NO.:
DATE:

John Emil I. Bragais
BSCS 3-1N

COMP 016
Web Development

Assignment #1

1.-Code Optimization

a. Efficient Code Structure - write clean, modular, and reusable code to enhance maintainability and reduce redundancy. For example in CSS, it's best to write the styles of all containers that have the same styles into one selector instead of separating them into different section despite having no distinction between them.

b. Minification - reduce the size of the code files by removing whitespace, comments, and unnecessary characters in the code.

- Security Measures

a. Regular Update - keep all software, libraries, and dependencies up-to-date to mitigate vulnerabilities and reduce the risk of exploitation.

b. HTTPS - use more secured extended version of the Hypertext Transfer Protocol (HTTP), HTTPS with the S for Secure, to encrypt data transmitted between the server and the user's browser, ensuring secure communication.

c. Input Validation and Sanitization - validate and sanitize user inputs to prevent common security vulnerabilities, such as SQL injection and cross-site scripting (XSS) attacks.

d. Authentication and Authorization - implement strong authentication mechanisms, such as multi-factor authentication (MFA), and ensure proper authorization to control access to sensitive data and functionalities.

- Accessibility Standards

a. Contrast and Readability - maintain sufficient color contrast between text and background, and use readable fonts and sizes to ensure content is easily readable for all users.

b. Alt Text for Images - provide descriptive alt text for images to ensure they are accessible to users with visual impairments who rely on screen readers.

c. Keyboard Navigation - ensure that all interactive elements can be accessed and operated using a keyboard, catering to users with motor impairments.

NO.:
DATE:

d. ARIA (Accessible Rich Internet Application) Roles - Use ARIA roles and attributes to enhance the accessibility of web applications and provide additional context to assistive technologies.

2. * Progressive Web Apps

Progressive Web Apps (PWAs) combine the best features of web and mobile applications. They offer offline functionality, fast loading times, and improve user engagement through push notifications. PWAs provide a native app-like experience while being accessible directly from the web, eliminating the need for app store downloads. This trend is reshaping web development by enabling developers to create highly responsive and engaging applications that work seamlessly across different devices and platforms.

Opportunities for Developers:

- Develop a single codebase that works on multiple platforms, reducing development time and costs.
- Create apps that offer a smooth and engaging user experience, leading to higher user retention and satisfaction.
- PWAs can be indexed by search engines and shared via URLs, making them easily discoverable and accessible.

* Web Assembly

Web Assembly (wasm) is a binary instruction format that allows code written in multiple programming languages to run on the web at near-native speed. It provides a secure and efficient execution environment, enabling high-performance applications that were previously not feasible on the web. WebAssembly is designed to work alongside JavaScript, allowing developers to leverage its performance benefits while maintaining the flexibility of web development.

Opportunities for Developers:

- Build applications that require intensive computations, such as games, simulations, and data processing tools.
- Use languages like C, C++, and Rust to develop web applications, expanding the pool of available development tools and libraries.

NO.:
DATE:

- Benefit from WebAssembly's memory-safe and sandboxed execution environment, reducing the risk of security vulnerabilities.

* Serverless Architecture

It allows developers to build and deploy applications without managing server infrastructure. Cloud providers handle server management, scaling, and maintenance, enabling developers to focus on writing code.

Opportunities for Developers:

- Automatically scale applications based on demand, ensuring optimal performance and cost-efficiency.
- Eliminate the need for server management, reducing operational complexity and costs.
- Accelerate development cycles by focusing on code and functionality rather than infrastructure.

3. Backend development refers to the server-side aspect of web development, focusing on managing the behind-the-scenes functionality that powers a website or web application. It involves handling server-side logic, such as processing requests, executing business logic, and managing data storage through databases. Backend developers create APIs for seamless communication between different parts of an application and ensure security measures like authentication and authorization. They also manage server configurations, deployments, and optimize performance through techniques like caching and load balancing.

4. * Node.js (JavaScript):

Performance - It is known for its high performance due to its non-blocking, event-driven architecture. It can handle a large number of concurrent connections with minimal overhead, making it ideal for real-time applications.

Scalability - excels in scalability, allowing horizontal scaling through clustering and load balancing. Its lightweight nature and ability to handle asynchronous operations make it suitable for large-scale applications.

Ease of Use - uses JavaScript, which is a widely known and used by

NO.:
DATE:

developers. This makes it easier for front-end developers to transition to backend development. The extensive npm (Node Package Manager) ecosystem provides a vast array of libraries and tools.

* PHP

Performance - PHP has improved significantly in terms of performance with the introduction of PHP 7 and later versions. However, it may not match the performance of Node.js for real-time applications.

Scalability - PHP can be scaled effectively using techniques like load balancing and caching. However, it may require more effort to achieve the same level of scalability as Node.js due to its synchronous nature.

Ease of Use - PHP is known for its simplicity and ease of use, making it a popular choice for beginners. It has a large community and extensive documentation, which helps a big time in learning and troubleshooting.

* Django (Python):

Performance - Django, a high-level Python web framework, offers good performance for most web applications. However, it may not be as fast as Node.js for handling a large number of concurrent connections.

Scalability - It is a highly scalable and can handle large-scale applications with proper optimization. It supports various scaling techniques, including database optimization, caching, and load balancing.

Ease of Use - It emphasizes rapid development and clean, pragmatic design. It comes with built-in features like an admin panel, authentication, and ORM (Object-Relational Mapping), which simplify development.

Integration with Front-end Frameworks

All three server-side technologies can integrate seamlessly with front-end frameworks like React, Angular, and Vue.js. They typically communicate via RESTful APIs or GraphQL, allowing the front-end to send requests and receive data from the backend. This separation of concerns enables developers to work on the front-end and backend independently, promoting modularity and flexibility in development.

NO.:
DATE:

5. HTML (HyperText Markup Language)

It is the standard markup language used to create and structure content on the web. It defines the structure of web pages using a series of elements and tags, which specify how text, images, links, and other content should be displayed in a browser.

Its roles include embedding various types of content, creating hyperlinks for navigation, enhancing accessibility through proper semantic elements, and improving search engine optimization (SEO) by providing meaningful context to the content.

6. Semantic Markup in HTML involves using elements such as header, main, and section, instead of using div, that convey meaning and context about the content they enclose. This approach enhances both accessibility and search engine optimization (SEO) of web pages by providing clearer information to both users using assistive technologies to interpret and navigate the content, and search engines for better indexing of the web page's content more accurately and rank it appropriately in search results.

Examples of Semantic Elements:

<header> - Represents the introductory content or navigation links for a section or page.

<nav> - Defines a set of navigation links.

<section> - Groups related content together, typically with a heading.

<aside> - Contains content that is tangentially related to the main content, such as side bars.

<footer> - Represents the footer for a section or page; often containing copyright or contact information.

<article> - Encapsulates a self-contained piece of content that could be independently distributable or reusable.