УДК 004.738.5

M. V. Lavrov

FRONTEND VS BACKEND: UNDERSTANDING THE DIFFERENCE AND THEIR INTERACTION IN WEB DEVELOPMENT

Вінницький національний технічний університет

Анотація

У роботі досліджено фундаментальні відмінності між фронтенд та бекенд розробкою, їхні технологічні стеки та архітектурні особливості у сучасній веб-розробці. Розглянуто роль клієнтської частини як інтерфейсу взаємодії з користувачем та серверної частин як основи логіки та зберігання даних. Показано ключову роль API у забезпеченні комунікації між фронтендом і бекендом через механізм запит-відповідь. Детально проаналізовано принципи клієнт-серверної архітектури, сучасні фреймворки та інструменти розробки, а також перспективи кар'єрного зростання для спеціалістів обох напрямків.

Ключові слова: фронтенд, бекенд, веб-розробка, інтерфейс користувача, серверна логіка, зберігання даних, бази даних.

Abstract

The work investigates the fundamental differences between frontend and backend development, their technology stacks, and architectural features in modern web development. The role of the client side (HTML, CSS, JavaScript) as a user interaction interface and the server side (programming languages, databases) as the foundation of logic and data storage is considered. The key role of API in ensuring communication between frontend and backend through the request-response mechanism is shown. The principles of client-server architecture, modern frameworks and development tools, as well as career prospects for specialists in both fields are analyzed in detail.

Keywords: frontend, backend, web development, user interface, server logic, data storage, databases.

Introduction

Modern web applications function through the interaction of multiple technological components operating on both the client and server sides. When users log into social networks, order food through online delivery services, or watch video content on streaming platforms, complex processes occur that involve both visible and hidden system components. In information technology, these components are commonly referred to as frontend and backend. Frontend represents the part of the application responsible for the user interface and interaction with the user, while backend provides data processing, application logic, and data storage on remote servers. Understanding the interaction between these components is fundamental for studying modern web development. The purpose of this study is to analyze the differences between frontend and backend development, examine their core technologies, and demonstrate how they interact through Application Programming Interfaces (APIs) to create modern web applications used by millions of users.

Research results

In the modern digital age, web applications have become an integral part of daily life. However, behind every button click or page load lies a complex interaction between two fundamentally different components: the frontend and the backend. Understanding how these components work and communicate is essential for anyone involved in software development [4].

Frontend development, often called "client-side" development, encompasses everything users experience directly in their web browsers [1]. It is the visual layer of an application – the colors, layouts, buttons, and interactive elements. The primary goal of frontend development is to create an intuitive and aesthetically pleasing user interface (UI) that provides a positive user experience (UX) [6]. The foundation of frontend development rests on three core technologies. HTML (HyperText Markup Language) serves as the structural skeleton of web pages, defining the hierarchy of content. CSS (Cascading Style Sheets) controls the visual styling – colors, typography, layout, and responsive behavior [7]. JavaScript adds dynamic behavior and interactivity, enabling features such as form validation, animated transitions, and real-time content updates. Beyond these fundamental languages, modern frontend development relies heavily on frameworks like React, Angular, and Vue.js, which streamline development and enhance functionality for complex applications [3]. Frontend developers must also consider cross-browser compatibility, performance optimization, and accessibility standards, ensuring that applications work for all users regardless of their device or abilities.

Backend development, or "server-side" development, focuses on the behind-the-scenes logic, data processing, and infrastructure that power web applications [4]. If the frontend is the visible exterior of a building, the backend is its foundation, electrical wiring, and plumbing – invisible but absolutely essential [1]. The backend architecture consists of several interconnected components. The server is a powerful computer

that hosts the application and responds to client requests. The application itself contains the business logic written in programming languages such as Python, Java, PHP, Ruby, or JavaScript (using Node.js). Databases store and organize all application data – relational databases (SQL) like MySQL and PostgreSQL structure data into tables, while non-relational databases (NoSQL) like MongoDB offer greater flexibility for evolving data structures [4]. The backend also manages user authentication, session handling, input validation, and protection against security threats such as SQL injection. Performance optimization through caching and load balancing falls under backend responsibility, as does integration with external services. Backend frameworks like Django, Express.js, Laravel, and Spring Boot accelerate development by providing pre-built components and established patterns [3].

The magic of web applications lies in the seamless communication between frontend and backend components. This interaction follows a structured pattern known as the request-response cycle [1]. The cycle begins with a user action in the browser – clicking a link or submitting a form. The frontend captures this action and formulates an HTTP request containing relevant information such as the desired resource and input data. This request travels across the internet to the server where the backend resides. Upon receiving the request, the backend processes it according to the application logic – querying databases, validating permissions, or performing calculations. After processing, the backend constructs an HTTP response and sends it back to the client. Modern applications increasingly transmit data in JSON (JavaScript Object Notation) format, which is lightweight and easily parsed by JavaScript. The frontend receives this response and immediately updates the user interface accordingly. Crucially, this entire exchange happens without requiring a full page reload, creating the smooth, app-like experience users expect. Application Programming Interfaces (APIs) serve as the formal contract governing frontend-backend communication. REST has become the dominant architectural style, organizing interactions around resources accessed through standard HTTP methods, while GraphQL offers an alternative where clients specify exactly what data they need [4].

Understanding the distinctions between frontend and backend development helps aspiring developers choose appropriate career paths [5]. Frontend developers typically possess strong visual sensibilities and empathy for end users, enjoying the immediate visual feedback of their work [2]. Backend developers tend to excel at logical thinking and system design, finding satisfaction in building robust, scalable systems that operate reliably under load. Full-stack development has emerged as a popular career option, combining competencies from both domains – full-stack developers understand the entire application lifecycle and can work independently on complete features, making them valuable team members particularly in startups and small organizations [2].

The distinction between frontend and backend continues evolving with technological advances. Server-side rendering frameworks like Next.js blur traditional boundaries by executing frontend code on servers. Edge computing pushes certain backend processing closer to users. Microservices architectures decompose backend functionality into smaller, specialized services [4]. These trends suggest that while fundamental concepts remain relevant, the practical implementation of client-server architecture will continue adapting to new requirements and capabilities.

The most successful applications result from effective collaboration between frontend and backend specialists who respect each other's expertise while working toward shared goals. By appreciating both the differences and the interdependence of these domains, developers can build systems that are not only technically excellent but also genuinely valuable to the people who use them [1].

Conclusion

Frontend and backend development represent two complementary halves of modern web applications, each with distinct responsibilities, technologies, and expertise requirements. The frontend creates engaging user experiences through visual design and interactive elements, while the backend ensures reliable data processing, security, and scalability. Their interaction through carefully designed APIs enables the seamless functionality that users have come to expect from digital services .

Understanding the interaction between frontend and backend components provides a critical conceptual foundation for further study and professional development in information technology. Specialization may focus on the creative, user-facing aspects of frontend development, the logical and systematic challenges of backend engineering, or the comprehensive perspective of full-stack development. Regardless of specialization, knowledge of how these components interact remains essential. As web technologies continue to evolve, the principles of client-server architecture will persist, adapting to new paradigms while maintaining the essential separation of concerns that ensures the functionality, scalability, and reliability of modern web applications.

REFERENCES

1. Романюк О. Н., Катєльніков Д. І., Косовець О. П. Веб-дизайн і комп'ютерна графіка : навчальний посібник. – Вінниця : ВНТУ, 2007. – 147 с.
2. Пасічник О. В., Пасічник В. В. Веб-дизайн : підручник. – 2-ге вид., стер. – Львів : Магнолія 2006, 2012. – 520 с. – (Серія «Комп'ютинг»).
3. Frontend and backend [Електронний ресурс] – Режим доступу: https://en.wikipedia.org/wiki/Frontend_and_backend
4. Intuit Blog. Front End vs Back End vs Full Stack Developers [Електронний ресурс] – Режим доступу: https://www.intuit.com/blog/innovative-thinking/front-end-vs-back-end-vs-full-stack-developers/
5. BrowserStack. Web Frameworks: All You Should Know About [Електронний ресурс] – Режим доступу: https://www.browserstack.com/guide/web-development-frameworks
6. AWS. Frontend та backend: різниця в розробці додатків [Електронний ресурс] – Режим доступу: https://aws.amazon.com/compare/the-difference-between-frontend-and-backend/
7. EPAM Campus. Career Test Results: Frontend Software Engineer [Електронний ресурс] – Режим доступу: https://campus.epam.ua/en/careertest/result/W1stMSwtNCwyLDUsMV0sWy0xLDAsLTEsLTIsLTJdLFstMSwtMywtNCwwLDBdLFswLC0xLDAsMSwyXSxbMSwtMywtMSwxLDBdXQ==

Лавров Микола Васильович - студент групи 1ПКТ-24б, факультету інтелектуальних інформаційних технологій та автоматизації, Вінницький національний технічний університет, м. Вінниця, Хмельницьке шосе, 95, e-mail: first.kreox@gmail.com

Науковий керівник: Кобилянська Ірина Василівна, канд. пед. наук, старший викладач кафедри іноземних мов, Вінницький національний технічний університет. E-mail: kobylianska@gmail.com

Lavrov Mykola Vasylovych - student of group 1PKT-24b, Faculty of Intellectual Information Technologies and Automation, Vinnytsia National Technical University, Vinnytsia, e-mail: first.kreox@gmail.com

Scientific supervisor: Kobylianska Iryna Vasylivna, PhD in Pedagogy, Senior Lecturer, Department of Foreign Languages, Vinnytsia National Technical University. E-mail: kobylianska@gmail.com