

# Web Application Programming and Hacking – Lab 1 Report

**Instructor:** Dr. Phu Phung

**Student Name:** Sai Pranay Gundu

**Email:** gundus1@udayton.edu

**Short-bio:** Cybersecurity student with interest in network security, ethical hacking, and web programming.



Figure 1: Headshot

**Repository URL:** [<https://github.com/pranay2503/waph-pranay2503.git>]

## 1. Overview

This project involved creating a comprehensive, professional portfolio website from the ground up and deploying it to the cloud using GitHub Pages. The core of the assignment was to build a dynamic and interactive front-end experience by leveraging modern web technologies. This included structuring the site with HTML5, styling it with a responsive CSS framework (Bootstrap), and bringing it to life with advanced JavaScript functionalities.

**Key learning outcomes:** - **Front-End Frameworks:** Practical experience using Bootstrap to create a professional, mobile-first, and responsive layout

without writing extensive CSS from scratch. - **Advanced JavaScript:** Implemented several dynamic features, including real-time clocks and interactive UI elements, reinforcing skills in DOM manipulation. - **Asynchronous Operations:** Mastered `async/await` with the `fetch` API to integrate third-party web services, handling asynchronous data fetching, parsing JSON, and updating webpage content. - **Client-Side State Management:** Used JavaScript cookies to store data on the client-side, allowing the website to “remember” users and personalize their experience across sessions. - **Cloud Deployment:** Hands-on experience with the development-to-deployment workflow using Git and GitHub Pages, making a web application publicly accessible.

**Project URLs:**

- **Live Deployed Website:** <https://pranay2503.github.io/>
  - **Project GitHub Repository:** <https://github.com/pranay2503/pranay2503.github.io>
  - **WAPH Project Repo:** <https://github.com/pranay2503/waph-pranay2503>
- 

## 2. Project Tasks and Implementation

### Task 1: Professional Website and Non-Technical Requirements

**1.1 Use of an Open-Source CSS Framework** Bootstrap 5 was used to ensure the website is fully responsive and visually appealing. The theme is based on the “Simply Me” template from BootstrapMade, integrated via CDN links in the HTML `<head>`.

```
<link href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap/5.3.3/css/bootstrap.min.css" rel="stylesheet">
<link href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-icons/1.11.3/font/bootstrap-icons.css" rel="stylesheet">
```

**1.2 Page Tracker** A Flag Counter is embedded in the footer to track the number and origin of visitors.

```
<a href="https://info.flagcounter.com/YgqF">
  
</a>
```

### Task 2: WAPH Course Page

A separate `course.html` page introduces the “Web Application Programming and Hacking” course, with instructor, institution, and a link to the project repository.

---

### Task 3: Technical Requirements - JavaScript Functionality

#### 3.1 Use of JavaScript Libraries

- **jQuery:** For simplified DOM manipulation and event handling.

- **Typed.js:** For animated typing effect in the hero section.

**3.2 Live Digital and Analog Clocks** The `initializeClocks` function in `assets/js/main.js` creates and updates two clocks in real-time. The digital clock updates every second, and the analog clock is drawn on a `<canvas>` and redrawn every second.

```
// assets/js/main.js - Analog Clock Drawing Logic
function drawClock() {
  // ... clear canvas ...
  const now = new Date();
  const hour = now.getHours();
  const minute = now.getMinutes();
  const second = now.getSeconds();
  // Calculate angle for each hand
  let hourAngle = (hour * Math.PI / 6) + (minute * Math.PI / (6 * 60)) + (second * Math.PI / (6 * 60 * 60));
  drawHand(hourAngle, radius * 0.5, radius * 0.07, '#333');
  // ... draw minute and second hands ...
}
setInterval(drawClock, 1000);
```

**3.3 Show/Hide Email Address** An interactive feature in the “About” section allows toggling the visibility of the email address using a jQuery script in `index.html`.

---

## Task 4: Technical Requirements - Web API Integration

**4.1 JokeAPI Integration** Fetches a new programming or pun joke every 60 seconds from the JokeAPI.

```
// assets/js/main.js
async function fetchJoke() {
  const jokeContainer = $('#joke-text');
  try {
    const response = await fetch('https://v2.jokeapi.dev/joke/Programming,Pun?type=single');
    const data = await response.json();
    jokeContainer.text(data.joke || 'No joke found...');
  } catch (error) {
    console.error("Failed to fetch joke:", error);
    jokeContainer.text('Could not load joke.');
  }
}
fetchJoke();
setInterval(fetchJoke, 60000); // Fetch a new joke every minute
```

**4.2 XKCD API Integration** Displays a random XKCD comic using a public CORS proxy (`api.allorigins.win`) to handle cross-origin issues.

```
// assets/js/main.js
async function fetchComic() {
  const comicContainer = $('#comic-container');
  const randomComicId = Math.floor(Math.random() * 2500) + 1;
  const apiUrl = `https://xkcd.com/${randomComicId}/info.0.json`;
  const proxyUrl = `https://api.allorigins.win/get?url=${encodeURIComponent(apiUrl)}`;

  try {
    const response = await fetch(proxyUrl);
    const data = await response.json();
    const comicData = JSON.parse(data.contents);
    comicContainer.html(`
      <strong class="d-block mb-2">${comicData.title}</strong>
      
    `);
  } catch (error) {
    // ... error handling with a fallback comic ...
  }
}
```

---

## Task 5: Technical Requirements - JavaScript Cookies

**5.1 Remembering the Client** JavaScript cookies are used to track user visits. The `friendlyGreeting` function checks for a `lastVisit` cookie and displays a personalized welcome message.

```
// assets/js/main.js
function friendlyGreeting() {
  const welcomeBanner = $('#welcome-banner');
  const lastVisitCookie = document.cookie.split('; ').find(row => row.startsWith('lastVisit'));
  let message = 'Welcome to my portfolio!';

  if (lastVisitCookie) {
    const lastVisitDate = new Date(decodeURIComponent(lastVisitCookie.split('=')[1]));
    message = `Welcome back! Last visit: ${lastVisitDate.toLocaleString()}`;
  }

  welcomeBanner.text(message);
  document.cookie = `lastVisit=${encodeURIComponent(new Date().toISOString())};path=/;max-age=300000`;
}
```

---

## Features Implemented

- **Responsive Design:** Bootstrap 5 for mobile-first, professional layout.
  - **Professional Sections:** About Me, Skills, Portfolio, Contact, and WAPH Course.
  - **Animated Hero Text:** Typed.js for dynamic typing effect.
  - **Live Clocks:** Real-time digital and analog clocks.
  - **Show/Hide Email:** Toggle button for email privacy.
  - **Cookie-Based Greeting:** Personalized welcome message for returning visitors.
  - **Page Tracker:** Flag Counter for visitor analytics.
  - **Joke of the Minute:** Fetches a new joke every minute from JokeAPI.
  - **Random XKCD Comic:** Displays a random XKCD comic using a CORS proxy.
- 

## Technologies Used

- HTML5
  - CSS3
  - JavaScript (ES6+)
  - Bootstrap 5
  - jQuery
  - Typed.js
  - JokeAPI
  - XKCD API
- 

## Setup and Usage

No local setup is required.

**Live Site:** <https://pranay2503.github.io/>

---

## Credits

The visual theme of this website is based on a template by BootstrapMade.