



Duc Vo

Developing a Vision Web application Service

Metropolia University of Applied Sciences

Bachelor of Engineering

Information Technology

Bachelor's Thesis

15 April 2022

Abstract

Author: Duc Vo
Title: Developing a vision web application service
Number of Pages: 54 pages
Date: 15 April 2022

Degree: Bachelor of Engineering
Degree Programme: Information Technology - Smart Systems
Supervisors: Janne Salonen, Head of Department

Artificial intelligence is advancing at a rapid speed in today's world, due to its sophisticated and widespread use in almost every aspect of life. As more organizations and businesses join the artificial intelligence revolution, a new horizon for the computer industry in particular and all other specialized industries ,in general, is opened. Along with this trend, this thesis aims to encourage undergraduate study into the application of cloud-based artificial intelligence services to real-world situations.

The primary objective of this thesis was to develop a web application that utilizes artificial intelligence and machine learning techniques. More precisely, this research investigated Azure Cognitive Services, a collection of application programming interfaces, software development kits, and Microsoft Cloud services that enables programmers to create, test, and deploy computer vision projects. The project concentrated on constructing the technological side to conduct a large-scale interface application that enables customers to install some of the Face API's capabilities from Azure without needing to learn directly from Microsoft or possess programming or machine learning skills.

In conclusion, this project required significant effort to broaden knowledge and expertise in constructing a demonstration large-scale online application. Despite the time constraint, this project accomplished sufficient functionality to be used in a real-world application. Finally, from a technological standpoint, this thesis may contribute to research and development efforts aimed at broadening the use of these types of approaches in a variety of systems.

Keywords: Artificial intelligence, machine learning, Azure Cognitive Service, computer vision, Node.js, React

Contents

List of Abbreviations

1	Introduction	1
2	Theoretical Background	2
2.1	Cloud Computing	2
2.1.1	Fundamentals of Cloud Computing	2
2.1.2	Cloud Computing Architecture	4
2.2	Artificial Intelligence and Machine Learning	6
2.2.1	Machine learning	7
2.2.2	Neural Network and Deep Learning	8
2.2.3	Computer Vision	10
2.3	Cloud-based Artificial Intelligence Service	11
2.3.1	REST API	11
2.3.2	Software Development Kit	12
2.3.3	Application Programming Interface Key	13
2.3.4	JavaScript Object Notation	13
2.3.5	Frameworks	14
2.4	Microsoft Azure – Cloud Computing Services	14
2.5	Azure Cognitive Services	17
2.5.1	Fundamentals of Azure Cognitive Services	17
2.5.2	Cognitive Vision API	18
2.6	Web application Service	19
2.6.1	Node.js	19
2.6.2	ExpressJS	22
2.6.3	Node Package Manager – NPM	23
2.6.4	React	24
2.6.5	Heroku	27
2.6.6	Netlify	28
3	Project Specifications	28
3.1	Project Objective	29
3.2	Project Architecture	29
4	Application Implementation	31

4.1	Azure Cognitive Services Configuration	31
4.2	Azure Cognitive Online API	33
4.2.1	Recognize and Identify Face	33
4.2.2	Compare and Verify Two Faces	37
4.2.3	Object Identity	39
4.3	Face Detection Console Application	44
4.4	Vision Web Application Implementation	46
4.4.1	Server Implementation	47
4.4.2	Front-end Implementation	50
4.4.3	Application Testing	52
5	Conclusion	54
	References	55

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
CNN	Convolutional Neural Network
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IaaS	Infrastructure as a Service
IT	Information Technology
NPM	Node Package Manager
PaaS	Platform as a Service
REST	Representational State Transfer
SaaS	Software as a Service
SDK	Software Development Kit
SQL	Structured Query Language

1 Introduction

Nowadays, an increasing number of businesses acquire and analyze massive volumes of data. Numerous innovative business models based on data have arisen. Despite the increased demand, data processing is a difficult and time-consuming procedure. Deriving new insights and actionable tactics from huge amounts of data is a big hurdle and machine learning aims to address this. Applications can utilize machine learning to build models that process data quickly and generate actionable information. However, machine learning is typically used in conjunction with in-depth mathematical understanding; building up private applications is time-consuming and costly. To address these issues, cloud-based artificial intelligence services were made available. These services eliminate the need for significant additional work and also provide the well-known and much-desired benefits of cloud computing. Cloud systems, at their core, offer computation and storage services.

With the rising interest in cloud computing, an increasing number of studies are being undertaken to determine how cloud computing affects the software development industry. The variety of cloud service alternatives available is staggering. IT giant companies such as Amazon, Microsoft, and Google have also developed their own cloud solutions. Microsoft Cognitive Services enables the use of machine learning to tackle common problems such as evaluating text for emotional sentiment or analyzing photos for object or face recognition. With the help of SDKs and REST APIs for client library SDKs, a wide range of companies may now embed cognitive intelligence into their applications using Azure Cognitive Services on the Microsoft Azure cloud platform. No prior understanding of machine learning, data science, or artificial intelligence is necessary when using Azure Cognitive Services to integrate cognitive capabilities into applications.

The purpose of this bachelor's thesis is to build a vision web application utilizing cloud-based artificial intelligence (AI) services offered by Azure cloud computing to discover hidden patterns or insights. For the first time ever, computers are

capable of performing better than humans at analyzing visual images and pinpointing specific details inside them. Among other things, the computer vision service assists with image labeling, image description generation, moderated content, video analysis, and automatic text extraction. Using Microsoft Azure's vision service, the project uses advanced cognitive algorithms to process photos and return information. The application has three key features: face detection, two-face verification, and object identification. By combining a variety of unique technologies to solve a real-world problem, this project is feasible in the research sector, as well as for future development.

2 Theoretical Background

This chapter delves deeper into the fundamental features of machine learning, artificial intelligence, and cloud-based services. This section discusses many components of cloud-based artificial intelligence services in general. Additionally, relevant technologies are discussed that can be employed in conjunction with cloud-based artificial intelligence services.

2.1 Cloud Computing

2.1.1 Fundamentals of Cloud Computing

Cloud computing has been critical in the digital transformation that has occurred over the last few years. Businesses rely on cloud technologies and have adapted their business operations to accommodate them. Interest in and money generated by the cloud industry will continue to expand in the future as well [1]. The goal of this concept is to deliver automated services for quickly providing scalable, low-cost resources and minimizing the need for resource management. It is difficult to precisely define cloud computing as a whole, as no generally accurate definition exists. The NIST (National Institute of Standards and Technology) definition, which is also published by ENISA (European Network and Information Security Agency), is one of the most often used and generally recognized: “cloud computing is a model for enabling ubiquitous, convenient, on-

demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction" [2, p. 2]. Cloud computing may offer many services that enable customers to run their businesses completely in a cloud environment, which implies that not only data can be kept in a cloud environment, but also software programs and other processes can be operated on it.

According to NIST, this cloud model is comprised of a collection of five fundamental characteristics [2, p. 2]. To begin, cloud infrastructures must provide self-service. The primary distinction between conventional data centers and cloud environments is that in the latter, services may be ordered, managed, and scheduled without the client or supplier interacting with each other. Automatic provisioning leads to enhanced efficiency and cost savings for both parties, as opposed to clients waiting several days for the cloud service provider to assess their request. The second fundamental attribute is that the cloud environment's network connectivity must be extensive. Customers must be able to access cloud services and resources through conventional protocols and on a variety of platforms, such as laptops, mobile devices, and so on. Thirdly, cloud resources are shared among numerous clients in accordance with their requirements. Moreover, the resources may be physically situated in different data centers, and the customers may not have complete control over the location of the resources assigned to them. However, the client may give an approximate geographic location. The term "resources" refers to a variety of things, including storage, memory, and network bandwidth. Elasticity is the fourth essential characteristic of cloud systems. To create an easily scalable environment, cloud resources must be assigned and released automatically. Finally, both parties' resource utilization must be automatically monitored, managed, and reported [3, pp. 9-11]. The wide range of cloud computing services encourages competition amongst cloud service providers, resulting in an even wider range of services and, ultimately, increased customer benefits.

2.1.2 Cloud Computing Architecture

The architecture of cloud computing systems is composed of two critical elements : three service delivery models and four deployment models [2, p. 2]. Each delivery service and deployment model combination offers distinct advantages and targets unique consumer segments [3, p. 43]. The most commonly acknowledged classification approach for cloud computing is the Software-Platform-Infrastructure (SPI) paradigm. Cloud computing delivers three basic service models: software as a service (SaaS), platform as a service (PaaS), and infrastructure as a service (IaaS) [2, pp. 2-3].

The three service models are classified hierarchically according to the client's level of control and security over the cloud services. SaaS solutions provide the least flexibility and carry the least responsibility for service security from the customer's perspective. PaaS has a central position in the hierarchy, providing some flexibility while delegating certain security responsibilities to the customer. Due to the high level of customization possible with IaaS, the cloud service provider cannot guarantee the security of the applications and must instead provide security at the platform level. [3, pp. 34–37.]

The SaaS model is one in which the supplier makes software available to the consumer on-demand. The provider hosts the software on its infrastructure and has complete control over the cloud architecture and application operating settings. The customer often accesses the applications through a website browser and, in certain circumstances, has restricted access to user-specific setting options. By using a SaaS solution, the client avoids the requirement to assure software-hardware compatibility and the responsibility for application maintenance, since the provider is responsible for both. Additionally, the SaaS model allows service providers to monitor and prevent illegitimate application copies from being distributed. [3, pp. 37-39.]

IaaS suppliers offer customers real and virtual computing resources. The client does not manage or control the cloud infrastructure at its core. The client, on the other hand, has the ability to install, operate, and configure any kind of software,

including operating systems. The IaaS architecture enables rapid scaling and may significantly minimize the expenses associated with acquiring, deploying, and maintaining physical resources. Additionally, owing to the automated nature of cloud services, the time required to bring the infrastructure online for real usage is significantly decreased when compared to acquiring and setting up physical equipment. [2, p. 3.]

PaaS, or Platform as a Service, is a third cloud service paradigm that fits between SaaS and IaaS. This model offers much more than the ability to leverage an existing software application. PaaS services offer the opportunity to design, create, and maintain applications. Furthermore, PaaS offers the cloud infrastructure required to store and operate applications. As with other cloud service models, the user is not responsible for managing the cloud infrastructure's essential components. As with other cloud service models, the user is not responsible for managing the critical components of the cloud infrastructure. Nevertheless, the user has complete control over the applications installed, and customization of the application environment is possible depending on the service provider. [2, pp. 2-3.] The primary downside of PaaS is that many providers restrict clients to proprietary development environments and tools, limiting their ability to deploy applications created in a certain programming language. To address these constraints, several PaaS systems are open source. The Microsoft Azure platform, for example, is an example of an open PaaS solution since it supports both Microsoft proprietary and non-proprietary programming languages and tools. [4, pp. 206-212.] The Microsoft Azure platform will be examined in further detail.

In addition, the NIST identifies four cloud deployment models: private, public, community, and hybrid. Private clouds are often hosted on-premises and owned by the cloud's operator. The private cloud infrastructure can be owned, managed, and operated by the same business or by a third-party cloud service provider. The major goal of private clouds is to increase data security, which is accomplished by restricting access to the cloud to the beneficiary firm. [2, p. 3.]

Public clouds are widely used cloud systems in which the cloud service provider owns and manages the infrastructure. Additionally, the cloud infrastructure must be located in the region of the cloud service provider. Private clouds are derived from public clouds. The primary distinction between the two is that private clouds are dedicated to a particular organization or business, while public clouds are accessible to the entire public. [2, p. 3.]

When two or more firms share interests, operate in the same industrial region, and agree to share cloud infrastructure, community clouds are formed. Simply said, community clouds are public clouds that are provided to a specific community of customers or enterprises for their exclusive usage. A crucial characteristic of community clouds is their independence from the continuing existence of any of the enterprises involved in the cloud's structure. [2, p. 3.]

A hybrid cloud is formed by integrating any of the above-mentioned cloud deployment options. A hybrid cloud is made up of two or more cloud infrastructures connected by technology that facilitates data and application mobility. Hybrid clouds provide the optimal cloud solution without sacrificing security or cost and are the most often used deployment strategy for organizations or major corporations. [2, p. 3.]

2.2 Artificial Intelligence and Machine Learning

The capacity for learning is a critical aspect of human intelligence. Several decades ago, that was considered the sole context in which the term "learning" could be applied. Nowadays, artificial intelligence (AI) is a large branch of computer science that focuses on the creation of intelligent computers capable of performing tasks that are traditionally performed by humans. The term "machine learning" refers to the development of systems that perform tasks associated with artificial intelligence.

2.2.1 Machine learning

For a long time, a significant distinction between people and computers has been that human beings tend to naturally improve their approach to a problem. Humans learn from their errors and attempt to remedy them or find new ways to the issue. Due to the fact that traditional computer programs do not consider the result of their job, they are incapable of improving their behavior. The discipline of machine learning tackles this exact issue by developing computer programs that are capable of learning and thereby improving their performance as a result of more data and experience.

Arthur Samuel was the first scientist to construct a self-learning software in 1952 when he built a program that improved the game checkers as the number of games played increased [5, pp. 1122-1123]. In 1967, the first pattern recognition software was developed. It was capable of detecting patterns in data by comparing new input to previously stored data and identifying similarities. Machine learning has been employed in data mining, adaptive software systems, and text and language learning disciplines since the 1990s. As an example, computer software that collects data on the users of an e-commerce store and uses this data to make more tailored adverts has the potential to gain new knowledge and gets dangerously near to having artificial intelligence.

Machine learning is a subfield of computer science and a type of artificial intelligence application. Machine learning is primarily concerned with the development of applications/algorithms that adapt and improve on their own in response to fresh input. Daily, around 2.5 exabytes of data are generated, the majority of it is unstructured such as audio, videos, and documents [6]. It is a significant challenge to process this data efficiently and effectively. Machine learning enables the examination of this volume of data in the shortest amount of time possible in order to identify patterns and develop insights. These results can be generalized using machine learning technologies and hence utilized to solve new problems or analyze previously unknown data. The “changes” might be either improvement to already existing systems or building whole new systems.

Nevertheless, forecasts or predictions from machine learning make applications and devices smarter and represent an impressive added value for many projects.

2.2.2 Neural Network and Deep Learning

Neural networks, sometimes referred to as artificial neural networks or simulated neural network, are a subset of machine learning that form the basis of deep learning techniques, that employs a network of functions to recognize and transform one type of data input into another type of data output. The neural network is comparable to the human brain in that it is capable of performing tasks similar to those performed by the human brain. The neural network can be utilized in many algorithms to analyze complicated data inputs in such a way that the computer can readily interpret them. Neural networks are used to solve a variety of real problems, including image processing, pattern identification, speech recognition, and medical diagnosis. Machine learning employs a variety of methodologies and technologies, the most important and widely utilized of which is the neural network.

Convolutional neural networks (CNNs) are a subclass of artificial neural networks that are one of the most often used deep learning designs for image recognition applications. A CNN (cf. figure 1) is composed of input and output layers, as well as many hidden layers. Each unit in a layer is linked to each unit in the subsequent layer in a fully connected neural network. The photos to categorize is an input layer, from which the network receives all necessary data, in this case. Hidden layers exist between the input and output layers. Each hidden layer is used to identify a distinct collection of characteristics in a picture, starting with the simplest and progressing to the most detailed. Convolutional layers are among these hidden layers; they simulate the response of a neuron to visual stimuli. After the input layer is processed in a CNN, the picture is processed in many connected convolutional hidden layers. The initial hidden layer will catch rough edges and color, while the subsequent layers will capture finer details and characteristics. The network produces predictions at the output layer. The predicted picture categorizations are compared to the human-added labels. If they are inaccurate,

the network employs a process called backpropagation to rectify its learning, allowing it to generate more accurate estimates in the subsequent iteration. Backpropagation is a technique that enables neural networks to learn their parameters, mostly via prediction failures. [7.] After sufficient training, a network can generate classifications automatically and without the assistance of humans.

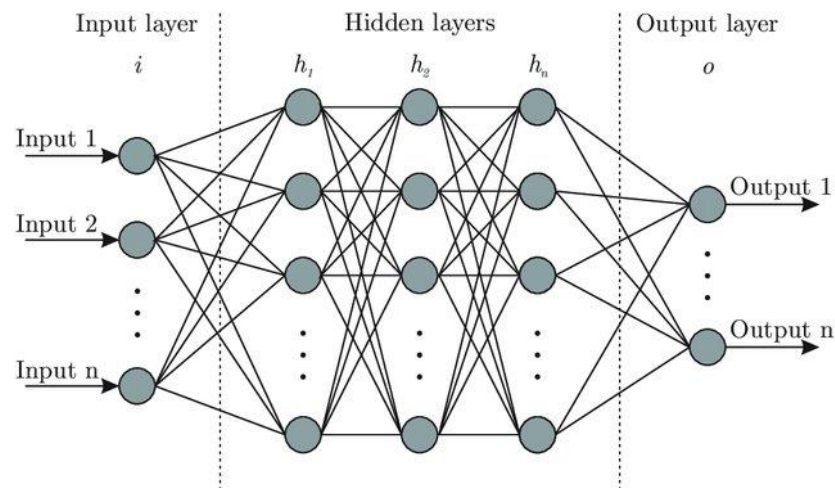


Figure 1: Neural network structure. Copied from Bre, Facundo et al [8, p. 4].

Deep learning is a subfield of machine learning and deep learning techniques are built on neural networks. Deep learning is a type of machine learning method that enables computers to learn from their experiences and comprehend the structure of abstract concepts. Due to the fact that the computer gains knowledge from prior experience, a human-computer operator is not required to describe any of the information required by the computer. The hierarchy of conceptions enables the computer to acquire complex concepts by creating them from smaller concepts and a graph of these hierarchies has multiple layers of depth. [9, p. 6.] Due to its primarily hierarchical structure and extensive learning capability, deep learning models are particularly effective in categorization and forecasting, while also being scalable and adaptable to a wide variety of critically crucial challenges. The figure below illustrates that a branch of artificial intelligence is machine learning, and deep learning is a subset of machine learning that is particularly well-suited for tasks such as feature extraction.

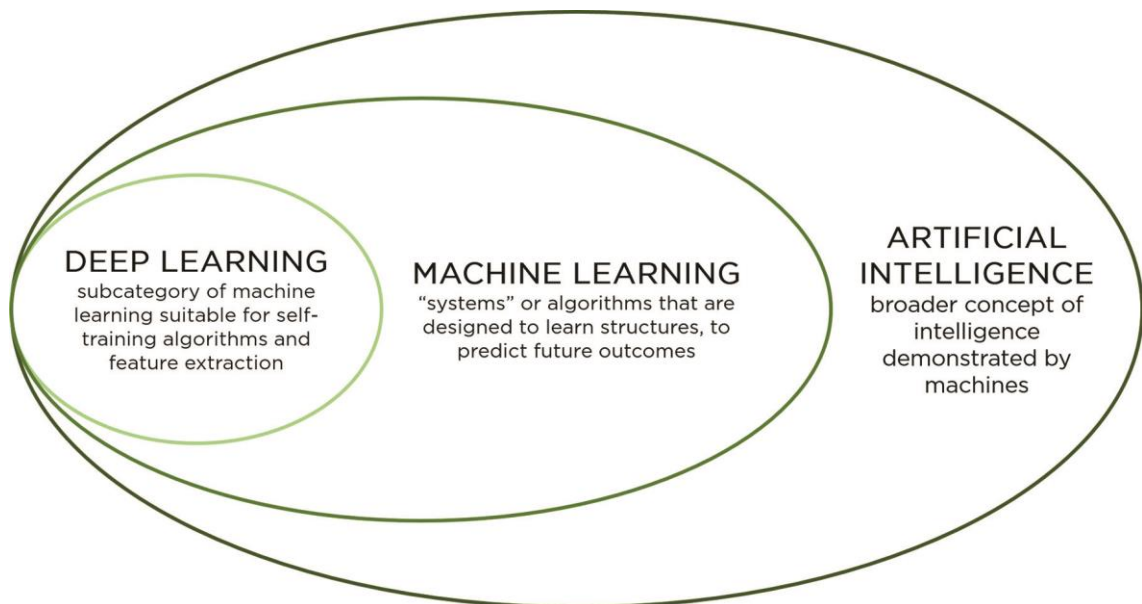


Figure 2: The relationship between artificial intelligence, machine learning, and deep learning.

Copied from Townend, Oliver et al [10].

2.2.3 Computer Vision

Computer vision is a branch of artificial intelligence and computer science concerned with assisting computers in obtaining the ability to see and comprehend in the same manner that humans do. Humans seem to be adept at extracting information from visual inputs. People are capable of interpreting distances and color information, distinguishing fore- and background objects, as well as identifying persons and classifying items. They can track moving objects in a series of photos and extract precise spatial information from the video. Convolutional neural networks underpin modern computer vision algorithms, providing a huge performance boost over older image processing techniques.

Developing a machine that appears to sense in the same way as humans does not come easily, not just due to the time commitment, but also because people do not fully understand how the viewing process works. Nobody believes this is straightforward, with the possible exception of artificial intelligence pioneer Marvin Minsky [11]. However, computer vision was only capable of performing limited tasks. The discipline has made great progress in recent years, exceeding humans in various tasks involving object recognition and categorization as a

result of advancements in artificial intelligence and discoveries in deep learning and neural networks.

For instance, two datasets may be provided to a neural network: one named "cat" includes photos of cats, while the other named "no cat" contains images empty of cats. On the basis of these datasets, the Neural Network may be trained to recognize images of cats without being taught what a cat is. It establishes guidelines for determining what each picture set has in common with the rest of the photos and how the image sets vary from one another. This is the fundamental method by which computers use computer vision to accomplish previously performed tasks by humans. As the dataset expands, the neural network's algorithm may be refined, boosting its accuracy and ability to execute successfully in more difficult tasks.

2.3 Cloud-based Artificial Intelligence Service

“Cloud-based artificial intelligence services” is a general concept for all cloud computing services related to AI. Either one or more artificial intelligence-based technologies are employed, or these are services dedicated to the development of AI applications. Different technologies are utilized to access these services. Through easy API requests, users can access high-quality AI models for vision, voice, language, and decision-making, as well as develop their own machine learning models using tools such as Jupyter Notebooks, Visual Studio Code, and open-source frameworks. The next section discusses the many technologies that are utilized to evaluate cloud-based artificial intelligence Services.

2.3.1 REST API

An API is a software interface or communication protocol that determines the manner in which computer programs connect with one another via a network. A contract between a protocol, a data format, and an endpoint is defined by an API. The primary distinction between APIs and websites is that, although websites

publish content that is consumed by users, APIs do not. The appearance, structure, and content of the site are subject to change at any moment without prior notification to users. On the other hand, an API is similar to a contract that cannot be altered once it is released due to the fact that it may be used by several other programs.

REST is an acronym for Representational State Transfer. It is a style of software architecture developed by Fielding that defines the principles for developing web services. REST makes considerable use of HTTP capabilities to work. A complete set of HTTP verbs (GET, POST, PUT, PATCH, DELETE), header-based authentication, and cache ability indication in answers are all examples [12]. It was extensively accepted because of its simplicity and performance, ease of implementation, and lack of specialized software. Typically, URL endpoints indicate a data object with which a client wishes to interact. RESTful web services (cf. figure 3), by virtue of their necessity for a consistent interface, provide cross-system interoperability and are hence suited for cloud environments. REST APIs or RESTful APIs are the terms used to refer to these interfaces. In most situations, these interfaces make use of the HTTP protocol, which eliminates the need for extra libraries or applications. URLs are used to access data. Each URL corresponds to a request, and the data returned corresponds to the response. REST APIs are the most commonly offered interfaces for cloud services.

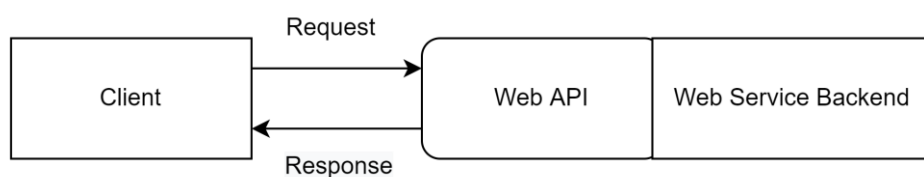


Figure 3: Request and response on web API.

2.3.2 Software Development Kit

A software development kit (SDK) is a collection of software development tools with the purpose of developing customized software. These can be customized for a particular piece of hardware or platform. Additionally, it contains a set of

tools, libraries, relevant documentation, code samples, methods, and guidance that enable developers to create software applications for a specific platform. Often, at least one API is provided in the SDK, as without the API, programs cannot exchange data or collaborate. Certain cloud providers offer SDKs for a variety of programming languages, providing an alternate interface for their services while comparing to REST APIs.

2.3.3 Application Programming Interface Key

Application programming interface (API) keys are widely used in cloud computing for authenticating access to online services. The API key is often allocated to a specific service and acts as both an identity and a security token for authentication. If API keys are utilized, they must also be sent when a service is accessed. APIs are interfaces that assist in the development of software and describe how various pieces of software communicate. The API key regulates the requests made between programs, the manner in which it is made, and the data formats that are utilized. It is often used to collect and analyze data in Internet-of-Things applications and websites, as well as to let people submit data. For instance, customers may get a Google API key or a YouTube API key through an API key generator.

2.3.4 JavaScript Object Notation

JSON is a standardized data interchange format that is simple to understand by humans. It enables serializable objects to be exchanged. JSON is a lightweight, standardized file format for structured communication that originated with the JavaScript programming language. Although JSON has native JavaScript capabilities, it is JavaScript-independent. JSON is widely used and has a natural syntax similar to those of other computer languages. JSON has become the standard for modern programming interfaces due to its simplicity and convenience of use. JSON is a data format composed of two types of data: collections of key-value pairs and ordered lists of items. The majority of REST

APIs represent payloads in JSON format. Additionally, a JSON file (cf. figure 4) may include tables and nested JSON objects.

```
{
  "id": "46ca813a-23c5-4844-a2cd-a751cb544bdc",
  "name": "Lorem elit 6",
  "description": "enim incidunt dolore aute velit",
  "targetIds": [
    "e54f0ca2-03d5-4b2e-98e8-0d4dc5dee13e"
  ],
  "labelIds": [
    "ac03ed0d-dd94-4ae1-9ef0-2eafcca4b285"
  ],
  "createdAt": "1953-03-28T22:29:17.684Z",
  "updatedAt": "1942-06-16T13:08:02.437Z",
  "deletedAt": "1979-10-23T01:07:24.702Z",
  "organizationId": "urn:uuid:81ae83e1-103e-409e-c1c4-92a768ccf3a5",
  "creatorId": "8fa21f6b-7590-a06d-14c8-6b1354128382",
  "ruleType": "level of information"
}
```

Figure 4: JSON file.

2.3.5 Frameworks

Frameworks provide developers with a programming framework to build applications. These are domain-specific and facilitate application development in their particular areas of application. They include libraries with well-defined interfaces that abstract away portions of the application's functionality. Numerous machine learning frameworks are available today that enable developers to design and train their own models quickly and easily. In general, a framework is more extensive than a protocol and more prescriptive than a structure.

2.4 Microsoft Azure – Cloud Computing Services

Microsoft Azure is a cloud computing platform and website portal for managing and accessing Microsoft's cloud services and resources. These services and resources include storing and transforming data toward the requirements of the

user. It delivers software, platforms, and infrastructure systems, as well as support for a variety of programming languages, frameworks, and tools. These services vary from those designed to address certain minor use cases to those that provide infrastructure for a variety of situations. Microsoft has established a worldwide network of data centers to assure the highest possible availability of Azure services. Multiple regions may be used to assign resources for any service. Unlike Microsoft's rivals' offerings, which either supply raw resources or very restricted development tools, the Azure platform provides a more controlled experience. Azure's services and tools are intended to aid developers in creating readily scalable and managed applications.

Microsoft Azure is composed of two critical structures: IaaS, and PaaS. With the IaaS structure, virtual machine, and server offerings from Microsoft are all accessible. Azure will handle both the logic and the operating system of the device via a single virtual machine. Following that, PaaS Azure's cloud services handle all operating systems, allowing users to concentrate only on mission-critical operations. It gives a high-quality application to the users. PaaS Azure structure is the most straightforward and effective way to migrate an application to the cloud. Because it offers more than just IaaS service, Azure differs from Microsoft's main rival and the most popular supplier of cloud services, Amazon. Rather than just delivering hardware resources, Azure also assists in the development of software applications using the available tools. [13.] Additionally, unlike Google's cloud services offering, Azure development possibilities are not limited to a few chosen programming languages. Azure is a highly scalable open platform, which means that software developers can choose from a diverse set of development tools. As indicated in figure 5, the Azure infrastructure is composed of three primary components: computation, storage, and the fabric controller. Additionally, Microsoft has said expressly that it intends to provide Azure services to both independent software sellers and business developers [14, p. 2].

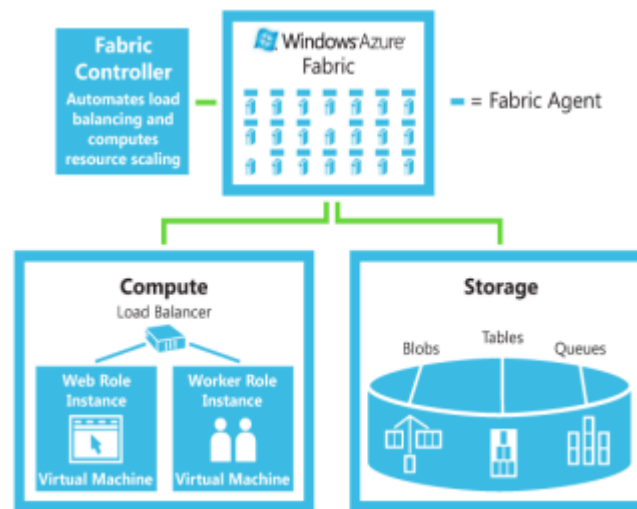


Figure 5: Azure components. Copied from Maini, Raman et al [15, p. 75].

As is the case with the majority of comparable cloud environments, Azure provides its services through a website-based interface that is used to install and configure various components of the environment. Due to the way Azure services are constructed, developing applications that use them need a unique methodology. The Azure platform's technologies and capabilities may be summarized as follows. To begin, the compute service offers processing power for the applications, which are hosted in separate virtual machines (VMs) running the developer's preferred operating system, such as Windows Server or Ubuntu. However, the virtual machines themselves operate on the Windows Azure operating system. The storage service then offers storage capacity on servers in Microsoft-owned data centers. Finally, the fabric controller represents the application that is responsible for managing communication between computing and storage services.

Microsoft Azure is being embraced by businesses to boost their efficiency and competitiveness due to its many exceptional features and flexibility. Microsoft businesses may now produce and operate, demand flexibility, and avoid paying for user fees by just paying for what they use. Azure also integrates easily with other Microsoft products and services, including Windows Server, SQL Server, Exchange, and Share Point, to securely back up and store critical business data. Furthermore, Azure enables enterprises to migrate programs swiftly from on-premises servers to the cloud and also bring their products as well as services to

the market easily. This enables businesses to avoid the high cost of acquiring new servers. Azure backs up business data and applications in a simple and dependable manner. Additionally, Azure is a beneficial tool for developers since it supports a diverse set of operating systems, programming languages, frameworks, databases, and devices. Consistency in application development, administration, and security, as well as identity management and data platform development, is supported by Azure.

2.5 Azure Cognitive Services

2.5.1 Fundamentals of Azure Cognitive Services

Microsoft Cognitive Services is a collection of artificial intelligence application APIs that enables programmers at all levels, from students writing their first program to professionals working for large enterprises and organizations, to enhance the intelligence, engagement, and discoverability of their applications. Cloud Cognitive Computing is a concept that refers to cloud computing systems that use machine learning, reasoning, natural language processing, speech, and vision in order to deliver data insights and support people in making better decisions [16]. It is intended to provide more customized computing experiences and better productivity via the use of increasingly competent computers that can see, hear, communicate, understand, and even think. Cloud Cognitive Computing is offered as a set of application programming interfaces that allow the storage, transformation, analysis, and presentation of data. With REST APIs and client library SDKs, Azure Cognitive Services enable developers to incorporate cognitive intelligence into their applications. Cognitive Services APIs are designed as REST APIs, which enables programmers to integrate them on a wide variety of platforms, including iOS, Android, and Windows, simply by connecting to the Internet. Microsoft Cognitive Services now comprises 21 APIs organized into five categories:

- Vision: content and other important information can be analyzed from photographs and videos.
- Speech: it enhances voice recognition and identifies the speaker.
- Language: it comprehends phrases and meaning rather than only words.
- Knowledge: this category identifies and collects research from scientific publications.
- Search: machine learning is used for online searches.

2.5.2 Cognitive Vision API

It is an API group dedicated to image processing; Microsoft offers the following four APIs under this category:

1. Computer Vision API: this API enables the extraction of important information from photographs, such as identifying the sort of item in the picture or, if the character in the image is a person, determining the figure's gender. Additionally, this API facilitates the recognition of well-known characters and the extraction of text from photographs.
2. Face API: as the name implies, this is the API for detecting faces in photographs. Moreover, this API offers information on the face's features, such as its age, gender, the brilliance of the grin, and even the length of the hair. Face matching, face attributes, and characteristic analysis are all possible uses of the service.
 - Face Recognition API: based on facial detection, it analyzes two or more photos to identify whether or not they include the same face. It may be used to verify and identify a person's face.
 - Face Detection API: in order to identify a particular face in a picture, this API gives information about the faces that have been identified.

- Emotion API: the Emotion API accepts an image as input and provides an expression for each face in the picture across a range of emotions. This API enables users to assess the mood of the individual in a photograph by determining whether they are happy, sad, angry, fearful, or neutral.
3. Custom Vision Service API: its purpose is to develop picture categorization models that "learn" from the labeled photos provided by users. Images of dogs may be used to determine their breed, for example.
 4. Video API: this API is a collection of Microsoft's sophisticated video processing techniques. Developers may use the Video API to integrate video editing tools such as picture stabilization, face recognition, motion detection, and thumbnail creation.

2.6 Web application Service

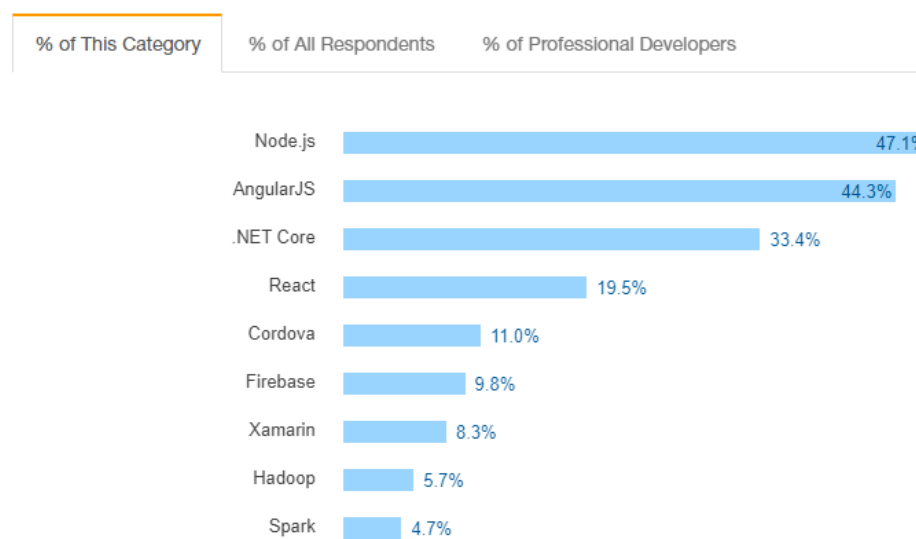
A web application, commonly shortened as a web app, is an interactive computer program created in web technologies such as HTML, CSS, and JS that stores and manipulates data through the use of four fundamental functions known as CRUD. It is used by a team or a single user to execute activities over the internet. CRUD is a well-known acronym that is important to develop an application that utilizes persistent storage. It is an acronym that stands for Create, Read, Update, and Delete. This section presents the tools used to develop and test web applications.

2.6.1 Node.js

Using the Chrome browser's V8 JavaScript Runtime Environment, Node.js is a JavaScript transpilation platform that is exceptionally fast. Web and server-side applications are only two examples of how Node.js may be used to create a wide range of applications. PayPal, LinkedIn, eBay, and Walmart are just a few of the firms that have recently made the transition to Node.js. [17, p. 11.]

Node.js's single-thread event-driven design and asynchronous functions make it ideal for large-scale network applications. Furthermore, Node.js is built to deal with the most difficult and significant problems that arise while building out applications. It utilizes JavaScript on the server-side and asynchronous non-blocking I/O. Instead of the more traditional Java, Python, PHP, Ruby, Node.js is seen as an effective option in today's online architecture. For a variety of reasons, Node.js has been the most popular framework for developing websites in recent years. Because Node.js is written in JavaScript, which was also introduced with ES6, it is straightforward and convenient from the start. As a result, no more expertise is necessary. Node.js has also developed itself effectively to achieve high execution speed and scalability, as previously discussed. [17, pp. 14-17.] Last but not least, the large Node.js community may be of benefit to developers in any case. The figure below shows that Node.js is the most commonly used in different fields.

Frameworks, Libraries, and Other Technologies



20,229 responses; select all that apply

Figure 6: Node.js is the most commonly used technologies in this category.

Copied from Stack Overflow [18].

However, it is possible that Node.js has limits that cannot be overcome. Node.js is really not meant to be just a substitute for Apache, for example, code written in

Node.js is less comfortable to deal with as its complexity increases. This implies that programmers must have low-level access to HTTP and other protocols in order to do their work. Many frameworks in the community, such as Express, Rails, and Meteor, are available to aid with this problem by supporting the higher level for programmers.

Node.js has several drawbacks, such as the fact that it is in the process of being developed. Certain features may need to be revised as a result. On the other hand, the popularity of Node.js as a development tool has exploded in recent years. Node.js is a game-changer in website development since before it, backend developers had to determine which server-side language was best for their projects, such as PHP, Python, or even Ruby, while frontend developers still utilized JavaScript on the client-side. [17, pp. 19-20.]

Because Node.js was developed in JavaScript, it resolved conflicts between client and server programming. As a result, full-stack development became possible, from the first step of the user interface to the last stage of the database operations. In general, the code is more compact when utilizing just one language across the system, and the same data formats and tools are used for server-side and client-side development, as well as for testing. [17, pp. 22-23.] In addition, Node.js is based on Google's V8 engine, a JavaScript engine that is faster and more efficient.

However, as previously indicated, Node.js is a single-thread environment that uses the non-blocking I/O method in order to improve system performance. Traditional web programming, in particular, relies on blocking I/O to call back data. As a result, a backlog of open threads has built upon the server, causing a slowness. As a result of this, Node.js is built on a non-blocking I/O rather than a standard I/O, and an event loop with asynchronous functions is established to handle queueing jobs more efficiently. With this event-driven architecture, Node.js provides a more powerful tool for backend development.

Node.js, on the other hand, is a suitable candidate for microservice design, which is good for agile projects. Essentially, microservices are a software architecture

in which modules are broken down into microservices and hosted on separate servers. And Node.js is an excellent framework for implementing microservices.

2.6.2 ExpressJS

ExpressJS is a Node.js framework. It comes with several sophisticated features that make it ideal for a website or mobile development. It offers HTTP and middleware methods, which enhances the API's capability and simplicity of usage. As a result, Express has grown in popularity due to its many advantages, such as package support and increased performance when programming. [19, pp. 2-3.]

Two ideas established Express's reputation: minimalism and flexibility. The minimalization property enables developers to work more efficiently while maintaining a high level of effectiveness, whilst the flexibility attribute simplifies execution by accepting HTTP requests from the client and returning HTTP replies. Express is regarded as the greatest suited for high-traffic online applications and a diversified ecosystem that provides a flexible middleware system due to its superior performance. [19, pp. 2-3.]

Express is a routing layer made of a large number of modular processing units known as middleware. It has a collection of middleware applications that provide out-of-the-box support for common web application needs. Node relays HTTP requests/responses to Express, whose middleware handles the processing. Prior to sending requests to handlers, middleware functions process them. As can be seen from figure 7, a request-response cycle is capable of invoking a sequence of middleware functions. A middleware application has the ability to access and alter request and response items.

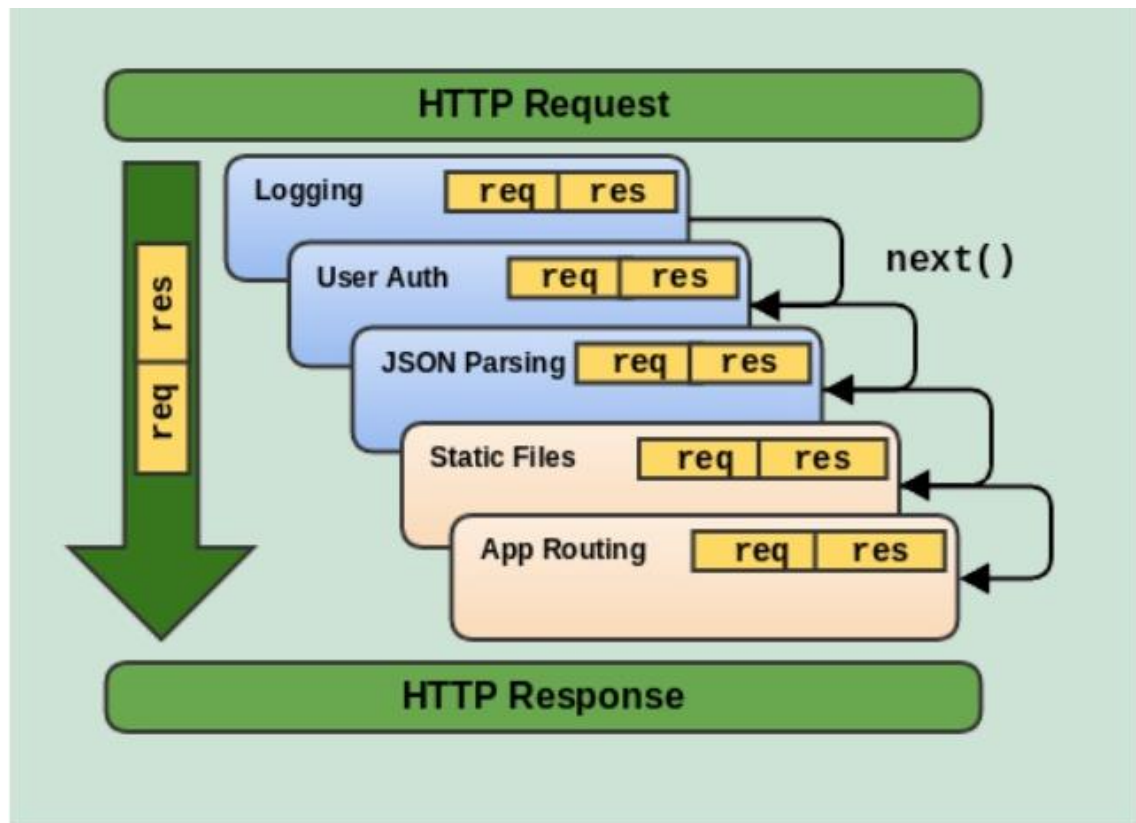


Figure 7: A layer stack in middleware stack of ExpressJs workflow. Copied from Grewe, Lynne [20].

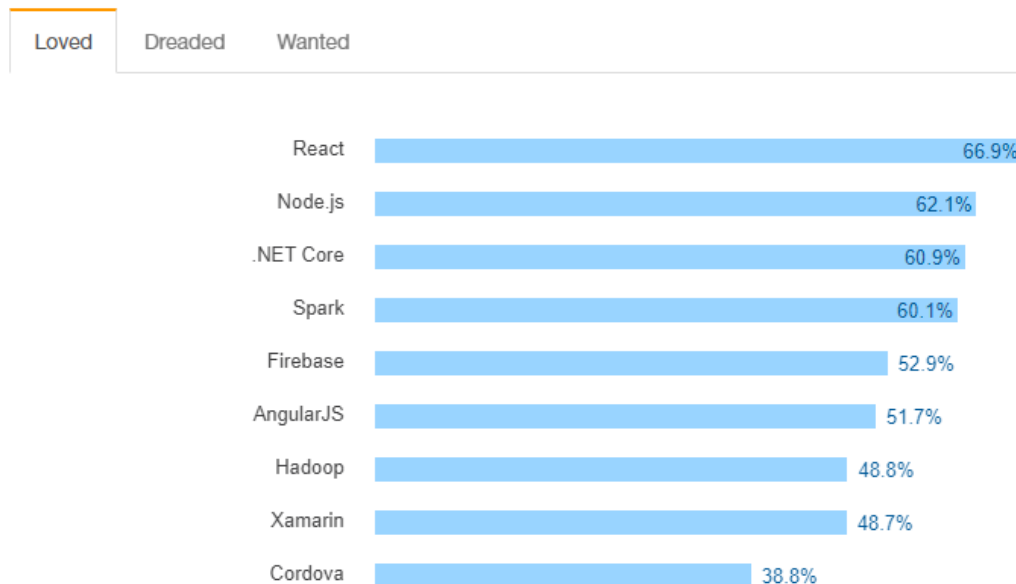
2.6.3 Node Package Manager – NPM

Several solutions for managing JavaScript libraries for Node.js have been developed in combination with Node.js development. Node Package Manager (NPM) is the most common JavaScript package management system. NPM was first published on January 13th, 2010, and was incorporated as part of the Node.js framework, making it the main package manager for JavaScript by market share, with more than half of the userbase at 60% when last verified in 2018. Nowadays, NPM Inc., a corporation created in 2014, maintains the NPM software as well as the NPM package repository, while providing paid services and support for corporate usage. [21.] The most recent stable version, 6.9.0, was published on March 6th, 2019. Since 2017 the NPM registry, the service that hosts packages contributed by developers, has served more than twice as many packages as the second-largest registry, Apache Maven [22].

There is a reason why NPM is called Node Package Manager: it was initially designed for use with the Node.js runtime environment for server-side programming. Nevertheless, NPM is often used for creating JavaScript user interfaces. An NPM package is a folder that contains a JavaScript program and a description of the program in the package.json file. Information about the package, its dependencies, and other NPM-related settings is included in the package.json file. NPM consists of an “npm” registry that provides numerous packages, a website for convenient package retrieval and browsing, and a command-line utility, named **npm**, for managing the packages. To install packages on a device or on a project-by-project basis, **npm install <package name>** is the most widely used command. NPM publishes and installs open-source packages for free, and it does not even need a login. With a premium ID, users may manage private packages. More than 1.3 million open-source packages are available on the NPM website.

2.6.4 React

Since its inception, React has evolved from a front-end framework with built-in JavaScript to a robust JavaScript library that is compatible with mobile applications, server-side, client-side, and perhaps virtual reality applications as well [23, p. 24]. Jordan Walke, a Facebook software developer, created React. It was first utilized on Facebook in 2011 and on Instagram in 2012. In 2013, React was formally published as an open-source framework for developers [24, p. 1]. React has become increasingly popular in recent years because of its numerous advantages for application development. Microsoft, PayPal, Tesla, Dropbox, Twitter, Disney, Netflix, Uber, or even Salesforce might be included on this list, except Facebook – the foundational one [23, p. 34]. It was voted the most loved web framework in 2019 by developers on Stack Overflow, as detailed in figure 8.



% of developers who are developing with the language or technology and have expressed interest in continuing to develop with it

Figure 8: The most loved Web Frameworks. Copied from Stack Overflow [18].

React contains many noteworthy features that may entice other developers to utilize it, including the speedy Virtual DOM, component-oriented development, and document-model abstraction. First of all, one of the primary reasons for React's success is Virtual DOM. The majority of web development strategies make use of the DOM, which consumes more resources as it fetches and uploads data and then reloads pages. This advancement may be more manageable, scalable, and integrated into the system. However, React overcame this limitation by implementing an exceedingly sophisticated technique for generating a Virtual DOM. By requiring an in-memory representation of the DOM, virtual DOM is more effective and faster than the DOM. Thus, whenever users interact or data is fetched, React will compare the current page to the newer one and then capture the new page, rather than doing the whole process as the DOM. As a result, applications built using React may hit 60 frames per second regardless of the device. [25, p. 3.]

Additionally, React is composed of several blocks that perform distinct functions inside a single layout. Components are the building pieces that may be used to create any kind of user interface in React, such as buttons, text fields, and forms.

Written in JavaScript and geared on components, React is believed to be the most advantageous to programmers when it integrates all of these features into one to target certain areas of a website. It is much better if the components are reusable in React. This implies that it has altered the way programming is conducted somewhat by focusing on components rather than distinct HTML, CSS, and JavaScript activities. It has the potential to alleviate the limitations of several technologies. Additionally, developers may easily edit the content of each section. [25, p. 3.]

The last benefit of React stems from the preceding concepts. That is the reduction of the document model to a user interface representation that is very lightweight. Thus, React adheres to standards, reduces inconsistencies across devices or browsers, and increases performance via component rendering. [25, p. 3.] Moreover, React is designed without regard for any presentation conventions. React is primarily responsible for rendering Views in the majority of models.

Before learning how React works, it is important to understand that React is an exceptionally powerful toolkit that enables developers to easily alter the user experience using JavaScript and possibly XML. If JavaScript is optimized for browsers, Node.js will act as a bridge between JavaScript applications and the system through the command-line interface. Besides that, Node.js often collaborates with **npm** to achieve maximum performance. [25, pp. 4-5.]

By evaluating differences between the recently altered page and the prior one and then performing activities, React employs the Virtual DOM to edit the representation with fewer resources than the conventional DOM [25, p. 45]. This advancement has resulted in React gaining a larger user base in the web development industry. Next, React guides the user interface by assigning specialized and focused content blocks called components to lower-level user interface responsibilities. React supports the optional JSX language, which is handy. By incorporating HTML code into JavaScript, React becomes very straightforward to use for Java developers. The only slight disadvantage of React is that it needs a translator to convert the JSX to JavaScript. Specifically, a typical React project includes a source folder that contains all necessary JavaScript

modules, an index.html page or a landing page that determines how the user interface should be displayed, a **package.json** file that contains all project information, and a module packager that is typically used to transform JSX and other related files. [25, pp. 4-5.]

2.6.5 Heroku

Heroku is a cloud platform for developing, building, delivering, monitoring, and scaling applications. It is a Platform as a Service cloud. [26.] It lets developers construct, execute, and manage cloud-based applications. Heroku began operations in 2007. Heroku is both the title of the object and the name of the firm that created it, and Heroku incorporated is the name of the corporation. Heroku is a Salesforce Incorporated subsidiary and is included in the Salesforce App cloud product portfolio.

Heroku, according to its description, is a platform as a service that supports a wide variety of programming languages. When Heroku was founded in 2007, it supported just Ruby; now, it supports Java, Node.js, Scala, Python, Clojure, PHP, and Go. Heroku is a cloud application development platform that enables users to create and manage applications without having to worry about hardware or servers. Heroku enables users by offering a platform that aids in the arrangement, setup, maintenance, and scalability of applications. [26.] Additionally, Heroku provides other management capabilities, such as data persistence tools.

Heroku is regarded as simple to use. Heroku is most advantageous for businesses under certain conditions. Heroku offers a free service model for small businesses, although tiered service packages are available for those with more particular business needs. Heroku's cloud service platform is based on a container architecture with integrated data services and a rich environment for deploying and operating contemporary applications. Heroku applications often have a unique domain name that is used to route HTTP requests to the appropriate container.

2.6.6 Netlify

Developers must manage the following activities when adding Continuous Deployment into their software development pipeline: DNS service, hosting service, security protection, and server for the website. Each of the tasks listed above requires a huge amount of development work on the part of developers. Historically, this was accomplished manually, which proved to be inefficient and costly. Nowadays, each of these discrete jobs is served by a variety of service providers. However, Netlify is the most robust option since it combines hosting, DDOS security, DNS services, and web application development into a single package. [27.]

Netlify is a San Francisco-based provider of website hosting infrastructure and automation solutions. Netlify is a better solution for automating current website development tasks. It integrates the hosting infrastructure, continuous integration, and deployment pipeline with a single process. Software developers no longer need to construct and manage a deployment service when they use Netlify. Netlify takes care of everything. Each time a software developer publishes an application, Netlify builds and deploys the web application automatically using just the configuration file supplied by the developer.

3 Project Specifications

Microsoft AI and Azure services may be used to develop both simple and advanced applications. User queries may now be handled in the shortest amount of time possible. A new level of development convenience and customer satisfaction has been achieved thanks to Microsoft AI - Azure Cognitive Services. As a matter of fact, its simplicity enables accurate information to be channeled for prompt service and suggested actions. Each service may be accessed individually, allowing users to adapt them to their application purposes. This thesis aims to create a web application that utilizes computer vision to perform a variety of activities with the assistance of Azure Cognitive Services.

3.1 Project Objective

Azure provides a variety of options for creating static web applications, depending on the level of functionality and control required by the program. Using computer vision and Azure Cognitive Services, this thesis aims to build a simple model that leverages machine learning and artificial intelligence to perform a variety of tasks. The goal is to create a user-friendly program that offers basic image execution functionality. In order to give the intended experience, the program includes everything a user sees and processes quickly. For example, the users enter a URL for a picture, which is then sent to Microsoft Azure, where the server processes it and returns JSON data that contains the following features:

- Identifying and recognizing the face of the person,
- Comparing and verifying two faces,
- Identifying, tagging, and describing objects in photographs.

3.2 Project Architecture

Figure 9 shows how the users may submit data as an image URL and choose the input object processing function. It also can be seen that all of the application's components and how they interact with the development environment. As soon as the data is received, it will be converted to a JSON object and sent directly to the Node Express Server. JSON objects are sent and received on the Azure Computing Cloud using HTTP / HTTPS protocols via the Node Express Server. The server functions as a mediator. All data will be stored in Microsoft Azure and given a unique identifier that can be used to call the database. Additionally, Azure Cognitive Service will manage user queries, apply Azure AI algorithm to evaluate photos, and deliver data in JSON form through HTTP/HTTPS protocol. The users will see the results of their input in the online application. With Netlify Cloud, the frontend website is developed in React, which is a simple and easy-to-use programming language. The Node.js and Node Express Server APIs include

methods for POST, GET, DELETE, and PUT data on the server. In addition, Heroku is the cloud platform that hosts the API.

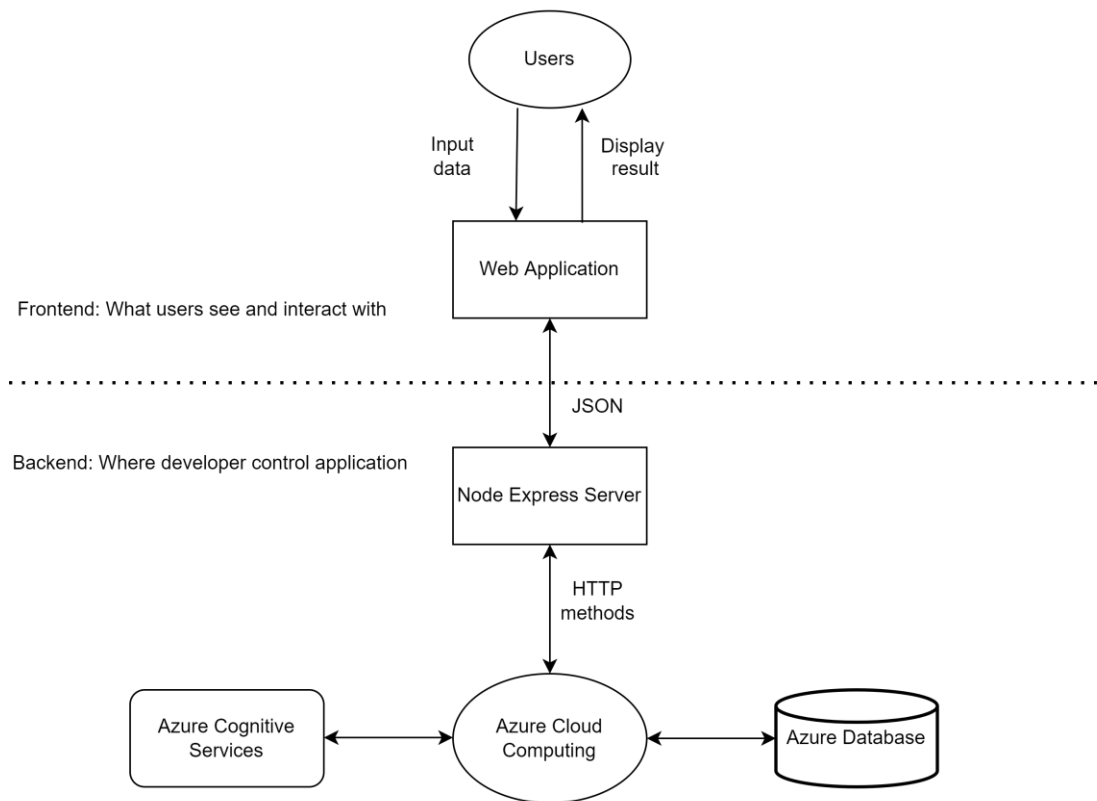


Figure 9: Project diagram.

The URL of the picture to be processed must be entered by the user, and the image must offer enough information for the program. JPEG (Joint Photographic Experts Group), PNG (Portable Network Graphic), GIF (Graphics Interchange Format), and BMP (Bitmap Image) are the image formats supported by the Azure Face API. The maximum file size for a photo is 1MB. A 1920x1080 pixel picture must include a minimum discernible face size of 36x36 pixels in order to be used. The application's primary interface enables users to view the data they have received. Bounding boxes enclosing entire faces or objects accompany outputs delivered as pictures. When users use the program, no picture is saved to the server; just the extracted facial characteristics are. After 24 hours from the initial detection call, these data will be destroyed. As a result, the photographs of individual users will remain private and not be used for other purposes.

4 Application Implementation

This chapter explains how to deploy web application components, including how to build the application architecture and configure Azure cloud servers. It outlines the general design, the way components are linked together, and the building blocks required. The specifics of each component's implementation will be discussed in the following sections.

4.1 Azure Cognitive Services Configuration

Developers may quickly add machine learning functionality to their applications using cognitive services. The Azure portal's Cognitive Services will be set up in this part. This section implies that users already have an Azure account in order to configure it. Vision API is responsible for managing data related to photos and videos. Building interactive applications that can identify photos and videos and extract useful data from them is possible with the help of this API.

To begin, users must get the Endpoint URL and Subscription Key for the Face APIs set from the Azure Cognitive Service marketplace in order to correctly use the server APIs. In computing, an endpoint is a distant computer that interacts with the network to which it is attached. When an API communicates with another system, the touchpoints are known as endpoints. An API's endpoint may be a server or service's URL, for example. Utilizing the Endpoint URL, a user may access all of Face APIs' services and features.

An API key or an application programming interface key is the code transmitted by computer applications. With the help of the API or application programming interface, the software or application then identifies the caller. In order to monitor and manage the use of interfaces, application programming keys are often used. In most cases, this is obtained to guard against the misuse of the aforementioned API. The API key may be used as both a unique identification and a secret authentication token. Access to the API to which the key is tied often comes with the key. Currently, Azure offers two sets of keys, with key 1 serving as the main

key and taking precedence in processing requests to the cloud. Figure 10 below illustrates the key and endpoint of Face API from cognitive services resources.

The screenshot shows the 'Keys and Endpoint' page for a Face API resource named 'DucVo-FaceAPI'. The page includes a search bar, 'Regenerate Key1' and 'Regenerate Key2' buttons, and a navigation sidebar with categories like Overview, Activity log, Access control (IAM), Tags, Diagnose and solve problems, Resource Management, and Monitoring. The 'Keys and Endpoint' section is active, displaying a warning message about key security, a 'Show Keys' button, and fields for KEY 1, KEY 2, Location/Region (westeurope), and Endpoint (https://ducvo-faceapi.cognitiveservices.azure.com/).

Figure 10. API key and endpoint of Face API.

Additionally, the total API calls (cf. figure 11) users have made, and the amount permitted to the price tier users choose may be seen at the very bottom of the overview page.

The screenshot shows the 'Overview' page for the Face API resource. The navigation sidebar is visible, with 'Overview' selected. The main content area includes a survey prompt, 'Essentials' information (Resource group: ducvo-resource, Status: Active, Location: West Europe, Subscription: Azure subscription 1, Subscription ID: 8f65e66f-598e-4e51-b51a-6b04512b0344, Tags: Click here to add tags), and a 'Monitoring' section. The 'Monitoring' section displays 'Quota info' with a bar chart showing 'Free tier Total' and 'Free tier Remaining' at 5000 Calls, and 'Free tier Rest period' at 30.00 Days.

Figure 11: Total number of API calls in Azure.

4.2 Azure Cognitive Online API

4.2.1 Recognize and Identify Face

Azure's online API testing console is one of its strongest features. The developer will be able to perform a test call to the server and see the structure of the JSON object that is received and transmitted, as well as any arguments that may be included in the call. POST, PUT, GET, DELETE, and other fundamental HTTP methods are used to call features that fulfill a common function. Only the extracted facial features are kept on the server when using Azure Cognitive Services, therefore there is no need to keep a picture. After 24 hours after the initial detection call, these data will be destroyed. Each picture has a faceID that may be used to identify it in all Face APIs services.

JPEG, PNG, GIF, and BMP are the image formats supported by the Azure Face API. A photo's file size is between 1KB and 1MB. To be used, a 1920x1080 pixel image must contain a minimum visible face size of 36x36 pixels. The minimum size for the face region with dimensions more than 1920x1080 pixels will increase correspondingly. A picture may be returned with the faces of up to 100 people. Faces are rated according to the size of their facial rectangles.

Http Method
POST
Host

Name

Query parameters

returnFaceId	<input type="text" value="true"/>	✕ Remove parameter
returnFaceLandmarks	<input type="text" value="false"/>	✕ Remove parameter
returnFaceAttributes	<input type="text" value="Value"/>	✕ Remove parameter
recognitionModel	<input type="text" value="recognition_04"/>	✕ Remove parameter
returnRecognitionModel	<input type="text" value="false"/>	✕ Remove parameter
detectionModel	<input type="text" value="detection_03"/>	✕ Remove parameter
faceIdTimeToLive	<input type="text" value="86400"/>	✕ Remove parameter

[+ Add parameter](#)

Headers

Content-Type	<input type="text" value="application/json"/>	✕ Remove header
Ocp-Apim-Subscription-Key	<input type="text" value="Value"/> !	

[+ Add header](#)

Figure 12: Face detection parameter.

When it comes to computer vision, face identification is the simplest application. The six fundamental parameters (cf. figure 12) for an API request to Azure Face Detection may be observed in the figure: face id, face landmarks, recognition model, detection model, face attributes, and face id time to live.

- Face ID: picture recognition software faceID uses a set of characters to identify the characteristics of a face taken from the image. FaceIDs are generated for every picture that is submitted to Azure Face APIs and may be used to access any other Face API service.
- Face landmarks: it is a method for identifying a person's facial traits. Pixel coordinates and their distinctive names, such as corners of the mouth and the corners of the eyes, as well as the silhouette of the jaws, may be

returned by the system, which can be used in a variety of ways. Using facial landmarks and identification algorithms, the system can identify emotions via facial movements and forecast how the face will evolve over time.

- Face attributes: facial expressions such as a smile or a frown are also included in this category. Occlusion, accessories, blur, exposure, and noise all play a role in the final image created by the subject's hair, spectacles, and other personal style choices. Attribute-specific returns may not be entirely correct.
- Recognition model: the default model is 'recognition 01', which was developed before March 2019. It is advised that users use 'recognition 02' rather than 'recognition 01', which has lower overall accuracy.
- Detection model: 'detection 01', which is recommended for face detection towards the front, is the default. Faces may not be identified in some situations, such as those with unusually big angle (head-pose) faces, obscured faces, or images with improper orientation. 'Detection 02', a new version published in 2019, is better at detecting faces with blurriness and reduced size. Face traits and landmarks, on the other hand, are not taken into consideration by this model.
- Face id time to live: the amount of time it takes for the face ID to be stored. A time span of 60 seconds to 864400 seconds is supported. 86400 is the default setting (24 hours).

The request body to Face API Detection mode is shown in figure 13. The JSON object field in the request contains the URL of the input picture. All the parameters described in the previous section are included in an example HTTP request provided by the online API testing console.

Figure 14 above describes a response form that might be sent to the developer. This section has three properties, gender, age, and emotion as well as four face rectangle values: height, width, and top. Developers may then create face-bounding boxes in photographs using this tool.

In order to do an object detection task, a given input picture contains a large number of potential targets, all of which must be recognized as well as their precise locations determined. Bounding boxes are used to specify the target places in the picture and in the text. For images, the box width and height coefficients, or the top-left/bottom-right x/y coordinate pairs, may be used to define the bounding box, which is an imaginary rectangular box around the predicted item. A limit box is thus represented by a collection of four real numbers. It is common practice to use bounding boxes in image processing when attempting to detect many objects in a single frame. Bounding boxes are often represented using one of two conventions:

- It defines the box in terms of its upper left and lower right coordinates.
- It defines the dimensions of the box, including the width and height measured from the box's center.

4.2.2 Compare and Verify Two Faces

Besides selecting face detection, the user opts for verification. The primary use of this characteristic is to determine whether two faces belong to the same person or if one belongs to a person in a certain group of people. Users just need to focus on the first goal: the resemblance of two faces in this scenario. It is easier to identify two faces with a high-quality photograph. If possible, users can choose faces that are frontal, crisp, and include a resolution of at least 200x200 pixels, or 100 pixels between each eye. The result indicates the degree of confidence in determining whether or not two faces belong to the same person.

When testing in detection mode, the user simply knows only the URL of the picture. While in verification mode, the server does not need a URL address but instead requests the image's faceID. If users wish to utilize the same faceID for all of the features of Azure Face Vision, they must provide it in the JSON object's request body (cf. figure 15), which is a feature.



Figure 15: Input images and their content.

This time, it must employ the face detection mode mentioned above in order to discover the faceID of the photographs. As seen in figure 12 above, two images of a football player are included, each with a faceID. Verifying mode will then employ these two faceIDs in the following manner (cf. figure 16). The POST method is used, as well as the URL for the request and a JSON example.

Request body

JSON fields in face to face verification request body:

Fields	Type	Description
faceId1	String	faceId of one face. comes from Face - Detect .
faceId2	String	faceId of another face. comes from Face - Detect .

```

1 {
2   "faceId1": "19cfeefa-ae73-4087-a261-548b2a3ee808",
3   "faceId2": "3bd65360-078e-449b-b7b7-59d4584d908b"
4 }
5

```

Request URL

```
https://westeurope.api.cognitive.microsoft.com/face/v1.0/verify
```

Figure 16: The request body and URL of Face Verify API.

'IsIdentical' and 'confidence' are two of the factor of response form's two objects. The value of 'isIdentical' indicates whether or not the two faces are similar. When a system calculates a confidence level, it is a value between 0 and 1, which represents a face-to-face match. From figure 17, it can be seen that the photographs are obviously pointing towards the same individual, as shown by the 93,5 percent confidence rating.

Response content

```

x-envoy-upstream-service-time: 5
apim-request-id: 0a652cdd-34b3-4efb-8909-d29514a1f226
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
x-content-type-options: nosniff
CSP-Billing-Usage: CognitiveServices.Face.Transaction=1
Date: Wed, 06 Apr 2022 08:39:05 GMT
Content-Length: 41
Content-Type: application/json; charset=utf-8

{
  "isIdentical": true,
  "confidence": 0.93539
}

```

Figure 17: The output of Face Verify API.

4.2.3 Object Identity

To set up this API in the Azure portal, the user selects Analyze Image from the Computer Vision API (cf. figure 18) guide online. Additionally, this feature is

designed to identify and deliver a list of the names of the things found in a picture, as well as provide a short description of each object in the image depending on its location. Image dimensions must be at least 50x50 pixels and the image size must not exceed 4MB.

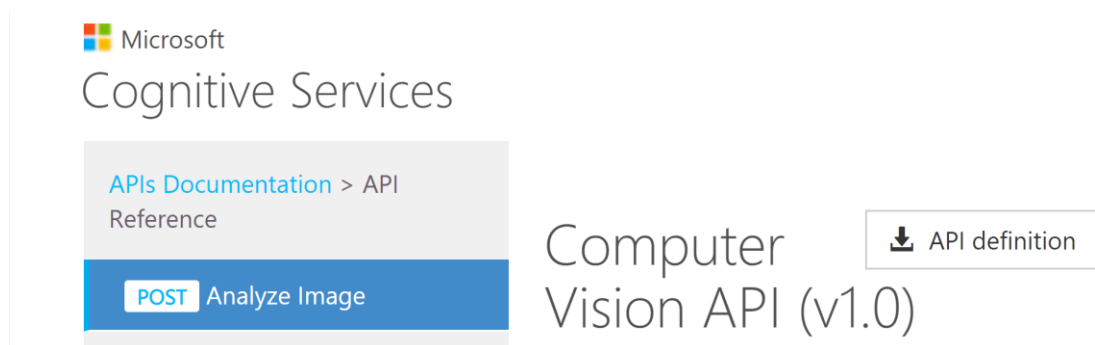


Figure 18: Computer Vision - Analyze Image mode feature.

In order to analyze photos and produce information, the Computer Vision API uses the most advanced algorithms. Users can use this function to determine whether a photograph contains adult content or if it contains any faces. In addition, it can estimate the dominant and accent colors, categorize photographs, and describe an image with entire English words. It is also capable of intelligently generating thumbnail pictures for the effective presentation of huge photographs. Figure 19 below describes three main parameters of this API.

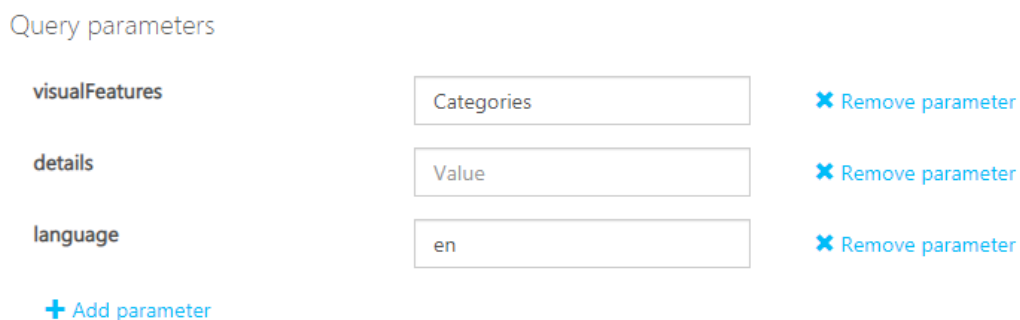


Figure 19: Computer Vision API parameters.

An Azure Analyze Image API request contains three fundamental parameters: visualFeatures, details, and language. For 'visualFeatures', it is included the following options:

- **Adult:** it is possible for a user to block adult content from being shown in their program by using an adult detection tool. In order for developers to interpret the outcomes of content flags in their own way, they are applied with a score between 0 and 1.
- **Brands:** this option identifies commercial brands in photos or videos from a database of thousands of worldwide logos. It is possible to utilize this function, for example, to find out which companies are the most popular on social media or the most often featured in media product placements. When a system service recognizes brand logos in a given picture, it provides the brand name, a level of confidence, and the location of a bounding box around the logo.
- **Color:** foreground color, background color, and the overall dominant color set are all provided by analyzing the colors in an image. Green, orange, yellow, pink, purple, and teal are among the hues that have been returned.
- **Description:** this feature analyze a picture and develop a statement that conveys what is in it in plain language. Numerous descriptions are generated by the system, each with a confidence score. Lists of descriptions, ranked from greatest to lowest in confidence, are the result of the final stage.
- **Faces:** this option detects faces in a picture and offers information about each face found. Each face recognized by this API is returned with its coordinates, rectangle, gender, and age.
- **ImageType:** characteristics such as whether an image is drawn by hand and how likely it is to be an image of clip art may all be determined using ImageType.

- Tags: hundreds of items, living organisms, landscapes, and behaviors may be used to identify and tag visual aspects in a picture. The API response includes clues to explain the context of the tag when the tags are unclear or not widely known. It is not just the primary subject, such as a person in the front, that is tagged, but also the location (indoors or outdoors), the equipment and materials used in the shot, the flowers or animals in the shot, and the clothing worn by the subject.

Only two possibilities are available for the detail parameter. The first is 'Celebrities', which uses an Azure database to keep track of all the famous individuals and picturesque spots in the photo. In addition, 'Landmarks' demands the system to produce findings based on data and algorithms collected from Azure accessible photographs and images. It is the user's responsibility to provide the language of the tags and the description in this parameter.

```
1 {"url": "https://i0.wp.com/www.muovijalelu.fi/wp-content/uploads/39514.jpg?fit=1400%2C1925&ssl=1"}
```

Request URL

```
https://westeurope.api.cognitive.microsoft.com/vision/v1.0/analyze?visualFeatures=Description&details=Landmarks&language=en
```

Figure 20: The request URL of Computer Vision API.

An HTTP call to the Computer Vision API Analyzing mode is shown in the image above. The method to be used is POST, and the URL for the request as well as a sample of JSON data will be provided. The Eiffel Tower URL (cf. figure 20) is used as the input picture in this example. In this procedure, the image's content is used to extract a wide range of visual characteristics. Uploading an image or supplying an image URL are the only two ways of submitting an image. Users may specify which features to return in their request using an optional argument. In the default answer, picture categories are included.

```

{
  "categories": [{
    "name": "building_",
    "score": 0.984375,
    "detail": {
      "landmarks": [{
        "name": "Eiffel Tower",
        "confidence": 0.9994469285011292
      }]
    }
  }],
  "description": {
    "tags": ["outdoor", "clock", "building", "tower", "large", "city", "front", "water", "sitting", "tall", "train", "cloudy",
    "track", "standing", "bridge", "old", "river", "church", "skiing"],
    "captions": [{
      "text": "a large clock tower towering over Eiffel Tower",
      "confidence": 0.7747231288138562
    }]
  },
  "requestId": "b3c5e613-1e95-425a-941d-0e0215ad5491",
  "metadata": {
    "height": 1925,
    "width": 1400,
    "format": "jpeg"
  }
}

```

Figure 21: The result of Computer Vision API.

JSON will be returned if the query is successful. If the request fails, an error number and message will be included in the response to explain what went wrong. The response data form contains four types: categories, a description, a requestID, and metadata (cf. figure 21):

- RequestID: faceID and requestID serve the same purpose. Basically, it is a string of letters and numbers used to identify a picture.
- Metadata: it provides information about the image's dimensions and format.
- Categories: the image's name, location, and a confidence rating for each category are all included in this array. This array delivers output tags (based on the objects, living beings, and behaviors recognized in the picture) and a human-readable statement that summarizes its contents with a confidence rating of 95% or above.

With a confidence ratio of up to 99.4% and the description "a large clock tower towering over Eiffel Tower," Azure has successfully located the precise Eiffel Tower in this case.

4.3 Face Detection Console Application

Adding face recognition to an application is a huge challenge for many developers since it is not often the major focus of the program, but it is nevertheless a crucial component. Microsoft Azure offers a variety of cognitive services, one of which is the face API. A client SDK or a REST API may be used locally or in the cloud to identify faces. In addition, face API provides client SDKs for .NET, Python, Java, Node.js, Go, and iOS. After gaining a grasp of the HTTP request's structure, the next step is to develop a small example demo for the face recognition program. A console application has been constructed here using the Azure Cognitive Services software development kit (SDK) and the **faceClient** library for .NET (cf. figure 22).

```

1  using System;
2  using System.Collections.Generic;
3  using System.IO;
4  using System.Linq;
5  using System.Threading;
6  using System.Threading.Tasks;
7  :
8  using Microsoft.Azure.CognitiveServices.Vision.Face;
9  using Microsoft.Azure.CognitiveServices.Vision.Face.Models;
10

```

Figure 22: The needed libraries and packages for .NET.

```

// API key and endpoint in the Azure portal.
private const string faceEndpoint =
    "https://ducvo-faceapi.cognitiveservices.azure.com/";

private const string apiKey = "2ccac62cb1134122875ea611970c7368";

private const string url1 = "https://www.suomifutis.com/wp-content/uploads/2021/12/Getty_messi-2-696x452.jpg";

```

Figure 23: API key, endpoint, and URL used for console application.

To authenticate the client, three objects must be initialized: the endpoint, the key, and the URL (cf. figure 23). In C#, these three objects are specified as private const strings and given three equivalent values. **FaceEndpoint** and **apikey** have been specified before; **url1** is the image address to be processed.

```

static void Main(string[] args)
{
    var faceClient = new FaceClient(new ApiKeyServiceClientCredentials(apiKey),
                                   new System.Net.Http.DelegatingHandler[] { })
    {
        Endpoint = faceEndpoint
    };

    FaceAttributeType[] faceAttributes = { FaceAttributeType.Age, FaceAttributeType.Gender,
                                           FaceAttributeType.Emotion, FaceAttributeType.Glasses};

    var faceResult1 = faceClient.Face.DetectWithUrlAsync(url1, true, false, faceAttributes).Result;

    Console.WriteLine($"FaceID: {faceResult1.First().FaceId}");
    Console.WriteLine($"Age: {faceResult1.First().FaceAttributes.Age}");
    Console.WriteLine($"Gender: {faceResult1.First().FaceAttributes.Gender}");
    Console.WriteLine($"Emotion: {faceResult1.First().FaceAttributes.Emotion.Happiness}");
    Console.WriteLine($"Glasses: {faceResult1.First().FaceAttributes.Glasses}");
}

```

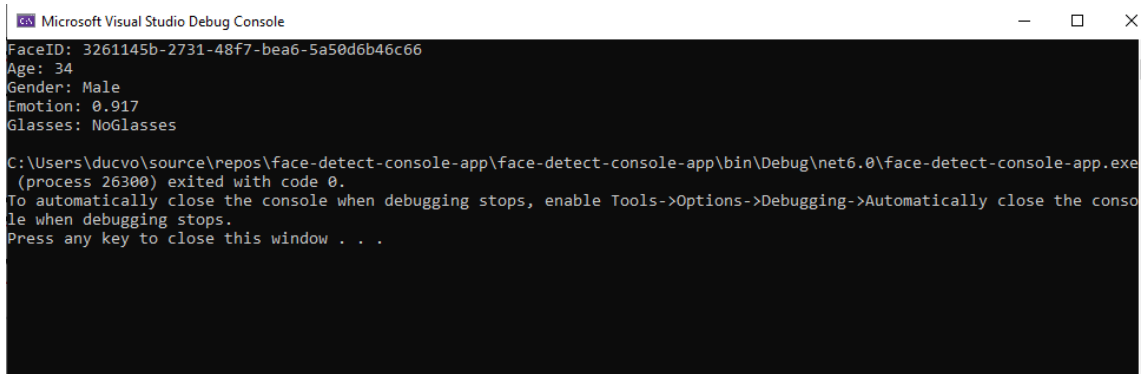
Figure 24: Main function of console application.

The main function (cf. figure 24) will comprise the major classes of the program. Firstly, **FaceClient** is the class that represents authorization to access the Face service, and it is required for the full operation. It is instantiated using the originator's subscription information and used to generate instances of other classes. As seen in the picture, this class includes a constructor named **ApiKeyServiceClientCredentials**, which enables authentication to the API using a simple API Key mechanism, with the `apiKey` argument being the subscription key previously stated. The value of **faceEndpoint** is set to `Endpoint` in the **FaceClient** class since this is the default parameter name for the API call route in the Azure Cognitive Services SDK.

Second, **DetectWithUrlAsync** is the name of the method that will be used for the **faceClient** object. This method provides four parameters:

- `url1`: a string contains the URL of the picture to be processed.
- `true`: a Boolean value returns the faceID.
- `false`: a Boolean value does not return the faceID confront landmarks.
- `faceAttributes`: this method returns all face attributes.

Finally, the **console.WriteLine** function displays all of the user's information, including their faceID, Age, Gender, Emotion, and Glasses.



```
Microsoft Visual Studio Debug Console
FaceID: 3261145b-2731-48f7-bea6-5a50d6b46c66
Age: 34
Gender: Male
Emotion: 0.917
Glasses: NoGlasses

C:\Users\ducvo\source\repos\face-detect-console-app\face-detect-console-app\bin\Debug\net6.0\face-detect-console-app.exe
(process 26300) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console
when debugging stops.
Press any key to close this window . . .
```

Figure 25: Result of console application.

The image above illustrates all of the data in detail of a football player, Messi. This is precisely what a console program that leverages the Azure API to recognize faces and deliver accurate data accomplishes.

4.4 Vision Web Application Implementation

The project organizes the application's primary components into distinct folders. The front-end makes use of a template created by **create-react-app**, which encloses the **node_modules** folder containing all required modules, and the public folder containing static files. The **src** folder includes source code, and the root directory includes the general configuration files for React application. All the configuration files are also located in the root directory of the project as displayed in figure 26. The backend server is a Node.js application that is interactively initialized using the **npm start** command. Additionally, the Axios HTTP client enables the creation of API applications in a variety of ways, from the simplest to the most complicated, on both browsers and Node.js servers.

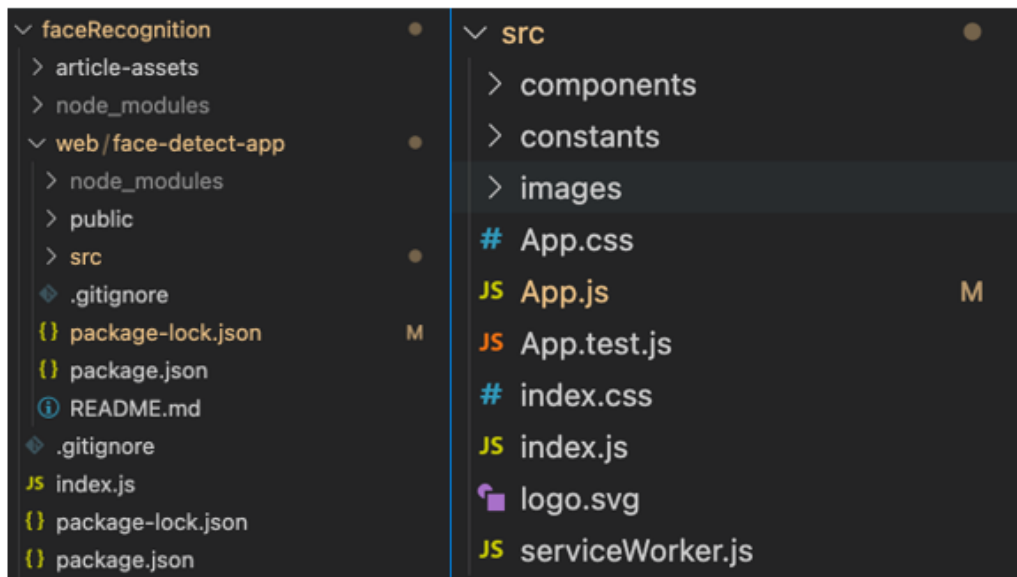


Figure 26: Application structure.

4.4.1 Server Implementation

After configuring Azure Cognitive Services, the next critical step is to set up the application's server environment. The first plan was to develop the application using the QT framework, however during the research phase, the Node.js platform was chosen to develop the complete application. There are two reasons why Node.js is an ideal match for this project:

- Node.js delivers a comprehensive library that is well-suited for this project's requirements. It includes a React library that simplifies the development of the web application frontend; the Express.js framework enables the creation of a robust server layer for establishing API interaction with Azure.
- Node.js runs cross-server and is lightweight and efficient due to its event-driven design and non-blocking I/O technique. Node.js applications are fast to respond in real-time and are cross-platform as well as multi-device compatible.

The following are some of ExpressJS 's most important functions:

- It creates intermediate classes that will respond to HTTP queries.
- It defines a router that enables it to be used with a variety of various actions depending on the HTTP method and URL.
- It allows for the retrieval of HTML pages in response to parameters.

ExpressJS may be installed using NPM or third-party package management. Once the ExpressJS download is complete, it may be imported into a Node.js project in the same manner as other Node modules. The first line retrieves the primary Express module from the installed package. This module is a function that is executed on the last line in order to initialize **app** variables. The server manages a large number of separate REST APIs for various purposes and executes distinct data.

```
2  const express = require('express')
3  const axios = require('axios')
4  const bodyParser = require('body-parser')
5  const path= require('path');
6  const app = express();
```

Figure 27: Needed modules and libraries for Axios.

Axios is an HTTP client library based on Promises that enables the development of basic to complicated API applications in the browser or Node.js. **Path** and **body-parser** are all the required libraries for parsing JSON, buffer, string, and URL encoded data supplied and for dealing with file and directory paths, as can be seen from figure 27. This module instantiates an **axios** instance with a custom **baseURL** to the API server. This is advantageous for data retrieval from a server and avoids the need to write the settings for each call.

```
// API key from Azure
const ApiKey1 = `2ccac62cb1134122875ea611970c7368`
const ApiKey2 = `584835fa01c941ebb7a20105dd6d2959`

// Azure endpoint URL
const AzureEndpoint1 = `https://ducvo-faceapi.cognitiveservices.azure.com/`
const AzureEndpoint2 = `https://ducvo-computervision.cognitiveservices.azure.com/`
```

Figure 28: API keys and endpoint URLs.

After that, the developer initializes the API key and the application's endpoint. Figure 28 illustrates that **APIKey1** and **AzureEndpoint1** are used to access the Face API, whereas **APIKey2** and **AzureEndpoint2** are used to access the Computer Vision API.

Because the server is based on Node.js and ExpressJS, it can handle both RESTful API calls and HTTP queries. ExpressJS supports HTTP protocols and middleware in order to provide a comprehensive and useful API. By using **middlewares**, ExpressJS enables developers to execute activities directly on HTTP requests and answers. The HTTP request and response have a lifecycle, which is completed when the **end()** function is used to provide a response to the client. Using middleware functions, the HTTP request or response may be retrieved throughout the request-response cycle.

These middlewares act as validators or authenticators, authorizing and validating access to and sharing of resources. Thus, rather than utilizing its own server directly like a conventional Express application would, it is wrapped in the HTTP server. To detect preflight requests, a **CORS** middleware (cf. figure 29) function was constructed. **Access-Control-Allow-Methods** is a list of HTTP methods that are currently supported. This header's value in replies contains all methods that are currently supported: GET, PATCH, PUT, POST, DELETE, and OPTIONS. Middleware functions have access to **req** (the request object), **res**(the response object), and the following function in the request-response cycle of the application. The **next()** function in the Express router is a function that, when triggered, runs the middleware that follows the current middleware. If the current middleware function does not complete the request-response cycle, it must execute **next()** to transfer control to the next middleware function.

```

// Allow cors middleware
app.use((req, res, next) => {
  res.header('Access-Control-Allow-Origin', '*')
  res.header('Access-Control-Allow-Headers', 'Origin, X-Requested-With, Content-Type, Accept')
  next()

  app.options('*', (res, req) => {
    res.header('Access-Control-Allow-Methods', 'GET, PATCH, PUT, POST, DELETE, OPTIONS')
  })
})

```

Figure 29: CORS middleware.

A request handler is responsible for receiving, processing, and transmitting requests to the Azure cloud. Specifically, the function sends a POST request to the Azure API services endpoint with the URL of the input picture, the value of which will be given from the application's frontend. After receiving the response, the function assigns all **response.data** to the data variable, which the developer may utilize to obtain the result. With the application's features, a total of three request handler methods will be created. All of these functions contain the same format as the previous one and all employ the POST method, with the exception of the Endpoint URL and the request body's attributes. Finally, if **PORT** is not provided in the environment variables, the server listens on port 5000 by default

4.4.2 Front-end Implementation

The web application's concept is to provide three possibilities for each feature: face detection, comparison between two faces, and object identification with description. When a user selects the feature, uploads a picture from the URL input field and presses the submit button, the program returns the identical results. The majority of the code in this area will be written in Node.js, and will be distributed using the Netlify platform. The homepage (cf. figure 30) briefly describes all the functions of the web application, and it is the first page when the user opens the application. Additionally, URLAPI is initialized inside the **src/constant** folder with the address of the server that has been deployed to, in this case, the Heroku platform.

The server is responsible for registering the faces with Microsoft Cognitive Services' Face API.

Upload a URL from an image on JPG format, the app is responsible for the following:

- Show the attributes of the image, in this case: gender and age.
- Identity the object.
- Recognize two faces.

Figure 30: Homepage overview.

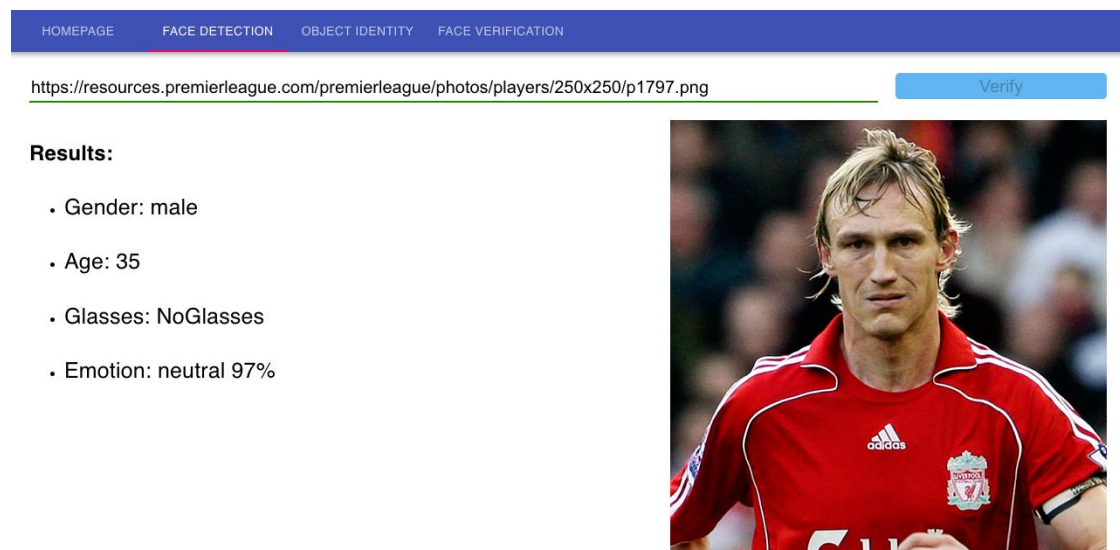
The function **handelDetectImage** (cf. figure 31) manages the first feature, Face Detection, as can be seen from the figure below. When called, it makes a request to the Express Server to receive the API call's details, including the POST method, the name of the header, and the image URL. Following that, the answer data will be processed and stored in an array of persons. It also logs any problems that occur throughout the operation.

```
const handelDetectImage = async (event) => {
  event.preventDefault();
  setReady(false);
  try {
    const fetchOptions = {
      method: "POST",
      headers: {
        Accept: "application/json",
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        image: image,
      }),
    };
    const resp = await fetch(`${URLAPI}/create-facelist`, fetchOptions);
    const people = await resp.json();
    console.log("data of people: ", people.data[0]);
    const emotionObject = people.data[0].faceAttributes.emotion;
    const emotionArray = Object.values(emotionObject);
    const indexEmoMax = Math.max(...emotionArray);
  }
}
```

Figure 31: The function **handelDetectImage** overview.

The program provides three more functions that perform identical duties: **handleFaceIDs**, **handleVerify**, and **handleIdentity**. The **handleFaceIDs** and **handleVerify** methods will be used to handle event dispatching in the face comparison feature; **handleFaceIDs** will take care of uploading photographs to Azure and retrieving the associated faceID, before passing these data to **handleVerify** for processing and returning the final result. And **handleIdentity** will manage the last feature, object identity.

4.4.3 Application Testing



The screenshot displays a web application interface with a blue navigation bar at the top containing the following menu items: [HOMEPAGE](#), [FACE DETECTION](#), [OBJECT IDENTITY](#), and [FACE VERIFICATION](#). Below the navigation bar, the URL <https://resources.premierleague.com/premierleague/photos/players/250x250/p1797.png> is shown, followed by a blue button labeled "Verify".

Results:

- Gender: male
- Age: 35
- Glasses: NoGlasses
- Emotion: neutral 97%




Figure 32: Testing example for face detection.

Figure 32 illustrates the face detection function focusing on the detection of frontal human faces. The result includes three properties including gender, age, and emotion of the Finnish football player, Sami Hyypiä.

5 Conclusion

Cloud computing is a significant area of information technology, with an enormous number of services accessible. In comparison to operating programs locally, using a cloud option for application deployment benefits companies specializing in software development. As cloud computing environments continue to evolve at a fast pace, some IT organizations have established their own cloud solutions. Microsoft Windows Azure is a cloud computing platform developed and maintained by Microsoft.

The purpose of this project was to investigate Azure cloud computing services by constructing a web application that used the Microsoft Cognitive Services API and had three primary features: face recognition, two-face verification, and object identification. The whole application was built on the Node.js platform, with the frontend powered by the React framework and the API calls handled by the ExpressJS library. The web application was hosted on the Netlify platform, while the server was created on Heroku Cloud. All data entered is saved in a Microsoft Azure database and is immediately deleted 24 hours after the first import.

The thesis work provided the foundation for the subject and demonstrated the feasibility of implementing a web application. However, as a product created by a student of embedded systems, this seems to be an inferior software product. Some limitations of this project still remain such as security concerns and unconventional interfaces as well as less noticeable features of the project.

To summarize, further issues and features will need to be resolved and implemented in the future. Nonetheless, this final year project was a reasonably effective effort that addresses the premise's difficulty. With the growth of technology, artificial intelligence and machine learning applications will undoubtedly establish a firm basis for human society's evolution. Thus, along with the website industry continuing to expand, this thesis was written as a fundamental study into the aforementioned concerns. However, further research on how to improve web applications using face recognition, two-face verification, and object identification are needed.

References

1. Gartner forecasts worldwide public cloud end-user spending to grow 23% in 2021. Online. April 2021.
<<https://www.gartner.com/en/newsroom/press-releases/2021-04-21-gartner-forecasts-worldwide-public-cloud-end-user-spending-to-grow-23-percent-in-2021>>. Accessed: 25 March 2022.
2. Mell, Peter & Grance, Tim. The NIST definition of cloud computing. Technical report NIST. Special publication 800-145. Electronic book. September 2011.
<<https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf>>. Accessed: 25 March 2022.
3. Krutz, Ronald & Vines, Russell. Cloud security: a comprehensive guide to secure cloud computing. Electronic book. 2010. USA: Wiley Publishing, Inc.
<https://23510310jarinfo.files.wordpress.com/2011/08/ebooksclub-org_cloud_security_a_comprehensive_guide_to_secure_cloud_computing.pdf>. Accessed: 25 March 2022.
4. Sarna, David. Implementing and developing cloud computing applications. Electronic book. 2011. USA: Auerbach Publications.<<https://doc.lagout.org/Others/CRC.-.Implementing.and.Developing.Cloud.Computing.Applications.pdf>>. Accessed: 26 March 2022.
5. Sammut, Claude & Websiteb, Geoffrey. Encyclopedia of machine learning and data mining. Electronic book. 2017. NY: Springer.
<<https://github.com/sanjaysheel/Data-Science-Books-1/blob/master/Encyclopedia%20of%20Machine%20Learning%20and%20Data%20Mining%2C%20Second%20Edition.pdf>>. Accessed: 26 March 2022.

6. Marr, Bernard. How much data do we create every day? The mind-blowing stats everyone should read. Online. May 2018. Forbes. <<https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/>>. Accessed: 26 March 2022.
7. Convolutional neural networks for visual recognition. Online. CS231n github. <<https://cs231n.github.io/convolutional-networks/>>. Accessed: 27 March 2022.
8. Bre, Facundo; Gimenez, Juan & Fachinotti, Victor. Prediction of wind pressure coefficients on building surfaces using artificial neural networks. Online. 2017. Scindirect. <https://www.researchgate.net/publication/321259051_Prediction_of_wind_pressure_coefficients_on_building_surfaces_using_Artificial_Neural_Networks>. Accessed: 27 March 2022.
9. Kim, Kwang. Deep learning. 2016. Goyang, Korea: National Cancer Center.
10. Townend, Oliver; Nielsen, Jens & Ramsgaard, Jesper. Real-life of the applications for machine learning in hearing aids. Online. 26 March 2018. Hearingreview. <<https://hearingreview.com/hearing-products/hearing-aids/real-life-applications-machine-learning-hearing-aids>>. Accessed: 28 March 2022.
11. Knight, Will. What Marvin Minsky still means for AI. Online. 26 January 2016. Technologyreview. <<https://www.technologyreview.com/2016/01/26/163622/what-marvin-minsky-still-means-for-ai/>>. Accessed: 28 March 2022.

12. Roy, Fielding. Architectural styles and the design of network-based software architectures. Doctoral dissertation. Online. 2000. Irvine, USA: University of California.
<https://www.ics.uci.edu/~fielding/pubs/dissertation/fielding_dissertation_2up.pdf>. Accessed: 29 March 2022.
13. Sturt, Robert. What is the difference between Microsoft Azure vs Amazon AWS?. Online. 01 December 2021. Netify. <<https://www.netify.com/learning/what-is-the-difference-between-microsoft-azure-vs-amazon-aws>>. Accessed: 30 March 2022.
14. Chappell, David. Windows Azure and ISVs – a guide for decision makers. Online. 2009. Microsoft Corporation.
<<https://download.microsoft.com/download/e/7/4/e74d55e6-d156-404f-b6c5-a53a9a4b1d42/windows%20azure%20for%20isvs%20v1%2011--chappell.pdf>>. Accessed: 30 March 2022.
15. Maini, Raman & Virk, Ishpreet. Cloud computing: Windows Azure platform. Journal of global research in computer science. Online journal. 2012. <<https://www.rroj.com/peer-reviewed/cloud-computing-windows-azure-platform-37688.html>>. Accessed: 30 March 2022.
16. Fried, Jeff. How knowledge management will change with the advent of machine learning and cognitive search. Online. CIORevue.
<<http://knowledgemanagement.cioreview.com/cxoinsight/how-knowledge-management-willchange-with-the-advent-of-machine-learning-and-cognitive-search-nid-27521-cid-132.html>>. Accessed: 31 March 2022.
17. Herron, David. Node.js website development. Fourth Edition. Electronic book. May 2018. Packt Publishing.
<<https://www.packtpub.com/product/node-js-website-development-fourth-edition/9781788626859>>. Accessed: 01 April 2022.

18. Developer survey results. Online. 2017. Stackoverflow.
<<https://insights.stackoverflow.com/survey/2017#technology>>. Accessed: 01 April 2022.
19. Ethan, Brown. Web development with Node and Express. 2nd Edition. Electronic book. November 2019. O'Reilly Media, Inc. URL:
<<https://www.pdfdrive.com/web-development-with-node-and-express-leveraging-the-javascript-stack-d176027720.html>>. Accessed: 03 April 2022
20. Lynne, Grewe. Express: middleware. CS6320: SW engineering of web based systems. Online. 2019. Cal State East Bay.
<<http://borg.csueastbay.edu/~grewe/CS6320/Mat/NodeJS/ExpressMiddleware.html>>. Accessed: 04 April 2022.
21. Node.js user survey. Online. 2018. Nodejs organization.
<<https://nodejs.org/en/user-survey-report/>>. Accessed: 04 April 2022.
22. Paul, Brown. State of the union: npm. Online. January 2017. Linux.com.
<<https://www.linux.com/news/state-union-npm/>>. Accessed: 04 April 2022.
23. Roldan, Carlos. React cookbook. Electronic book. August 2018. Packt Publishing. <<https://libribook.com/ebook/14627/react-cookbook-create-dynamic-web-apps-pdf/?bookid=45368>>. Accessed: 04 April 2022.
24. Eve, Porcello & Alex, Banks. Learning React. 2nd Edition. Electronic book. July 2020. O'Reilly Media, Inc.
<<https://media.graphcms.com/HrM5QEgWSweYEQBwCISG?dl=true>>. Accessed: 04 April 2022.

25. Antonio, Cássio. Pro React. Electronic book. December 2015. Apress, Springer Nature. <https://github.com/LeuisKen/react-collection/blob/master/ebooks/Pro%20React.pdf>. Accessed: 05 April 2022.
26. What is Heroku. Online. Heroku Inc. San Francisco, CA: Salesforce Inc. <<https://www.heroku.com/what>>. Accessed: 05 April 2022.
27. Varty, Joel. What is Netlify and what are its benefits?. Online. 4 August 2020. Agilitycms. <<https://agilitycms.com/resources/posts/what-is-netlify-and-why-should-you-care-as-an-editor>>. Accessed: 05 April 2022.

