

waph-atmakugh

WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Atmakuri Ganesh

Email: atmakugh@mail.uc.edu

Short-bio: A masters student with communication,organizational, and technical skills seeking opportunities. A hand-working and motivated engineering student with authentic skills in user application development and design thinking,dedicated to leveraging my abilities as a capable and diligent student



Figure 1: Ganesh headshot

Lab Overview

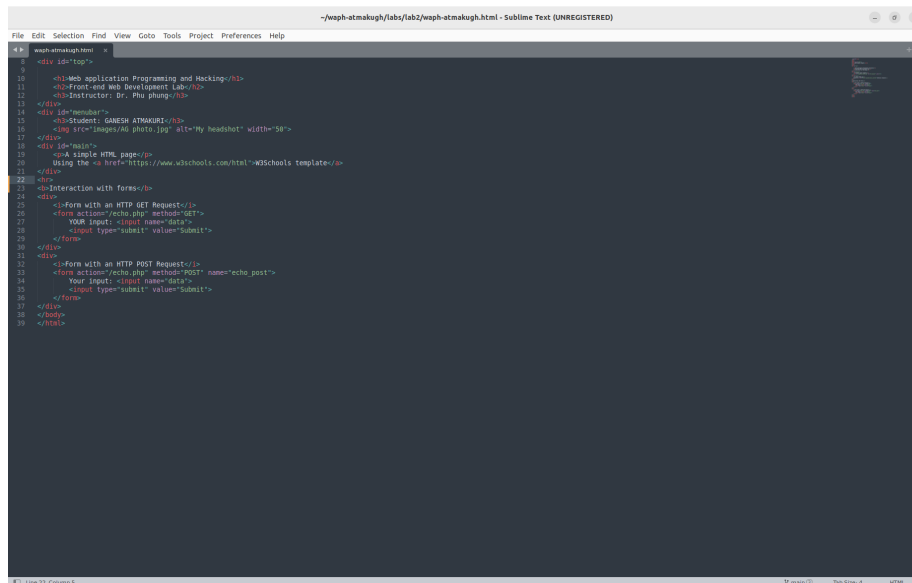
- This lab covers Frontend web development
- Task 1 primarily focuses on developing simple html web page with basic tags
- This lab also covers using echo.php to handle GET and POST requests
- Task 1 also covers using inlined and external javascript
- Task 2 covers about the ajax, CSS, jquery, and web api integration
- Types of CSS is also covered in CSS
- Fetch(), JSON(), async, await functions are seen in the Task 2

Repository Information

Repository's URL: <https://github.com/ATMAKURIGANESH3009/waph-atmakugh/tree/main/labs/lab2>

Task 1 - Basic HTML with forms and Javascript

- **A.HTML**
- In this task, I have developed a basic html code with basic tags and forms
- After creating a lab2 folder. I have created a waph-atmakugh.html and written html code inside it.
- It contains course name, instructor id, student details, student headset using h and img tag
- Next, I have developed a form with HTTP get request
- In this, I have used echo.php file which was generated in the last lab for the request purpose
- Similarly, I have created a code for POST request using form tag
- Code for this task (fig 2) :



```
1 <div id="header">
2
3     <h1>Web application Programming and Hacking</h1>
4     <h2>Front-end web Development Lab</h2>
5     <h3>Instructor: Dr. Ph. phung</h3>
6 </div>
7
8 <div id="student">
9     <h3>Student: GANESH ATMAKUR</h3>
10    
11 </div>
12
13 <div id="main">
14    <p>A simple HTML page</p>
15    Using the <a href="https://www.w3schools.com/html/w3schools_template/">
16
17 </div>
18
19 <div>
20    <p>Interaction with forms</p>
21 </div>
22
23 <div>
24    <p>Form with an HTTP GET Request</p>
25    <form action="echo.php" method="GET">
26        Your input: <input name="data">
27        <input type="submit" value="Submit">
28    </form>
29 </div>
30
31 <div>
32    <p>Form with an HTTP POST Request</p>
33    <form action="echo.php" method="POST" name="echo_post">
34        Your input: <input name="data">
35        <input type="submit" value="Submit">
36    </form>
37 </div>
38 </body>
39 </html>
```

Figure 2: Basic HTML

- The output of this task looks as here (fig 3):
- **B.Simple Javascript**
- After the forms, I have performed an inline javascript exercise. If I click on show the date it will display the current date
- In a div tag onclick="document.getElementById('date').innerHTML=Date()" includes a functionality to display the date. (fig 4)

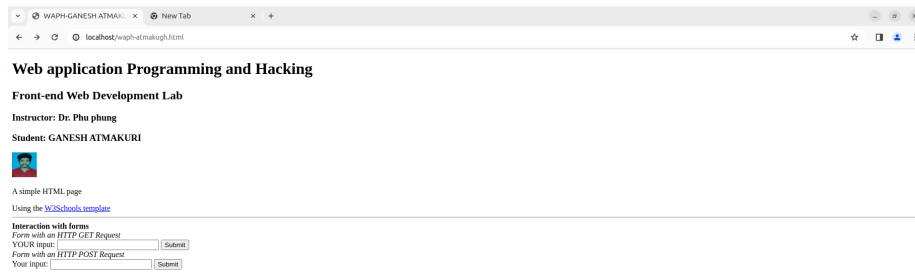


Figure 3: HTML web page with forms

```
<hr>
<b>Experiments with JavaScript code</b><br>
<i>Inlined Javascript</i>
<div id="date" onclick="document.getElementById('date').innerHTML=Date()">Click here to show Date()</div>
```

Figure 4: Inline Javascript Date()

- Next task is to creat digital clock as well as analog clock
- I developed a function to display clock and I have set the interval to change the time every 500ms (fig 5)

```

10 <!--Web application Programming and Hacking-->
11 <!--Front-end Web Development Lab-->
12 <!--Instructor: Dr. Phu phung-->
13 </div>
14 <div id="navbar">
15 <div id="student">GANESH ATMAKURI</div>
16 <div id="email"><button onclick="showEmail()">Show my email</div>
17 <script src="email.js"></script>
18 
19 </div>
20 <div id="digital-clock"></div>
21 <canvas id="analog-clock" width="150" height="150" style="background-color:#999"></canvas>
22 <script src="https://wepb-uc.github.io/clock.js"></script>
23 </div>
24 function displayTime(){
25     document.getElementById(digital-clock).innerHTML="Current time: " + new Date();
26 }
27 setInterval(displayTime,500);
28

```

Figure 5: Digitalclock

- Next I written a code to show my email id when I clicked on show my email
- For this, I have included javascript code inside a new file “email.js” (fig 6)

```

1 var showEmail = function(){
2     function showEmail(){
3         document.getElementById(email).innerHTML = "Show my email";
4         showEmail = false;
5     }
6     var myEmail = "a href='mailto:atmakuri@uc.edu'>atmakuri@uc.edu";
7     document.getElementById(email).innerHTML=myEmail;
8     showEmail();
9 }
10
11
12
13

```

Figure 6: Email

- Also, I included a external javascript file “clock.js” and included a code inside the same script tag
- I used canvas to draw the clock image and a functions to draw the clock (fig 7)

```

24 function displayTime(){
25     document.getElementById(digital-clock).innerHTML="Current time: " + new Date();
26 }
27 setInterval(displayTime,500);
28
29 var canvas = document.getElementById("analog-clock");
30 var ctx = canvas.getContext("2d");
31 var radius = canvas.height / 2;
32 ctx.translate(radius, radius);
33 radius = radius * 0.5;
34 setInterval(drawClock,1000);
35
36 function drawClock() {
37     drawFace(ctx,radius);
38     drawHands(ctx,radius);
39     drawTime(ctx,radius);
40 }
41

```

Figure 7: Analogclock

- Total combined code for Task 1: (fig 8)
- Output for Task1 (fig 9):

```
File Edit Selection Find View Goto Tools Project Preferences Help
waph-atmakugh.html - Sublime Text (UNREGISTERED)

1 <!DOCTYPE html>
2 <html>
3 <meta charset="utf-8">
4 <title>WAPH-GANESH ATMAKURI</title></head>
5 <body>
6 <div id="top">
7 <div>Web application Programming and Hacking</div>
8 <div>Front-end Web Development Lab</div>
9 <div>Instructor: Dr. Phu phung</div>
10 <div id="member">
11 <div>Student: GANESH ATMAKURI</div>
12 <div id="email" onclick="showEmail()">Show my email</div>
13 <script src="email.js"></script>
14 
15 </div>
16 <div id="digital-clock"></div>
17 <script id="analog-clock" width="150" height="150" style="background-color: #999;"></script>
18 <script src="https://waph-us.github.io/clock.js"></script>
19 </script>
20 function displayTime() {
21   document.getElementById("digital-clock").innerHTML="Current time: " + new Date();
22   setInterval(displayTime, 500);
23 }
24 var canvas = document.getElementById("analog-clock");
25 var ctx = canvas.getContext("2d");
26 var radius = canvas.height / 2;
27 ctx.translate(radius, radius);
28 radius = radius * 0.54;
29 setInterval(drawClock, 1000);
30
31 function drawClock() {
32   drawFace(ctx, radius);
33   drawNumber(ctx, radius);
34   drawTime(ctx, radius);
35 </script>
36 <div id="nav">
37 <div>A simple HTML page</div>
38 Using the <a href="https://www.w3schools.com/html/w3schools_template">w3
39 </div>
40 <div>
41 <div>Interaction with Forms</div>
42 <div>
43 <div>Form with an HTTP GET Request</div>
44 <form action="/echo.php" method="GET">
45   YOUR input: <input name="data">
46   <input type="submit" value="Submit">
47 </form>
48 </div>
49 <div>Form with an HTTP POST Request</div>
50 <form action="/echo.php" method="POST" name="echo_post">
51   Your input: <input name="data">
52   <input type="submit" value="Submit">
53 </form>
54 </div>
55 <div>Experiments with JavaScript code</div>
56 <div>
57 <div>Inlined Javascript</div>
58 <div id="date" onclick="document.getElementById('date').innerHTML=date()">Click here to show Date</div>
59 </div>
60 </body>
61 </html>
62
63 Line 60, Column 5
```

Figure 8: Task1

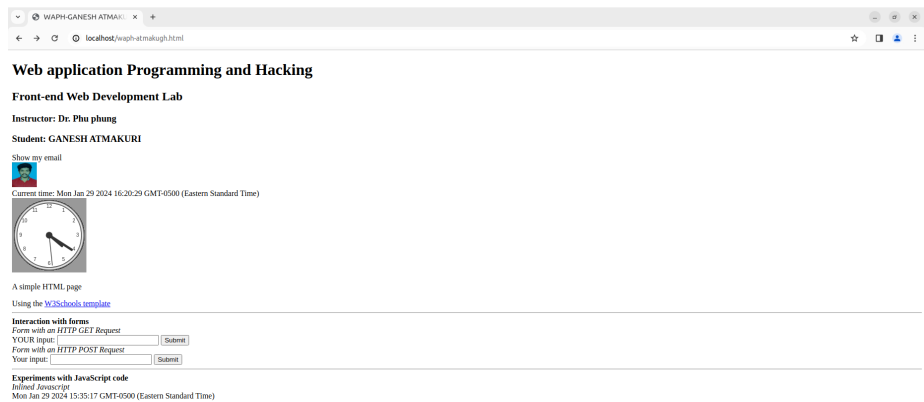
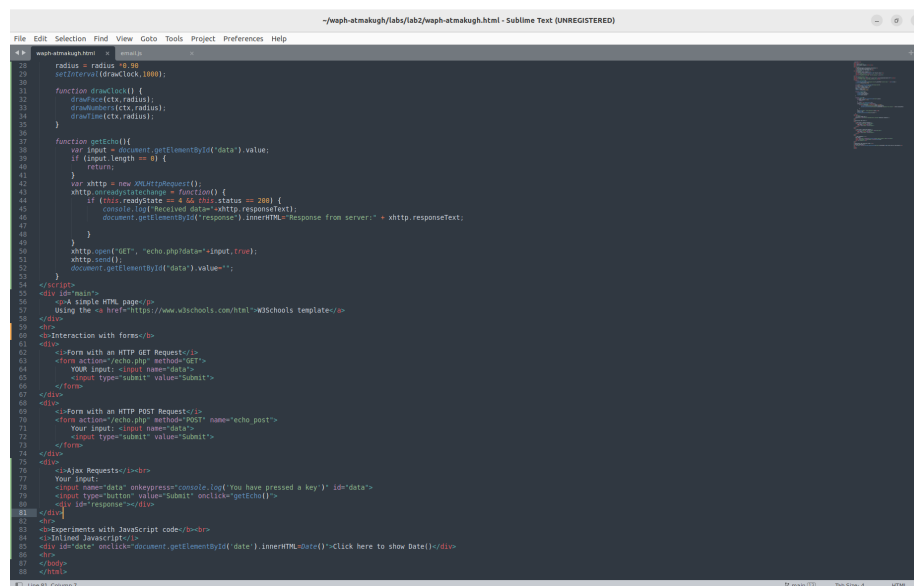


Figure 9: Task1_output

Task 2 - Ajax, CSS, jQuery and web API integration

- **A.Ajax**
- Ajax stands for Asynchronous Javascript and XML
- It is a standard for data to be collected in JS for the web browser to send/receive data from the web without reloading the page
- I added a input tag for taking the user input and a button for submitting the request and a div element to write the JS code. All together after the form.
- I have written a function called getEcho() which takes the input and checks if the length of the input is zero or not to process the request
- Next, a new ajax object is created and onreadystatechange function is set up.
- If the ready state is 4 and status is 200 it will print the response text after handling the request
- A code is written to create an ajax request and sending the request to the server
- echo.php file handles the GET request initialized by xmlhttp.open
- Code for the getecho function (fig 10):



```
File Edit Selection Find View Goto Tools Project Preferences Help
wasp-atmakugh.html - [Untitled]
10 radius = radius + 0.50;
11 setinterval(drawClock, 1000);
12
13 function drawClock() {
14   drawSec(cts, radius);
15   drawMin(cts, radius);
16   drawHr(cts, radius);
17 }
18
19 function getEcho() {
20   var input = document.getElementById('data').value;
21   if (input.length == 0) {
22     return;
23   }
24   var xmlhttp = new XMLHttpRequest();
25   xmlhttp.onreadystatechange = function() {
26     if (this.readyState == 4 && this.status == 200) {
27       console.log('Received data: ' + xmlhttp.responseText);
28       document.getElementById('response').innerHTML = 'Response from server: ' + xmlhttp.responseText;
29     }
30   }
31   xmlhttp.open('GET', 'echo.php?data=' + input, true);
32   xmlhttp.send();
33   document.getElementById('data').value = '';
34 }
35
36 </script>
37
38 <div id="main">
39   <!-- Simple HTML page -->
40   Using the <a href="https://www.w3schools.com/html/w3schools_template">
41
42   <!-- Interaction with forms -->
43   <div>
44     <!-- Form with an HTTP GET Request -->
45     <form action="echo.php" method="GET">
46       <input type="text" name="data">
47       <input type="submit" value="Submit">
48     </form>
49   </div>
50   <div>
51     <!-- Form with an HTTP POST Request -->
52     <form action="echo.php" method="POST" name="echo_post">
53       Your input: <input name="data">
54       <input type="submit" value="Submit">
55     </form>
56   </div>
57   <div>
58     <!-- Ajax Request -->
59     Your input:
60     <input name="data" onkeypress="console.log('You have pressed a key ')" id="data">
61     <input type="button" value="Submit" onclick="getEcho()" id="response">
62   </div>
63   <div>
64     <!-- Experiments with Javascript code -->
65     <!-- Inlined Javascript -->
66     <div id="date" onclick="document.getElementById('date').innerHTML=date()">Click here to show Date</div>
67   </div>
68 </div>
```

Figure 10: getEcho function

- Output of the ajax response (fig 11):
- I have noticed how the ajax request/response showing in the network window
- When I started a new capture, after entering the text in the field and when the request is submitted then I can see there is a response printing in a

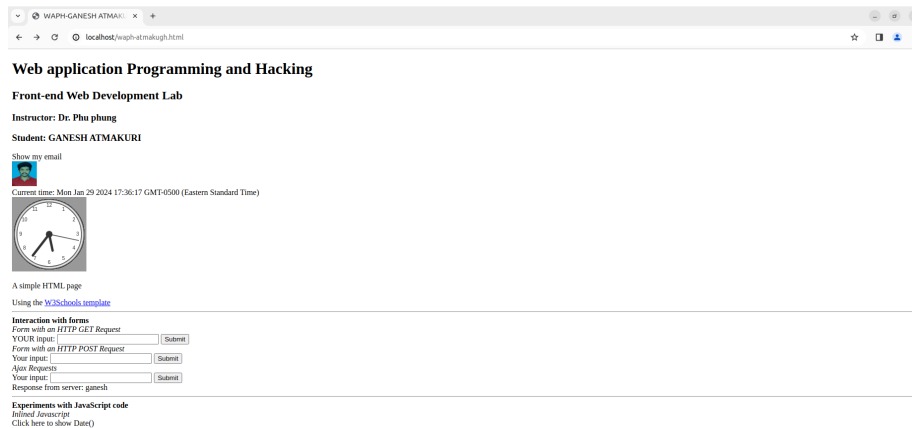


Figure 11: getEcho function output

console window as Response + ourtext

- When I inspect through the echo.php response it shows the status code as 200
- Each time when I ran the request the response message is changing in the console and the number of times the request executed is changing
- We can also see if there are any errors (fig 12)

• B.CSS

- External CSS:
- External CSS is giving a external style sheet in our html page. In this CSS code is written in external css file and the output is rendered for the html page
- I have included one of the remote CSS provided in the class to my page in the head tag
- Next, I made changes to my code accordingly with the div class related to the external CSS file. I arranged different div tags inside a main div tag container wrapper
- Code and rendered output: (fig 13)
- Internal CSS:
- I have added a style tag in the head tag as an internal css
- I edited background color of body to powder blue and h1 tag color to blue (fig 14)
- Next, I removed the code and defined a style for ajax request button in

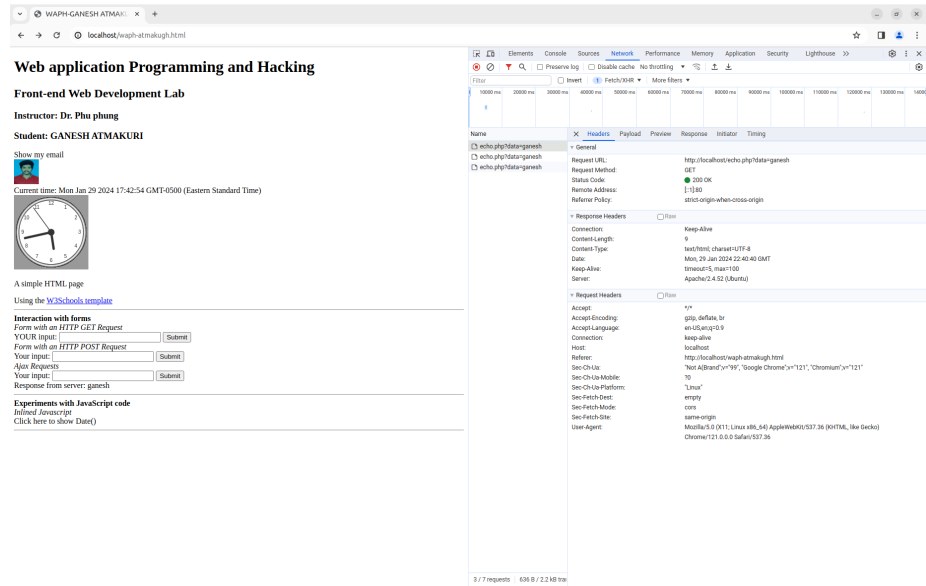


Figure 12: Network Window

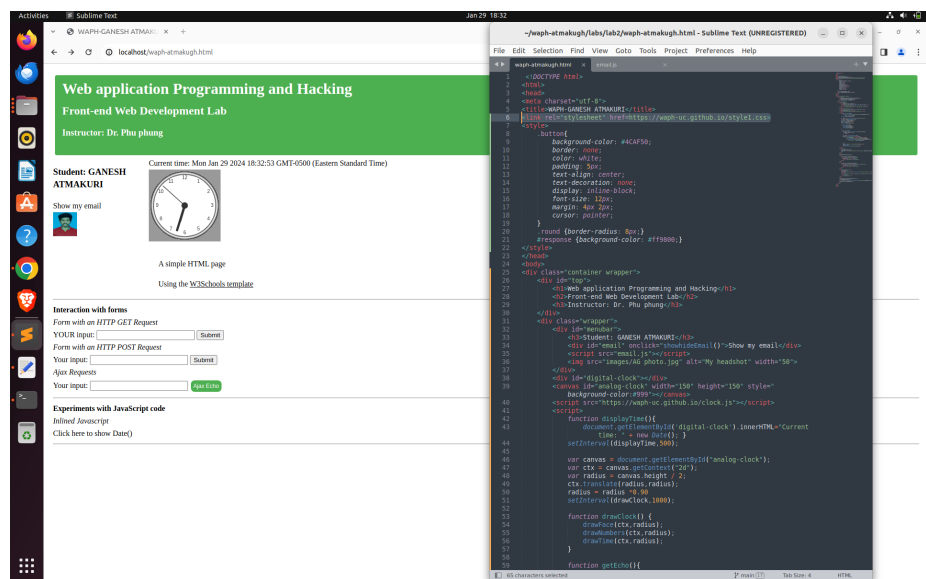


Figure 13: External CSS

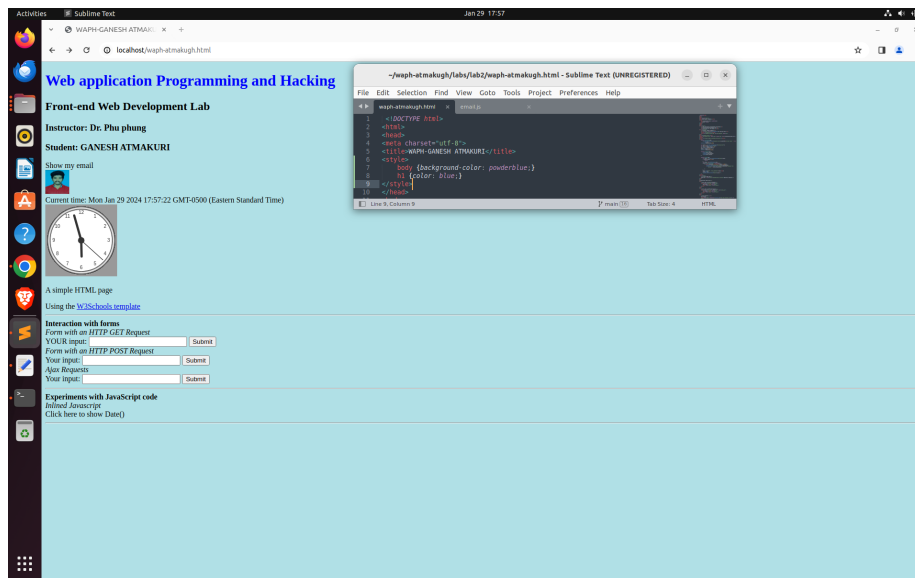


Figure 14: Internal CSS

the head tag as an internal css

- Added the class name to the ajax input button and changed the value from submit to Ajax Echo
- Code and output (fig 15):
- **C.jQuery**
- jQuery is a popular javascript library that provides easy way to access APIs and working on it. Particularly it can simplify the complex tasks.
- I copied the jquery script code into the head section. It is required for the jquery to run
- **i.jQuery \$.get():**
- First a new button is added at the bottom to call the function jQueryAjax() when it is clicked
- A new function jQueryAjax() is created for an ajax get request and it will prints the response back
- Data which we entered is fetched in a variable. A general test is performed whether the data is empty or not by length function
- Next, jQuery selector gets the echo.php file and it reads the input from the container and print back the response by selecting #response id
- Code is as follows (fig 16):

- **ii.jQuery \$.post():**
- A new button is added similarly like above at the bottom to call the function jQueryAjaxPost() when it is clicked
- A new function jQueryAjaxPost() is created for an ajax post request and it will prints the response back
- Data which we entered is fetched in a variable. A general test is performed whether the data is empty or not by length function
- Next, jQuery selector gets the echo.php file and it reads the input from the container and print back the response by selecting #response id
- Code is as follows (fig 17):



Figure 17: jQuery - \$.Post()

- **D. Web API integration**
- We can integrate any free APIs inside our html page using jQuery.
- **i.Ajax on API:** The idea of this is to integrate a joke api by sending a request and to display the response of a random joke
- A ajax request code is written in an old script tag
- \$.get() fetches the api for the response and JSON is used for formatting the response data
- There is no button created for handling the joke. Therefore this request will execute everytime when the page is reloaded.
- code and output (fig 18):
- After refreshing a browser, I have inspected the network window
- Everytime, when a browser is reloaded a random joke is fetched and printed in the console window as API code
- In request windows, status is showing as 200 ok and in the response tab, it is displaying the api code which is fetched (fig 19)

-ii.Using fetch api:

- Guessing the age based on name is an another api I have fetched in this sub task
- I have created a input button guess age to execute an api when the button is clicked

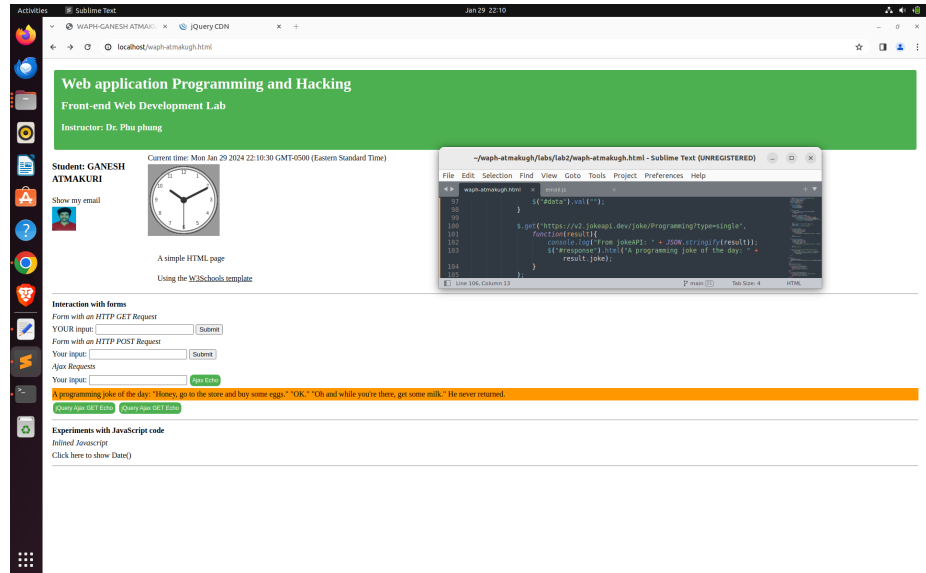


Figure 18: Joke API

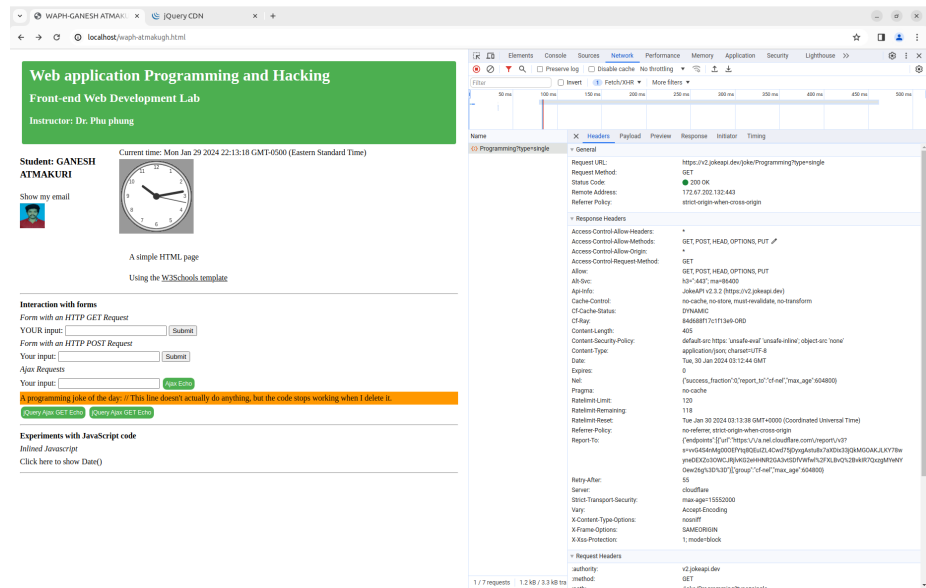
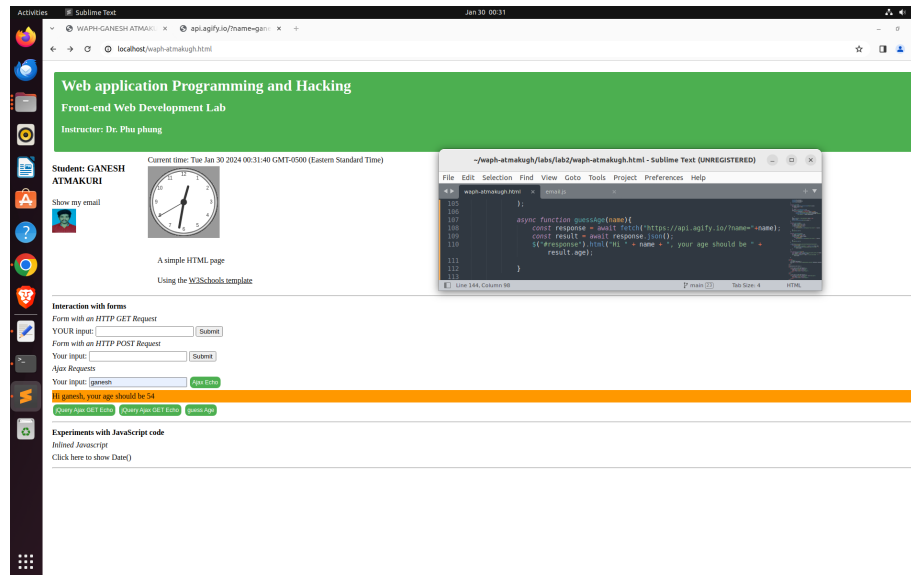


Figure 19: Joke API Network window

- Next I have created a async function guessAge
- I used fetch() which is a javascript method for fetching results across the network
- It will return a promise
- Now the api will respond and code will handle the response
- Code and output: (fig 20)



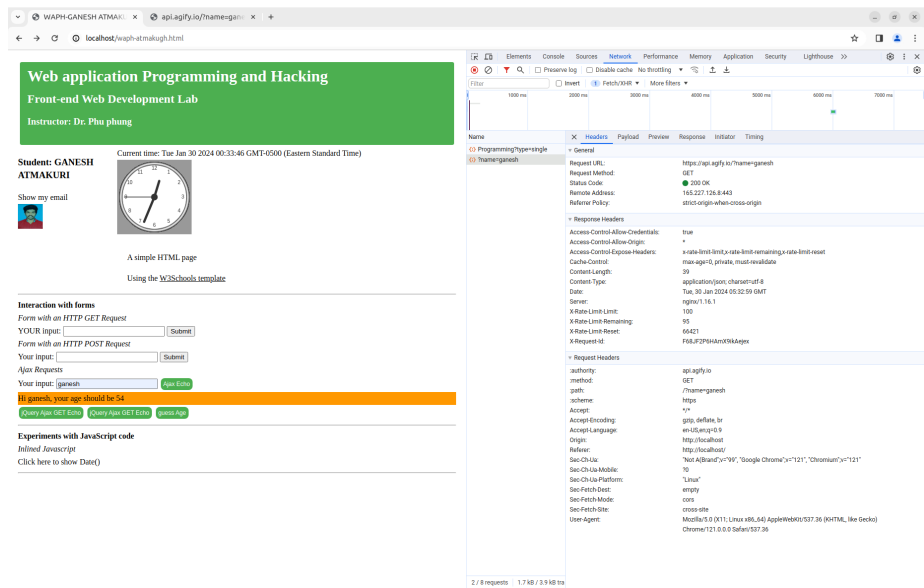


Figure 21: Name API Network window