WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Bheemreddy Vikhyath Reddy

Email: bheemrvy@mail.uc.edu

Figure 1: Vikhyath Reddy

Repository Information

Repository's URL: (https://github.com/Vikhyath-Reddy/waph-bheemrvy)

The codes for all topics taught in the Computer Systems course remain in this private repository. This section outlines the structure of this repository.

The lab's overview

This extensive lab covers both basic and advanced web programming principles and takes place throughout Lectures 4-6. Using both inline and external JavaScript, Task 1 instructs participants to create a basic HTML file with tags, a headshot picture, and a form. They also learn how to incorporate JavaScript for inline functionality, such as displaying the current date and time, logging keystrokes, creating an analog clock, and implementing show/hide features for emails. Advanced subjects like Ajax, CSS, jQuery, and Web API integration are covered in Task 2. Building Ajax GET requests, examining network connections, and applying inline, internal, and external CSS styling are among the tasks that participants perform. Library integration is used by the jQuery component to send Ajax GET requests & display responses when a button is clicked.The objective of the Web API integration work is to use jQuery Ajax to retrieve and display an odd programming joke, and to use the fetch API on a different endpoint with JavaScript and HTML to display user input and responses. The lab emphasizes the real-world application of web design concepts at every turn, ranging from fundamental HTML structures to complex subjects like Ajax & API integration.

This lab provides a concise yet comprehensive learning experience in HTML tags, CSS, AJAX, jQuery, and API calls. Participants will gain practical skills in using these fundamental web development tools, covering everything from basic HTML structure to advanced topics like asynchronous request handling and API integration.

Link to Lab2 code : [https://github.com/Vikhyath-Reddy/waph-bheemrvy/tree/main/Labs/Lab2]

Task 1: Basic HTML with forms, and JavaScript

a. Create a straightforward HTML file titled "waph-yourusername.html" featuring essential tags, an embedded headshot image, and a simple form.

The HTML file "waph-bheemrvy.html" was written by me and has a basic structure with tags like "<!DOCTYPE html>," "

," "

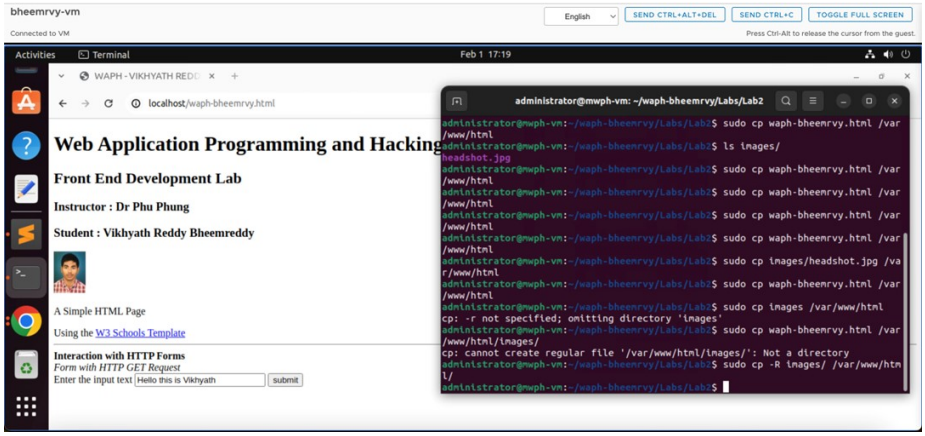
," and "

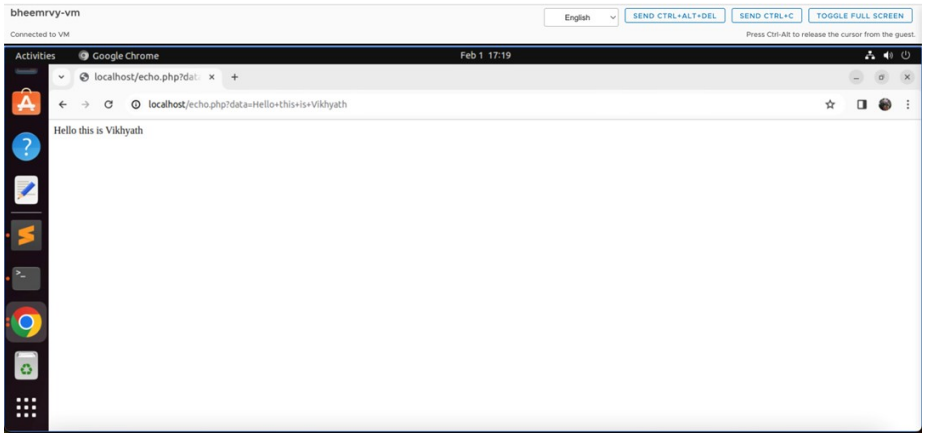
." A title and meta tags for the viewport and character set are contained in the

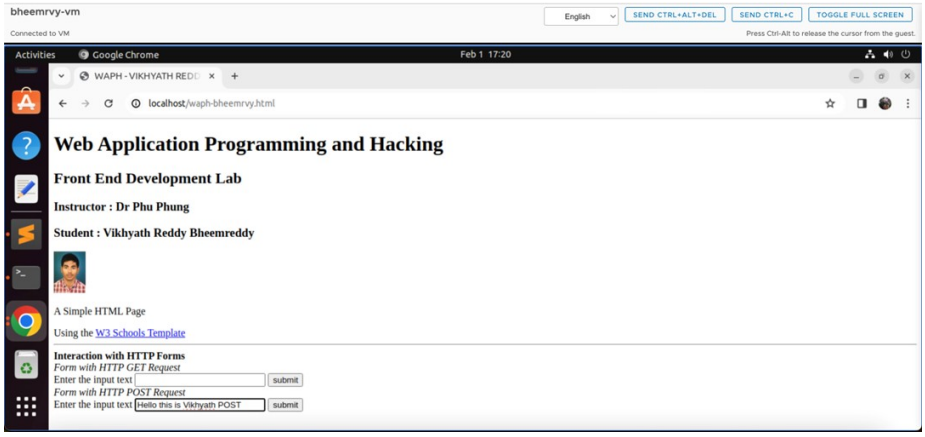
element. I inserted an image using '', a form with '

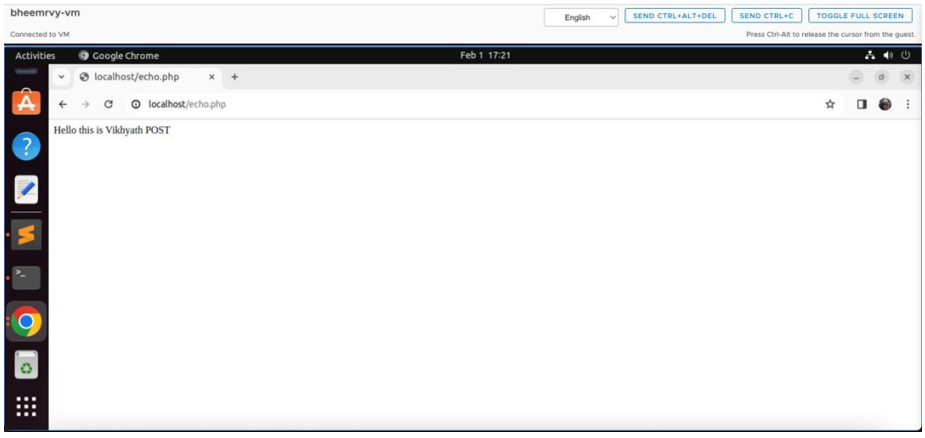
', and the heading tags with names in the

section. Customized text was added, including "Your Name," "Your Webpage Title," and "Professor's Name". I saved the file and opened it in a browser to see my basic webpage.




bheemrvy-vm
Connected to VM
English
SEND CTRL+ALT+DEL
SEND CTRL+C
TOGGLE FULL SCREEN
Press Ctrl-Alt to release the cursor from the guest.
Activities
Google Chrome
Feb 1 17:19
localhost/echo.php?data=Hello+this+is+Vikhyath
Hello this is Vikhyath

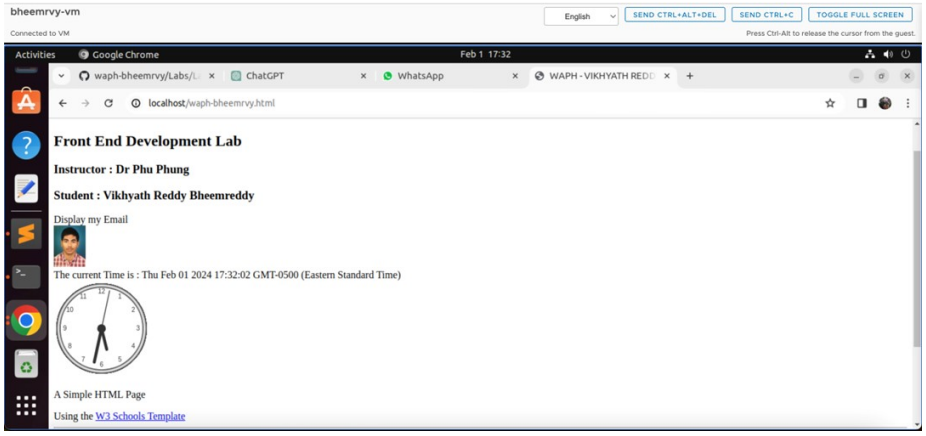

bheemrvy-vm
Connected to VM
English
SEND CTRL+ALT+DEL
SEND CTRL+C
TOGGLE FULL SCREEN
Press Ctrl-Alt to release the cursor from the guest.
Activities
Google Chrome
Feb 1 17:20
WAPH - VIKHYATH REDD
localhost/waph-bheemrvy.html
Web Application Programming and Hacking
Front End Development Lab
Instructor : Dr Phu Phung
Student : Vikhyath Reddy Bheemreddy
A Simple HTML Page
Using the W3 Schools Template
Interaction with HTTP Forms
Form with HTTP GET Request
Enter the input text
submit
Form with HTTP POST Request
Enter the input text
Hello this is Vikhyath POST
submit

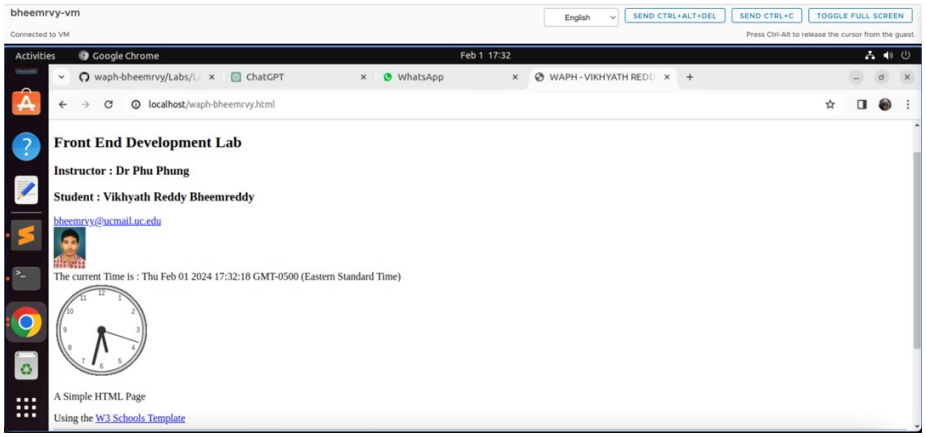

bheemrvy-vm
Connected to VM
English
SEND CTRL+ALT+DEL
SEND CTRL+C
TOGGLE FULL SCREEN
Press Ctrl-Alt to release the cursor from the guest.
Activities
Google Chrome
Feb 1 17:21
localhost/echo.php
localhost/echo.php
Hello this is Vikhyath POST

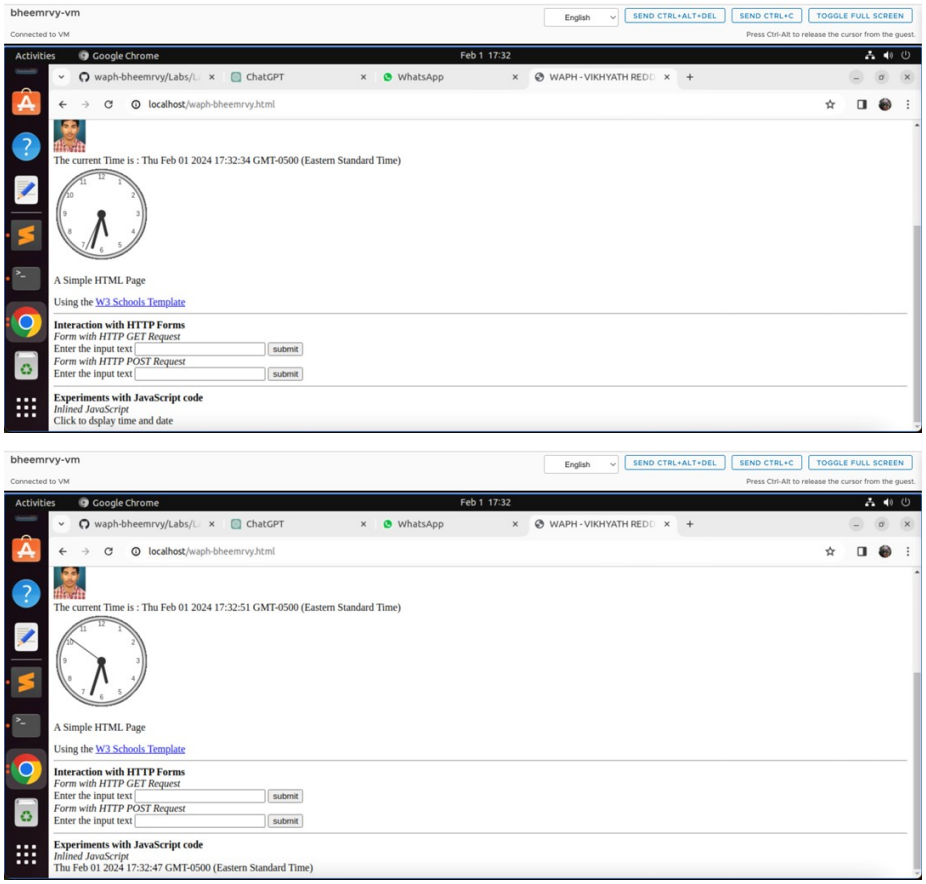
b. Simple JavaScript

I used many JavaScript features to improve user interaction when creating my HTML page. The current date can be shown dynamically and keystrokes can be recorded in real time when a click event occurs thanks to inline JavaScript. An internal JavaScript clock that updates in real time is created within a ‘

} element. JavaScript code organized into an external file enables an email show/hide feature and presents an HTML canvas-based interactive analog clock with visually appealing graphics. An HTML page that is rich in features and interactive is produced by combining inline, internal, & external JavaScript.





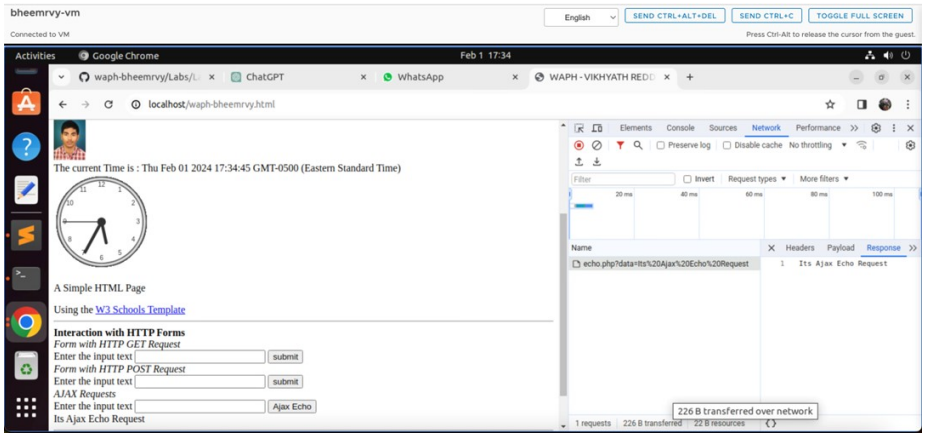


Task 2: Ajax, CSS, jQuery, and Web API integration

a. Ajax

I improved my HTML page by including a button, a ’

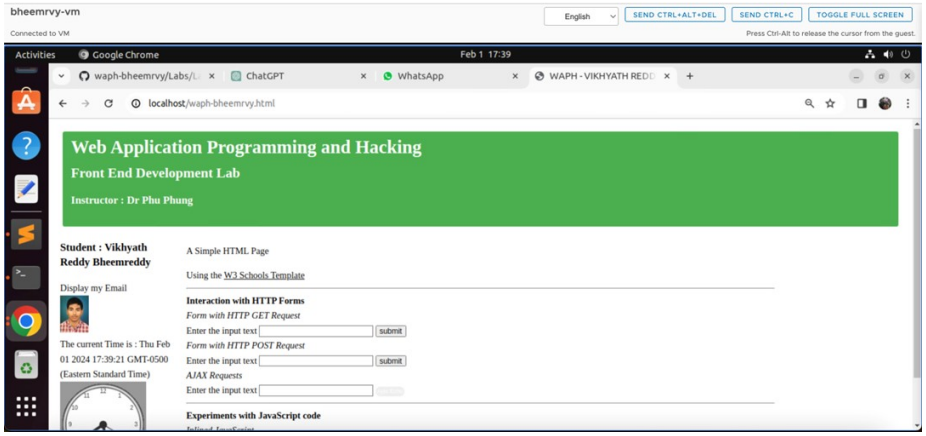
’ element, and a user input box. I programmed the button to communicate user input to the server by sending an Ajax GET request to the echo.php web application using JavaScript. The HTTP response is captured by an event listener, which then shows what it contains in the webpage’s specified element.



b. CSS

By using CSS for internal formatting in the ’

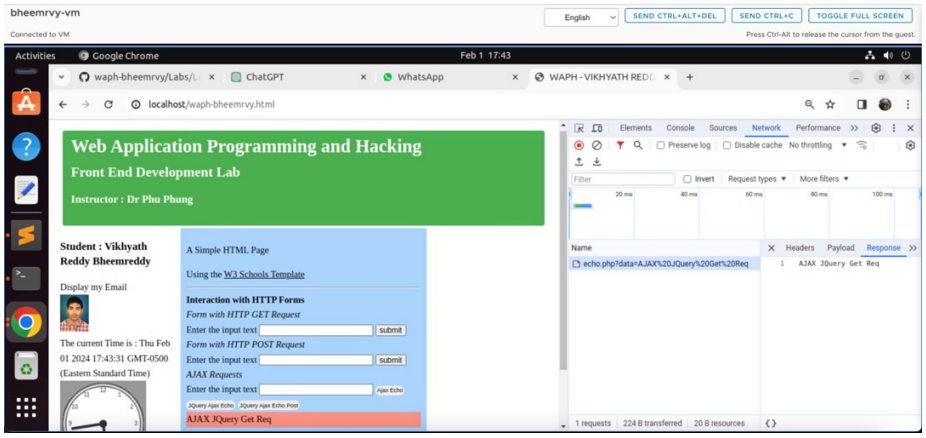
’ section and inline styles for specific elements, I enhanced the visual appeal of my webpage. Using inline CSS, the background color of the analog clock was set. For organized styling, I also generated an external CSS file (like styles.css), which I linked to the HTML using the ‘’ tag to make it more maintainable. In addition, I improved the design process by using a remote CSS file to add pre-existing styles.



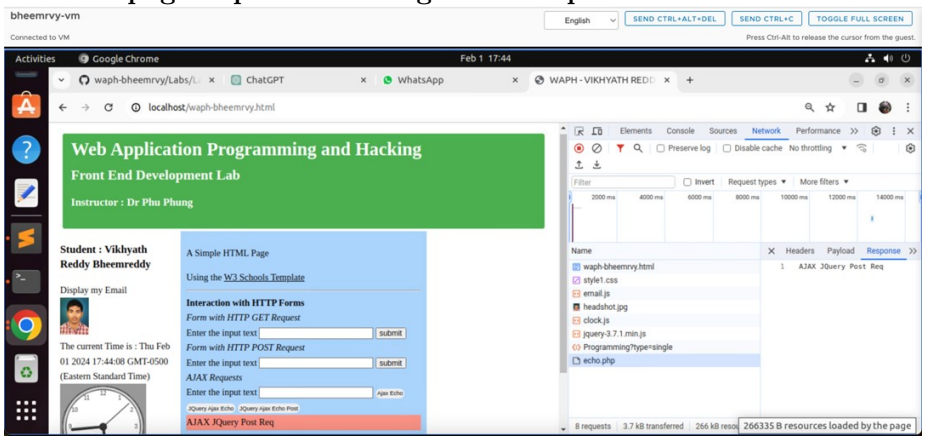
c. jQuery

1) When the corresponding button is clicked, send an Ajax GET request to the echo.php web application and display the response content I used JavaScript to construct a button to launch an Ajax GET request, giving my webpage a dynamic feature. After the code receives the response from the server, it interacts with the echo.php web page to get and

dynamically display data. In addition to facilitating immediate interaction for seamless data extraction and webpage presentation, this improves user engagement.



2) I added a feature that makes my webpage more user-responsive. A particular button's click initiates JavaScript code, which sends an Ajax POST request to echo.php and transfers data for processing. The code dynamically loads the material onto the webpage upon receiving the response from the server.



d. Web API integration

1) Using Ajax on https://v2.jokeapi.dev/joke/Programming?type=single
I used jQuery Ajax to add a captivating feature to my webpage. My JavaScript code fetches single-line jokes about programming from the https://v2.jokeapi.dev API by launching an Ajax call upon page load. The code manages the response well, incorporating a random joke into the page with ease. I examined the request as well as the reply details using browser network tools, which gave me

important insight into the data flow with the external API. This adds levity to the user experience and demonstrates how well I can use Ajax to retrieve and deliver dynamic data.

2) Using the fetch API on https://api.agify.io/?name=inputLinks to an external site. I interacted with the https://api.agify.io/ endpoint via the fetch API, which allowed me to submit user-input names for examination. I created code using HTML and JavaScript that uses fetch() to call the external API and pass in user input. The outcomes are displayed live on the website, and I used the network tools in the browser to look up request details. But I ran into trouble when the agify API hit its daily limit, which resulted in an error message (429: Request Limit Reached) and no answer.