



## Accessibility in Web Development

**Daksh Purohit<sup>1</sup>, Himanshu Vyas<sup>2</sup>, Mayank Gupta<sup>3</sup>, Yogita Punjabi<sup>4</sup>**

Department of Artificial Intelligence & Data Science  
 Jaipur Engineering College & Research Centre  
 dakshpurohit2003@gmail.com

### Abstract

The Food Ordering System is a web application designed to streamline the process of ordering food from restaurants. It offers a convenient and user-friendly platform for customers to browse menus, place orders, and make payments online. This project aims to enhance the dining experience by bringing the entire ordering process online. User Registration and Authentication: Users can create accounts and log in securely, ensuring a personalized experience. Browse a wide range of restaurants, view their menus, and access essential details. Menu Customization: Customize food orders by adding or removing items, specifying preferences, and choosing delivery or pickup options. Online Payments: Securely pay for orders using various payment methods, including credit/debit cards and digital wallets. Order Tracking: Track the status of orders in real-time, from preparation to delivery. Reviews and Ratings: Users can leave reviews and ratings for restaurants and dishes to help others make informed choices. Admin Dashboard: Restaurant owners have access to an admin dashboard to manage menus, orders, and customer reviews. The front end is built using React, providing a dynamic and responsive user interface. JSX creates interactive components and manages the presentation layer. JSON is utilized for data exchange between the front-end and back-end. The back-end is powered by Node.js, providing a scalable and efficient server environment. Express.js is used to build robust APIs and handle HTTP requests. MongoDB is the database system used for storing user profiles, restaurant data, menu, and order information.

*Keywords: Web Accessibility, Inclusive Design, WCAG, User Experience, Assistive Technologies, Accessibility Compliance, Usability Testing, Inclusive Web Development.*

### Article Status

Available online :

*2024 Pratibodh Ltd. All rights reserved.*

### 1. Introduction

The internet, a ubiquitous and versatile resource, is accessible across a multitude of platforms, rendering it an essential tool in the contemporary digital landscape. While the internet operates based on established web standards, numerous vendors have devised proprietary technologies that offer enhanced functionalities surpassing the capabilities of these standards. These technologies enable the creation of sophisticated web applications, providing an interactive and immersive user experience. Examples of such proprietary formats include Adobe Systems' Flash, Apple's QuickTime, Microsoft's Silverlight, Google's Gears, and Oracle's JavaFX. These platforms have expanded the horizons of web development, offering unique and innovative solutions for building rich internet applications. This paper delves into the intricacies of these proprietary technologies, examining their impact on the web development landscape and exploring the opportunities and challenges they present to developers and users alike.

### HTML5

Inaugurated in 2004, HTML5 has revolutionized web development, bridging the gap between traditional web

design and application development. Unlike its predecessors, HTML5 enables the creation of dynamic and interactive web pages, integrating multimedia elements such as audio, video, and animations seamlessly. This standard not only expands the horizons of web content but also introduces offline functionality, allowing users to access web applications without an active internet connection. Moreover, HTML5 incorporates geolocation services, enhancing user experiences based on their physical locations, and facilitates local storage in client-side databases, ensuring efficient data management. In essence, HTML5 stands as a cornerstone in the evolution of web technologies, ushering in an era where interactivity, multimedia, and offline accessibility converge, shaping a more engaging and responsive digital landscape. HTML5 stands as a cornerstone in modern web development, providing a robust framework designed to deliver consistent and immersive web experiences across diverse platforms. Tailored for compatibility with PCs, Tablets, Smartphones, and Smart TVs (refer to Fig. 1), HTML5 redefines the concept of platform versatility. As a working draft, HTML5

has garnered significant traction, with discerning browser designers and forward-thinking websites embracing its elements. One of its hallmark features lies in its support for location-based services, accentuating the importance of geospatial context in web applications.

Furthermore, HTML5 embraces open standards, exemplified by its compatibility with Scalable Vector Graphics (SVG) and open XML file formats, ensuring seamless integration of high-quality graphics. This amalgamation of cross-platform compatibility, forward-thinking design, and support for open standards underscores HTML5's pivotal role in shaping the contemporary digital landscape.



Fig. 1

#### HTML5 Features

HTML5 incorporates pioneering features, redefining the landscape of web development with following features

- Canvas – 2D/3D Graphics
- Audio & Video
- Location based Services
- Working Offline
- Web Workers
- Drag & Drop
- New Input Types
- New Elements
- Form Elements

#### Canvas

The HTML5 element stands as a pivotal tool in contemporary web development, enabling the dynamic rendering of graphics through JavaScript on web pages enriched with video and animations.

Functioning as a versatile container, the element serves as the canvas upon which intricate 2D and 3D graphics come to life. It distinguishes itself by offering a robust framework where scripts meticulously craft graphics, showcasing a

seamless integration of artistry and technology.

This canvas interface provides developers with an array of sophisticated methods, facilitating the creation of intricate paths, geometric shapes, textual elements, and images. Noteworthy is its exceptional efficiency in handling

graphics-intensive content, ensuring rapid rendering and optimal performance.

This amalgamation of precision and speed positions the element as an indispensable asset for crafting visually engaging web experiences.

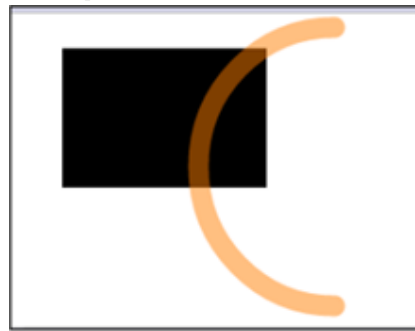


Fig.2

#### Svg (Scalar Vector Graphics)

SVG, or Scalable Vector Graphics, is a cornerstone in web design due to its exceptional clarity and adaptability. One of its standout features is that SVG images remain sharp and clear, regardless of their size. Whether enlarged or compressed, the image quality remains intact [4]. Moreover, SVG images can be effortlessly searched, indexed, scripted, and compressed, showcasing their versatility. They are not only printable in high quality at any resolution but also allow seamless zooming without compromising their sharpness. In the realm of digital graphics, SVG stands as a testament to both simplicity and precision, making it an essential tool for web designers aiming for exceptional visual impact. (Fig-3,4). There are more tools as well as given above.



Fig.4

### CSS: An Overview

In Cascading Style Sheets (CSS), a style sheet (C) is comprised of rules (R) that determine the visual properties of document elements. Each rule (R<sub>i</sub>) contains a single selector (S<sub>i</sub>) and a pair defining properties (P<sub>i</sub>) and their values (V<sub>i</sub>). These selectors specify which elements the properties will be applied to, shaping their appearance in web browsers. Selectors, structured as sequences of simple selectors linked by combinators, identify targeted elements. Simple selectors include the universal selector '\*' and type selectors (e.g., 'h1').

Despite CSS allowing comma-separated lists of selectors and multiple properties per selector, we focus on single selectors and properties for clarity. These selectors, often referred to as patterns in CSS, define boolean functions determining if an element is selected, forming the foundation for CSS rules. In this overview, we explore key CSS concepts: combinators, structural pseudo-classes, and property inheritance, which are essential elements shaping the visual presentation of web documents.

### CSS Property and Inheritance

In the world of web development, CSS property inheritance operates in a structured manner, ensuring a coherent style flow. When defining a property for an element, CSS follows a specific sequence: if a value is explicitly set, it takes precedence. If not, the property's inheritable nature allows it to inherit the parent element's value. If no inheritance is specified, the default initial value comes into play. This mechanism, exemplified through examples like background and text color inheritance, maintains a consistent design language throughout a website.

Additionally, understanding the specificity of selectors becomes pivotal when resolving conflicts in styles, ensuring the accurate rendering of web elements. These rules, although nuanced, form the backbone of seamless and harmonious web design. For example, with this style sheet and this HTML fragment:

```

Div
{
background-color:white;color:blue;font-weight: normal;
}
p
{
background-color:inherit;color: inherit;
}

```

```
<div><p>Hello, world.</p></div>
```

The background color of the div element is set to white. The background color of the paragraph is also white, because its background-color property is set to inherit and the background color of the div parent element is white. The inherit value does not require that the parent element have the same property set explicitly; it works from the computed value. In the above example, the color property of the paragraph has value inherit, but the computed value is blue because it inherits. The font-weight property of the element is also set to normal since it is inherited by default. Fig 5 is shown below to represent the style of the website of food delivery app and styling is done by the help of CSS.

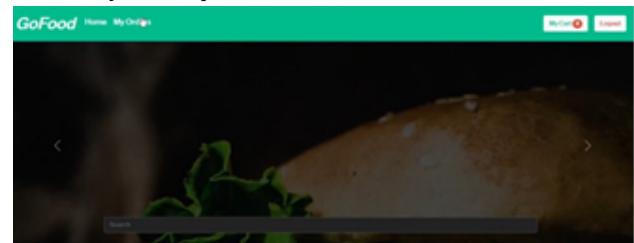


Fig.5

### Bootstrap: An Overview

Bootstrap is a highly popular front-end framework used for developing responsive and mobile-friendly websites. It is a free and open-source framework that incorporates HTML, CSS, and JavaScript, making web development faster and more accessible. One of its key features is the mobile-first approach, ensuring that websites are designed to be responsive on mobile devices first and then scaled up for larger screens. One of Bootstrap's strengths is its wide browser support, ensuring a consistent user experience across different browsers. It is beginner-friendly, allowing anyone with basic knowledge of HTML and CSS to get started. Bootstrap provides a responsive CSS grid system, enabling websites to adapt seamlessly to various screen sizes, from desktop to tablets and mobile phones. Being open source, Bootstrap encourages collaboration and community-driven development. It offers a variety of pre-designed components like typography, forms, buttons, and modals, simplifying the development process. Additionally, Bootstrap includes JavaScript plugins that enhance website functionality without extensive coding. In summary, Bootstrap's appeal lies in its user-friendly nature, extensive documentation, mobile-first design philosophy, broad browser

compatibility, and the availability of pre-designed components and JavaScript plugins. Its

open-source status promotes community collaboration and continual improvement, making it a top choice for web developers. Fig 6 is shown to get an idea that with the help of bootstrap it can be made in this way having the different categories.

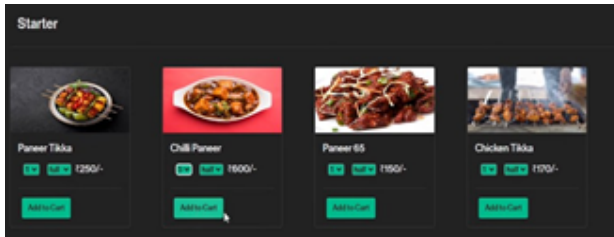


Fig.6

### Bootstrap Grid System

It is widely used to design layout and content structure in print design. In web design, it is a very effective method to create a consistent layout rapidly and effectively using HTML and CSS. Bootstrap includes a responsive, mobile-first fluid grid system that appropriately scales up to 12 columns as the device or viewport size increases. It includes predefined classes for easy layout options, as well as powerful mixins for generating more semantic layouts. This is explained by the fig.7

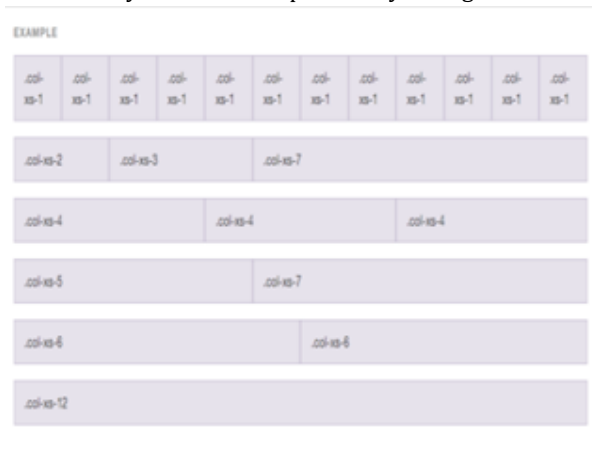


Fig.7

### Bootstrap Typography

Bootstrap establishes basic global display, typography, and link styles. When more specific styling is needed,

- Developers can refer to textual utility classes.
- Bootstrap utilizes a native font stack, ensuring the selection of the best font-family for each operating system and device.
- To enhance inclusivity and accessibility, Bootstrap assumes the browser's default root font-size (typically 16px), allowing visitors to customize their browser defaults as necessary.
- Typographic base styles are applied to the element using the attributes `$font-family-base`,

`$font-size-base`, and `$line-height-base`.

- The global link color is set via `$link-color`, and link underlines are applied only on hover.
- The background-color for the element is set using `$body-bg` (default is #fff).
- These styles are defined within `_reboot.scss`, with global variables specified in `_variables.scss`.
- Developers are advised to set `$font-size-base` in rem units to maintain consistency and scalability.

### Bootstrap: Conclusion

Bootstrap enables fast and responsive development, offering consistency and robust support from the development and design community. As the framework evolves, the reasons to embrace Bootstrap become increasingly compelling. If you haven't explored this framework yet, now is a great opportunity to give it a try. Bootstrap offers numerous advantages, including swift and responsive development, along with steadfast support from the thriving development and design community.

### JavaScript: An Overview

With the rise of the internet, new web technologies have emerged, transforming web applications from static content to dynamic and interactive platforms. Unlike HTML, which is static, JavaScript has played a pivotal role in enabling dynamic web experiences, allowing active communication with users. JavaScript is widely used in contemporary web applications, both as a client-side and server-side scripting language. JavaScript supports the Model View Controller (MVC) architecture, facilitating clean and organized code by separating different parts of the program. All these frameworks are built on the MVC or similar architectural principles. In this study, the research explores the strengths and weaknesses of each framework, their impact on application speed, methods for testing JavaScript applications, and strategies to enhance code security.

### JavaScript

Understanding the Foundations of Web Development: At the core of every web page lies a trio of essential building blocks: Hypertext Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript. HTML forms the bedrock, delineating the structure and fundamental content of a webpage. It provides the scaffolding upon which the rest of the elements rely. CSS steps in as the artist, meticulously crafting the visual appeal of components and the overall layout. It adds the colors, styles, and aesthetics, transforming a plain structure into an engaging user interface. JavaScript, on the other

hand, functions as the dynamic force, empowering developers to manipulate and interact with various elements on the webpage. It breathes life into static pages, enabling responsiveness and interactivity, making the user experience vibrant and engaging.

**The Universal Language of Web Technologies:** No matter how intricate the applied technologies might be, during the compilation phase, everything boils down to the fundamental trio: HTML, CSS, and JavaScript. These components serve as the universal language of web development. Regardless of the complexities of underlying architectures, frameworks, or libraries, everything ultimately translates into these fundamental elements. This universality underscores the importance of mastering these core technologies, as they form the foundation upon which all web-based applications and interfaces are constructed.

**Defining JavaScript's Complex Characteristics:** JavaScript, in the realm of programming languages, is often described with technical jargon: high-level, multi-paradigm, non-blocking, and asynchronous. Terms like garbage-collected, interpreted, single-threaded, and concurrent are frequently used to encapsulate its nature. Yet, for the average reader, especially those new to programming, these descriptors can be overwhelmingly abstract. Understanding the depth and breadth of JavaScript can be a daunting task, especially without a programming background.

**Demystifying JavaScript for All Audiences:** Recognizing this challenge, this chapter endeavors to unravel the complexities of JavaScript. Instead of delving into technical intricacies, it aims to provide a lucid, accessible exploration of JavaScript's main characteristics. By dissecting these features in simple, relatable language, the chapter strives to offer a comprehensive understanding. Whether you're a novice venturing into the programming world or a seasoned developer seeking a clear perspective, this analysis aims to bridge the gap, ensuring that the richness of JavaScript is accessible to all.

## Popular JavaScript Framework

### A. AngularJS

AngularJS is a frontend web application framework initially developed by Google in 2010. The active version is 1.7, released in 2018. It is a vital component of the MEAN stack, which includes MongoDB for the database, Express.js for the web application server framework,

AngularJS itself, and Node.js as the server runtime environment.

AngularJS follows the MV\* (Model View Whatever) architecture, a variation of the traditional MVC model. Unlike MVC, where model, view, and controller components are distinctly separated, in MV\*, the connection between the model and view is essential. Any change in the model reflects in the view and vice versa. Controllers, defined using the ``ng-controller`` directive, play a significant role in AngularJS. They accept a ``$scope`` object, acting as an intermediary between the controller and view, facilitating techniques like dependency injection. Key AngularJS directives include ``ng-app``, ``ng-init``, ``ng-include``, ``ng-model``, ``ng-bind``, ``ng-repeat``, ``ng-show``, ``ng-hide``, and ``ng-click``.

### B. BackboneJS

BackboneJS is a JavaScript library offering a minimalistic approach to separating business logic from the user interface. It operates on an architecture akin to MVC, abstracting data through models, DOM views, and event binding. Backbone relies on the JavaScript library `underscore.js`, which must be included alongside the `backbone.js` file. Its architecture includes components such as HTTP Request (for client-server communication), Router (for connecting actions and events through URLs), View (for presentation), Events (for triggering functions based on object events), Model (containing application data and logic), Collection (a list of models), and Data Source (for database connections).

### C. EmberJS

EmberJS is a JavaScript framework employing MVC architecture. Its components consist of Model (associated with routes and data presentation), View (rarely used due to Handlebar templates), Template (standard template with header, footer, and `{{outlet}}` expressions for dynamic content loading), Controller (managing display logic and interactions between route, model, and view), and Route (defining application representation and loading necessary templates and data based on URLs). Ember's notable features include computed properties, observers, and bindings.

### D. KnockoutJS

KnockoutJS, based on the MVVM architecture derived from MVC, is a JavaScript library supporting the development of interactive applications. The View represents the user

interface created in HTML, which binds to the ViewModel. Changes in the ViewModel are automatically reflected in the View through two-way data binding. ViewModel acts as a bridge between real data and user representation. Core concepts of KnockoutJS include observables, data binding, templates, and components, ensuring structured applications and code reusability.

### **Prong and Consof JavaScript Framework**

Backbone focuses primarily on the core application structure, leaving room for enriching functionalities through plugins. However, without these plugins, it remains a relatively small library with limited capabilities. AngularJS, although highly dynamic and fast-evolving, has yet to see widespread adoption in the industry due to its rapid changes, making many IT companies hesitant to fully embrace it. Ember, while functionally similar to AngularJS, often presents challenges in practical application despite the wealth of online resources available. Knockout lacks integrated routing, a feature readily available in AngularJS. Furthermore, AngularJS stands out by incorporating essential concepts like dependency injection, offering a plethora of services (\$http, \$log, etc.), modules, filtering, form validation, and more, features that are not fully implemented in Knockout. Knockout, however, can serve as a suitable framework for simpler applications where intricate user interface control is not a primary concern. In terms of architectural analysis, all four frameworks operate based on variations of the MVC architecture. Backbone aligns with the MVP architecture, where HTML and DOM serve as the View component, and the Backbone View acts as the Presenter component. When it comes to template handling, Backbone utilizes the underscore.js library, Ember employs handlebars.js, and Knockout integrates its own template handling system. Regarding the binding of views and models and page refreshing methods, AngularJS, Ember, and Knockout offer automatic view refresh when the model undergoes changes. In contrast, Backbone necessitates calling the render function each time the View requires refreshing, granting programmers more control but demanding careful handling. A challenge identified during research pertains to memory leaks in Backbone. Consequently, for users less experienced in JavaScript, it is advisable to consider alternative frameworks among the remaining three options to avoid encountering these complexities.

### **References**

- [1] Research on HTML5 in Web Development by Ch Rajesh, K S V Krishna Srikanth, Department of IT, NITS, Visakhapatnam.
- [2] On the Analysis of Cascading Style Sheets from Pierre Geneves, CNRS, Nabil Layaida, INRIA and Vincent Quint, INRIA.
- [3] J. J. Barton and J. Odvarko. Dynamic and graphical web page breakpoints. In Proceedings of the 19th international conference on World Wide Web, WWW'10, pages 81-90, New York, NY, USA, 2010. ACM.
- [4] B. Bos, T. , Celik, I. Hickson, and H. W. Lie. Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification. W3C Recommendation, World Wide Web Consortium, June 2011.
- [5] T. , Celik, E. J. Etemad, D. Glazman, I. Hickson, P. Lines, and J. Williams. Selectors level 3. W3C recommendation, World Wide Web Consortium, September 2011.
- [6] A Review Paper on Bootstrap Framework by Suraj Shahu Gaikwad, Prof Pratibha Adkar, MCA Department, PES Modern College of Engineering Pune, India. [7] C. Beem, W. Peelaers, and L. Rastelli explored the concepts of deformation quantization and superconformal symmetry in three dimensions in their 2016 study.
- [8] In 1982, Efron, B. published a work titled "The Jackknife, the Bootstrap, and Other Resampling Plans" through the Society for Industrial and Applied Mathematics in Philadelphia.
- [9] The Nature and Evolution of JavaScript by Van Chau Dao.
- [10] Survey on web technology present usage statistics of JavaScript as a client-side programming language on websites. The data was retrieved on October 7, 2020.