MODERN TECHNOLOGIES FOR BUILDING GRAPHICAL USER INTERFACES ON THE INTERNET

Nikolay Nikolov[1]

[1] *Assist. Prof., Department of Informatics, University of Economics, Varna, Bulgaria*

E-mail: nikolay.nikolov@ue-varna.bg

JEL: C69, C88

Received: 7.11.2023

Accepted: 7.12.2023

Published: 22.12.2023

ABSTRACT

This article presents a comprehensive overview of modern technologies for creating graphical user interfaces on the internet, with a focus on key aspects such as HTML, CSS, and JS. Libraries and frameworks such as React.js, Vue, and Angular are analyzed to unfold their advantages and disadvantages. The first part of the article examines fundamental concepts related to user interfaces (types of user interfaces) and explores technologies like HTML, CSS, and JS, while also conducting a comparison between different JavaScript libraries and frameworks. The primary goal is to highlight their applications and advantages in the development of enhanced user interfaces. The second part of the article concentrates on React.js, introducing its core characteristics and methods of utilization in new or existing web applications. The process of creating React applications is demonstrated, encompassing the creation of components and state management. Key aspects of React are analyzed to provide a thorough understanding of its capabilities in constructing contemporary graphical interfaces on the internet. In the conclusion of the article, the main findings are summarized, emphasizing the significance of modern technologies in improving the user experience in the online environment.

Keywords: *User Interfaces, Web Technologies, GUI Development*

Citation*: Nikolov, N. (2023). Modern Technologies for Building Graphical User Interfaces On The Internet. Journal „Човешки ресурси & Технологии = HR & Technologies", Creative Space Association, 2, pp. 90 – 104.*

INTRODUCTION

In the era of information technology, web applications play an increasingly important role in our daily lives. Modern web technologies are evolving rapidly, with constantly increasing demands placed on them. They allow us to receive and share information, communicate with people from various parts of the world, and optimize a significant portion of modern businesses. Every modern company utilizes numerous information products in its operations. Therefore, building an effective and intuitive user interface is of crucial importance for the success of the information product. Very often, these products are web-based applications with numerous features. One of the directions of development in web technologies is user interfaces, they are becoming increasingly complex, requiring time for development, and simultaneously aiming for simplicity, intuitiveness, and ease of use. Understanding the new modern technologies to achieve these goals is of utmost importance for developers.

Various technologies are used for the development of web applications, depending on the needs and requirements of the project. **The purpose** of the current article is to present the fundamental technologies for building modern web-based user interfaces. It aims to outline the advantages and disadvantages of using the tools and to highlight the implementation of interface elements through a selected library.

1. OVERVIEW OF MODERN TECHNOLOGIES FOR BUILDING GRAPHICAL USER INTERFACES ON THE INTERNET AND LITERATURE REVIEW

User Interface (UI) is the way in which users (people) interact with a particular device. The user interface includes both hardware and software components. The user interface exists for various processes and provides a means of input and output (Deacon, 2020).

User Interface (UI) is the point of interaction between a person and a computer and the communication within the device. This can include screens, keyboards, mice, and the visual appearance of the desktop. It is also the way in which the user interacts with an application or website. There are several types of user interfaces:

- **Command Line Interface (CLI)** – This interface uses text commands to interact with a computer system. It is often used by developers, system administrators, and advanced users who prefer the efficiency and flexibility of command-based tools.

- **Graphical User Interface (GUI)** – This interface utilizes icons, buttons, and other visual elements to allow users to interact with a computer system. This is the most encountered type of user interface, used on desktop and mobile devices.
- **Natural Language Interface (NLI)** – This interface uses natural language, such as spoken or written words, to interact with a computer system. It is often used in virtual assistants and chatbots.
- **Touchscreen Interface** – This interface allows users to interact with a computer system by touching the screen. It is commonly used on mobile devices and tablets.
- **Gesture-Based Interface** – This interface uses physical gestures, such as swipes, taps, and pinches, to interact with a computer system. It is commonly used on touchscreens and in virtual reality systems.
- **Voice-Based Interface** – This interface allows users to interact with a computer system using their voice. It is commonly used in virtual assistants and smart home devices.

Every type of user interface has its advantages and disadvantages, and the choice of interface will depend on factors such as user needs, the device being used, and the context of use (Churchville, 2021).

When it comes to creating websites, multiple software technologies are employed. All web pages in the virtual space are text documents written in the **Hypertext Markup Language (HTML).** This technology serves as a fundamental markup language for describing and designing web pages, outlining the structure of the document (Sulova, 2018).

HTML is the primary language for visualization in the web space. It defines the meaning and structure of web content. "Hypertext" refers to the links that connect web pages, both within a website and between different websites. Links are a fundamental aspect of the web (MDN, 2023). HTML (HyperText Markup Language) is a metalanguage for describing formatted documents. The description of documents is done through special elements called tags. Tags are used to format individual elements of the text, such as headings, quotes, sections, hyperlinks, etc. (Borisova, 2014). HTML markup includes special "elements" such as <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , , <img>, <aside>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <output>, <progress>, <video>, <ul>, <ol>, <li>, and many others. An HTML element is isolated from the rest of the text in its document with "tags," consisting of the element's name surrounded by "<" and ">" (MDN, 2023).

HTML5 allows us to build diverse internet content without resorting to the inclusion of plugins or an interface provided by other manufacturers (Varbanov, et al., 2014). Cascading Style Sheets (CSS) are used to describe the format of web pages, specifying the visual appearance of elements. This way, the development of internet pages separates content from their presentation (Sulova, 2018).

CSS is a descriptive language through which rules are created, indicating how an HTML element should appear when visualized on a web page (Varbanov, et al., 2014). The official specification of CSS is maintained by the World Wide Web Consortium (W3C). CSS allows the definition of the appearance of elements on an HTML page, including fonts, sizes, colors, backgrounds, and more. CSS code consists of a sequence of style rules, each representing a selector followed by properties and values (Borisova, 2014).

With its development, CSS becomes more detailed and complex. By applying the technology, various views can be created, tailored to the screen size and device orientation. This allows the construction of web applications with so-called adaptive or responsive design, enabling web interfaces to look good and be accessible across devices with different resolutions (Sulova, 2018).

JavaScript (JS) is a lightweight, interpreted programming language with object-oriented capabilities that enable the building of interactivity in static HTML pages. JavaScript was developed by Brendan Eich from the Netscape team under the name Mocha. Later, it was renamed to LiveScript, and this was the official name when it was first released in beta versions of the Netscape Navigator 2 browser. In 1995 (version 2 of Netscape Navigator), it was renamed JavaScript. Microsoft also introduced a JavaScript-like language called JScript in its Internet Explorer 3. The JavaScript language is supported by all web browsers and standardized by the European Computer Manufacturers Association (ECMA) (Borisova, 2014).

For client-side JavaScript applications DOM manipulation is a key ability that enables dynamic manipulation of HTML and CSS resources. However, DOM manipulation is expensive and the performance cost can potentially have a negative impact on the user experience. DOM manipulation is central to all the selected frameworks, but the frameworks have taken different approaches to implementing DOM manipulation. can give insights to better understand any differences in DOM manipulation performance.(Persson, 2020)

Websites of 2020 are often feature rich and highly interactive applications. JavaScript is a popular programming language for the web, with many frameworks available. A common

denominator for highly interactive web applications is the need for efficient methods of manipulating the Document Object Model to enable a solid user experience. (Persson, 2020)

Although best known as a scripting language for web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB, and Adobe Acrobat. JavaScript is a prototype-based, single-threaded, dynamic language that supports object-oriented, imperative, and declarative styles (MDN, 2023).

Through JavaScript, content can be dynamically inserted into web pages, elements can be modified, information about the current date, browser type, form content can be gathered, and the collected information can be visualized. JavaScript enables reacting to events occurring in the browser, such as writing a message when a page loads or when the user selects an image (Sulova, 2018).

In dynamic web pages, the AJAX (Asynchronous JavaScript and XML) technique is used for faster loading of web pages. AJAX allows updating parts of a web page without reloading the entire page. This reduces the amount of information exchanged between the server and the client, increasing the speed of loading and functionality of web pages (Sulova, 2018).

For the development and maintenance of complex projects, software frameworks are used. They contain basic software modules with ready-to-use code snippets that developers use to solve common programming tasks, such as managing AJAX requests or defining file structures. They also dictate the rules for building the architecture of the application: you get a skeleton that needs to be extended and modified according to specific requirements (Sakovich, 2023).

Software frameworks can include utility programs, code libraries, scripting languages, and other software that facilitates the development and integration of various components into a large software product (Sakovich, 2023). Thanks to software frameworks, developers don't start projects from scratch; they have the foundation for implementing other project-specific functions. This helps accelerate development and improve the performance and reliability of solutions (Sakovich, 2023).

JavaScript software frameworks are an essential part of modern front-end web development, providing developers with proven tools to build scalable, interactive web applications. Many contemporary companies use frameworks as a standard part of their toolkit, so many front-end development positions now require experience with JavaScript libraries and frameworks (MDN, 2023).

HTML, CSS, and JavaScript are technologies used to create user interfaces, but the use of libraries and software frameworks facilitates the creation of more complex, interactive, and reactive user interfaces.

A software library is a collection of program code that provides functionalities and procedures needed to solve specific tasks. This includes a set of functions, classes, or methods that can be integrated into other programs. When a programmer uses a library, they add a reference to it in their program code and interact with the provided functionalities. On the other hand, a software framework represents a more extensive and structured foundation for software development, where basic components and architectural structure are already embedded. This streamlines the process of creating an application, as the programmer uses pre-defined components and follows specific rules for building their application on this foundation (Goel, 2023).

In summary, the library provides specific functionalities, while the framework provides a holistic structure and approach to application development, already containing the necessary elements for creating larger and more complex software projects (Goel, 2023).

Some of the most popular libraries and frameworks are:

- **React.js,** developed by Facebook, is a popular library for building user interfaces. It allows developers to create reusable UI components and offers excellent performance. React uses a virtual DOM, which enables more efficient updates and reduces the number of necessary DOM manipulations (Meta Open Source, 2023).
- **Angular** is a platform and framework for building client-side single-page applications using HTML and TypeScript. Angular is written in TypeScript and implements essential and optional functionality as a set of TypeScript libraries imported into applications (Angular (Google), 2022).
- **Vue** (pronounced /vjuː/, like view) is a JavaScript framework for building user interfaces. It is based on standard HTML, CSS, and JavaScript, providing a declarative and component-based programming model to help you efficiently develop user interfaces, whether they are simple or complex. Vue was created by Evan You in 2014 as a personal side project. Currently, it is maintained by both a full-time team and volunteers from around the world (Vue, 2023).
- **jQuery** is a fast, lightweight, and feature-rich JavaScript library that simplifies traversing and manipulating HTML documents, handling events, and facilitating AJAX

interactions. jQuery does not impose a structural framework for organizing code; it is often used to enhance existing applications with interactive functionality (jQuery, 2023).

In Table 1, some data, functions, and characteristics of the libraries and frameworks are compared.

Table 1.
Comparing key features of popular libraries and software frameworks for building user interfaces

Feature	jQuery	React	Vue	Angular
Classification	Library	Library	Framework	Framework
Development date	August 2006	March 2013	February 2014	September 2016 (Angular.js 2010)
Author	John Resig	Jordan Walke (facebook)	Evan You	Miško Hevery (google)
Open source	Yes	Yes	Yes	Yes
Learning Difficulty	Easy	Average	Easy	Hard
Performance	Low	High	High	High
Data binding	Limited	One-way	Two-way	Two-way
Virtual DOM	No	Yes	No	Yes
Server rendering	No	Yes	Yes	Yes
Reusability	Low	High	High	High
Templates	Basic	JSX	HTML	HTML
Language Support	JavaScript	JavaScript/ TypeScript	JavaScript	TypeScript/ JavaScript
Community Support	Big	Big	Big	Big
State management	No	Yes	Yes	Yes
Routing	No	React Router	Vue Router	Angular Router
Dependency Injection	No	No	No	Yes
Form handling	Limited	Uncontrolled	V-Model	Reactive Forms
Testing	Limited	Jest	Jest	TestBed
Mobile development	Limited	React Native	Weex	NativeScript

Source: Own Elaboration

The analysis of the data in Table 1 shows that jQuery lags in characteristics and functionalities compared to the other listed libraries and frameworks. It can be concluded that jQuery is less relevant to modern methods of developing user interfaces and is not recommended for use due to the availability of better software solutions.

The front-end libraries and frameworks mentioned so far are just some of the most popular ones used by developers today. The choice of a library or framework depends on the project requirements, as well as the experience and preferences of the developers. There are several reasons why react,js should be choosed instead of other frameworks. As shown in “Table 1” react.js is preferred for its declarative syntax, allowing for code that is easier to understand and debug. Its efficient virtual DOM enhances rendering performance by updating only the components that have changed. The component-based architecture of React promotes code reusability and maintainability, fostering collaboration within development teams. With a large and active community, React enjoys extensive support, resources, and a plethora of third-party libraries. React is flexible and versatile, easily integrable into existing projects, and supports server-side rendering for various application types. Widely adopted in the industry, React provides reassurance in terms of long-term stability and sustainability.

2. REACT.JS – KEY FEATURES AND USAGE

React is a JavaScript library for building user interfaces (UI). The user interface is constructed from small units like buttons, text, and images. React allows them to be combined into reusable, nested components. From websites to mobile applications, everything on the screen can be divided into components (Meta Open Source, 2023). React can be used both to create a single-page application entirely built with React and to integrate it into existing web applications when developing new features (Meta Open Source, 2023). When creating a new React application, it is recommended to use frameworks that leverage React. These frameworks provide functionalities that most applications and websites would eventually need (Meta Open Source, 2023).

Such software frameworks include:

- **Next.js** is a software framework for React. Next.js is versatile and allows the creation of React applications of any size—from primarily static blogs to complex dynamic applications.
- **Remix** is a software framework for React with built-in routing. It allows you to divide your application into nested parts that can load data in parallel and update in response to user actions.

- **Gatsby** is a React framework for quickly creating websites with support from CMS systems. Its rich ecosystem of plugins and a GraphQL data layer make it easy to integrate content, APIs, and services into a website.
- **Expo** is a React framework that enables the creation of universal Android, iOS, and web applications with genuine Native user interfaces.

When adding React to an existing web application, several different approaches can be used. Embedding React components in existing HTML pages—this method uses React components as parts of HTML pages embedded in existing HTML code. This is a clever way to gradually introduce React into existing websites. React can be added as a script in HTML files and used to visualize dynamic parts of the web page. Another approach is building new React pages, including them in the existing web application at new URLs.

At the core of the library is the principle of components, as every user interface in the application is built from components. Each particle of the interface in a React application is a component. The advantages of this component-based approach to application development are reusability and separation of concerns.

Components are one of the fundamental concepts in React. They form the foundation on which user interfaces (UI), their functionalities, and state are built (Meta Open Source, 2023).

Components are the fundamental building blocks in a React application. They can be functional or class-based (the use of class components is currently considered outdated and not recommended by the official React documentation, but it is still permissible and supported). React components can receive parameters, called props (short for properties), allowing them to be personalized and reused in various contexts. Components can also maintain their own state, used to manage and update their user interface.

React components can be nested, forming a component tree. The highest-level component is usually called the "App" component, responsible for rendering and managing other components.

An example of defining a component can be seen in Fig. 1, where the StudentCard component is defined. This component displays the student's name, major, shows their picture, and states which course they are in. Another component can be seen in Fig. 2, where the StudentGallery component is defined, displaying a title and using the StudentCard component three times. Fig. 3 shows how the StudentGallery component participates in the definition of

the "App" component, which is at the highest level of the component tree. The screen after launching the application can be seen in Fig. 4.

```
export function StudentCard(props) {
  return (
    <section style={{ margin: '1rem', display: 'inline-block' }}>
      <h2>{props.fullName}</h2>
      <h4>{props.specialty.toUpperCase()}</h4>
      <img
        width={320}
        height={320}
        src={props.imgURL}
      />
      <div style={{ fontWeight: 'bold' }}>Курс: {props.year}</div>
    </section>
  );
}
```

Fig. 1. Definition of StudentCard component

Source: Own Elaboration

```
export function StudentGallery() {
  return (
    <main style={{ display: 'flex', flexDirection: 'column', alignItems: 'center' }}>
      <h1>Невероятните студенти</h1>
      <section>
        <StudentCard
          specialty='Информатика'
          year='2'
          fullName='Иван Иванов'
          imgURL="https://faces-img.xcdn.link/image-lorem-face-5759.jpg" />
        <StudentCard
          specialty='Мобилни и уеб технологии'
          fullName='Мария Маринова'
          year='3'
          imgURL="https://faces-img.xcdn.link/image-lorem-face-5238.jpg" />
        <StudentCard
          specialty='Бизнес информационни системи'
          fullName='Георги Григоров'
          year='4'
          imgURL="https://faces-img.xcdn.link/image-lorem-face-4796.jpg" />
      </section>
    </main>
  );
}
```

Fig. 2. Definition of StudentGallery component
Source: Own Elaboration

```
export default function App() {
  return (
    <div className="App">
      <StudentGallery/>
    </div>
  )
}
```

Fig. 3. Definition of App component
Source: Own Elaboration

Fig. 4. Browser Screen when a React Application is Launched
Source: Own Elaboration

JSX is a JavaScript syntax extension that allows writing markup, similar to HTML, within a JavaScript code file. Despite other ways to write components, most React developers prefer the conciseness of JSX, and most codebases use this syntax (Meta Open Source, 2023).

An example of JSX code can be seen in Figures 1, 2, and 3. When writing JSX, the return statement always returns only one parent element. If there are multiple adjacent elements, they must be wrapped in a single parent, which is then returned as the component's value. Another rule is that all tags must be closed, unlike HTML where some tags like **`<img>`**, **`<li>`**, etc., can be self-closing. In JSX, this is not allowed. Another rule is that it's recommended, and in some cases mandatory, to write in camelCase style (all words are written together, with the first letter of the first word in lowercase and the first letter of the remaining words in uppercase).

The primary feature of JSX is the ability to write JavaScript within the markup language. JavaScript code can be included using curly braces “**{}**”. For example, you can write the following **<StudentCard year={3+1}>**, where the value of the year attribute in the **StudentCard** component is set as a JavaScript expression that needs to be evaluated. After evaluation, the value will be four. In Figure 1, all attributes of the elements have values as JavaScript expressions. For instance, in**<h4>{props.specialty.toUpperCase()}</h4>** this expression uses the JavaScript function that converts all letters in the text to uppercase. In the same Figure 1, you can see how parameters are passed to the function (props) and used in the JSX code. Figure 2 shows how the values of these parameters are passed to the **StudentCard** component.

JSX can be conditionally rendered using JavaScript syntax such as if-else statements and ternary operators, as shown in Figure 5. Using if-else constructs in JSX allows you to check a specific condition and render a specific component or element based on whether the condition is true or false. This can be especially useful when you need to render different components or elements depending on some state of the application (Meta Open Source, 2023).

```
if(boolean){
  return <h1>True </h1>
}
else{
  return <h1>False </h1>
}
```

Fig. 5. JSX - if-else
Source: Own Elaboration

The ternary operator is another tool that can be useful for conditional rendering in JSX. It allows writing a concise condition and executing one or another code block depending on the result of the condition. The ternary operator can be particularly helpful when we want to render components or elements that depend on some state of the application but don't want to write long if-else constructs (Meta Open Source, 2023).

React is one of the most popular libraries for creating web applications. It allows developers to create components that are easy to maintain, reuse, and test. This makes it an ideal choice for building single-page (SPA) and mobile applications. React is actively evolving, and it has a large community of developers who provide useful tools and libraries to facilitate the development process. Additionally, many companies use React to build their web applications, making knowledge of this technology valuable for developers.

3. RESULTS AND DISCUSSION

Building an app with React is straightforward due to its declarative and component-based architecture. React's ability to intelligently update only the components that have

changed, rather than re-rendering the entire DOM, leads to improved performance. This feature is particularly beneficial for creating dynamic and responsive user interfaces, ensuring a smoother user experience. The virtual DOM, combined with the reconciliation algorithm, sets React apart by optimizing rendering processes and enhancing the overall speed and efficiency of web applications. Its reusable components simplify code organization and maintenance. Additionally, React's JSX syntax allows developers to write UI components using a syntax that closely resembles HTML, making it more intuitive for front-end developers.

One notable advantage of React is its extensive and active community support. The vast ecosystem of libraries, tools, and resources enables developers to find solutions to common challenges easily. React's unidirectional data flow and state management via props and state contribute to predictable and manageable application state.

Compared to other frameworks, React's component reusability, one-way data flow, and the virtual DOM contribute to a more efficient and maintainable development process. Its ability to seamlessly integrate with other libraries and frameworks also enhances its versatility. While the preference for a particular framework may vary based on project requirements, React's simplicity, scalability, and strong community support make it a popular choice for building modern, interactive web applications.

CONCLUSION

Libraries and software frameworks are of great benefit in web application development, as they provide ready-made solutions for many of the tasks associated with software development. This allows developers to focus on the core functionality of the application without having to solve problems that have already been addressed by others.

With the help of libraries and frameworks, developers can avoid creating new functionalities from scratch, reducing development time, and increasing project efficiency. This leads to faster development of web applications, which is crucial for business and the competitiveness of companies.

Libraries and software frameworks also ensure that the code is easier to maintain and can be understood by other developers, which is essential for the sustainability and future of applications.

The most well-known libraries and frameworks, such as React, Angular, and Vue, have extensive development and active communities, meaning there is plenty of documentation,

guides, and examples for use. This makes their use easy and accessible, even for beginner developers.

In conclusion, the use of libraries and software frameworks is of utmost importance for the efficient and rapid development of modern web applications. They reduce development time, ensure easier maintenance, and sustainability of applications, and are accessible to a wide community of developers.

REFERENCES

1. Angular, (2022). Introduction to Angular concepts. [Online] Available at: https://angular.io/guide/architecture [Accessed 21 март 2023].
2. Borisova, D. I., (2014). Osnovi na web programirane. Sofia: UniBIT.
3. Churchville, F., (2021). What is user interface (UI)?. [Online] Available at: https://www.techtarget.com/searchapparchitecture/definition/user-interface-UI [Accessed 22 Февруари 2023].
4. Deacon, P., (2020). *UX and UI strategy: A step by step guide on UX and UI design.* USA. Independently published
5. Ekta Goel, (2023), Software Framework vs Library [Online] Available at: https://www.geeksforgeeks.org/software-framework-vs-library [Accessed 2023].
6. jQuery, (2023). What is jQuery?. [Online] Available at: https://jquery.com/ [Accessed 2023].
7. MDN, (2023). HTML: HyperText Markup Language. [Online] Available at: https://developer.mozilla.org/en-US/docs/Web/HTML [Accessed 01 Март 2023].
8. MDN, (2023). JavaScript. [Online] Available at: https://developer.mozilla.org/en-US/docs/Web/JavaScript [Accessed 20 Март 2023].
9. MDN, (2023). Understanding client-side JavaScript frameworks. [Online] Available at: https://developer.mozilla.org/en-US/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks [Accessed 20 Март 2023].
10. Meta Open Source, (2023). Describing the UI. [Online] Available at: https://react.dev/learn/describing-the-ui [Accessed 21 Март 2023].
11. Meta Open Source, (2023). Installation. [Online] Available at: https://react.dev/learn/installation [Accessed 17 Март 2023].

12. Persson, M. (2020), JavaScript DOM Manipulation Performance : Comparing Vanilla JavaScript and Leading JavaScript Front-end Frameworks (Dissertation). Available at: https://urn.kb.se/resolve?urn=urn:nbn:se:bth-19531 [Accessed 2023].
13. Sakovich, N., (2023). Top Most Popular Frontend Frameworks 2023. [Online] Available at: https://www.sam-solutions.com/blog/best-frontend-framework/ [Accessed 20 март 2023].
14. Sulova, S., (2018). Web saitove – planirane, proektirane i tehnologii za sazdavane. In: Internet tehnologii. Varna: Univ. izd. Nauka i ikonomika, pp. 103-135.
15. Varbanov, R., Vasilev, S., Petrov, P. & Nacheva, R., (2014). WEB tehnologii. Varna: Univ. izd. Nauka i ikonomika.
16. Vue, (2023). Introduction. [Online] Available at: https://vuejs.org/guide/introduction.html [Accessed 2023].