CS 337

Wrap Up & Review

Benjamin Dicken (modified by Samantha Cox)

SCS

- Response Rate: 73.28% (as of 12:30pm)
  - Reminder: we need to reach **80%** for a PA drop!

Presentation Recording

- 5-8 minutes
- Spend the majority of the time showing the application and it’s functionality
- If you would like, can spend ~1 min at the end giving an overview of the file structure and code
- You should have a screen recording + audio explanation
    - Video camera is optional
- Can use quicktime or OBS

Wrap up

- Do you feel more confident with your web programming knowledge?
- What topics did you find most interesting?

Review

- Cumulative exam
- Wednesday, December 13th, 3:30-5:30pm
- Lets go over a few review questions

Question 1: Password security

1. In class, we talked about password salting and hashing. Answer each of the following with 1-2 sentences:

    a. There are many hash functions, but only some are suitable for password hashing. What are 3 important properties of a password hashing function?

    b. Why is salting needed? Why is salting + hashing better than just hashing on its own?

Question 1: Password security

1. In class, we talked about password salting and hashing. Answer each of the following with 1-2 sentences:

    a. One-way, fast, uniqueness, consistent length

    b. So that same passwords can lead to different hashes

Question 2: MongoDB and Mongoose

In this question, you should write the complete schema that can model a book. The schema should include information about the **name** (string), **ISBN number** (string), and **page count** (Integer). It should also have a list of **Author** schema types (by ID). Write the code to create the schema.

Question 2: MongoDB and Mongoose

```
var AuthorSchema = new Author({
    name: String,
    . . . .
});
var Author = mongoose.model('Author', AuthorSchema );

var BookSchema = new Schema({
    name: String,
    isbnNumber: String,
    pageCount: Number,
    authors: [ {type: mongoose.Types.ObjectId, ref: 'Author'} ]
});
var Book = mongoose.model('Book', BookSchema );
```

Question 3: AJAX

Rewrite this code using fetch for the AJAX

```
function getItemsForList(listName) {
  var httpRequest = new XMLHttpRequest();
  if (!httpRequest) { return false; }

  httpRequest.onreadystatechange = () => {
    if (httpRequest.readyState === XMLHttpRequest.DONE) {
      if (httpRequest.status === 200) {
        let msgs = document.getElementById('items');
        msgs.innerHTML = httpRequest.responseText;
        msgs.scrollTop = msgs.scrollHeight;
      } else { alert('Response failure'); }
    }
  }

  let url = '/items/' + listName;
  console.log(url)
  httpRequest.open('GET', url);
  httpRequest.send();
}
```

Question 3: AJAX

Rewrite this code using fetch for the AJAX

```
function getItemsForList(listName) {
  let p = fetch('/items/' + listName, {method:'GET'});
  p.then((result) => {
     return result.text()
  }).then((result) => {
    let msgs = document.getElementById('items');
    msgs.innerHTML = httpRequest.responseText;
    msgs.scrollTop = msgs.scrollHeight;
  }).catch( (error) => {
    alert('Response failure');
  });
}
```

Question 4: Forms

- Write the HTML form code that has the following functionality:
  - Has a username field, password (with security dots), and an email (that ensures a properly formatted email), and a favorite color
  - Has labels for each field
  - Sends the form data via POST to **`http://localhost:80/post/data`**

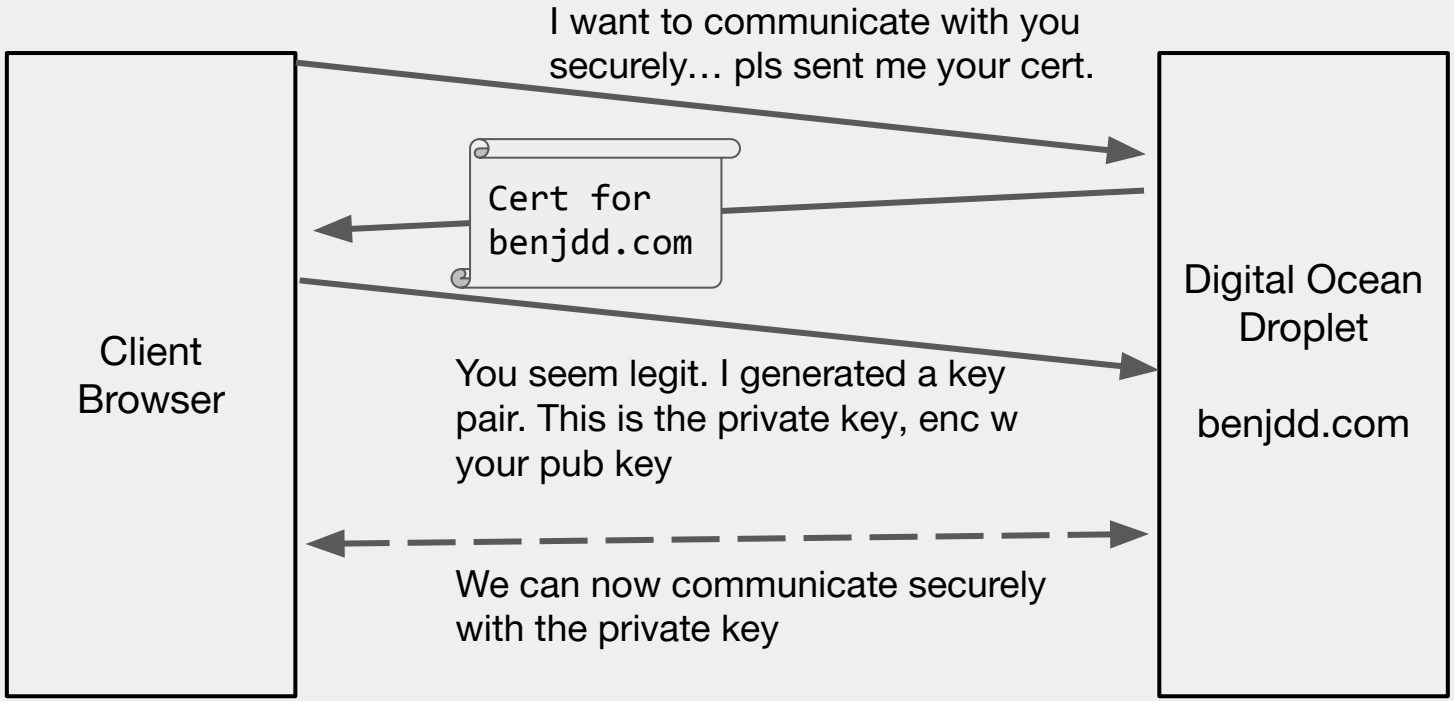
Question 4: Forms

```
<form method="POST" action="http://localhost:80/post/data" >
    <label for="username">Username:</label>
    <input type="text" name="username" id="username" />

    <label for="password">Password:</label>
    <input type="password" name="password" id="password" />

    <label for="email">Email:</label>
    <input type="email" name="email" id="email" />

    <label for="color">Color:</label>
    <input type="color" name="color" id="color" />

    <input type="submit" value="submit" />
</form>
```

Question 5: HTTPS

- When a client wants to communicate with a web server securely via HTTPS, there is a process at the beginning of the communication for establishing trust between the client and the server. Draw a diagram that shows the steps of communication leading up the the client and the server being able to communicate with each-other.

Question 5: HTTPS



Question 6: Web Application Vulnerabilities

- In class, we talked about various web application vulnerabilities / attacks. In a couple of sentences, please describe:
  - Cross Site Scripting (XSS)
  - Database Injection
  - Denial of Service

Question 6: Web Application Vulnerabilities

- Cross Site Scripting (XSS)
  - Inject malicious JavaScript code into a website
  - Used by malicious users to gain access to information
- Database Injection
  - When data from client input (form) is sent to a server and “injected” into a database query
  - Risk of giving a user information that they should not have
- Denial of Service
  - Overwhelm a web server, API, etc with high volume of requests
  - Externally force service degradation

Question 7: Uniform Resource Locator

- Define each the following parts of a URL:
  - Scheme (Protocol)
  - Domain
  - Path

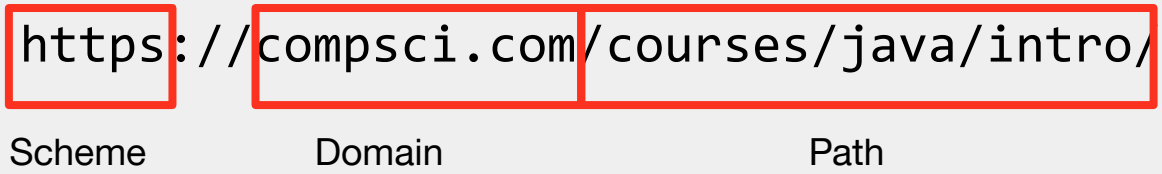
- Label the Scheme, Domain, and Path in the following URL:

  `https://compsci.com/courses/java/intro/`

Question 7: Uniform Resource Locator

- Scheme (Protocol): Used to communicate between the browser, and the source of the page
- Domain: Used to identify the location to get the resource from
- Path: The path to the specific resource or file within the specified domain

- Label the Scheme, Domain, and Path in the following URL:



Any other questions or comments?