Development of a Responsive Web Application for Human Resource Management Using a NoSQL Database

2025 24th International Symposium INFOTEH-JAHORINA (INFOTEH) | 979-8-3315-1579-9/25/$31.00 ©2025 IEEE | DOI: 10.1109/INFOTEH64129.2025.10959255

Gorana Mišković
Dwelt d.o.o.
Banja Luka, BiH
gorana@dwelt.net

Dragoljub Krneta
Dwelt d.o.o.
Banja Luka, BiH
dragoljub@dwelt.net

***Abstract* - The topic of this paper is the development of a responsive web application for human resource management using a NoSQL database. The technical goal of the paper is to demonstrate the possibility of developing a modern three-tier web application with the option of using MongoDB, a NoSQL database management system. From an economic perspective, the goal is to support business processes and assist in most activities related to modern human resource management in companies or organizations, thus creating conditions for faster and higher-quality decision-making by management concerning human resource management.**

Keywords - web application, management, human resources, NoSQL, database.

I. INTRODUCTION

The development of a software application for human resource management represents one of the most significant steps in the modernization, optimization and improvement of business processes within organizations. Human resource management includes all aspects of employee management, employment planning, recruitment, training, evaluation, salary calculation, reporting and other tasks related to labor relations. Considering the increasing complexity and number of activities in human resources, many companies and organizations today recognize the importance of automation and digitization of these processes. Application software for human resource management enables companies to manage employee data more efficiently, improve productivity, reduce administrative tasks and provide a high degree of automation in human resource management.

In the second chapter of the paper, a brief overview of the basic concepts of human resource management, information systems in that area and software concepts for the implementation of information systems for human resource management is given. The third chapter briefly describes the stages of development and used technology, as well as the basic aspects related to database design, server-side application development and client-side application development. In the final chapter, the main advantages of the mentioned approach and directions for further research are listed.

II. SOFTWARE SUPPORT FOR HUMAN RESOURCE MANAGEMENT

A. Human resource management

The concept of human resources management can be interpreted in several ways, but it is best described through its characteristics in relation to the personnel department with which it is often identified. It should be noted that the personnel department usually refers to administrative tasks related to employees, such as, for example, payroll and other related operational tasks focused on ensuring the daily performance of employee management functions. On the other hand, human resource management is a much broader concept that includes certain aspects of strategic human resource management, so it could also be viewed as a kind of management philosophy. This scientific discipline developed from the fact that human resources together with their knowledge and abilities are the most valuable resource of the organization [1].

The main traditional activities of human resource management include all processes that enable effective management of human resources in the organization. The mentioned activities are job analysis and human resource planning, candidate recruitment, candidate selection, employee training, employee development, performance management, employee rewards and labor relations. On the other hand, contemporary human resource management activities relate to knowledge management, talent management and intellectual capital management. These activities relate to the development of talented individuals and the maximum use and increase of knowledge, its management in general, which further leads to the creation of intellectual capital [1].

B. Human resource management information systems

In the digital era, HR systems and technology in general have profoundly transformed various aspects of HR management. The integration of these technologies into its practices has changed and advanced the way organizations manage their workforce, streamline operations and improve the employee experience. The use of human resource information

systems has led to the automation of these processes, reducing administrative burden and minimizing human error [2].

Human Resource Information System (HRIS) could be defined as a tool that enables human resource management to perform its main activities more easily while increasing efficiency. This tool enables comprehensive management of all data on the organization's human resources [1].

A human resource management information system encompasses a wide range of applications and tools designed to support human resource management functions. By consolidating HR management data into a single, integrated platform, organizations can generate detailed reports and analytics that provide insight into workforce trends, performance metrics and employee satisfaction. This data-driven approach enables professionals to make better-informed decisions, align strategies with organizational goals and ultimately achieve better business results [3].

C. *Key components of the human resource management information system*

A human resource management information system is used to manage all aspects related to employees, such as recruiting, training, performance monitoring, payroll administration, benefits and more. The HR management system enables the automation, analysis and management of all key processes in human resource management. Key modules of these systems typically include recruitment planning, recruitment management, employee management, time management, performance management, employee relations management, payroll support and various reporting [1].

Software support for human resource management is implemented with several different concepts:

- as a subsystem, module or submodule of the organization's general software, such as ERP system
- as a stand-alone software system that provides support to most activities in human resource management
- as a separate software unit that supports only certain activities of human resource management (e.g. a system for recording working hours) [1]

ERP (Enterprise Resource Planning) is the infrastructure of business planning, execution and analysis. ERP systems are ready-made information systems that are oriented towards the informational support of most of the most common business processes. ERP brings together different departments or functions into a single unified system, which helps improve efficiency, reduce costs and provide better insights into business performance [4]. Common ERP components include accounting and financial management, human resources (HR), customer relationship management (CRM), procurement and supply chain management (SCM). Each component collects data from and supports the processes of its respective business department [5].

An HRIS, with all these key modules integrated, enables organizations to optimize HR processes, improve employee engagement, reduce errors and increase efficiency. By choosing the appropriate modules, the organization can adapt the system to specific needs and strategies.

The topic of this work is the development of an independent software system, i.e. a responsive web application, which is intended to support most activities in human resource management, but above all to demonstrate optimization and employee management efficiency through the introduction of software.

III. Development of a Web Application to Support Human Resource Management

The development of a responsive web application for human resource management took place through the following stages: analysis of business processes and user requirements, creation of a database model, development of the middle layer of the application, development of the user interface, application testing, verification and validation. In the phase of analyzing business processes and user requirements, the usual activities related to this phase took place: understanding and documenting existing business processes, workflows and organizational structures, as well as understanding and analyzing user requirements. The goal is to identify opportunities to improve efficiency, automate tasks, or reduce costs through new software. The output from this phase is a use case diagram and a sequence diagram. Application testing, verification and validation are key activities in the software system development process that ensure that the application meets user requirements and functions as expected. Each of these activities has a specific purpose and is applied during different stages of development. The phase of analyzing business processes and user requirements and the phase of application testing, verification and validation are briefly mentioned in this paper because the focus of the work is on the creation of the database model, the development of the middle layer of the application and the development of the user interface.

Most web applications that process data have a multi-tier or three-tier architecture. The three-tier application architecture used here has a client layer, an application logic layer and a database layer. In this case, the creation of a web application can be divided into database design, server-side programming (backend) and client-side programming (frontend).

The following technologies were used to develop the application:

- MongoDB, a database management system
- Node.js, a platform for server-side application development
- Angular, a platform for client-side application development

A. *Database design*

The development of data management systems led to the emergence of the NoSQL database, which significantly influenced the way of handling large amounts of unstructured and semi-structured data. The term NoSQL is also interpreted as non-SQL or not only SQL and implies non-relational

databases. Among them, MongoDB stands out as a leading solution due to its flexibility, scalability and performance [6]. The basic types of NoSQL databases, according to the model they use, are key/value, column-based databases, document-based databases and graph databases. The most significant representative of document-based databases is MongoDB [7]. Unlike the SQL databases, that consist of tables with rows and columns, the MongoDB architecture implies storage in the form of documents, which are flexible and like JSON objects. Each document contains a pair of fields and values. Values refer to standard data types. A set of several documents constitutes a collection [8]. SQL databases, by having primary and foreign keys in relations are great for quickly storing and retrieving data, but it should be noted that it also requires significant memory. SQL databases can be scaled vertically, but not horizontally, i.e. memory can be added but is again limited to the existing hardware. In contrast, non-relational databases do not require shared keys with available horizontal scaling, i.e. storage capabilities on multiple servers [9]. As it is possible to store data on multiple servers, latency is also reduced for users, regardless of where the data resides. Another advantage of the MongoDB is that there is no rigid schema, so it is possible to create and modify the data structure without restrictions. Such agile modeling enables and facilitates faster iterations in software development. In addition, the rich query language further differentiates it from traditional databases. This includes filtering, sorting, aggregation, etc. Also, indexing is available, which provides an additional level of efficiency, optimization and query speed [6]. However, when it comes to the choice between relational or non-relational databases, the answer lies in the nature of the requirements and the individual case. As the main difference is that the data is unstructured or semi-structured, this is the biggest determinant when choosing between these two approaches. Although it is possible to relate collections in MongoDB, there is nothing that enforces referential integrity unless it is done on the application layer, unlike in relational databases. When it comes to query complexity, despite the rich language of the MongoDB, the preference is still given to SQL databases because they offer complex joins, sub-queries and nesting that NoSQL does not. There are no limits when it comes to choosing between these two approaches, combinations of relational and non-relational models are also possible. This hybrid model offers great flexibility while providing read and write consistency and the same performance [10].

MongoDB was chosen for the database management system because of its flexibility for working with structured, semi-structured and unstructured data. Data used by information systems for human resource management, due to the nature of business processes, can come from structured, semi-structured and unstructured data sources. Structured data is data that is organized into clearly defined formats, usually in tabular form (e.g. databases). Semi-structured data is data that is not in a fully structured format, but still contains some form of tags, metadata, or organization that allows it to be partially organized and analyzed. They are usually in formats like XML, JSON, or email. Unstructured data is data that does not have a predefined organization and is usually in a text format. This data is more difficult to analyze, but as technology advances, tools like artificial intelligence and machine learning can help extract and analyze it. Sources of structured data in human resource management can be:

- databases that contain data about employees (personal data, positions, employee seniority history, etc.),
- systems for recording working hours, data from other systems (calculation of salaries, sick leave, vacations, etc.).

Sources of semi-structured data can be:

- e-mail (communication between managers and employees, requests for training or training information, etc.),
- employee performance evaluations that are often written in semi-structured formats (e.g. open responses in rating systems, surveys),
- chat communication (data from team chat platforms that may contain important information, but are not completely organized in a structured format) and others

Sources of unstructured data can be:

- resources on social networks,
- notes or transcripts from interviews with employees or candidates, where the data is not structured,
- videos (training, meetings or interviews),
- documents in Word, PDF files, Power Point slides, unstructured data that comes in the form of open answers on surveys or during informal conversations

The database scheme serves as a visual representation of the tables themselves, data collections in this case and the way they are organized. Thus, each data collection reflects a specific application module, and its fields correspond to the data that individual records of a collection can contain. Accordingly, the database consists of ten collections:

- departments, representing departments inside the organization itself
- positions, with the relation to the department to track employee positions in each department
- projects, containing data such as start date and end date
- employees, employee data with ability to track employees per project, achieved by establishing foreign key
- absences, refers to data related to absence types, for example sick leave
- attendances, containing data to enable tracking of attendance types such as full-time and others

- attendance records, to store and manipulate data related to each employee's attendance or absence at a certain time
- users, collection for authentication purposes
- permissions, collection for authorization purposes

Therefore, the scheme describes the relations between the tables like in the case of a relational database. However, as MongoDB was used as a DBMS of non-relational databases, its flexibility is indicated by the possibility of creating relations and thus creating a relational database. It is important to note that preservation of referential integrity is possible only through the application layer and that there is no schema display functionality, so the DbSchema application was used to display the database schema. Figure 1 shows part of the database schema.

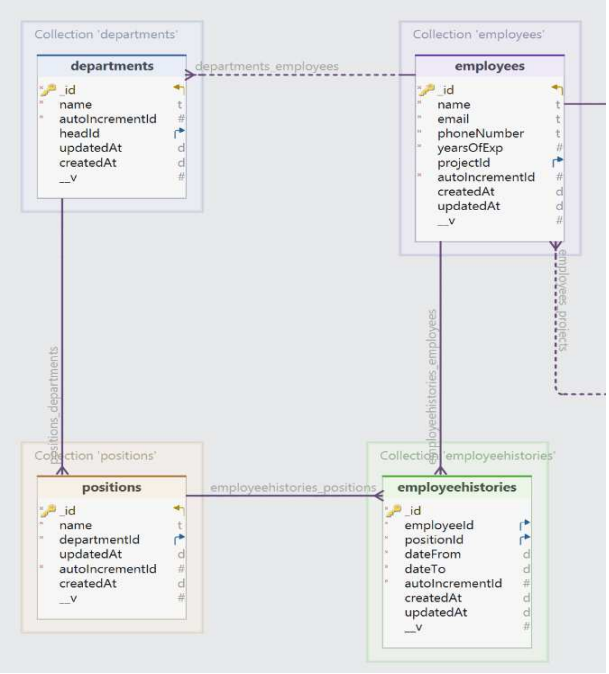

Figure 1. Part of the database schema

B. Server-side application development

There are numerous technologies used for the purpose of server-side application programming. One of them is Node.js, which allows JavaScript code to run in the background, outside of the browser. It is a free platform that uses Google's V8 virtual machine for the JavaScript language. In addition to the so-called runtime environment, Node.js also contains many useful modules which speed up development because it is not necessary to write code from scratch [11].

Node.js is distinguished by the speed of processing requests and asynchrony, which allows the processing of multiple requests at the same time. Also, it has a good mechanism for handling errors, where errors are already visible now of code execution. Another advantage is that it is free and rich in modules, thus enabling the development of high-performance applications and it is increasingly used for the creation of microservices [12].

The architecture of the server side of the application, developed with the help of Node.js technology, consists of controllers, models and services, which is shown in Figure 2.

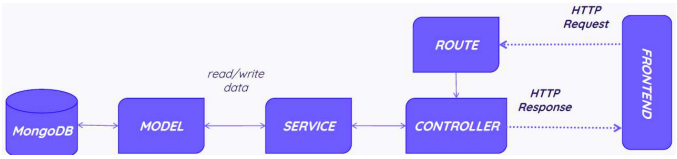

Figure 2. Backend architecture of the application

The server side of the application provides the necessary data to the user interface. This process is done with the help of HTTP requests. The model is based on a scheme and is defined with the help of Mongoose, an object-oriented JavaScript library, which is also responsible for the connection with the database, more precisely the one between MongoDB and the Node.js execution environment. Each model corresponds to the collection fields in the database.

The purpose of the model is to preserve the specific structure of the collection, as well as to use the collection within the code. In this way, it is possible to perform certain operations on the model or collection. This means that every request from the client to the server, which involves retrieving data from the database, involves executing a specific query over a specific collection. For this purpose, services are being developed that are responsible for processing requests and retrieving certain data or changing it. However, since there are certain rules for retrieving data, it implies that the request itself coming from the client must respect the given rules. For example, if data about a specific record is being retrieved, it is necessary for the client to explicitly state which record it is, by sending the unique identifier of that record as part of the request. Otherwise, the service itself will not be able to perform an operation on the model, so it is necessary to control these cases. This is precisely why controllers are important, i.e. to ensure the validity of the request itself before it even reaches the service. Figure 3 shows an example of a controller from the backend code of an application software for human resource management.

employeeController.js

```
const getAllEmployees = async (req, res) => {
    if (req.query.yearsOfExp)
        req.query.yearsOfExp = parseInt(req.query.yearsOfExp);

    try {
        const employees = await EmployeeService.getAllEmployees(req);
        return res.status(200).json({ status: 200, data: employees });
    } catch (e) {
        return res.status(400).json({ status: 400, message: e.message });
    }
};
```

Figure 3. An example of a backend controller

C. Client-side application development

The user interface of the application was developed with the help of the TypeScript language and the Angular framework.

Angular is a popular frontend framework, a framework for creating a user interface, developed by Google. This framework has a significant role in the modern development of three-layer applications, especially when it comes to the so-called MEAN stack (MongoDB, Express.js, Angular, Node.js) [13].

Angular is used to develop modular, interactive and dynamic applications. It is responsible for only displaying the user interface but also managing complex logic on the client side. Unlike Node.js technology, this technology uses Typescript language instead of JavaScript. The code itself is based on building components, services, directives and similar. The components are responsible for the content that the user sees, and each component is modular, which means it can be used again and again, but also combined with other components. In this way, larger parts of the application are developed, and modularity brings benefits for large applications in terms of code separation and code reuse. Fundamentally, services are the ones responsible for managing the logic, they can also be shared between components, and they are usually in charge of HTTP requests or e.g. authentication. So, in addition to components and services, there are also directives that represent classes that add certain behavior to elements such as forms, lists, styles and similar. Angular works seamlessly with backend technologies from the MEAN stack and its modularity and robust form handling make it popular in Web application development [13].

A similar architecture as well as at the backend of the application is applied in this part of the application code, so models, controllers and services can also be found. Redefining models ensures their consistency in client-server and server-database communication. Using Typescript instead of JavaScript further reduces the possibility of errors because data typing is provided, unlike the backend code where it is provided with the help of Mongoose. As in the backend part of the code, there are services on the client side, with the client-side services reflecting HTTP requests, which use models to define the return or input form of data, which is illustrated in Figure 4.

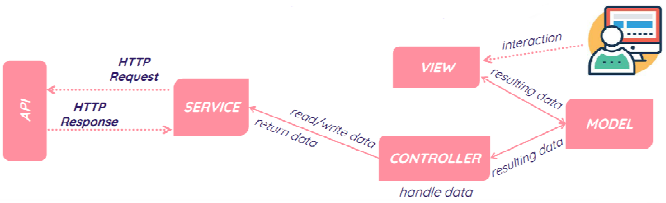

Figure 4. Client-side application

Given that it is necessary to ensure the validity of each request on the user interface side, it is handled in a similar way in this part of the code. Before each HTTP request, for example, calling a particular service, the validity and existence of the parameters expected by the backend and the database itself, is checked. Controllers are also responsible for that. Unlike the backend, on the client-side there are also visual representations in the form of components that reflect the model, so e.g. we have a view in the form of an add or edit form. Internet browsers convert the next part of the code, shown in Figure 5, into a form with appropriate fields visible to the user and in this way ensure user interaction with the application, i.e. its other layers.

```
<mat-toolbar class="mat-elevation-z1">
    <mat-toolbar-row>
        <span class="fs-6 fw-normal text-muted">{{isEditMode ? 'Edit employee history' : 'Add employee history'}}</span>
    </mat-toolbar-row>
</mat-toolbar>
<mat-card>
    <mat-card-content>
        <form *ngIf="myForm && data$ | async as data" [formGroup]="myForm" (ngSubmit)="onSubmit()">
            <div fxLayout="row" fxLayoutGap="10px">
                <autocomplete
                    label="Employee"
                    fxFlex.gt-sm="30"
                    [searchControl]="headControl"
                    [required]="true"
                    url="/employees/lookup"
                ></autocomplete>
                <autocomplete
                    label="Position"
                    fxFlex.gt-sm="30"
                    [searchControl]="positionControl"
                    [required]="true"
                    url="/positions/lookup"
                ></autocomplete>
                <mat-form-field fxFlex.gt-sm="30">
                    <input matInput readonly formControlName="dateFrom" [matDatepicker]="from" placeholder="Date from">
                    <mat-datepicker-toggle matIconSuffix [for]="from"></mat-datepicker-toggle>
                    <mat-datepicker #from></mat-datepicker>
                </mat-form-field>
                <mat-form-field fxFlex.gt-sm="30">
                    <input matInput readonly formControlName="dateTo" [matDatepicker]="to" placeholder="Date to">
                    <mat-datepicker-toggle matIconSuffix [for]="to"></mat-datepicker-toggle>
                    <mat-datepicker #to></mat-datepicker>
                </mat-form-field>
            </div>
            <div fxLayout="row" fxLayoutGap="10px">
                <button
                    mat-raised-button
                    fxFlex.gt-sm="10"
                    fxFlexOffset="90"
                    class="mt-2 submit-btn"
                    type="submit"
                >{{isEditMode ? 'Edit' : 'Add'}}</button>
            </div>
        </form>
    </mat-card-content>
</mat-card>
```

Figure 5. Example code for a user interface form

D. Application modules

The application consists of several program units: employees, employee history, sectors, projects, attendance and absences. The employee form enables creating records of employees through their entry into the system, editing, deletion, as well as table views and detail forms. Tabular view allows filtering and sorting of data.

As can be seen on this form, each employee can be a part of a specific project team, i.e. assigned to a specific project. Hence the need for records of the organization's projects themselves and the storage of project data.

Figure 6 shows the Employee web form of the human resource management web application.

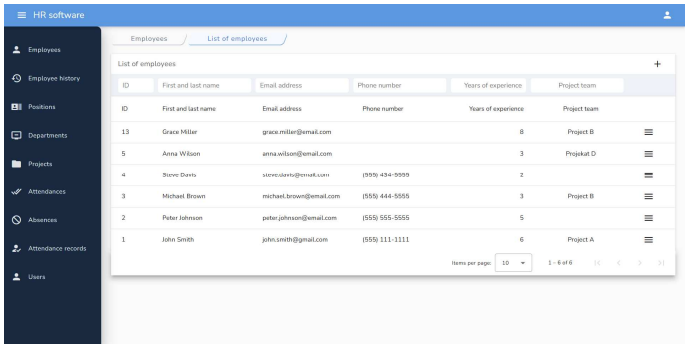

Figure 6. Employee form

Responsiveness of a web application refers to the ability of the application to automatically adapt to different screen sizes and devices, such as desktop computers, tablets and mobile phones. The main goal of responsive design is to provide users with consistent and optimized experience, regardless of the

device they use to access the website or application. Work on a specific device can be applied to application forms where it makes sense, e.g. intensive input is not suitable for mobile phones, while certain views are suitable for working on smaller screen sizes. Figure 7 shows the Position form displayed on a mobile phone with the iOS operating system.

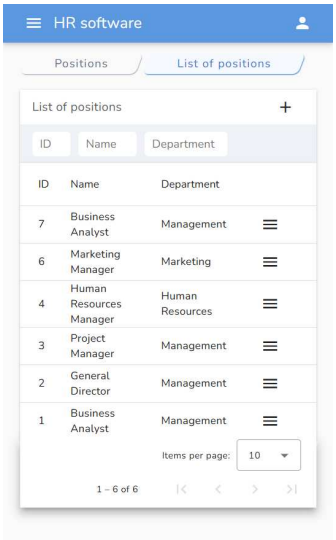

Figure 7. Position form displayed on a mobile phone

Responsive web applications have become a standard in web design because they enable wide accessibility and improve the user experience, making the application easier to use on any device.

The security aspects of the application are reflected through the implemented processes of authentication (the process of identifying and verifying the user of the application) and authorization (the process of determining the permissions of the identified user of the application).

IV. CONCLUSION

This paper offers one approach in the development of a software application for the management of human resources in companies and organizations. The main advantages of the proposed approach are:

- using a NoSQL database with the ability to work with structured, semi-structured and unstructured data sources
- responsiveness, i.e. the ability to work on different devices (computers, tablets, mobile phones)
- the application was developed on the latest technologies
- the application supports most activities in the field of human resource management
- the application can function independently of the ERP system

Developed web applications for human resource management in organizations can play a key role in optimizing the employee management process, improving efficiency and reducing the administrative burden. In addition, the application helps centralize and automate many key processes related to recruitment, training, business processes and employee evaluation. Using such applications contributes to increasing efficiency, reducing errors and improving employee satisfaction. The contribution of the paper from a technical perspective is to demonstrate the possibility of using a NoSQL database management system, using data of different structures, in the development of a responsive web application for human resources management. From an economic perspective, the contribution of the paper is to support business processes and assist in most activities related to modern human resource management in companies or organizations, thus creating conditions for faster and higher-quality decision-making by management concerning human resource management. Future work on the application implies the creation of a web service to connect it with other software systems within the company or organization.

LITERATURE

[1] B. Zolak Poljašević, Menadžment ljudskih resursa: Tradicionalni u susret savremenom pristupu. Univerzitet u Banjoj Luci. Banja Luka. 2021.

[2] T. Torres-Coronas and M. Arias-Oliva, E-Human resources management: managing knowledge people. Hershey, PA: Idea Group Publishing, 2005.

[3] T. Torres-Coronas and M. Arias-Oliva, Encyclopedia of human resources information systems: challenges in e-HRM. Hershey, PA: IGI Global. in 2009

[4] D. Krneta, Z. Marjanovic, Possibility Integration of Methodological Approaches Information Systems Development and Implementation ERP Solutions, INFOTEH-JAHORINA Vol. 10, Ref. E-I-7, p. 423-427, March 2011.

[5] https://www.netsuite.com/portal/resource/articles/erp/erp-systems-components.shtml [Date of access: 15 September 2024]

[6] P. Membrey, E. Plugge, and T. Hawkins, The Definitive Guide to MongoDB: The NoSQL Database for Cloud and Desktop Computing. New York, NY: Apress, 2010.

[7] D. Krneta, Baze podataka, drugo izmijenjeno i dopunjeno izdanje. My Book, Banja Luka. 2023.

[8] MongoDB. What is NoSQL? Available at: https://www.mongodb.com/resources/basics/databases/nosljl-explained [Date accessed: 18 September 2024]

[9] IBM. What is NoSQL database? Available at: https://www.ibm.com/topics/nosql-databases [Date of access: 18 September 2024]

[10] Oracle. What is NoSQL database? Available at: https://www.oracle.com/database/nosql/what-is-nosql [Date of access: 18 September 2024]

[11] M. Kiessling, The Node Beginner Book. Lulu Press, Raleigh, NC, 2012.

[12] Geeks for geeks. Pros and Cons of Node.JS in Web Development. Available at: https://www.geeksforgeeks.org/the-pros-and-cons-of-node-js-in-web-development [Date of access: 18 September 2024]

[13] P. Aluyege, MongoDB, Express, Angular and Node.js Fundamentals. Birmingham, UK: Packt Publishing, 2019.