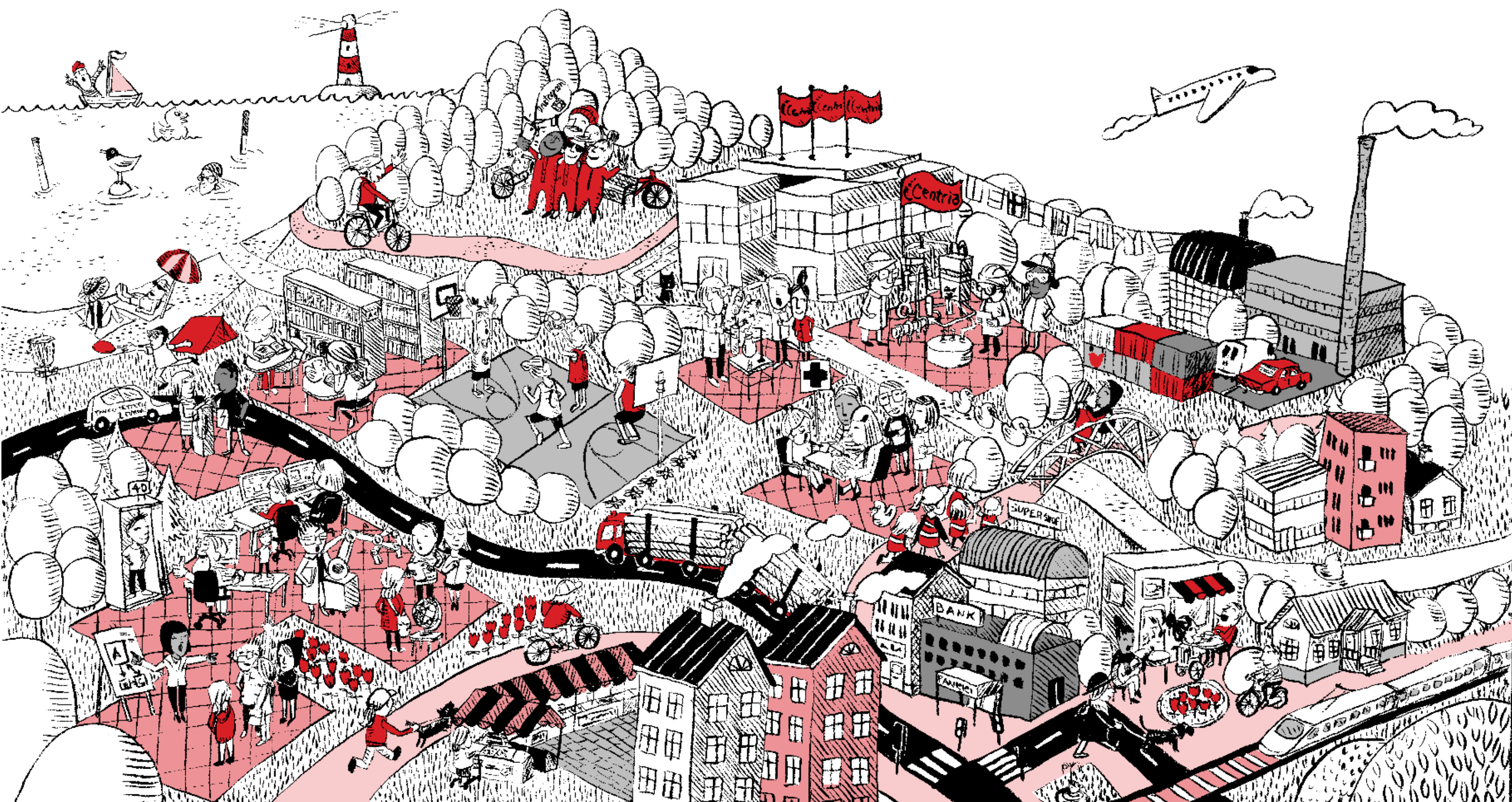
Vladimir Akhrem

JOB SEARCHING WEB SERVICE

REST API in React application

Thesis
CENTRIA UNIVERSITY OF APPLIED SCIENCES
Bachelor of engineering in information technology
January 2022



ABSTRACT

Centria University of Applied Sciences	**Date** 15 January 2022	**Author** Vladimir Akhrem
Degree programme Bachelor of engineering in information technology		
Name of thesis JOB SEARCHING WEB SERVICE. REST API in React application.		
Centria supervisor Jari Isohanni		**Pages** 30
Instructor representing commissioning institution or company Jari Isohanni		

The purpose of this thesis was to create a demo version of a web service using the React JavaScript library to find current vacancies for people who are looking for work. When writing the application, the REST API was used to connect to an external web service of the Russian job search site to obtain data on vacant vacancies to present them in the eazySearch application.

The reader is presented with a theoretical background for a general understanding of what works and how. This part describes the idea of web services as a whole and talks about their architecture as well as about RESTful interfaces, describing the architecture and features of REST, APIs, and CRUD methods. When a request is sent to the server, the response comes in JSON format – this is in more detail in the theoretical part as well as the JSON data structure. In addition to everything, at the end of the theoretical section, information is provided about the main figure of the thesis work – React, which is a JavaScript library for writing web applications with ready-made code.

In the practical part, the reader is presented with the process of creating a web application, which tells about the architecture of web applications. Before writing the service, use cases were created, as well as the diagram of it, which can be seen in the practical part. The description of each use case is presented after the diagram. There is also a diagram of the architecture of the easySearch application with a description of the work, where a REST request is sent using a specific URL, after which a REST response returns in JSON format. Moreover, the result of the work is presented, as well as a detailed description of the user interface and its design - what features does the interface have and what and how can the user interact with and what will it lead to. In addition, the technologies and tools that were used in the development of the web service are described for review. Finally, the conclusion is presented together with ideas for further development of the application, as well as with completion plans.

Key words
React JS, JavaScript, CSS, HTML, JSON, REST API, Front-end, URL

CONCEPT DEFINITIONS

AJAX

Asynchronous Javascript and Xml. This is the technology of accessing the server without reloading the page.

API

Application programming interface. This is a set of ways and rules by which different programs communicate with each other and exchange data.

CRUD

Create, Read, Update, Delete. These are the basic methods of working with databases. CRUD operations are designed to edit program data.

DOM

Document Object Model. It is an interface through which programs can work with the content, structure, and styles of a web page. Simply put, it is a set of methods that can be called and properties that can be accessed.

HTML

HyperText Markup Language. It is a standardized document markup language for web browsing in a browser.

HTTP

HyperText Transfer Protocol. A widespread data transfer protocol, originally intended for the transfer of hypertext documents, that is, documents that may contain links that allow to organize the transition to other documents.

JSON

JavaScript Object Notation. It is a format that stores structured information and is mainly used for data transfer between the server and the client.

JSX
JavaScript XML. It is an extension of the JavaScript syntax that allows to use HTML-like syntax to describe the structure of the interface.

REST
Representational State Transfer. This is a set of rules for how a programmer can organize the writing of server application code so that all systems can easily exchange data and the application can be scaled.

RFC
Request for Comments. This is a document from a series of numbered Internet information documents containing technical specifications and standards widely used on the World Wide Web.

RPC
Remote Procedure Call. It is a protocol for requesting services from a remote computer program over a network.

SOAP
Simple Object Access Protocol. It is a protocol for exchanging structured messages in a distributed computing environment.

TCP/IP
Transmission Control Protocol/Internet Protocol. This is a set of protocols that sets standards for communication between computers and contains detailed routing and interconnection agreements. The model describes the way data is transmitted from the source of information to the recipient.

UDDI
Universal Description Discovery & Integration. It is a tool for locating descriptions of web services (WSDL) for subsequent search by other organizations and integration into their systems.

URL
Uniform Resource Locator. This a system of unified addresses of electronic resources, or a uniform determinant of the location of a resource (file).

Use Cases

In software development and system design, it is a description of the behavior of a system when it interacts with someone or something from the external environment.

WSDL

Web Services Description Language. It is a web services description language based on XML. It describes methods, input and output data structures, data types, network addresses for accessing the service, and more.

XML

eXtensible Markup Language. It is a programming language for creating a logical structure of data, storing, and transmitting it in a form convenient for both a computer and a person.

ABSTRACT
CONCEPT DEFINITIONS
CONTENTS

1 INTRODUCTION

The Internet as known has changed a lot during the recent years, as have the numerous websites that it consists of. On the other hand, a network connecting several computers is a rather old idea that dates to the second half of the previous century, as a military idea. Later, highly specialized local networks appeared, where the scientific community exchanged thematic results of the latest research and development. However, it was too early to talk about a global network that would cover the whole world and allow millions of gadgets to be connected simultaneously. The key moment in the development of the Internet can briefly be considered the creation of two TCP/IP protocols that allow to easily connect to the Internet using a telephone line. A few years later, the Internet was becoming ever more publicly available, and since the early 90s, modern history began in the form in which the Internet is familiar to people now. (Independed Media 2022; Internet Society 2022.)

The first computer networks — military and civilian - preceded the first websites for decades. However, websites also need to be paid attention to — after all, it is thanks to them that there is an opportunity to freely communicate and receive information from all over the world. Today's web pages, as well as the underlying protocols, markup, and coding languages, force the data transmitted by links to take an accessible form. This makes the information understandable to the end-user. (Independed Media 2022; Internet Society 2022.)

With the expansion of the Internet, the possibilities have also expanded. The Internet provides users not only with the search for the necessary information, but also other services and resources that have long become an integral part of modern life. Nowadays there are many different websites and web applications that make our life easier. (Independed Media 2022; Internet Society 2022.)

This thesis focuses on a web application that was made by using React.js. The objective of this thesis was to develop a fast and efficient job seeking service with minimal requests for the client. The work is divided into two sections. The first section covers the theoretical part, which discusses web service architecture as well as restful interfaces. The topic of JSON data structure will be touched also. Moreover, the library React.js and its features will be described at the end of the theoretical part. The second section is practical and is about designing a job seeking app. It describes the architecture and what tools have been used for the creation of this service. More than that, a practical section is said about connecting to external web services, and the result will be demonstrated.

2 THEORETICAL BACKGROUND

This part introduces and makes it clear what the architecture of a web service is, describes what web services are, what features they have and what types there are, as well as how this system works. Moreover, the topic of data transmission and which protocols are used in this process is touched upon. In addition to everything, a few words are said about the development of a web service, how they can function. Furthermore, the reader will be able to learn about the REST API – what is an API and why REST, what it is in general, and what architecture it is built on. Equally important information about the difference between REST and another method is presented in the theoretical part, as well as the characteristics on which this architecture is based. Moreover, the chapter presents limitations and if the application meets them, then it has certain advantages, which are indicated as well as cases where it is recommended to use the REST API. In addition, the next section is about the JSON data structure, describing the basics of this format, what values JSON has and what the file is, what methods are used to work with it to send a converted JavaScript object over the network or to receive converted data in JSON format. More than that, the paragraph represents the General Rules for creating a JSON file and presents its advantages. The final section is about the JavaScript library – React. The reader is provided with information about what kind of tool it is and what developers use it for, what is the difference between a library and a framework and what functions this library has. Advantages of React is mentioned in this paragraph, as well as the distinctive features. It is time to go deeper.

2.1 General web service architecture

First, the web services are a composition of technologies. And like any other technology, they have a well-defined application environment. If we look at web services in the context of the network protocol stack, we will see that this, in the classical case, is nothing more than another add-on on top of the HTTP protocol. On the other hand, if we hypothetically divide the Internet into several layers, we will be able to distinguish at least two conceptual types of applications - computing nodes that implement non-trivial functions and applied web resources. At the same time, the latter are often interested in the services of the former. But the Internet itself is heterogeneous, which means that different applications on different network nodes operate on different hardware and software platforms and use different technologies and languages. Web services were invented to connect all this and enable one application to exchange data with others. (Walker 2021; Lozhko 2021.)

In fact, web services are the implementation of clear data exchange interfaces between various applications that are written not only in different languages, but also distributed on different network nodes. In addition, a web service can be designated as a software module designed to perform a specific set of tasks. The structural elements of the applications can be found on the network and can also be called accordingly. When called, the web service will be able to provide functionality to the client that requests this portal. This happens within seconds. (Walker 2021; Lozhko 2021.)

The system operates according to the following principle. The client makes a series of calls to the web service using requests to the server where the real web service is located. The requests themselves are made via Remote Procedure Calls (RPC). The data that is transmitted between the client and the server is the main component of the web service - XML. It is an extensible markup language that is like HTML and is easily understandable for an intermediate language. Therefore, when applications communicate with each other, they communicate in XML. For applications written in different languages, this provides a common platform for communication. (Walker 2021; Lozhko 2021.)

When transferring data between applications, web services use a Simple Object Access Protocol (SOAP). The data is sent via normal HTTP. SOAP messages are data that is transmitted from a web service to an application and these messages are nothing but an XML document. Given the fact that the document is written in XML, the application calling the web service can be written in any programming language. (Walker 2021; Lozhko 2021.)

There are basically two types of web services – SOAP and RESTful. For the developed environment to be fully functional, it is necessary to have certain components. These components must be present regardless of which development language is used to program a particular request. It is possible to create a web service yourself. To do this, a person needs to create a portal to host it and install a working programming environment. (Walker 2021; Lozhko 2021.)

The use of web services depends on the technologies used. With their help, the necessary conversion and configuration of outgoing and streaming data transmitted by the system takes place. SOAP is known as a version- and system-independent messaging protocol. It is based on the transmission of XML data in the form of encoded messages, where the message contains an XML document and the structure of the XML document follows a certain template, but not the content. The best part of web services and SOAP is that they are all sent over HTTP, which is a standard web protocol. A SOAP message contains a root element known as `<Envelope>`, which is the first element in an XML document. `<Enve-lope>` is divided into two parts. The first is a header that contains routing data. This data is information about the client to whom the XML document should be sent. The second part of the `<Envelope>` is the body that contains the actual message. (Walker 2021; Lozhko 2021.)

The client calling the service needs to know where its protocol is. In addition, the client application needs to know what a particular service is doing so that it can call the correct web service. This is done using WSDL, known as the Web services description language. The WSDL file is again an XML-based file that basically tells the client application what the web service is doing. Using the WSDL document, the client application will be able to understand where the web service is located and how it can be used. (Walker 2021; Lozhko 2021.)

As for universal description, discovery, and integration, UDDI is a standard for describing, publishing, and discovering web services provided by a specific service provider, in other words, it provides a specification that helps to post information about web services. UDDI provides a repository where WSDL files can be placed for the client application so that it understands the various operations that the web service provides. Thus, the client application will have full access to UDDI, which functions as a database containing WSDL files. (Walker 2021; Lozhko 2021.)

The development of web services is a simple task, but quite voluminous. It is necessary to develop own environment, where all working protocols will be uploaded in the future, as well as connect databases. Every framework needs architecture to make sure that the whole portal works as it should. Web services also have an architecture in which three instances can be distinguished. The first is a provider that creates a web service and makes it available for the client application to use. The second role is played by the requesting party. This is a kind of client application that needs to contact the web service. And it does not matter what language application it is. An application that is looking for operability through a web service can be written in any programming language. The third instance is the broker. A broker is an application that shares access to UDDI, where UDDI in turn helps the client application find a web service. (Walker 2021; Lozhko 2021.)

Applications will not be able to function without local and virtual databases. Web services allow to use protocols and requests to process and receive information from all media. Administrators can connect directories themselves and form new directions. There are three main types of requests for work and interaction between systems and applications. The first type is publishing, which is an action that makes the service available to the client. The provider makes it clear about the existence of the web service to the broker. "Find" is the second type of request, where the requester consults with the broker to find a web service. And binding is the third type. The requester can call the web service after receiving information from the web service that was received from the broker at the time. All incoming information in the system is redirected to the root directories. Every operation in the network is registered during the operation of the protocol. The description of web services on each specific portal makes it possible to quickly integrate and connect the necessary source of information. (Walker 2021; Lozhko 2021.)

It is important to note the following special characteristics of web services. Web services use XML to transmit data. This eliminates any dependencies on networks and OP, since it is a common language for everyone. Moreover, web services have a weakly connected system between the client and the web service. Using such an architecture makes software systems more manageable and simplifies integration between different systems. This means that the application should not change the way the web service is called if it changes over time. It is also worth noting that web services allow clients to call procedures for remote objects using an XML-based protocol, where remote procedures provide input and output parameters that the webservice must support. In addition to everything, the characteristics include synchronous or asynchronous functionality. The first one talks about linking the client to the execution of the service. In other words, the client will wait for the completion of the operation by the web service. The asynchronous operations allow parallel execution of other functions by the client after calling the service. This functionality is one of the most common and preferred methods that allows other web services to continue working while a certain operation is being performed. Web services also have support for document exchange. XML can represent not only data, but also complex documents, which is one of the key features. The listed parameters describe in detail the structure and properties of services on the Internet. With their help, application developers and portal owners can integrate various operation scenarios while using different languages. (Walker 2021; Lozhko 2021.)

2.2 Restful interfaces

Let is begin with the fact that the REST API is a way for websites and web applications to interact with the server. It is also called RESTful. The RESTful web service is built on the REST architecture and provides the client, who calls it API from the application in a secure way without saving state. The client can perform predefined operations using the Restful service. The REST API term consists of two abbreviations. API (Application Programming Interface) is code that allows two applications to exchange data from the server and REST (Representative State Transfer) is a way to create an API using the HTTP protocol. (Walker 2021.)

The REST API is used wherever the user of a website or web application needs to provide data from the server. These can be important documents, videos, or images. If a web browser as a client needs any of these resources, it must send a request to the server to have access to this very data, and here REST determines how to access them. For example, by clicking the video icon on the video hosting, the REST API performs operations and launches the video from the server in the browser. Currently, this is the most common way to organize an API. RESTful does not have a single standard of work, it is called an "architectural style" for server operations. (Walker 2021.)

As mentioned earlier, RESTful itself is not a standard or protocol. Developers are guided by the principles of REST API to create effective work from servers for their websites and applications. The principles allow to build a server architecture using other protocols: HTTP, URL, JSON and XML.
This distinguishes the REST API from the Simple Object Access Protocol (SOAP) method created by Microsoft in 1998. In SOAP, interaction for each protocol needs to be prescribed separately only in XML format. Also, in SOAP there is no caching of requests, more extensive documentation and implementation of a dictionary separate from HTTP. This makes the REST API style easier to implement than the SOAP standard. (Walker 2021.)

As for the architecture, the REST API is based on the HTTP Hypertext Transfer Protocol. This is a standard protocol on the Internet, created for the transmission of hypertext. Currently, any other types of data are sent using HTTP. Each object on the server in HTTP has its own unique URL in a strict sequential format. The REST API has four HTTP methods that are used for actions with objects on servers. One of the four methods – is GET, which is getting information about data or a list of objects. The next one is DELETE method, which is intended for deleting data. Another method called POST is used for adding or replace data. The last method in the list is PUT and it is needed for regular data update. Such requests are also called CRUD identifiers: create, read, update, delete. This is a standard set of actions for working with data. Each HTTP request has a header followed by a description of the object on the server — this is its state. (Walker 2021.)

The REST architecture is based on several characteristics. These characteristics are also known as design principles that must be followed when working with RESTful-based services. It is important to know that not every system whose components exchange data through request-responses is a RESTful system.

For the system to be considered RESTful, it must comply with the REST restrictions that are presented below. (Walker 2021.)

The first restriction is bringing the architecture to the client-server model. The basis of this limitation is the differentiation of needs. It is necessary to separate the needs of the client interface from the needs of the server storing the data. This limitation increases the portability of client code to other platforms and simplifying the server side improves the scalability of the system. The very differentiation into "client" and "server" allows them to develop independently of each other. (Walker 2021.)

The stateless is the next limitation. The REST architecture requires the following condition to be met. In the period between requests, the server does not need to store information about the client's status and vice versa. All requests from the client must be made so that the server receives all the necessary information to fulfil the request. Thus, both the server and the client can "understand" any received message without relying on previous messages. (Walker 2021.)

The system should be cacheable. Clients can cache server responses. Those, in turn, should have an explicit or implicit designation as cached or uncached, so that clients do not receive outdated or incorrect data in response to subsequent requests. Proper use of caching helps to eliminate some client-server interactions completely or partially, further increasing the performance and extensibility of the system. (Walker 2021.)

The fundamental requirements of the REST architecture include a unified, uniform interface. The client must always understand in what format, and to which addresses he needs to send a request, and the server, in turn, must also understand in what format he should respond to client requests. This unified format of client-server interaction, which describes what, where, in what form and how to send, is a unified interface. All data must be requested via a single URL by standard protocols, for example, HTTP. This simplifies the architecture of the site or application and makes interaction with the server clearer. (Walker 2021.)

Another restriction is Layered system. In this system, layers are understood as the hierarchical structure of networks. Sometimes the client can communicate directly with the server, and sometimes just with an intermediate node. The use of intermediate servers can increase scalability due to load balancing and distributed caching. For example, imagine some mobile application that is popular all over the world. Its integral part is downloading images. Since there are millions of users, one server could not withstand such a large load. Dividing the system into layers will solve this problem. The client will request an image from the intermediate node, the intermediate node will request an image from the server that is least loaded in the process and will return the image to the client. If caching is applied correctly here at each level of the hierarchy, then it is rea to achieve good scalability of the system. (Walker 2021.)

The restriction of providing a code on demand implies that the client can extend its functionality by downloading code from the server in the form of applets or scripts. Also Starting with the Null style takes place to be, where the client knows only one entry point to the server. Further opportunities for interaction are provided by the server. (Walker 2021.)

Applications that comply with all the above restrictions have advantages such as reliability, because there is no need to save information about the client's status, which may be lost. Also, performance due to the use of cache, scalability, transparency of the interaction system, simplicity of interfaces, portability of components, ease of making changes and the ability to evolve, adapting to new requirements. (Walker 2021.)

The REST API is recommended to be used in cases where the bandwidth of the connection to the server is limited, also if there is a need to cache requests, in case the application or site will scale significantly, in addition, if the application or site uses AJAX. REST APIs are used more often than alternative methods, for example SOAP. In addition to websites and web applications, RESTful is used for cloud computing. (Walker 2021.)

2.3 JSON data structure

JSON (JavaScript Object Notation) is a data transfer format that is used when a web server and a browser interact. Before the creation of the JSON format, websites worked slowly. Each user request to the server required the updated HTML page to be re-sent to the browser. An AJAX request using the JSON format is executed in the background, so the page does not reload. Today, JSON is the standard for data transmission on the Internet. The basis of JSON is the syntax of JavaScript objects that are used in the language to represent data. Data in JSON format is stored as "key: the value" pairs and ordered lists. (Austin 2020.)

JSON values can be JSON objects, these are unordered sets of "key: the value" pairs, separated by commas. This also includes arrays — ordered collections of values enclosed in square brackets, values separated by commas. Integers or floating-point numbers, as well as the logical data type — true and false, will be added to the list of JSON values. In addition to everything else, this also includes strings — specified sequences of Unicode characters, character encoding standard, enclosed in two double quotes, and a Null value, which shows the lack of information. JSON is supported by most modern programming languages. It is available either by default or with the help of special libraries. (Austin 2020.)

A JSON file is a string. To work with data in this format, it is needed to use the methods of the global JSON object. To send a JavaScript object over the network, it must be converted to JSON (string). To do this, use the `stringify()` method, which takes an object as a parameter and returns a JSON string. To turn the data received in JSON format from the server into a JavaScript object, the `parse()` method is used. It works similarly to `stringify()`, taking a string as an argument and returning an object. (Austin 2020.)

The general rules for creating a JSON file describes the RFC standard, where data must be written in the form of "key: the value" pairs, also the data should be separated by commas. The object is inside curly brackets {}, and the array is inside square brackets []. There are other points. In JSON format, double quotes (“) are used, single quotes (‘) are not suitable. But quotes are not needed for any non-string values — numbers, arrays, Boolean values. One extra or missing comma or parenthesis may cause the JSON file to fail. Moreover, the JSON format does not support comments. Adding a comment will cause an error. The automatically generated JSON file should not contain errors, it is also better to check it with a validator. These JSON rules ensure the reliability and speed of the implementation of the encoding and reading algorithm. (Austin 2020.)

If talk about the advantages of the JSON format, it has convenient and fast-to-use methods designed for converting, parsing a JSON string into a JavaScript object and back. Its data structure is simple and straightforward. However, this format has a very small size compared to other data formats (for example XML). This is because the JSON format contains the minimum possible formatting, it means that only a few special characters are used when writing it. This is a very important advantage, because the data presented in JSON format will load faster than if it were presented in other formats. Due to the fact that the JSON format has a lot of advantages, it has become used not only in JavaScript, but also in many other languages. (Austin 2020.)

If we compare JSON and XML, the JSON format has several advantages. When transmitting data, the JSON size will be significantly smaller than the XML size. JSON has more convenient conversion methods to JavaScript data structures than XML. It is also easier to create, storing and exporting data to JSON supports modern relational databases such as PostgreSQL and MySQL. (Austin 2020.)

Stored data can be organized as an object and an array. The base format is an object, as indicated by curly brackets. The object below (Code 1) has three pairs of keys and values separated by commas. The values in this example are strings, so they are enclosed in quotation marks, just like keys. An array is a way of storing data with a more complex structure. In the fourth key pair (Code 2), "hobby” is a key, and the value is an array in square brackets.

```
{
“firstName”:”Vlad”,
“lastName”:”Akhrem”,
“age”:”25”
}
```

Code 1. Example of JSON object_1

```
{
“firstName”:”Vlad”,
“lastName”:”Akhrem”,
“age”:”25”,
“hobby”:[“sport”,”programming”,”traveling”]
}
```

Code 2. Example of JSON object_2

Data in JSON format is transmitted from the server to the browser using APIs such as XMLHttpRequest or the more modern Fetch API. It works as follows. The user clicks on the title or product card and sends a request to the server. Next, the API generates a request using JavaScript, after which the server processes the received request and generates a response in the form of data in JSON format. At the end, after receiving data from the server, the browser uses JavaScript to wrap this data in HTML tags and refreshes the page in the background, without restarting. (Austin 2020.)

2.4 React.JS

Libraries and frameworks are one of the most frequently used tools for web developers. They help to create websites, native applications, and projects of other formats. Both tools are useful, but it will take a lot of time to study their features. There is a constant debate among programmers about what React is. React is a library written in JavaScript that is used to work with interfaces. (Skillcrush 2022; Meta Platforms 2022.)

Developers use React to create interfaces that can change content without reloading the page. Thanks to this, sites or native applications respond quickly to user actions. The products can be added to the cart without reloading the page or filling out forms without forwarding. React is considered the most popular library in the world written in JS. Its popularity is ensured by the fact that different tasks can be solved with the help of the component. Especially if additional tools are used that integrate into projects and open access to non-standard features. (Skillcrush 2022; Meta Platforms 2022.)

The difference between a framework and a library is as follows: imagine that a framework is the frame of a house, which can be made any way inside. In fact, it is a building material that gives complete freedom of action. The library assumes a different approach to construction. It must start almost from scratch and the set of materials is limited, but no one prevents us from going beyond and using other components. There are differences, so developers need to choose the right tools for a specific task. (Skillcrush 2022; Meta Platforms 2022.)

It has already been said that React is used for the user interface. With its help, for example, it is possible to write a plugin for a content management system and it will work well on the front-end. React often becomes the "core" of components for various content management systems. To change the source code, the component will first have to be disassembled, make changes to the code, and then pack it back. That is hopeless to change the source code on the fly, because it is generated according to its own rules. (Skillcrush 2022; Meta Platforms 2022.)

This library has several features. The declarative property is manifested in the fact that the developer can describe the appearance of different elements in any available states. The declarative approach allows to save the source code and make it visual. There is also imperative code that helps to describe the scenario of events. This approach allows to answer the question of what needs to be done on the page or in the application to fulfill certain conditions. If there are many such scenarios, there is a high risk of making a mistake and then the structure may fall apart at any moment. For example, in a native application there are 10 entities and each of them has 5 states. By simple arithmetic, we get 50 possible states that need to be described. For example, if the user clicks on the menu button, a block should appear. If there is no click, then the menu will not be displayed. Declarative code helps to solve such problems easily, because it does not describe the conditions of change, but shows what should change. This approach is ideal for entities with many conditions. (Skillcrush 2022; Meta Platforms 2022.)

One of the main advantages of the library is that it can be used anywhere. For example, on your server when creating non-standard websites that are difficult to implement based on ready-made content management systems. React is often used to develop services where speed is important, not a beautiful interface. React can also be used on mobile platforms. There is React Native, which is used to create cross-platform applications. (Skillcrush 2022; Meta Platforms 2022.)

Another distinctive feature of React is the component approach, which manifests itself in the use of "building" units — components. There are no templates, controllers, and other elements, but there are components that can be controlled. The component is something like concrete blocks, from which is real to build any building. Components can inherit properties from each other, moreover, they can be combined with each other and reused. Components are about the same as JS functions. They are described using properties and save the state, and at the right moment they return React elements to display them on the page. The developer uses components to create a structure for their tasks. (Skillcrush 2022; Meta Platforms 2022.)

An important feature of the React library is the virtual DOM. This is an object that contains information about the state of the interface. For example, after a user sends data in a form to the server, React detects a change in state and updates the appearance of the interface. Thus, the virtual DOM allows to update the real DOM and change the state of components on the screen without delay. The user does not see blinks and other visual artifacts. With this React property, everything is not so smooth, because the rendering function does not work as quickly as we would like. The algorithm for calculating the state is slow and not always accurate, so developers have to use special properties. (Skillcrush 2022; Meta Platforms 2022.)

The JavaScript syntax extension is similar to HTML, but it is JS. The main feature is that programmer can write code using ready-made components. This approach allows to speed up development. The library can be used without JSX, but most developers use the extension to provide visual interaction with the interface. One of the main advantages of JSX is native error notifications and warnings that help quickly learn about situations that are worth paying attention to. (Skillcrush 2022; Meta Platforms 2022.)

3 DESIGN OF JOB SEEKER SERVICE

In the practical part of this work, we will talk about creating a web service for finding vacancies for employment. This web application is called easySearch and it aims to provide a list of current vacancies in real-time for those people who are looking for a job. The only thing is that the search is carried out only in Russia, in all its regions, since the application uses the API of the Russian site hh.ru - the largest Russian Internet recruitment company, whose clients are over three hundred and fifty thousand companies.

The main idea of the application is to see the available jobs and the possibility to filtrate them. So, the search of the desired job will be done much faster. Each job post has a brief info and description about the position, such as name of vacancy, the city, the main requirements, and obligations are indicated, as well as a link to a more detailed description of the work with all the nuances. If the user will be interested in more detailed information, he can open the provided link and be redirected to the full-page description.

The application uses a fairly simple design. The main page of the application welcomes the client and gives him a description of the service, after which the user needs to either login or register to view vacancies. The authorization window looks standard, where it asks the user for his login and password. The situation is similar with the registration window, nothing unusual, only in addition to the login and password, the user is asked for his e-mail and asked to enter the password twice to verify the identity of passwords. After registration or authorization, a successful login window appears, otherwise, the user will receive error notifications. Upon successful entry, the client is provided with a list of available vacancies.

3.1 Application architecture

Application architecture is the structural principle by which an application is created. Each architectural type has its own characteristics, properties, and relationships between components, where the component is a small or large logical and independent part of the architectural system of the application. Every application around us is built according to planned architecture. Even in cases where they did not adhere to any of its types in principle, the application will still work within one of the types of architecture. It does not work any other way. As part of the application architecture, there will be both front-end and internal services. Front-end development is related to the user interface of the application, while internal development focuses on providing access to data, services, and other existing systems that enable the application to work. It is important to think over the architecture of applications before starting development. After all, a well-thought-out architecture is the key to the efficiency and convenience of the future program. The program's response to user actions, the ability to cope with high loads, freezes - all these and other potential problems depend on how well the application architecture was initially thought out. (Red Hat 2022.)

There are some different types of application architecture. For example, a Layered or N-tier architecture, which is a traditional architecture what is often used to create local applications. Layered or N-tier architecture is obsolete, as is a Monolithic architecture, which is a single application stack containing all the functionality within a given application. Modern application architectures are usually loosely coupled, using application programming interfaces (APIs) and microservices to connect services. With the help of microservices, applications are divided into small components that do not depend on each other and each of these components or processes is a microservice. Microservices do not affect each other, which gives advantages for both fault tolerance and dynamic scalability. (Red Hat 2022.)

3.1.1 Use cases

The main purpose of creating any software system is to create a software product that helps the user to perform their daily tasks. To create such programs, the first step is to determine the requirements that the system must meet. However, if let users write these requirements on paper, the result often gets a list of functions, according to which it is difficult to judge whether the future system will fulfil its purpose and whether it will be able to facilitate the user to do his work at all. It is unclear which of the functions performed are more important and for whom. (Indeed 2022.)

To understand more precisely how the system should work, a description of the system's functionality through use cases is increasingly used. Use cases are a description of the sequence of actions that the system can perform in response to external influences from users or other software systems. Use cases reflect the functionality of the system in terms of obtaining a meaningful result for the user, so they more accurately allow to rank functions according to the significance of the result obtained. Creating use cases is important. Indeed, in some situations, it is much more difficult without the described scenarios than with them. (Indeed 2022.)

Use cases may contain the following elements and their number depends on the complexity of the scenario. First, the actor is the one who uses the system. If take an online store as an example, there may be several actors: buyers, sellers, delivery companies. Moreover, it is a stakeholder is someone, who is interested in a certain behavior of the system. Often it is not the end user, but someone who benefits from the functioning of the system. In the case of an online store, it can be a partner - a payment platform. The next element is primary actor. It is a person or system whose goals are achieved with the help of our product. In an online store, this may be the main distributor whose products are sold on this online platform. Nevertheless, preconditions and postconditions take place to be. That describes what should be available or should happen before and after running the usage scenario. To add more, triggers are also an element of the use cases. Triggers are events that cause software developers to start researching use cases or a report. Furthermore, a successful scenario in this list of elements too, it is a use case, where everything goes according to plan, without errors and surprises. The last but not the least, the alternative paths that are variations of the main successful scenario in case something goes wrong at the system level. (Indeed 2022.)

3.1.2 Use cases diagram

Here is a diagram of the use cases. This diagram has actor and use cases, as well as the connection between these elements. It was created to resolve the functionality of the system (FIGURE 1).

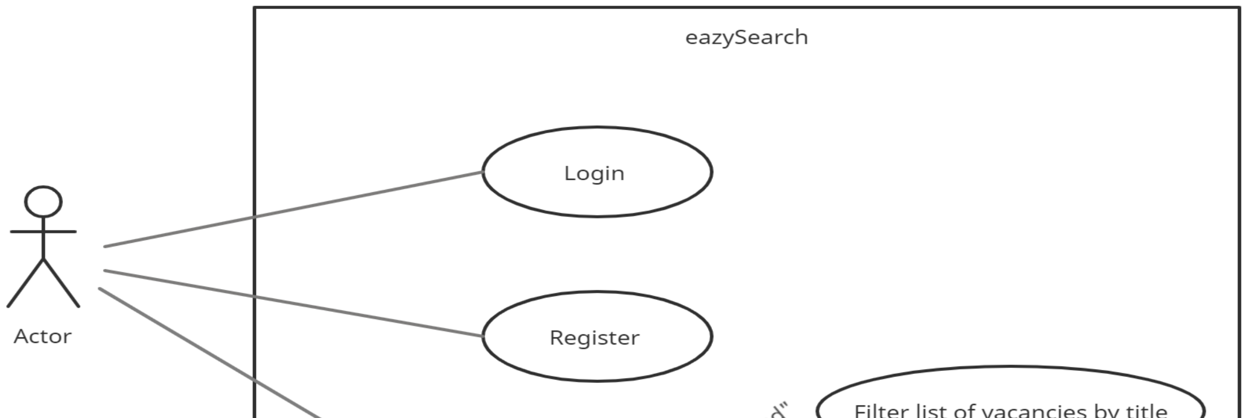


FIGURE 1. Use cases diagram

3.1.3 Use cases description

Use case ID:	001
Use case name:	Login
Actors:	User
Description:	User login into the system.
Pre-conditions:	User must be on the main page and click the login button.
Normal Flow:	User is provided with a form for entering own data: login and password. Processing is underway to log in. [**Exception:** Inputs are invalid], [**Exception:** Not registered before].
Alternative Flow:	User proceeds to registration.
Exceptions:	**Inputs are invalid:** Incorrect data entry. User is notified and asked to input data again. **Not registered before:** Non-existent account. User is notified and go to register.
Post-conditions:	Inputs are valid. Enter the system to see the list of vacancies.

Use case ID:	002
Use case name:	Register
Actors:	User
Description:	User registers to enter the system.
Pre-conditions:	User must be on the main page and click the register button or must on the login page and click the line for register.
Normal Flow:	User is provided with a form of registration: username, email address, password, and confirmation of the password. Processing is underway to enter the system. [**Exception:** Inputs are invalid]
Exceptions:	**Inputs are invalid:** The password does not meet the requirements. The lack of password identity. User is notified and asked to input data again correctly.
Post-conditions:	Inputs are valid. Enter the system to see the list of vacancies.

Use case ID:	003
Use case name:	Searching
Actors:	User
Description:	User views the list of available vacancies.
Pre-conditions:	User must be registered and log in into the system.
Normal Flow:	User is provided with a list of available vacancies with short information. User is able to filter the vacancies by the title for easy viewing. User can use the link to view detailed job information.
Alternative Flow:	User is provided with a list of available vacancies with short information. User can scroll down the list without filtering and use the link for viewing detailed job information.
Post-conditions:	The list of vacancies and information about them are received.

3.1.4 Architecture diagram

As mentioned earlier, the REST API is a way for websites and web applications to interact with the server. The use of the REST API in this application was needed to get data from the HeadHunter server. The REST API is based on the hypertext transfer protocol. Each object on the server in HTTP has its own unique URL in a strict sequential format. In this case, the unique URL is https://api.hh.ru/vacancies, which determines the resource we need on the service. To interact with this resource, the REST API uses the CRUD-command of the HTTP protocol to get information about data or a list of objects. After that, the resource is represented in JSON format (FIGURE 2; Code 3).

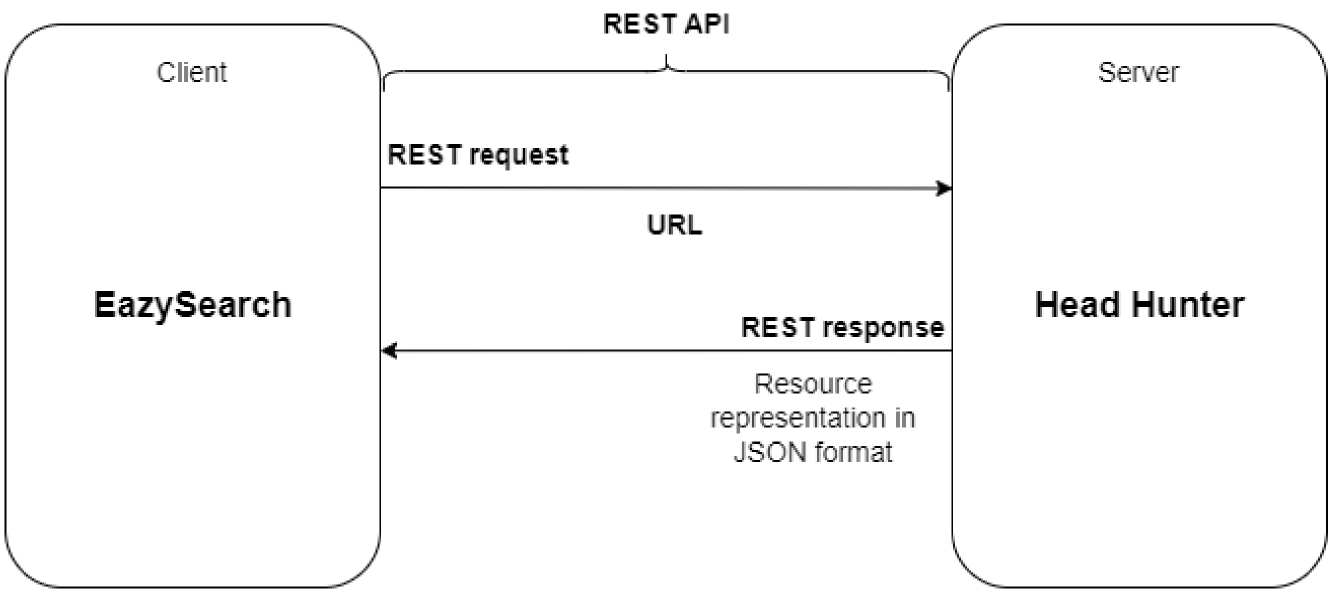


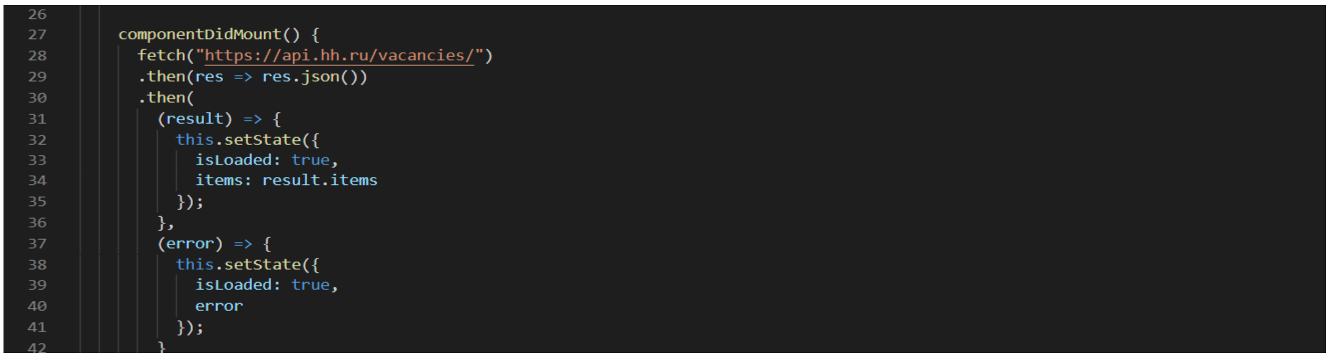
FIGURE 2. Architecture diagram

```
{"items":[{"id":"50677859","premium":false,"name":"Главный герой/главная героиня
рекламы hh.ru","department":{"id":"hh-1455-MARK","name":"HeadHunter::Департамент
маркетинга"},"has_test":false,"response_letter_required":true,"area":
{"id":"1","name":"Москва","url":"https://api.hh.ru/areas/1"},"salary":
{"from":null,"to":500000,"currency":"RUR","gross":true},"type":
{"id":"open","name":"Открытая"},"address":null,"response_url":null,"sort_point_dis
tance":null,"published_at":"2021-12-24T15:52:54+0300","created_at":"2021-12-
24T15:52:54+0300","archived":false,"apply_alternate_url":"https://hh.ru/applicant/
vacancy_response?
vacancyId=50677859","insider_interview":null,"url":"https://api.hh.ru/vacancies/50
677859?host=hh.ru","alternate_url":"https://hh.ru/vacancy/50677859","relations":
[],"employer":
{"id":"1455","name":"HeadHunter","url":"https://api.hh.ru/employers/1455","alterna
te_url":"https://hh.ru/employer/1455","logo_urls":
{"original":"https://hhcdn.ru/employer-logo-
original/907140.png","240":"https://hhcdn.ru/employer-
logo/4069248.png","90":"https://hhcdn.ru/employer-
logo/4069247.png"},"vacancies_url":"https://api.hh.ru/vacancies?
employer_id=1455","trusted":true},"snippet":{"requirement":"Объем текста истории —
```

Code 3. Response in JSON format

3.1.5 Presentation of results

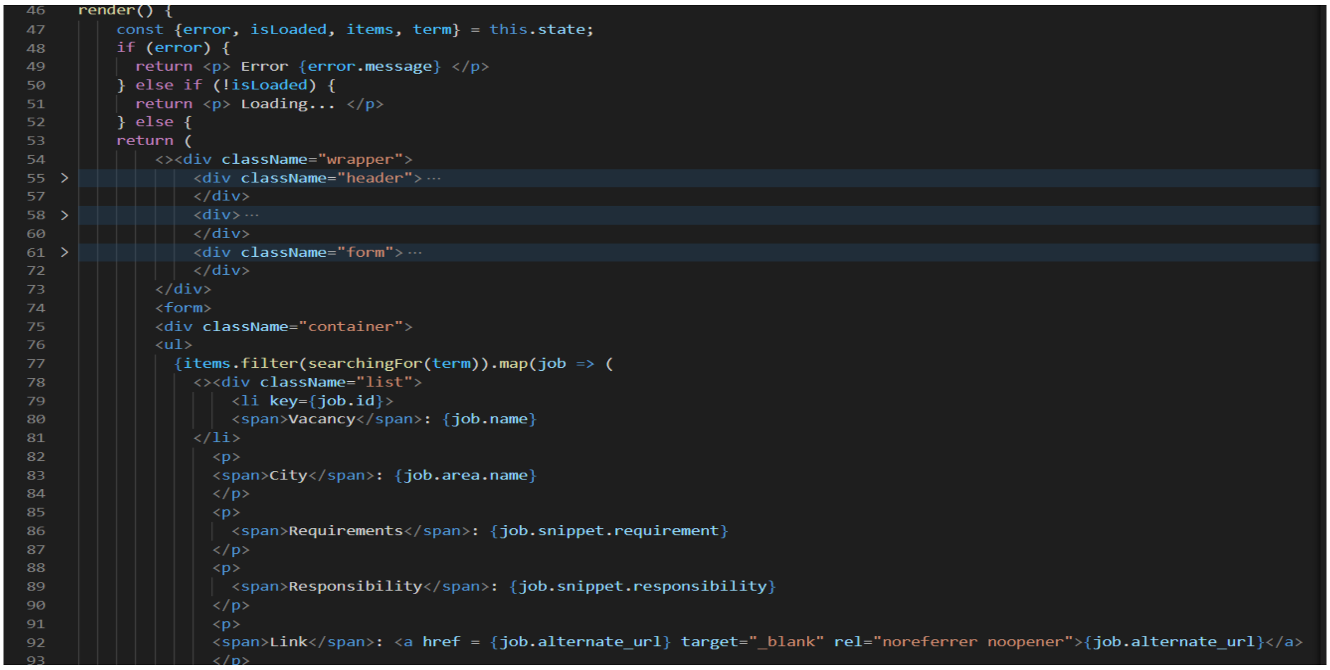
It was mentioned earlier that the API for this project was taken from the HeadHunter website. The lifecycle, “`componentDidMount`”, was used. It is called after mounting the component in the DOM. In the method, actions occur that require the presence of DOM nodes. This is the best place to create network requests. Inside it, a “`fetch`” (HTTP request of a new generation) is used, which provides an improved interface for making requests to the server and receiving data from resources. In the URL, filtering is selected by vacancies, which is exactly what we need. The “`res`” parameter in this case will receive a JSON file (PICTURE 1). The state will change in the result parameter, the empty array will be replenished. Then below is the error handling with the state change. In the render section there is conditional rendering, where a certain data structure is returned. Next, the elements with a certain final value are output (PICTURE 2).



```

componentDidMount() {
  fetch("https://api.hh.ru/vacancies/")
  .then(res => res.json())
  .then(
    (result) => {
      this.setState({
        isLoaded: true,
        items: result.items
      });
    },
    (error) => {
      this.setState({
        isLoaded: true,
        error
      });
    }
```

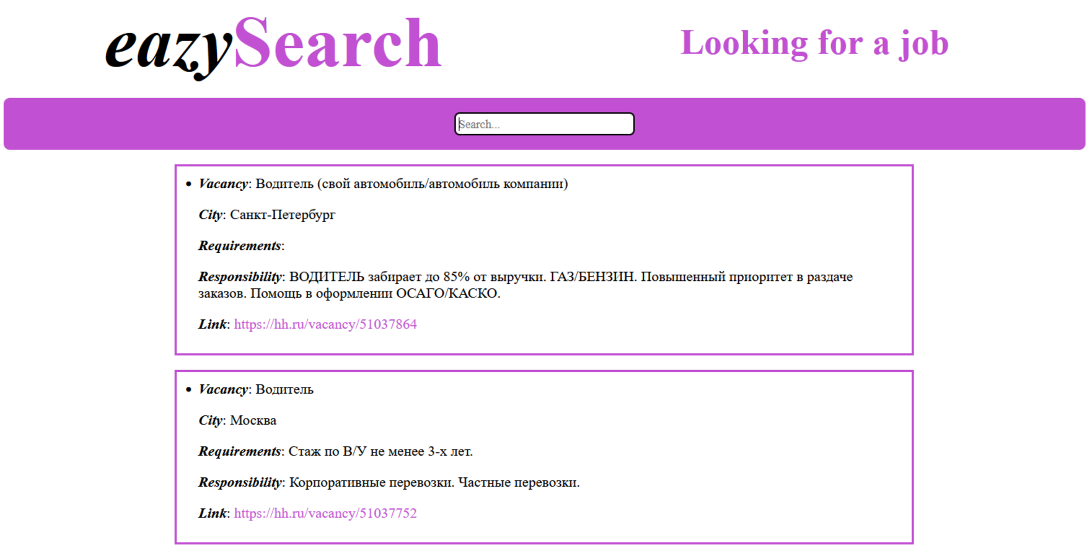
PICTURE 1. Fetch



```
render() {
    const {error, isLoaded, items, term} = this.state;
    if (error) {
      return <p> Error {error.message} </p>
    } else if (!isLoaded) {
      return <p> Loading... </p>
    } else {
    return (
        <><div className="wrapper">
            <div className="header">···
            </div>
            <div>···
            </div>
            <div className="form">···
            </div>
        </div>
        <form>
        <div className="container">
        <ul>
          {items.filter(searchingFor(term)).map(job => (
            <><div className="list">
                <li key={job.id}>
                <span>Vacancy</span>: {job.name}
            </li>
              <p>
              <span>City</span>: {job.area.name}
              </p>
              <p>
                <span>Requirements</span>: {job.snippet.requirement}
              </p>
              <p>
                <span>Responsibility</span>: {job.snippet.responsibility}
              </p>
              <p>
              <span>Link</span>: <a href = {job.alternate_url} target="_blank" rel="noreferrer noopener">{job.alternate_url}</a>
              </p>
```

PICTURE 2. Render

As a result, the page displays the resource data that was requested using the URL. In this case, not only the vacancy is rendered, but also information about the city, employer requirements, responsibilities as well as the link to go to more detailed information about the job, that is, the part of the data that was selected from the JSON file (PICTURE 3; PICTURE 4).



- ***Vacancy***: Водитель (свой автомобиль/автомобиль компании)

  City: Санкт-Петербург

  Requirements:

  Responsibility: ВОДИТЕЛЬ забирает до 85% от выручки. ГАЗ/БЕНЗИН. Повышенный приоритет в раздаче заказов. Помощь в оформлении ОСАГО/КАСКО.

  Link: https://hh.ru/vacancy/51037864

- ***Vacancy***: Водитель

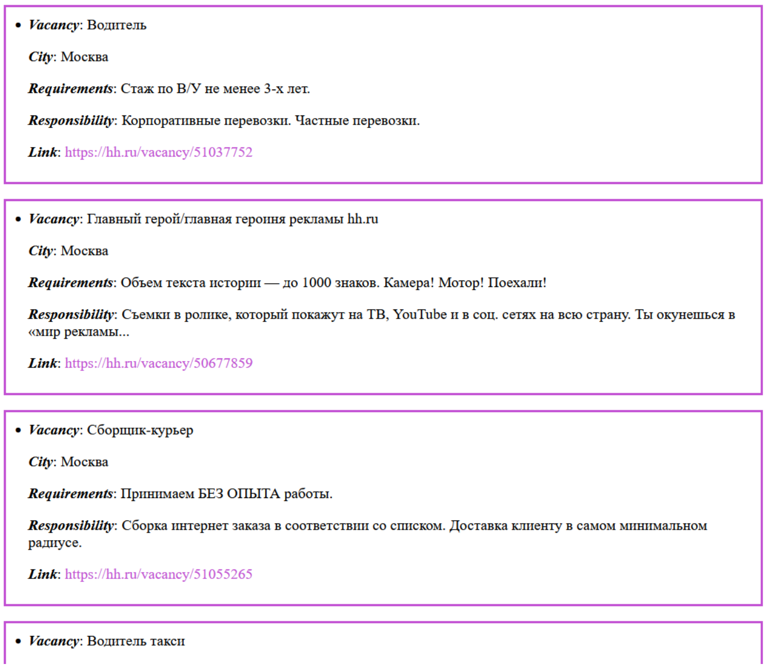
  City: Москва

  Requirements: Стаж по В/У не менее 3-х лет.

  Responsibility: Корпоративные перевозки. Частные перевозки.

  Link: https://hh.ru/vacancy/51037752

PICTURE 3. Result_1



- ***Vacancy***: Водитель

  City: Москва

  Requirements: Стаж по В/У не менее 3-х лет.

  Responsibility: Корпоративные перевозки. Частные перевозки.

  Link: https://hh.ru/vacancy/51037752

- ***Vacancy***: Главный герой/главная героиня рекламы hh.ru

  City: Москва

  Requirements: Объем текста истории — до 1000 знаков. Камера! Мотор! Поехали!

  Responsibility: Съемки в ролике, который покажут на ТВ, YouTube и в соц. сетях на всю страну. Ты окунешься в «мир рекламы...

  Link: https://hh.ru/vacancy/50677859

- ***Vacancy***: Сборщик-курьер

  City: Москва

  Requirements: Принимаем БЕЗ ОПЫТА работы.

  Responsibility: Сборка интернет заказа в соответствии со списком. Доставка клиенту в самом минимальном радиусе.

  Link: https://hh.ru/vacancy/51055265

- ***Vacancy***: Водитель такси

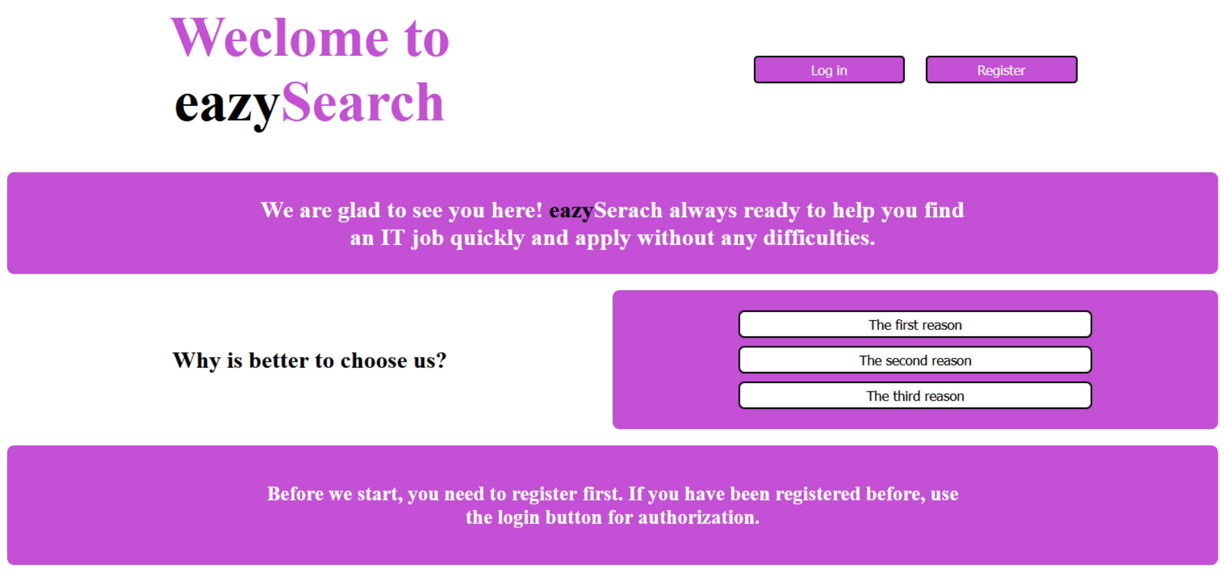
PICTURE 4. Result_2

3.1.6 User interface design

To begin with, that User Interface Design is an intermediary between a device and a person. This is not only the appearance, but also the interactivity of the product. User interface design is often confused with graphic design, branding, and even interface design, but these are all misconceptions. The User Interface is a structure that is responsible for transferring product development from research and mockups to an attractive and user-friendly experience. In other words, UI is an art that focuses on all the elements of a product that make it attractive: colors, button style, graphics, animation, typography, infographics, widgets, behavior, button responses. A successful combination of iconography, typography, color, space, and textures will help the user easily navigate in the finished application. The user should feel good using the product. It depends not only on the intuitive interface, but also on how this interface looks. Due to poor visual design, users stop using the app. Visual design itself is the individuality of the product. It manages the key elements that inspire confidence. Simply put, User Interface Design is the process of visual user management through the product interface using interactive elements, providing information exchange between a person and software components, which is responsible for transferring the visual aspects of the brand to the product interface in order to improve it. The user interface of the web service has been designed in such a way as to be simple, minimalistic, understandable, and most importantly, effective for using. (F5 Studio 2022.)

3.1.7 Welcome page

The welcome page is the starting point of the path for getting a list of vacancies by a user who is looking for a job. It contains short, but at the same time understandable and necessary information about this service. Moreover, the page includes several buttons with which the client can interact. Each button has a shadow effect on hover. These are buttons for login and registration, as well as buttons, when you click on each of it a pop-up window appears with a description of the reason why the client should choose this service. Given the fact that this is a demo version, these "reasons" buttons are slightly incomplete, so the visualization and text look raw at the moment. However, when clicking the registration button, the client will be directed to the registration page, while clicking the login button redirects to the login page (PICTURE 5; PICTURE 6).

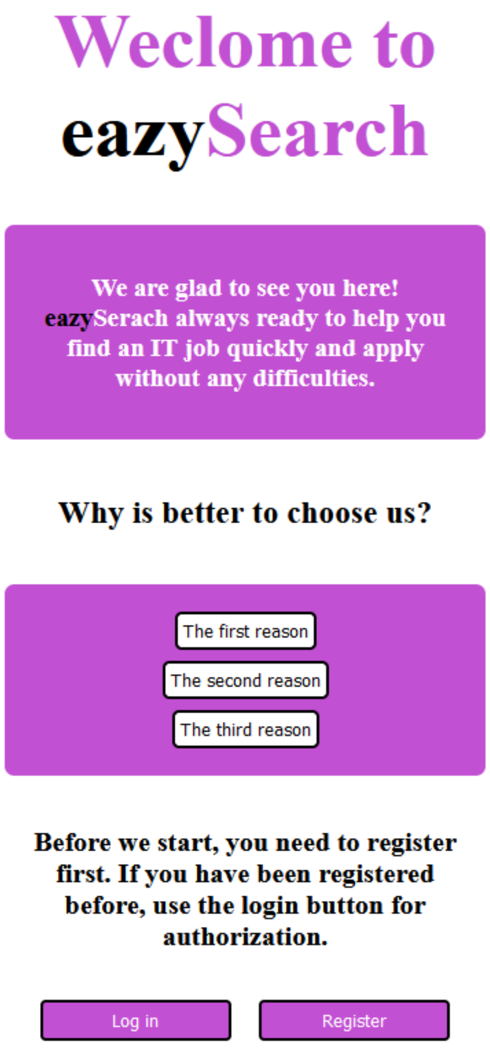

Weclome to eazySearch

We are glad to see you here! eazySerach always ready to help you find an IT job quickly and apply without any difficulties.

Why is better to choose us?

Before we start, you need to register first. If you have been registered before, use the login button for authorization.

PICTURE 5. Welcome page


Weclome to
eazySearch

We are glad to see you here!
eazySerach always ready to help you find an IT job quickly and apply without any difficulties.

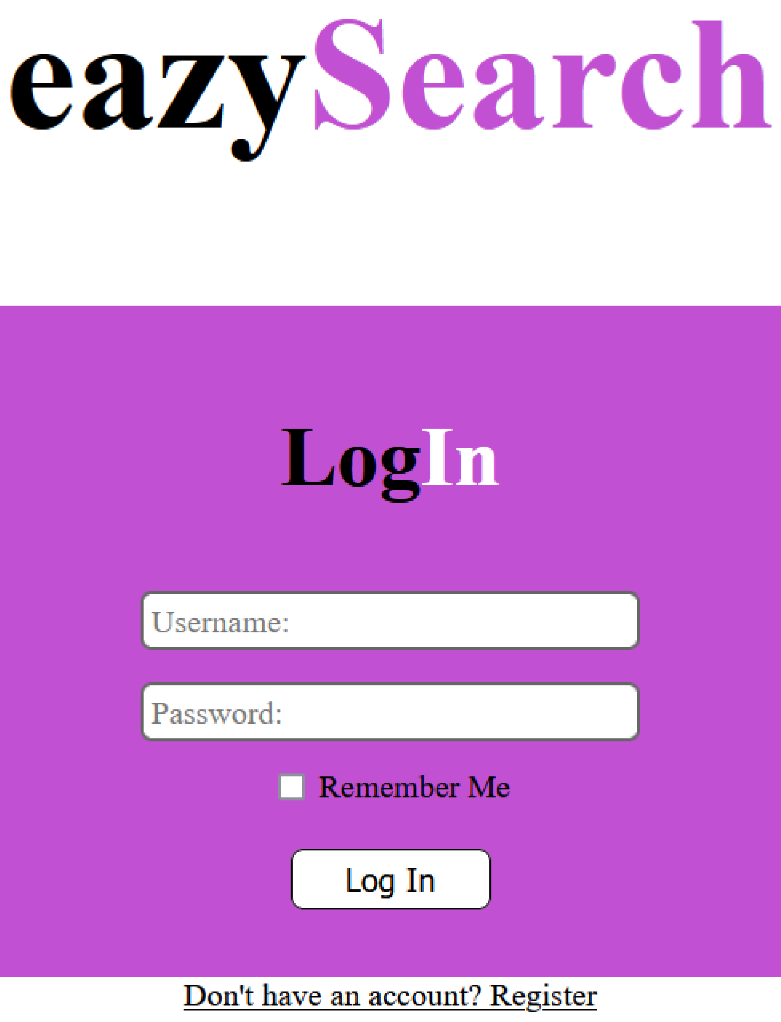
Why is better to choose us?

Before we start, you need to register first. If you have been registered before, use the login button for authorization.

PICTURE 6. Mobile welcome page

3.1.8 Login page

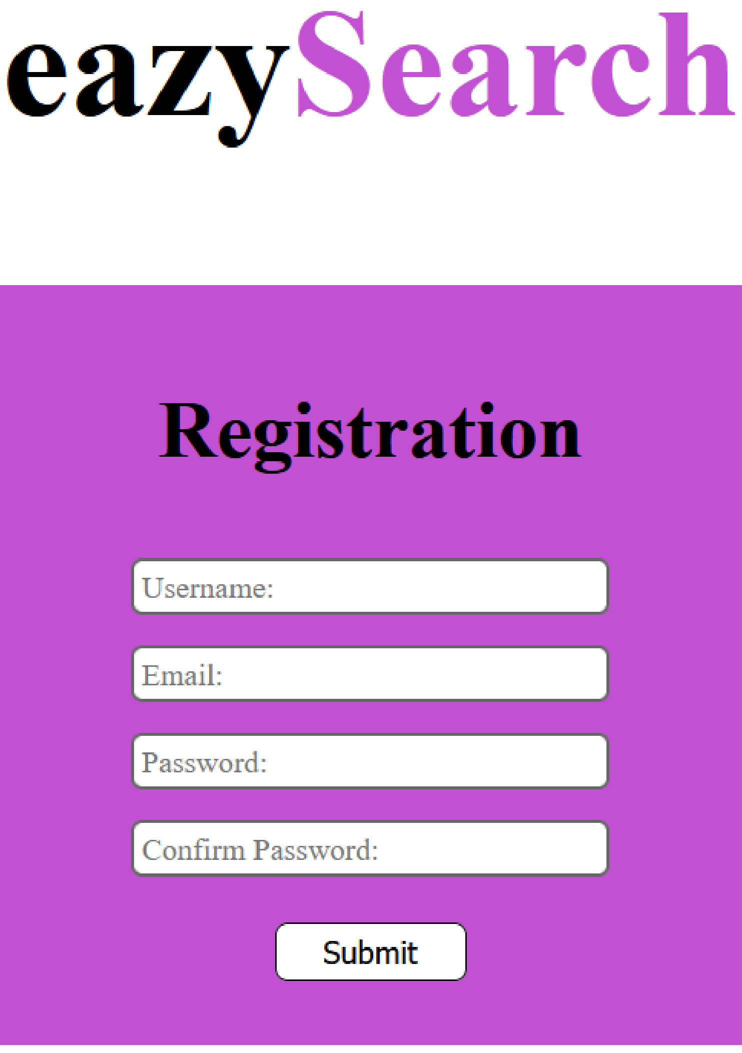
The login page has a fairly simple design. It includes a form for entering user data, that is, two windows, one of which is for entering a username and the other for entering a password, respectively. In addition, the client is able to use the "Remember me" function by checking the box after entering the data, so that the system remembers the inputs for the future re-entering. After successfully entering the username and password, when interacting with the login button, the user will be redirected to a page with a list of available vacancies. In case of entering incorrect data, the client will be notified about it by a pop-up window. In addition to everything, at the very bottom of the form, there is a transition to registration, if the user has not been registered before. After clicking, the client will be redirected to the registration page (PICTURE 7).



PICTURE 7. Login page

3.1.9 Register page

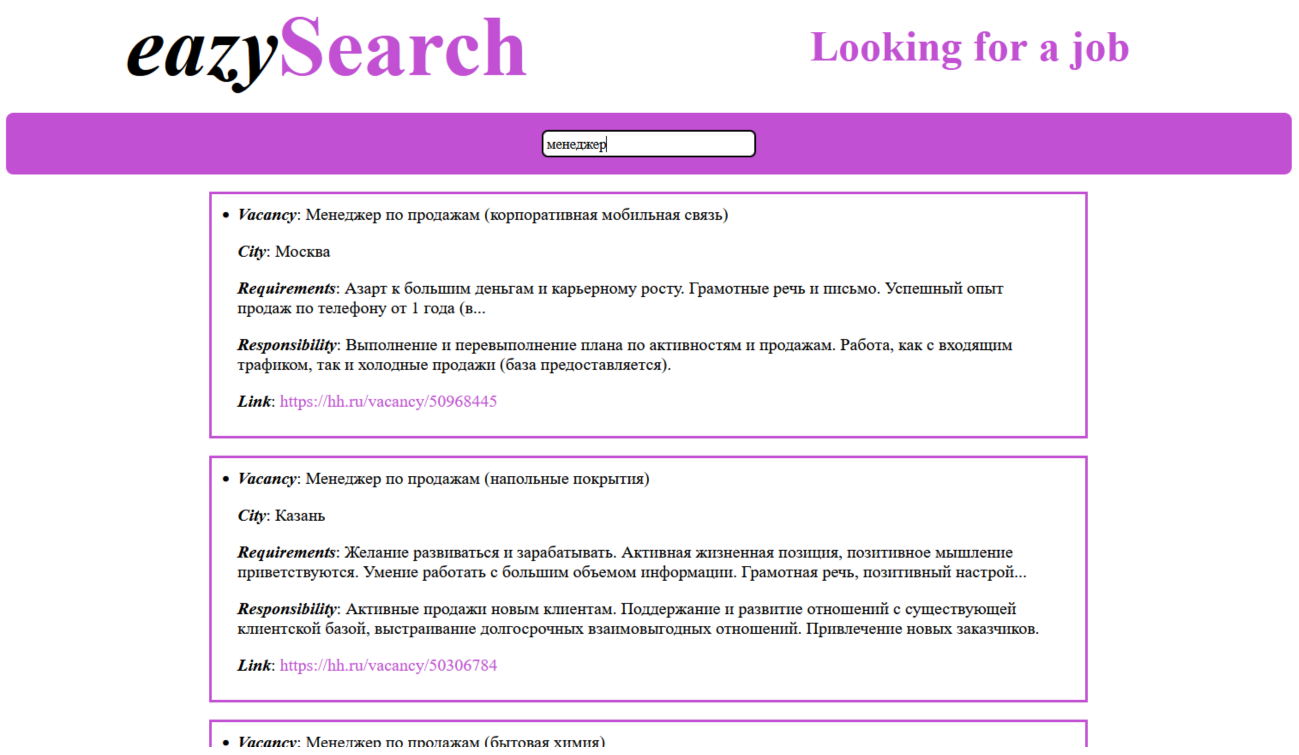
As for the design, the registration page looks similar to the login page. The client is provided with a form to fill in the data, which contains fields for entering a username, email address and password as well as a field for re-entering the password to verify identity of passwords. Moreover, the registration form requires the user to use a password of a certain length and symbolism. Otherwise, the client will be notified of unsatisfied requirements using a pop-up window. The user will also be notified if the entered passwords do not match. After successful completion, the user interacts with the “submit" button and receives a list of vacancies to view (PICTURE 8).



PICTURE 8. Register page

3.1.10 Job market page

This page represents a list of available vacancies. The user can view the proposed jobs by scrolling down the page. For easy viewing or to find the right vacancy, there is a search field on the page. This field filters the list by the job title. Also, to get more detailed information about the work, the user can click on the link that is present in each ad (PICTURE 9).



- *Vacancy*: Менеджер по продажам (корпоративная мобильная связь)

  City: Москва

  Requirements: Азарт к большим деньгам и карьерному росту. Грамотные речь и письмо. Успешный опыт продаж по телефону от 1 года (в...

  Responsibility: Выполнение и перевыполнение плана по активностям и продажам. Работа, как с входящим трафиком, так и холодные продажи (база предоставляется).

  Link: https://hh.ru/vacancy/50968445

- *Vacancy*: Менеджер по продажам (напольные покрытия)

  City: Казань

  Requirements: Желание развиваться и зарабатывать. Активная жизненная позиция, позитивное мышление приветствуются. Умение работать с большим объемом информации. Грамотная речь, позитивный настрой...

  Responsibility: Активные продажи новым клиентам. Поддержание и развитие отношений с существующей клиентской базой, выстраивание долгосрочных взаимовыгодных отношений. Привлечение новых заказчиков.

  Link: https://hh.ru/vacancy/50306784

- *Vacancy*: Менеджер по продажам (бытовая химия)

PICTURE 9. Job market page

3.2 Technologies used

As for the technologies that were used in this work, it would be better to start with the programming language that has been chosen. The scripting language, also known as JavaScript, is the main engine in programming interactive websites, used to write frontend and backend parts of websites, as well as mobile applications. It is quite simple, but it contains all the fundamental things: algorithms, object-oriented model, data structures. The JS code works immediately in the browser, therefore no costs are required to compile it. This is undoubtedly a benefit: there is no need to wait for anything, the result of the code can be immediately seen in the browser. Moreover, this language has many already developed frameworks and libraries, one of which is also involved in the project. When using them in web development, there is an advantage in the time of development itself and the graphical user interface improves. It is an open secret that JavaScript is the "best friend" for HTML and CSS. Without these technologies, the current project would not have taken place. Therefore, the role of HTML is to mark up the site, where CSS is responsible for the appearance. (Simplilearn 2022.)

Special attention should be paid to React JS library. React greatly facilitates the creation of interfaces by splitting each page into small fragments, they are also called components. React.js simplifies the creation of web applications by reducing the amount of manual code writing. In this work, version 17.0.2 was used. Interesting fact that the new versions of this library are fully compatible with previous releases. Old libraries do not become useless after updates are released. It can be considered as an advantage by using React JS. (Meta Platforms 2022.)

An important role is played by Node.js. It is a server platform for working with JS through the Chrome V8 engine. JavaScript performs the action on the client side, while Node performs the action on the server. This kind of platform works with external libraries, calls commands from JavaScript code, and acts as a web server. Nowadays, the Node.js platform is one of the most popular platforms for building efficient and scalable REST APIs. In time of work with app the version v14.17.6 was used. (Kinsta Inc 2022.)

As for the tools, Visual Studio Code version 1.63.2 acted as the development environment. This code editor is comfortable to use. For example, VS Code knows the syntax of different programming languages and helps to avoid errors in the semicolon or parenthesis. It is convenient because it substitutes some common code snippets itself. In addition, the dev environment remembers the names of variables and suggests them, so that there are no errors. (Microsoft 2022.)

Finally, Mozilla Firefox browser with advanced settings specifically for React was involved in this project. Analyzing the React developer tools it can be noted that this browser extension allows viewing the tree of components, their props, status, and context. It is also quite simple to catch inefficient components that are being re-rendered, see how much time it takes and build graphs to visualize the effectiveness of components. Thanks to this information, it is possible not only optimize the application, but also study React more deeply and understand all the subtleties of working with it. (Mozilla 2022.)

4 CONCLUSION AND FURTHER DEVELOPMENT

The thesis intended to create a web service for searching the vacancies by using the Java Script library — React. There were many challenges involved. It should be said that the project idea was abrupt, and it is still unclear how the idea came to mind to choose the development of exactly this type of service, where the API is present, the application of searching for something in general. It is always the hardest thing to get started. The motivation was the desire to master new skills related to frontend development of web applications, as well as to try new technologies that will be extremely useful for work in the future. Prior to the start of development, knowledge of working with the React library was not enough, so a decent amount of time was spent on studying. The lack of knowledge was the reason for minor difficulties with the React. It was necessary to understand how everything is connected and how it works. As a result, specific questions were answered. Before starting development directly, the graphic design of the application was created to better understand what needs to be developed, what should be present on the page, what it should all look like. With the design development using CSS, there were no difficulties at all, as well as working with markup — with JSX. Given the fact that this is a demo version, there is an unsolved problem with the multiple pages. They are not connected in any way because React is single-paged application and external libraries, such as React Router or Gatsby should be used, but this point can be attributed to further development. There were also some difficulties when writing the code that is associated with the search bar to filter the list by the name of the vacancy. But finally, it can be assumed that it turned out. And this is just the beginning, for me personally.

As for the further development of the web service, there are several features that can be applied for better use. First of all, the plan is to complete the demo version so that part of the above functions correctly. Nevertheless, it is a great idea to work on the design, the simpler the better, but it would be great to add some more features, so to speak, to delve into CSS to gain new skills. In addition, with the expansion of the market, the web service can be modified for other countries using different APIs. For another skill, it is worth trying to work with the API paired with the key. As mentioned earlier, it is better to transfer the project to Gatsby to connect the pages so that the application becomes multi-page and functions normally. Moreover, it is possible to add search windows to filter vacancies not only by name, but also by city or salary, for example. Initially, the idea was to add a form for filling out a resume by the user and attach it to the account so that the client could instantly use it and send it to the employer when looking for a job, as the addition this feature can be implemented too. It is safe to say that in the future there will be more ideas for further development of the web service.

REFERENCES

Austin, D. 2020. “What is JSON Used for in JavaScript Programming”. Available at: https://medium.com/swlh/what-is-json-used-for-in-javascript-programming-9d71284359a9. Accessed 15.01.2022.

Buna, S. 2019. “All the fundamental React.js concepts, jammed into this single Medium article”. Available at: https://medium.com/edge-coders/all-the-fundamental-react-js-concepts-jammed-into-this-single-medium-article-c83f9b53eac2 . Accessed 15.01.2022.

Evseev, A. 2019. “Android Application Development”. Available at: https://www.theseus.fi/bitstream/handle/10024/173534/aleksandr_evseev_thesis_2019.pdf?sequence=2&isAllowed=y. Accessed 15.01.2022.

F5 Studio. 2022. “What is UI Design and why is it important”. Available at: https://f5-studio.com/articles/what-is-user-interface-design-and-why-is-it-important/. Accessed 15.01.2022.

Indeed. 2022. “Use Cases: What They Are”. Available at: https://www.indeed.com/career-advice/career-development/list-of-use-cases-examples. Accessed 15.01.2022.

Independed Media. 2022. “The History of the Internet”. Available at: https://www.tech-faq.com/history-of-the-internet.html. Accessed 15.01.2022.

Internet Society. 2022. “Brief History of the Internet”. Available at: https://www.internetsociety.org/internet/history-internet/brief-history-internet/. Accessed 15.01.2022.

Kinsta Inc. 2022. “What Is Node.js and Why You Should Use It”. Available at: https://kinsta.com/knowledgebase/what-is-node-js/. Accessed 15.01.2022.

Lozhko, M. 2021. “Web Application Architecture: Best Practices and Guides”. Available at: https://lanars.com/blog/web-application-architecture-101. Accessed 15.01.2022.

Meta Platforms, Inc. 2022. “Tutorial: Intro to React”. Available at: https://reactjs.org/tutorial/tutorial.html. Accessed 15.01.2022.

Microsoft. 2022. “Why Visual Studio Code”. Available at: https://code.visualstudio.com/docs/editor/whyvscode. Accessed 15.01.2022.

Mozilla. 2022. “React Developer Tools by React”. Available at: https://addons.mozilla.org/en-US/firefox/addon/react-devtools/. Accessed 15.01.2022.

Red Hat, Inc. 2022. "What is an application architecture". Available at: https://www.redhat.com/en/topics/cloud-native-apps/what-is-an-application-architecture. Accessed 15.01.2022.

Simplilearn. 2022. “An Introduction to JavaScript: Here Is All You Need to Know”. Available at: https://www.simplilearn.com/tutorials/javascript-tutorial/introduction-to-javascript. Accessed 15.01.2022.

Skillcrush, Inc. 2022. “What Is React JS”. Available at: https://skillcrush.com/blog/what-is-react-js/. Accessed 15.01.2022.

Walker, A. 2021. “RESTful Web Services Tutorial”. Available at: https://www.guru99.com/restful-web-services.html. Accessed 15.01.2022.

Walker, A. 2021. “What are Web Services. Architecture, Types, Example”. Available at: https://www.guru99.com/web-service-architecture.html. Accessed 15.01.2022.

Wikipedia. 2022. “JSON”. Available at: https://en.wikipedia.org/wiki/JSON. Accessed 15.01.2022.