

Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Mahitha Kalaga

Email: kalagam1@udayton.edu



Repository Information

Repository's URL: <https://github.com/MahithaKalaga-cyber/waph-mahitha.git>

This is a private repository for Mahitha Kalaga to store all the code from the course. The organization of this repository is as follows.

Labs

Hands-on exercises in Lectures

- Lab 0: Development Environment Setup
- Lab 1: Foundations of the Web
- Lab 2: Front-end Web Development

Report

The lab's overview

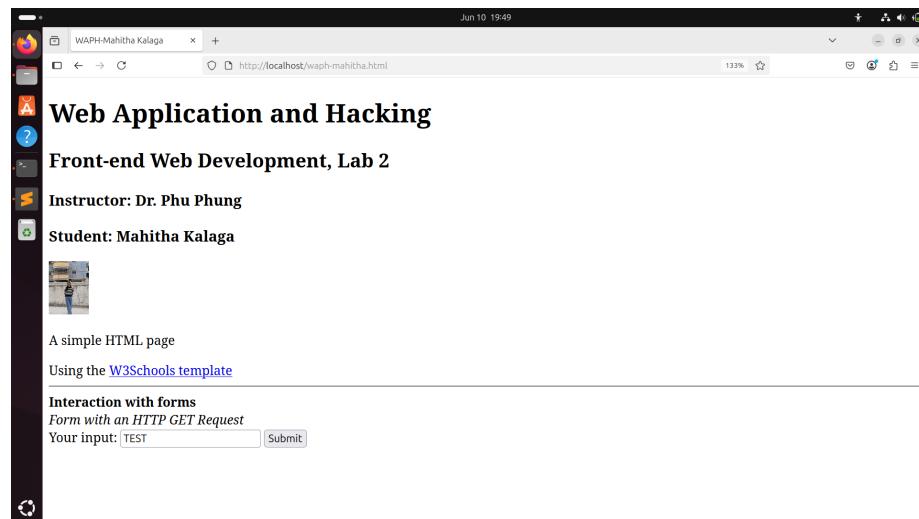
This lab focused on core front-end web development techniques using HTML, CSS, JavaScript, Ajax, jQuery, and Web API integration. The tasks reinforced lecture materials from weeks 4 to 6 and were implemented using a local Apache server. Through this lab, I gained experience in dynamic page updates, asynchronous requests, real-time clocks, and integrating third-party APIs into web interfaces.

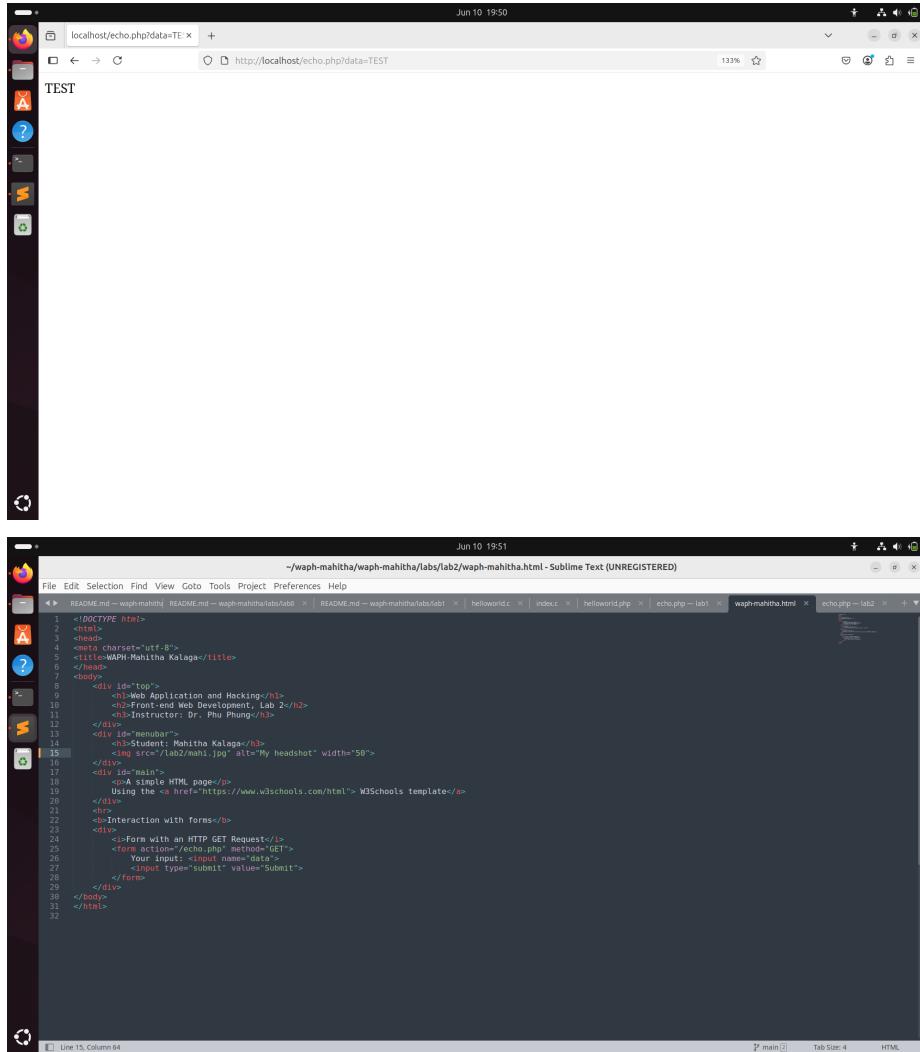
Lab's URL: Lab2

Part 1 - Basic HTML with Forms, and JavaScript

Task 1.A: A Basic HTML

Created waph-mahitha.html with proper HTML tags. Included:
- Headshot image (150x150 pixels)
- A simple form with and tags





Task 1.B.i: Inline JavaScript

For the inline JavaScript task, I added an onclick event to a button that, when clicked, displays the current date and time. I also used the onkeypress attribute in a text input to log what keys were pressed to the browser console.

Web Application and Hacking

Front-end Web Development, Lab 2

Instructor: Dr. Phu Phung

Student: Mahitha Kalaga

Current Time: Tue Jun 10 2025 20:07:18 GMT-0400 (Eastern Daylight Time)

A simple HTML page

Using the [W3Schools template](https://www.w3schools.com/html)

Experiments with JavaScript code

Inline JavaScript

[Click Here to Show Date\(\)](#)

Interaction with forms

Form with an HTTP GET Request

Your input:


```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>WAPH-Mahitha Kalaga</title>
6 </head>
7 <body>
8   <div id="title">
9     <h1>Web Application and Hacking</h1>
10    <h2>Front-end Web Development, Lab 2</h2>
11    <div>
12      <h3>Instructor: Dr. Phu Phung</h3>
13      <div id="menubar">
14        <a href="#">Home</a> | <a href="#">Mahitha Kalaga</a> | <a href="#">Logout</a>
15        
16      </div>
17      <div id="digit-clock"></div>
18      <script>
19        function displayTime(){
20          document.getElementById('digit-clock').innerHTML = "Current Time: " + new Date();
21        }
22        setInterval(displayTime, 500);
23      </script>
24    </div>
25    <div id="main">
26      <p>A simple HTML page</p>
27      <p>Using the <a href="https://www.w3schools.com/html"> W3Schools template</a></p>
28    </div>
29    <div>
30      <b>Experiments with JavaScript code</b><br>
31      <i>Inline JavaScript</i>
32      <div id="date" onclick="document.getElementById('date').innerHTML=Date()"><a href="#">Click Here to Show Date()

```

Task 1.B.ii: Digital Clock

Using setInterval() and JavaScript's Date() object, I created a live digital clock that updates every second. It displays the time in HH:MM,SS format. This was written inside a script tag and directly manipulated the inner content of a div.

The image consists of two vertically stacked screenshots of a Firefox browser window. Both screenshots show the same content: a web page titled "Web Application and Hacking" with sub-sections "Front-end Web Development, Lab 2", "Instructor: Dr. Phu Phung", and "Student: Mahitha Kalaga". Below this is a small thumbnail image of a person standing in front of a building. The page also contains a link to "W3Schools template", sections for "Experiments with JavaScript code" (including "Inlined JavaScript" and a button labeled "Click Here to Show Date()"), and a form for "Interaction with forms" with a "Form with an HTTP GET Request" section and a "Your input: TEST" field with a "Submit" button.

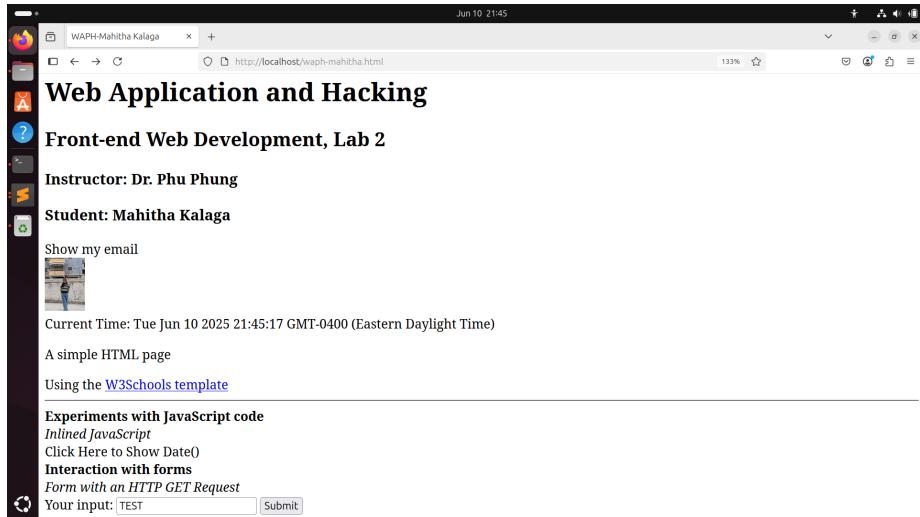
```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>WAPH-Mahitha Kalaga</title>
6   </head>
7   <body>
8     <div id="top">
9       <h1>Web Application and Hacking</h1>
10      <h2>Front-end Web Development, Lab 2</h2>
11      <h3>Instructor: Dr. Phu Phung</h3>
12    </div>
13    <div id="menubar">
14      <h3>Student: Mahitha Kalaga</h3>
15      
16    </div>
17    <div id="main">
18      <p>Using the <a href="https://www.w3schools.com/html"> W3Schools template</a></p>
19    </div>
20    <div>
21      <h4>Experiments with JavaScript code</h4>
22      <h4>Inlined JavaScript</h4>
23      <div><input type="button" value="Click Here to Show Date()" onclick="document.getElementById('date').innerHTML=Date()"/></div>
24      <h4>Interaction with forms</h4>
25      <div>
26        <form action="/echo.php" method="GET">
27          Your input: <input name="data">
28          <input type="submit" value="Submit">
29        </form>
30      </div>
31    </div>
32  </body>
33</html>
34
35

```

Task 1.B.iii: Show/Hide Email

Created a reusable and modular JavaScript file (email.js) that dynamically toggles the visibility of my email address. When the user clicks a div, the content switches between a label and a mailto: hyperlink. This component demonstrates external JS integration, conditional logic, and DOM element replacement.



Web Application and Hacking

Front-end Web Development, Lab 2

Instructor: Dr. Phu Phung

Student: Mahitha Kalaga

kalagam1@udayton.edu

Current Time: Tue Jun 10 2025 21:46:13 GMT-0400 (Eastern Daylight Time)

A simple HTML page

Using the [W3Schools template](https://www.w3schools.com/html/)

Experiments with JavaScript code

Inlined JavaScript

[Click Here to Show Date\(\)](#)

Interaction with forms

[Form with an HTTP GET Request](#)

Jun 10 21:47

~/waph-mahitha/waph-mahitha/labs/lab2/waph-mahitha.html - Sublime Text (UNREGISTERED)

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8">
5     <title>WAPH-Mahitha Kalaga</title>
6   </head>
7   <body>
8     <h1>EdTech</h1>
9     <h2>Web Application and Hacking</h2>
10    <h3>Instructor: Dr. Phu Phung</h3>
11    <div id="main">
12      <p>Student: Mahitha Kalaga</p>
13      <a href="#" onclick="showHideEmail()>Show my email</a>
14      <script src="lab2/email.js"></script>
15      
16      <div><Digital Clock></div>
17      <script>
18        function displayTime(){
19          document.getElementById('digital-clock').innerHTML = "Current Time: " + new Date();
20        }
21        setInterval(displayTime, 500);
22      </script>
23    </div>
24  </div>
25  <div id="main">
26    <p>A simple HTML page</p>
27    <p>Using the <a href="https://www.w3schools.com/html/"> W3Schools template</a></p>
28  </div>
29  <div>
30    <b>Experiments with JavaScript code</b><br>
31    <b>Inlined JavaScript</b><br>
32    <script>document.getElementById('date').innerHTML=Date()>Click Here to Sh
33    <b>Interaction with forms</b><br>
34  </div>
35  <div>
36    <form action="/echo.php" method="GET">
37      Your input: <input name="data">
38      <input type="submit" value="Submit">
39    </form>
40  </div>
41 </div>
42 </body>
43 </html>
44

```

~/waph-mahitha/waph-mahitha/labs/lab2/email.js - Sublime Text (UNREG... - Sublime Text (UNREGISTERED)

```

1 var shown = false;
2 function showHideEmail(){
3   if(shown){
4     document.getElementById('email').innerHTML = "Show my Email";
5     shown = false;
6   } else{
7     var myemail= "<a href='mailto:kalagam1' + \"\" + \"udayton.e
8     document.getElementById('email').outerHTML = myemail;
9     shown = true;
10   }
11 }
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44

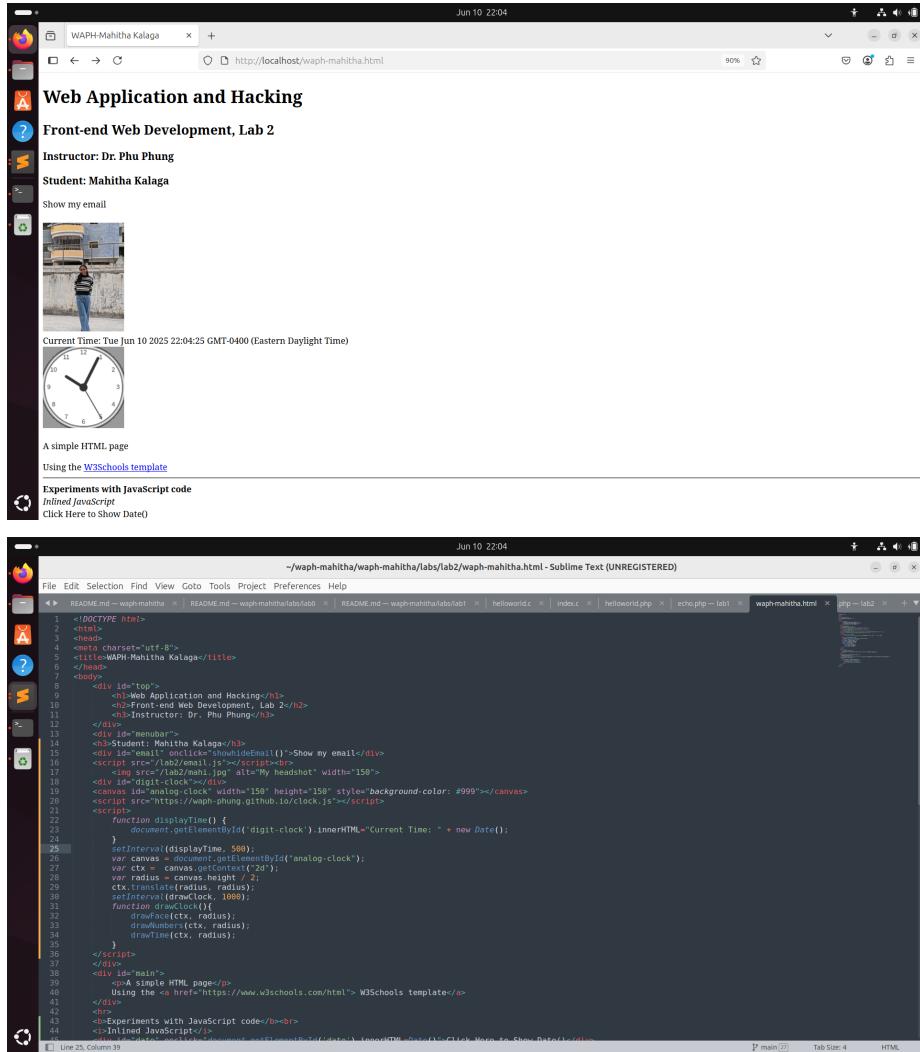
```

Line 13, Column 2. main [] Tab Size: 4. JavaScript.

Task 1.B.iv: Analog Clock

Added an analog clock using a

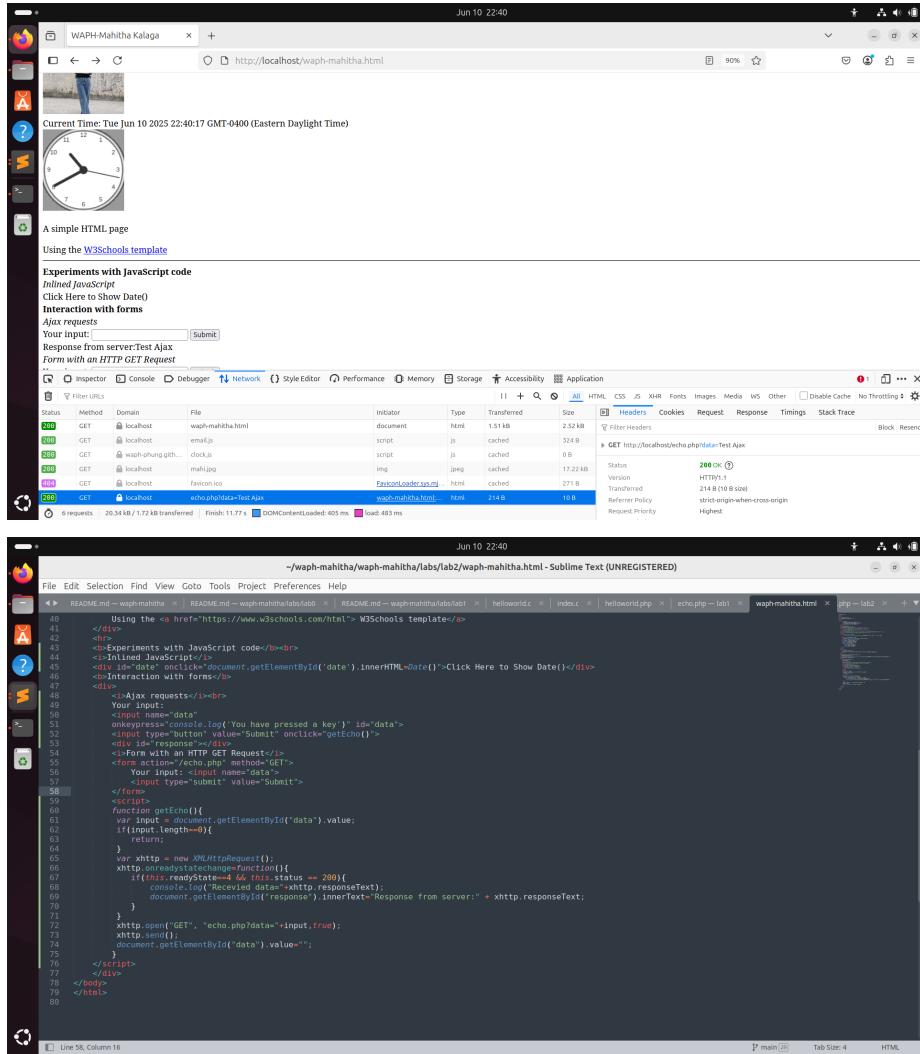
element and the external script hosted at <https://waph-phung.github.io/clock.js>. The script draws clock hands in real time using canvas rendering.



Task 2: Ajax, CSS, jQuery, and Web API Integration

Task 2.A: Ajax

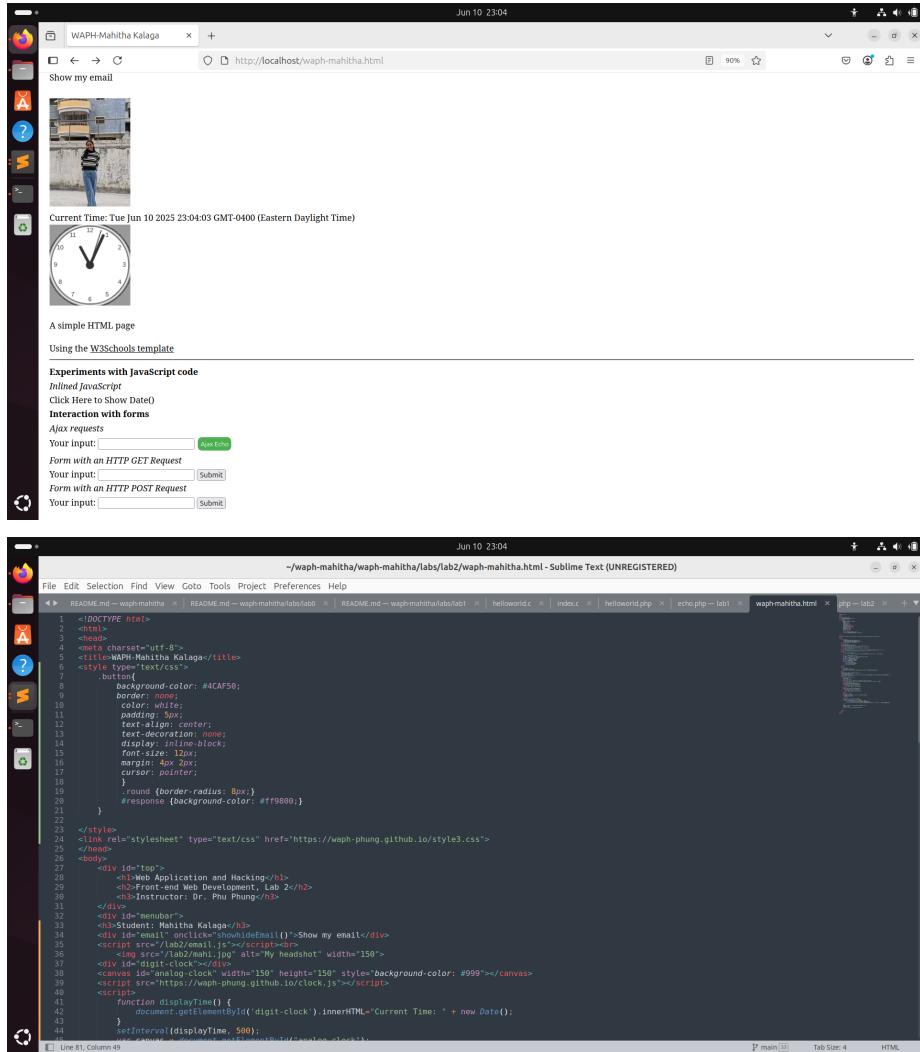
Implemented a form-driven Ajax request using XMLHttpRequest to send user input to echo.php. The servers response is retrieved and rendered within a target div element. By monitoring the request through browser developer tools, I gained insight into asynchronous communication and HTTP request/response lifecycles.



Task 2.B: CSS

Demonstrated an understanding of different CSS application methods:

- In-line CSS was used directly within HTML tags for quick styling.
- Internal CSS was defined within a style block in the head for layout consistency.
- External CSS was applied by linking to a remote stylesheet (<https://waph-phung.github.io/style3.css>).



Task 2.C: jQuery

Included the jQuery library via CDN and developed Ajax functions using both \$get() and \$post() methods to interact with the echo.php endpoint. The responses were dynamically injected into the DOM.

Jun 10 23:15

A simple HTML page
Using the [W3Schools](#) template

Experiments with JavaScript code

Initial JavaScript
Click here to Show Date()

Interaction with forms

Ajax requests

Your input:

Response from server:jQuery Ajax GET Testing

Form with an HTTP GET Request
Your input:

Form with an HTTP POST Request
Your input:

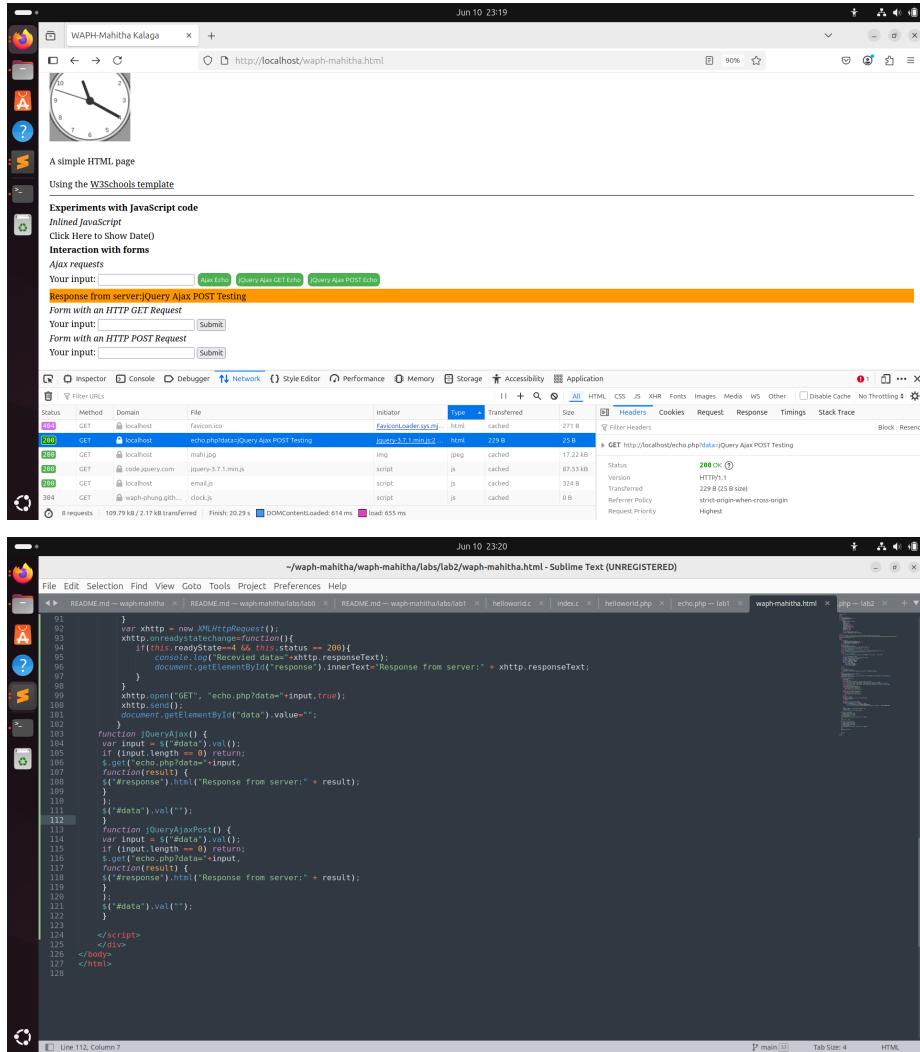
Jun 10 23:16

-waph-mahitha/waph-mahitha/labs/lab2/waph-mahitha.html - Sublime Text (UNREGISTERED)

```

67     <div>
68       <h3>Ajax requests</h3><br>
69       Your Input:
70       <input name="data">
71       <input type="button" value="Echo" onclick="getEcho()">
72       <input type="button" value="Ajax Echo" onclick="getAjaxEcho()">
73     </div>
74   </div>
75   <form with an HTTP GET Request/>
76   <form action="echo.php" method="GET">
77     Your input: <input name="data" type="text" value="" />
78     <input type="submit" value="Submit" />
79   </form>
80   <form with an HTTP POST Request/>
81   <form action="echo.php" method="POST">
82     Your input: <input name="data" type="text" value="" />
83     <input type="submit" value="Submit" />
84   </form>
85   <script>
86     function getEcho(){
87       var input = document.getElementById("data");
88       if(input.length==0) return;
89     }
90     var xhttp = new XMLHttpRequest();
91     xhttp.onreadystatechange=function(){
92       if(this.readyState==4&&this.status == 200){
93         console.log(Received data=>xhttp.responseText);
94         document.getElementById("response").innerText="Response From server:" + xhttp.responseText;
95       }
96     }
97     xhttp.open("GET", "echo.php?data=" + input, true);
98     xhttp.send();
99     document.getElementById("data").value="";
100   }
101   function getAjaxEcho(){
102     var input = $("#data").val();
103     if (input.length == 0) return;
104     $.post("echo.php", {data:input}, function(result) {
105       $("#result").html(result);
106       $("#response").html("Response from server:" + result);
107     });
108   }
109   $("#data").val("");
110 
```

Line 78, Column 49



Task 2.D: Web API Integration

- i. Joke API

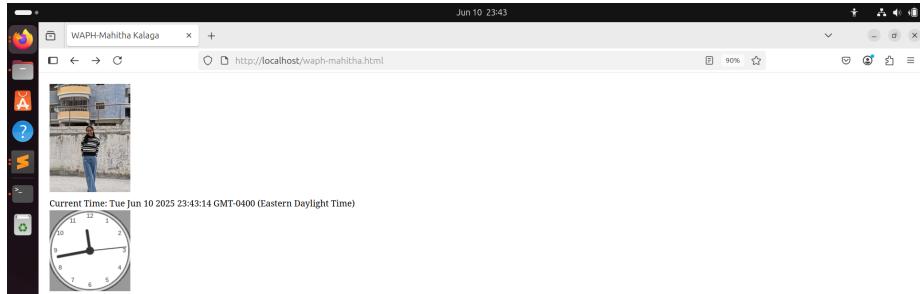
Used jQuery to fetch a random programming joke from <https://v2.jokeapi.dev/joke/Programming?type=single> on page load. The returned JSON was parsed, and the joke was displayed in a div element. This integration demonstrated the practical use of external APIs to enhance user engagement.

The screenshot shows a Firefox browser window with the title "WAPH-Mahitha Kalaga". The URL is "http://localhost/waph-mahitha.html". The page content includes a small image of a person, the text "Current Time: Tue Jun 10 2025 23:28:39 GMT-0400 (Eastern Daylight Time)", and a digital clock. Below this, there is a section titled "A simple HTML page" and "Using the W3Schools template". It contains a heading "Experiments with JavaScript code" and a sub-section "Inline JavaScript". It features a "Click Here to Show Date!" button and a "Clock" example. A yellow banner at the bottom says "A Programming Joke Of The Day: Java is like Alzheimer's, it starts off slow, but eventually, your memory is gone." There are two form sections for "HTTP GET Request" and "HTTP POST Request".

The screenshot shows a Sublime Text editor window with the title "-/waph-mahitha/waph-mahitha/labs/lab2/waph-mahitha.html - Sublime Text (UNREGISTERED)". The file content is a JavaScript file with several functions. One function, "joke", uses jQuery to get the value of an input field, sends an AJAX request to "jokeapi.dev/joke/Programming?type=single", and then logs the response to the console. Another part of the code handles the response by updating a "#response" element with the joke text.

- ii. Agify API with fetch()

Used JavaScript's modern `fetch()` API to retrieve age prediction data from <https://api.agify.io/?name=...> based on user input. The results were processed asynchronously and rendered within the page, providing a hands-on example of modern JavaScript promises and external API interaction.



A simple HTML page

Using the W3Schools template

Experiments with JavaScript code

Inlined JavaScript

Click Here to Show Date!

Interaction with forms

Ajax requests

Your input: [Ajax Echo](#) [Query Ajax GET Echo](#) [Query Ajax POST Echo](#) [Guess Age](#)

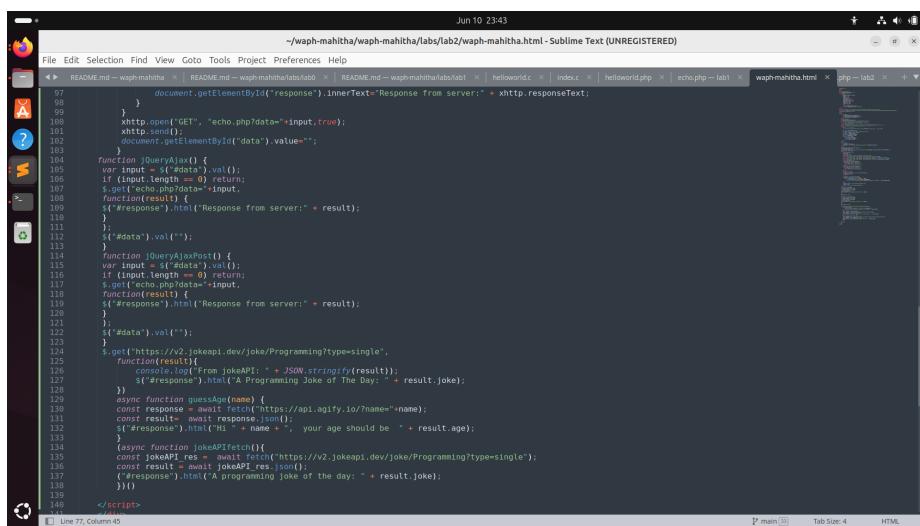
Hi mahitha, your age should be 28

Form with an HTTP GET Request

Your input: [Submit](#)

Form with an HTTP POST Request

Your input: [Submit](#)



```

97     document.getElementById('response').innerText="Response from server:" + xhttp.responseText;
98 }
99 xhttp.open("GET", "echo.php?data=" + input, true);
100 xhttp.send();
101 document.getElementById("data").value="";
102 }
103 function guessAge() {
104     var input = $('#data').val();
105     if (input.length == 0) return;
106     xhttp.open("POST", "echo.php?data=" + input,
107     function(result) {
108         $('#response').html("Response from server:" + result);
109     });
110     $('#data').val("");
111 }
112
113 function jqueryAjaxPost() {
114     var input = $('#data').val();
115     if (input.length == 0) return;
116     xhttp.open("POST", "echo.php?data=" + input,
117     function(result) {
118         $('#response').html("Response from server:" + result);
119     });
120     $('#data').val("");
121 }
122
123
124 $.get("https://v2.jokeapi.dev/joke/Programming?type=single",
125     function(result){
126         const jokeAPI = JSON.stringify(result);
127         $('#response').html("A Programming Joke of The Day: " + result.joke);
128     })
129     async function guessAge(name) {
130         const response = await fetch(`https://api.agify.io/?name=${name}`);
131         const result = await response.json();
132         $('#response').html(`Hi ${name} ! Your age should be ${result.age}`);
133     }
134     async function jokeAPIFetch() {
135         const jokeAPI = await fetch("https://v2.jokeapi.dev/joke/Programming?type=single");
136         const result = await jokeAPI.json();
137         $('#response').html(`A programming joke of the day: ${result.joke}`);
138     }()
139
140 </script>

```

Line 77, Column 45