WAPH - Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name : Vijaykumar Gandi

Email : gandivr@mail.uc.edu

Profile Pic : 

Repository information for Lab-1

Repository URL :

< https://github.com/gandivr/waph-gandivr/tree/main/labs/lab2/ >

In completing this Lab 2 Assignment - Front-end Web Development, I engaged in a comprehensive set of hands-on exercises covering HTML, JavaScript, Ajax, CSS, jQuery, and Web API integration. Let's delve into the detailed learnings following the process outlined in the lab description.

Task 1: Basic HTML with forms, and JavaScript

*** In this Task I Created a simple HTML file named waph-gandivr.html was my initial task.I learned to incorporate basic HTML tags, such as html, head, title, body, and others.I also included an image of my headshot using the img tag and designed a form with the Form tag. This process deepened my understanding of HTML structure and how to integrate multimedia and interactive elements into a webpage.

```
<!DOCTYPE html>
<html>
```

```
<head>
<meta charset="utf-8">
<title>WAPH- VIjaykumar Gandi</title>
</head>
<body>
<div >
    <div id="top">
        <h1>Web Application Programming and Hacking</h1>
        <h2>Front End Development Lab </h2>
        <h3>Instructor : Dr Phu Phung</h3>
    </div>
    <div>
        <div id="menubar">
            <h3>Student : Vijaykumar Gandi</h3>
            <img src="images/Screen-23.png" alt="Vijay Headshot image" width="50">
        </div>
        <div id="main">
            <p>A Simple HTML Page</p>
            Using the <a href="https://www.w3schools.com/html">W3 Schools Template</a>
            <hr>
            <b>Interaction with forms</b>
        <div>
            <i> Form with an HTTP GET request</i>
            <form action="/echo.php" method="GET">
                Your Input: <input name="input">
            <input type="submit" value="Submit">
        </form>
    </div>
    <div>
        <i> Form with an HTTP POST request</i>
        <form action="/echo.php" method="GET" name="echo_post">
            Your Input: <input name="input" onkeypress="console.log('You pressed a key')<inp
        </form>
    </div>
</div>
</body>
</html>
```

- Basic HTML Page:

  HeadShot:

  Form:

*** Simple JavaScript

In this sub-task i involved embedding JavaScript code into my HTML page.
Where I implemented inline JavaScript within HTML tags to display the current

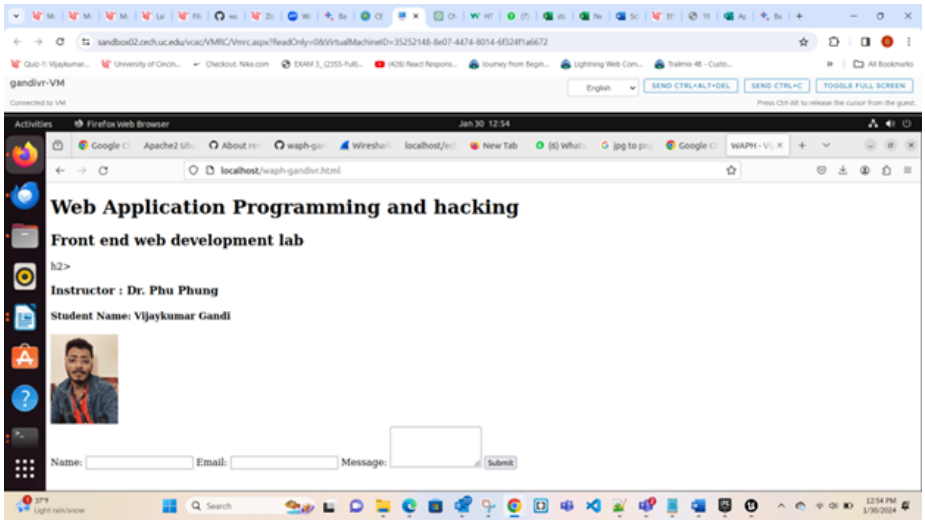


Figure 1: Screen-1

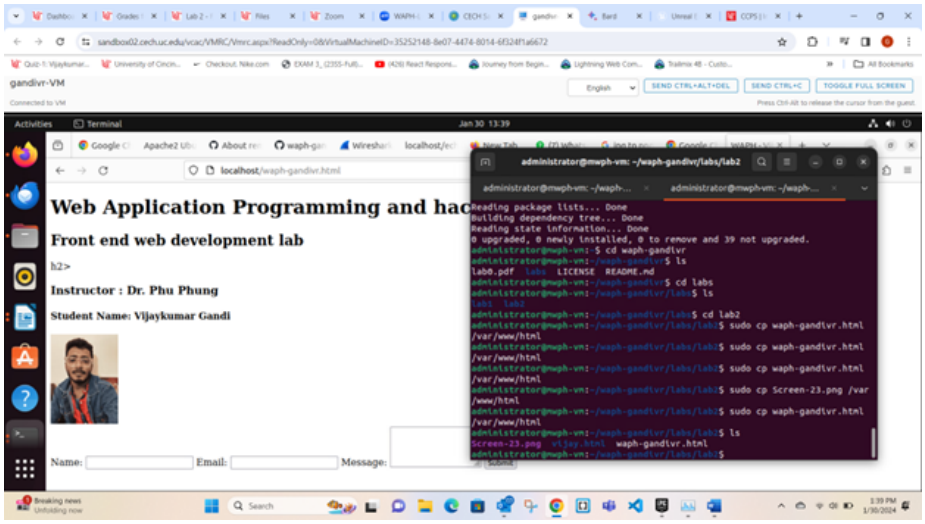


Figure 2: Screen-2

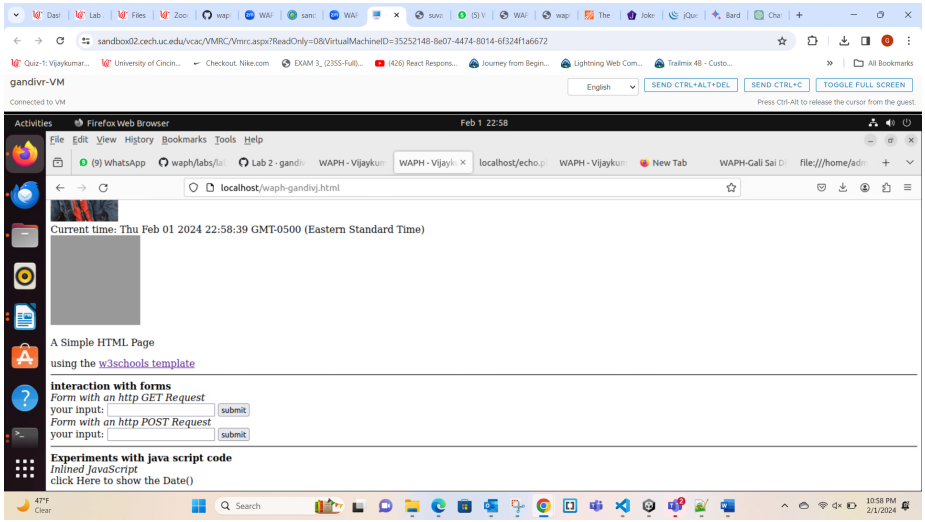


Figure 3: Screen-3

date/time on a click event and log key presses. And i Created a digital clock using a

```
<script type="text/javascript">
    function displaytime()
    {
        document.getElementById('digit-clock').innerHTML = "Current time: " + new Date();
    }
    setInterval(displaytime,500);
</script>
```

function drawClock() { drawFace(ctx, radius); drawNumbers(ctx, radius); drawTime(ctx, radius); }

Email:

Analog Clock:

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

a. Ajax

To understand Ajax, I added HTML elements for user input, a button, and a

element. I wrote JavaScript code to capture user input, construct an Ajax GET request to echo.php, and dynamically display the response content. Inspecting network connections in the browser allowed me to review and illustrate the Ajax request/response mechanism, enhancing my comprehension of asynchronous communication in web development.

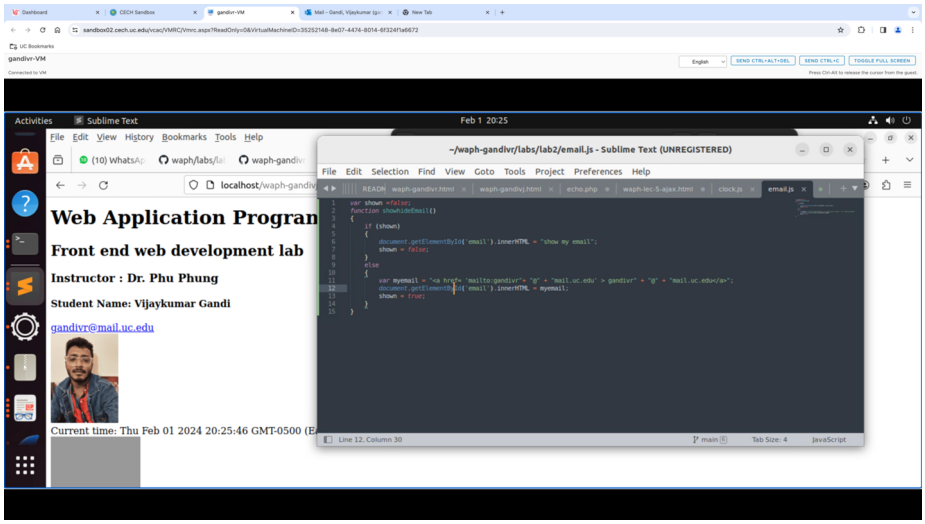


Figure 4: Screen-6

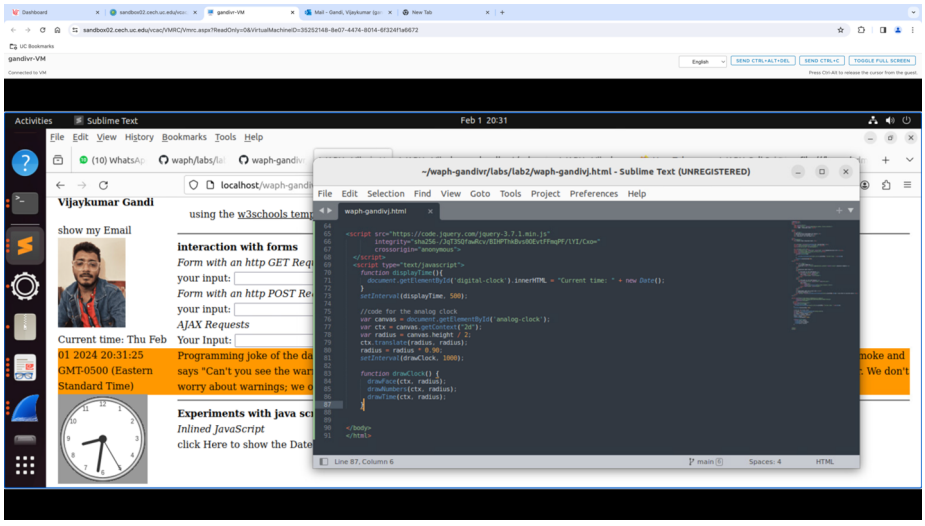


Figure 5: Screen-5

Ajax:

```
        <i> AJAX Requests </i><br>
        Your Input:
        <input name = "data" onkeypress="console.log('You have pressed a key')" id="data">
        <input type="submit" value="submit">
        <input class="button round" type="submit" value="Ajax Echo" onclick="getEcho()">
        <input class="button round" type="submit" value="Ajay Jquery GET" onclick="getJquery
        <input class="button round" type="submit" value="Ajay Jquery POST" onclick="getJquer
        <input class ="button round" type="submit" value="Guess Age" onclick="guessAge($('#c
        <div ID="response"></div>
</div>
```

- ![Screen-7](images/Screen-7.png)

b. CSS

Incorporating CSS into my page involved applying inline, internal, and external styles. I experimented with styling to understand how each method impacts the visual presentation of the webpage. This hands-on experience helped me grasp the nuances of CSS and its role in enhancing the aesthetics and layout of web content.

CSS:

```
 <link rel="stylesheet" type="text/css" href="https://waph-uc.github.io/style1.css">
<style>
  .button{
    background-color:#4CAF50;
    border:none;
    color:white;
    padding:5px;
    text-align:center;
    text-decoration:none;
    display:inline-block;
    font-size:12px;
    margin:4px2px;
    cursor:pointer;
  }
  .round{
    border-radius:8px;
  }
  #response{
```

```
    background-color:#ff9800;
  }
</style>
```

- `![Screen-8](images/Screen-8.png)`

c. jQuery

I added the jQuery library to my page and implemented HTML and JavaScript code to send Ajax GET requests to echo.php. Utilizing jQuery streamlined the process of DOM manipulation and interaction with the server. Creating buttons to trigger Ajax requests and display response content on click events enhanced my proficiency in using jQuery for efficient front-end development.

```
 function getJqueryAjax(){
  var input=$("#data").val();
  if(input.length==0)
    return;
  $.get("echo.php?data="+input,function(result){
    printResult(result);
  });
  $("#data").val("");
    }
function getJqueryAjaxPost(){
  var input=$("#data").val();
  if(input.length==0)
    return;
  $.post("echo.php",{data:input},function(result){
    printResult(result);
  });
  $("#data").val("");
}
```

JQUery:

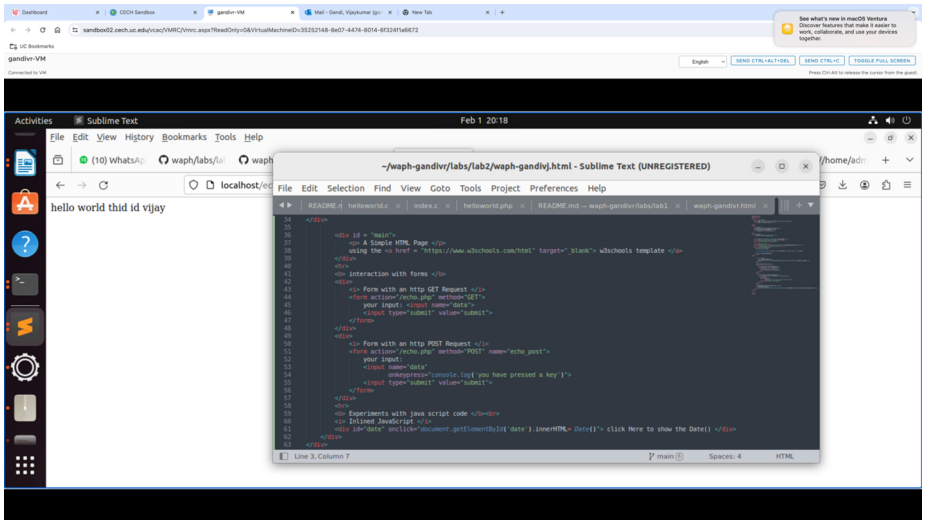


Figure 6: Screen-9

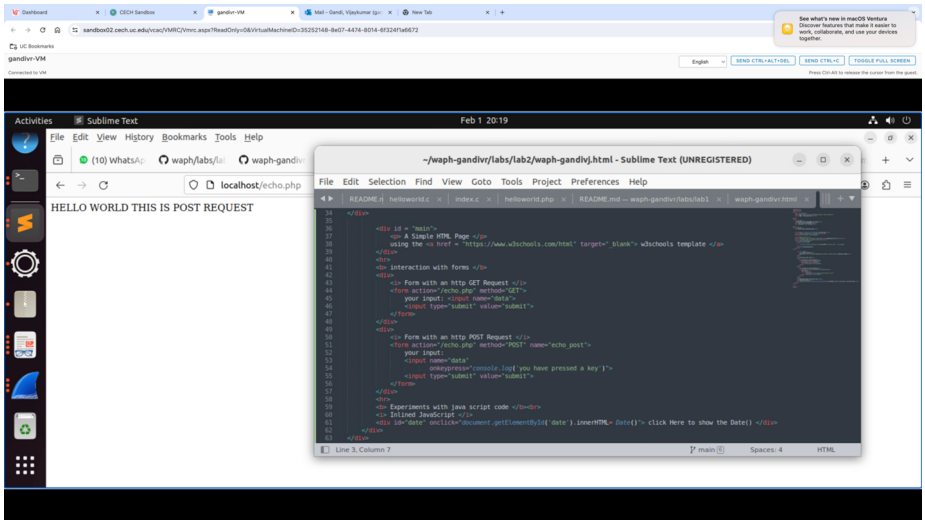


Figure 7: Screen-10

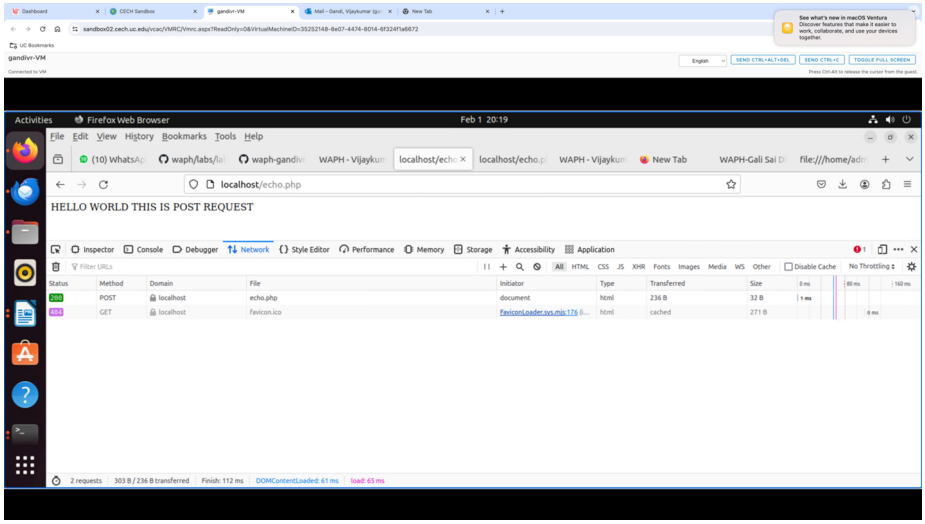
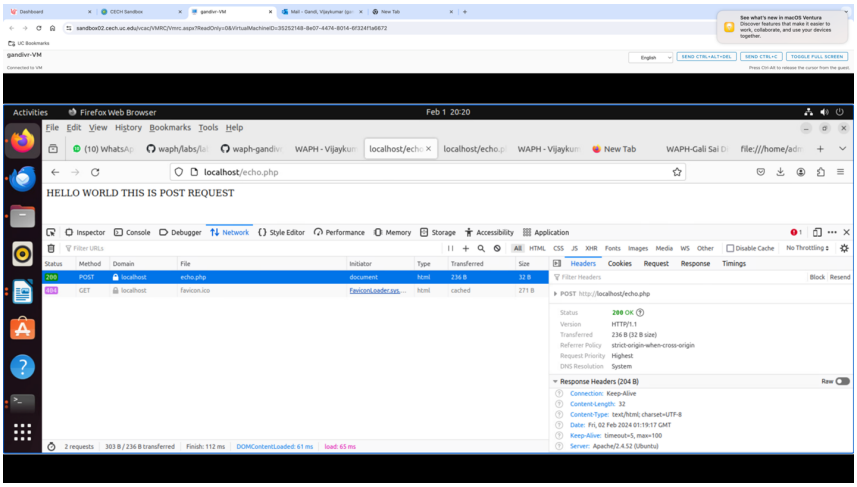


Figure 8: Screen-11



-

d. Web API integration

Using Ajax on https://v2.jokeapi.dev/joke/Programming?type=single, I wrote JavaScript code using jQuery Ajax to send a request and handle the response, displaying a random joke on page load. Inspecting the network in the browser allowed me to examine the request and response, deepening my understanding of integrating external APIs.

Similarly, using the fetch API on https://api.agify.io/?name=input involved writing HTML and JavaScript code to call the API with user input and display the response results. Inspecting the network connections provided insights into

the fetch API's request and response mechanism.

```
 function printResult(result){
        $("#response").html("Response From Server: " + result);
    }
$.get("https://v2.jokeapi.dev/joke/Programming?type=single",function(result){
        $("#response").html("Programming joke of the day: " +result.joke);
    });
async function guessAge(name){
  const response= await fetch("https://api.agify.io/?name="+name);
  const result= await response.json();
  $("#response").html("Hello "+name+" ,your age should be "+result.age);
    }
```

API:

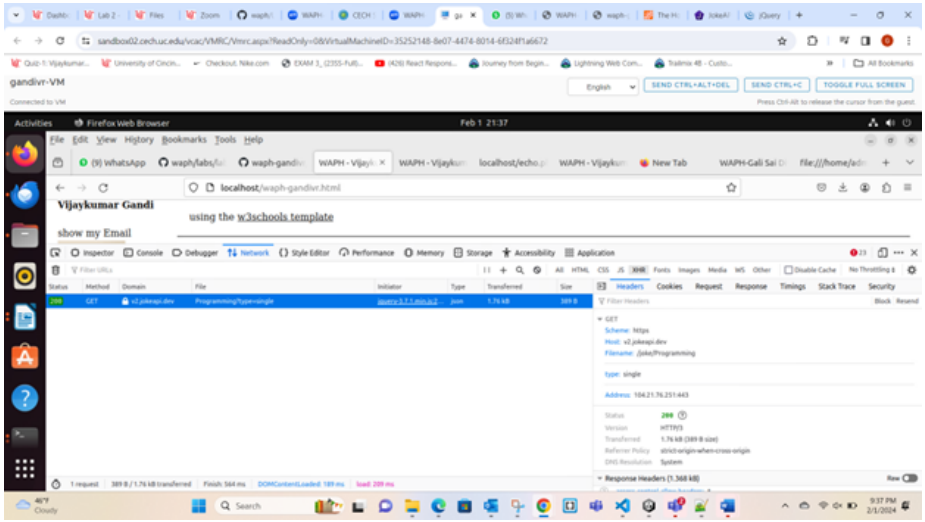


Figure 9: Screen-13