Hussein Ismael Ibrahim

Discussion 5

Web-Programming lab

1- **What are the differences between GET & POST methods?**
SOL/

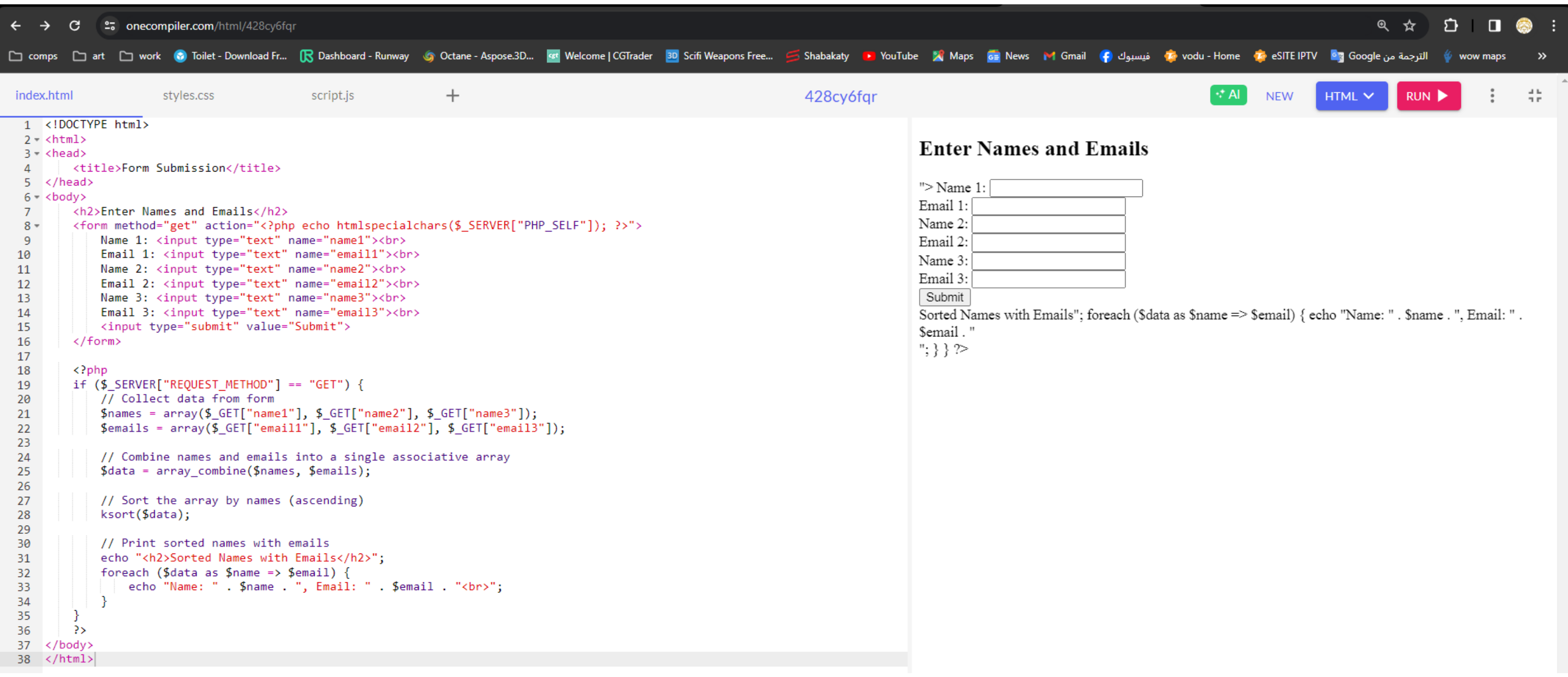
Feature	GET Method	POST Method
Purpose	Retrieving data from a server	Sending data to a server to create, update, or delete resources
Data Location	Appended to the URL as key-value pairs (query string)	Included in the HTTP request body
Security	Less secure (data visible in URL)	More secure (data hidden from URL)
Caching	Can be cached by browser	Not typically cached
Bookmarks	Can be bookmarked with data intact	Bookmarks don't include request body data
Browser History	Stored in browser history	Not typically stored in browser history
Data Size Limit	Limited by URL length (usually 2048 characters)	No inherent limit (can handle large amounts of data)

2- Write a PHP script to create a form for entering three names and their

emails and then print the names with the emails on the web page as sorting

ascending. (Use GET method in your form).

SOL/



onecompiler.com/html/428cy6fqr

index.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Form Submission</title>
</head>
<body>
    <h2>Enter Names and Emails</h2>
    <form method="get" action="<?php echo htmlspecialchars($_SERVER["PHP_SELF"]); ?>">
        Name 1: <input type="text" name="name1"><br>
        Email 1: <input type="text" name="email1"><br>
        Name 2: <input type="text" name="name2"><br>
        Email 2: <input type="text" name="email2"><br>
        Name 3: <input type="text" name="name3"><br>
        Email 3: <input type="text" name="email3"><br>
        <input type="submit" value="Submit">
    </form>

    <?php
    if ($_SERVER["REQUEST_METHOD"] == "GET") {
        // Collect data from form
        $names = array($_GET["name1"], $_GET["name2"], $_GET["name3"]);
        $emails = array($_GET["email1"], $_GET["email2"], $_GET["email3"]);

        // Combine names and emails into a single associative array
        $data = array_combine($names, $emails);

        // Sort the array by names (ascending)
        ksort($data);

        // Print sorted names with emails
        echo "<h2>Sorted Names with Emails</h2>";
        foreach ($data as $name => $email) {
            echo "Name: " . $name . ", Email: " . $email . "<br>";
        }
    }
    ?>
</body>
</html>
```

```
Enter Names and Emails

"> Name 1:
Email 1:
Name 2:
Email 2:
Name 3:
Email 3:
Submit
Sorted Names with Emails"; foreach ($data as $name => $email) { echo "Name: " . $name . ", Email: " .
$email . "
"; } } ?>
```

3- There are many other methods besides those mentioned in this lecture that could be used to avoid hacking operations. Mention some of them and give an example of one of them.

SOL/

- **Input Validation & Sanitization: Scrutinize user input (e.g., format, special characters) before use. Remove potentially harmful elements to prevent attacks like SQL injection or XSS.**
- **Prepared Statements: Secure way to interact with databases. Separates SQL code from user input, preventing SQL injection.**
- **User Authentication & Authorization: Verify user identity (authentication) before granting access (authorization). Use strong password hashing and secure session management.**
- **Regular Security Updates: Keep web server software, PHP, and libraries updated with the latest security patches to address vulnerabilities.**
- **CSRF Protection: Mitigate Cross-Site Request Forgery attacks (unauthorized actions) by using CSRF tokens to validate requests.**