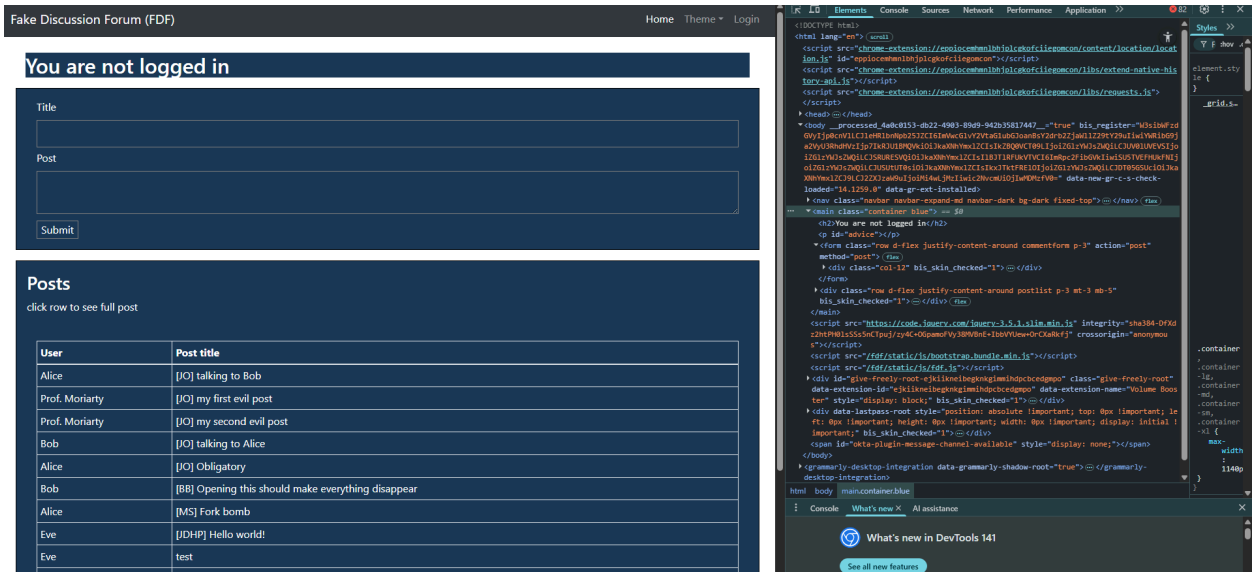
Cookies and Cross-site scripting (XSS) - Devin Dennis

Part 1: Cookies

a. There are cookies for cs338.jeffondich.com.
   i. Cookie Name: theme
   ii. Cookie Value: default
b. Yes, the value changed to red
c. I see the Cookie header set to the color I changed it to (red). Yes, I see the same cookie values
d. Yes
e. Through an HTTP GET request.
f. Through another HTTP GET request with an HTTP request header containing the changed cookie value.
g. You can modify the HTML by changing the class container for the page:



h. You can modify the HTTP request cookie header before it's sent to the server.
i. Burp Suite stores them in an internal cache called a cookie jar.

Part 2: Cross-site scripting.

a. There are 3 types of XSS attacks: Reflected XSS, Stored XSS, and DOM-Based XSS. Link
b. The first Moriarty attack is DOM-based. It's comprised of embedding an HTML tag into the post itself to change the color of the text. When a user views the post, it will be red. Moritarty's second attack uses the HTML script tag to embed JavaScript. Doing so allows the attacker to run malicious code on their client.
c. There's a post on the page that grabs the cookie data from the client's session and forwards it to the attacker's email address. This can be used maliciously to steal data from users of a site.

d. There's a post on the page that redirects the user to a new page. An attacker can use this redirected page to steal information from the user by getting IP's, making a spoofed site where they need to log in again to steal login credentials, or make them purchase something they think they're buying.
e. There are two ways I think you can prevent this.
    i. Prevent your page from executing injected HTML/JavaScript into your website. This prevents the malicious code from running.
    ii. On the server side, scrape all possible data from posts that contain code/HTML. Then the code will not be saved on the server and cannot be sent to the user.