(time, msg){
    return new Promise(function(resolve, reject){
        setTimeout(function(){
            console.log(msg);
            resolve();
        }, time);
    });
}
async function run() {
    await timer(9000, 'a');
    await timer(4000,'b');
    console.log('c');
};
run();
console.log('d');
```

(5.9) 2 points - ✓1 and **my answer**

Can you deterministically predict in which order the messages `a`, `b`, `c` and `d` will be printed on the console based on your understanding of Javascript non-blocking I/O and its event-loop execution model?

◯ no, we cannot predict the order (non deterministic execution)

√ yes, we can predict the order: 1. a, 2. b 3. c and 4. d

(5.10) 2 points - ✓1 and **my answer**

Analyzing the program above, can you deterministically estimate how long it will take for the program to terminate?

◯ no, we cannot predict the time

√ yes, we can predict that it will take 13 sec (+/-1 sec)

6. **Bonus**

There are no bad answers to the questions below. You will get points if the instructor believes you genuinely made some efforts to provide an answer. Keep in mind that quantity does not beat quality.

(6.1) 2 points - **my answer**

What are the topics that you did not know before taking this course and you enjoyed learning

(6.2) 2 points - **my answer**

What are the topics related to web development that you would like to learn and that was not covered in this course

(6.3) 2 points - **my answer**

Would you recommend this course to other students?

(6.4) 2 points - **my answer**

Share something with the course staff

Thank you for this great semester and have a good summer!