KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Technology

First cycle, 15 credits

Cloud Computing Pricing and Deployment Efforts

Navigating Cloud Computing Pricing and Deployment Efforts: Exploring the Public-Private Landscape

FREDRIK LUNDSTRÖM
CASPER KRISTIANSSON

Stockholm, Sweden, 2023

Cloud Computing Pricing and Deployment Efforts

Navigating Cloud Computing Pricing and Deployment Efforts: Exploring the Public-Private Landscape

FREDRIK LUNDSTRÖM

CASPER KRISTIANSSON

Degree Programme in Computer Engineering
Date: June 21, 2023

Supervisors: Johan Montelius, Mattias Kjörk
Examiner: Niharika Gauraha
School of Electrical Engineering and Computer Science
Host company: amaceit AB
Swedish title: Prissättning och Implementeringsinsatser för Molntjänster
Swedish subtitle: Att Navigera Molntjänsters Prissättning och Implementeringsinsatser: Utforska det Offentlig-Privata Landskapet

Abstract

The expanding adoption of cloud computing services by businesses has transformed IT infrastructure and data management in the computing space. Cloud computing offers advantages such as availability, scalability, and cost-effectiveness, making it a favored choice for businesses of all sizes. The aim of this thesis is to compare private and public cloud computing services in terms of pricing and implementation effort as well as comparing the cloud providers to each other. The top three cloud providers that will be examined are Google GCP, Microsoft Azure, and Amazon AWS. The study examines different pricing models and evaluates their effectiveness in different business scenarios. In addition, the thesis also discusses the challenges associated with building and maintaining private infrastructure and the deployment of applications to cloud computing service are examined. The research methodology involves data collection, analysis, and a case study of developing and deploying a ticketing system application on different cloud platforms. The ticket system helps to provide a realistic example and investigation of the cloud providers. The findings will help companies make informed decisions regarding the selection of the most appropriate cloud computing service based on pricing models and implementation efforts. The thesis provides valuable information on private and public cloud computing and recommends appropriate pricing models for different scenarios. This study adds to existing knowledge by analyzing current pricing models and deployment concepts in cloud computing. The thesis does not propose new solutions but follows a structured format compiling information on private, and public cloud computing and a comprehensive review of cloud computing pricing models and marketing efforts.

Keywords

Cloud computing, Private cloud, Public cloud, Cloud computing services, Cost-effectiveness, Implementation effort, Google GCP, Microsoft Azure, Amazon AWS, Pricing models, Cloud adoption, Cloud cost management, Cloud migration, Instance computing, Serverless computing, Data storage

Sammanfattning

Den växande adoptionen av molntjänster inom företag har förändrat IT-infrastrukturen och datahanteringen inom datorområdet. Molntjänster erbjuder fördelar såsom tillgänglighet, skalbarhet och kostnadseffektivitet, vilket gör det till ett populärt val för företag i alla storlekar. Syftet med denna avhandling är att jämföra privata och offentliga molntjänster med avseende på prissättning och implementeringsinsatser samt att jämföra molnleverantörerna med varandra. De tre främsta molnleverantörerna som kommer att undersökas är Google GCP, Microsoft Azure och Amazon AWS. Studien undersöker olika prismodeller och utvärderar deras effektivitet i olika affärsscenarier. Dessutom diskuterar avhandlingen också utmaningarna med att bygga och underhålla privat infrastruktur samt implementeringen av applikationer till molntjänster. Forskningsmetodologin omfattar datainsamling, analys och en fallstudie av utveckling och implementering av ett support system på olika molnplattformar. Supportsystemet hjälper till att ge ett realistiskt exempel och undersökning av molnleverantörerna. Resultaten kommer att hjälpa företag att fatta informerade beslut när det gäller valet av lämpligaste molntjänst baserat på prismodeller och implementeringsinsatser. Avhandlingen tillhandahåller värdefull information om privat och offentlig molntjänst och rekommenderar lämpliga prismodeller för olika scenarier. Denna studie bidrar till befintlig kunskap genom att analysera nuvarande prismodeller och implementeringskoncept inom molntjänster. Avhandlingen föreslår inga nya lösningar, men följer en strukturerad format genom att sammanställa information om privat och offentlig molntjänst samt en omfattande översikt av prismodeller och marknadsinsatser inom molntjänster.

Nyckelord

Molntjänster, Privat moln, Offentligt moln, Kostnadsjämförelse, Kostnadseffektivitet, Google GCP, Microsoft Azure, Amazon AWS, Molninförande, Molnkostnadshantering, Molnmigration, Instance computing, Serverless computing, Dataförvaring

Contents

List of Figures

List of Tables

Listings

List of acronyms and abbreviations

AD	Active Directory
AI	Artificial Intelligence
API	Application Programming Interface
AWS	Amazon Web Services
CAGR	Compund Annual Growth Rate
CAPEX	Capital Expenditures
CI	Continuous Integration
CORS	Cross-Origin Resource Sharing
CRM	Customer relationship management
CSS	Cascading Style Sheets
DBMS	Database Management System
EU	European Union
FaaS	Function as a Service
FTPS	SSL File Transfer Protocol
GCP	Google Cloud Platform
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IAM	Identity and Access Management
IP	Internet Protocol
IT	Information Technology
ML	Machine Learning
OPEX	Operating Expenses
ORM	Object-Reltaional Mapping
PaaS	Platform as a Service

REST	Representational State Transfer
SaaS	Software as a Service
US	United States
UX	User Experience
vCPU	Virtual Processor

Chapter 1

Introduction

The amount of cloud computing services used by businesses has rapidly increased over the past decade. Instead of using their own server for computing, businesses can use scalable and cost-effective resources available over the internet, provided by big Information Technology (IT) companies with a trusted reputation. Today, almost 40% of Microsoft's revenue comes from their "Intelligent Cloud" services including Azure [1] but only 12% from their operating system Windows [1], even though it is the most used operating system in the world [2]. This gives an insight into how big the business of cloud computing has become. Cloud computing has transformed the way businesses and organizations manage their IT infrastructure and data needs. Cloud computing has become an increasingly popular choice for organizations of all sizes because of the need for available, scalable, and cost-effective resources.

By developing and deploying an application to different public cloud providers, this thesis aims to explore and compare the pricing models and deployment efforts of private and public cloud computing services. This thesis seeks to help businesses make informed decisions about which computing service to adopt based on their unique needs and requirements. Additionally, this thesis will analyze the challenges that businesses face in deploying applications. This chapter will go through a brief background of the subject, state the problem, the purpose of the thesis, and the goals of the thesis.

1.1 What is Cloud Computing?

Cloud computing involves many services such as data storage, development tools, computing resources applications, network capabilities, and more over

the internet. Cloud computing aims to solve the availability, scalability, and cost-effectiveness of applications, websites, IT systems, and more. Some cloud computing providers have their own implementations of services, and some might offer services that other cloud computing providers do not, and it might become an obstacle for a business to choose which provider to use or if to build and host their own services. There are three main ways of implementing the cloud, namely private, hybrid, and public cloud. The thesis will explore the subject more deeply in Chapter 2.

1.1.1 Private Cloud

Private cloud is a choice for businesses that needs specific services that are not available via public cloud services, and therefore develop their own solutions with the scalability or ease of service delivery that private cloud can provide. A private cloud can also provide security with its isolated access, which is a good choice for businesses with workflows that deal with sensitive data such as confidential documents, intellectual property, personally identifiable information, medical records, or financial data.

1.1.2 Public Cloud

Public cloud providers offer computing power for customers' workloads in their own data centers. Unlike private cloud, they have full responsibilities for hardware, virtualization software, and network and therefore there is a chance they cannot offer custom services and isolated access to meet some customers' needs.

1.1.3 Hybrid Cloud

A hybrid cloud solves the integration, orchestration, and management of companies that need both private and public cloud services. Hybrid cloud should not get confused with hybrid multi-cloud which is the use of public cloud services from more than one cloud service provider.

1.2 Problem

Developers today face the challenge of choosing between private and public cloud computing. The challenge doesn't just stop there, even developers also phase difficulty in picking a cloud provider. There are also many different

services such as features, tools, data storage, and more that some cloud computing providers offer and some do not. These challenges present a significant problem for businesses, organizations, and independent developers looking to optimize their usage of cloud resources while managing costs and ensuring the smooth operation of their cloud infrastructure. How does a business, organization, or independent developer choose the most suitable private or public cloud computing service, depending on pricing models and deployment efforts?

1.3 Purpose

The purpose of this thesis is to help businesses to make informed decisions about which implementation of cloud and if public cloud is the best option, which public cloud provider is best suited for their specific needs and budget.

1.4 Goals

There are two primary goals of this study. The first goal of this study is to analyze the various pricing models offered by cloud service providers and evaluate their effectiveness in different business scenarios. The second goal is to understand the deployment efforts involved in building and maintaining cloud computing infrastructure and identify its associated challenges. This has been divided into the following three sub-goals:

1. Compare the pricing structures of Google GCP, Microsoft Azure, and Amazon AWS in order to determine their cost-effectiveness and suitability for different business scenarios.

2. Investigate the challenges and considerations involved in deploying applications to public cloud platforms, with a focus on Google GCP, Microsoft Azure, and Amazon AWS, and identify best practices for successful application deployment.

3. Examine the advantages and challenges associated with building and maintaining private cloud infrastructure, including considerations such as hardware, software, security, and scalability.

1.5 Research Methodology

The study is conducted by collecting and analyzing data from relevant sources and through a case study. The case study consists of the development and deployment of a ticket system application to different public cloud providers. The study is conducted in three stages:

1. The first stage involves a literature review to provide a comprehensive overview of cloud computing pricing models and deployment efforts.
2. The second stage involves a case study, which consists of the development and deployment of a ticket system web application to different public cloud providers.
3. The third stage involves data analysis, where the data collected will be analyzed.

1.6 Delimitations

This thesis aims to explore the benefits of cloud computing and examine various pricing models and deployment options associated with different types of cloud computing. The objective is to provide businesses with valuable insights to facilitate informed decision-making regarding the most suitable cloud computing solution for their needs.

However, it is important to note that this thesis has some delimitations.

1. The thesis focuses on specific cloud computing platforms; Amazon Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).
2. The thesis focuses also on a set of services that the different cloud providers provide. This means that a lot of services that the different cloud providers provide will not be included in the analysis.
3. Geographic scope, the research is mostly focused on the area of Europe, which means there could be quite large differences in other geographical locations.
4. The research primarily focuses on comparing and analyzing a specific set of services offered by different cloud providers, rather than comparing all services provided by the cloud providers.

Chapter 2

Background

In the past years, there has been a rapid increase in the usage of cloud computing services [3], and becoming more and more of a standard for businesses to use. The reason for this type of switch when a business decides on what type of infrastructure to use is that cloud computing has been able to provide cost-saving, scalability, flexibility, accessibility, and improved collaboration for the development team. The business has been able to in a more efficient and cost-effective way launch scalable applications. As more businesses adopt cloud computing, it is very likely that the trend of usage of cloud services will continue to grow.

In the past, it was standard for businesses to build and manage their own infrastructure [4]. This often required significant investment in hardware and software, as well as personnel to manage and maintain the infrastructure. This approach would also make it really hard to scale the usage of the hardware as it would take months of planning and executing This meant that for a lot of businesses, it required a lot of investment to launch and maintain access to products. However, the usage of cloud computing has made it easier for smaller businesses and even bigger ones to launch their products on the market.

With the help of cloud computing, small businesses, startups, and even large enterprise companies are able to leverage it for some type of service that they provide [5]. Cloud computing in many cases allows you to pay as you go, which has made it possible for businesses to find services to fit their specific requirements without any significant investment or long-term agreements. Among the larger enterprise companies, a lot of them have been utilizing cloud computing by leveraging both public and private computing as a hybrid model.

2.1 Introduction to Cloud Computing

The cloud computing industry is modeled on the foundation of delivering on-demand services to its customers using shared computing resources. This means that the customers do not need to rely on understanding or maintaining the infrastructure [6] and can mostly focus on the development process. The shared resources provided by the cloud provider can therefore be scalable and adjusted based on demand, where the customers can easily manage the resources as they see fit.

By utilizing cloud services, the developers of the IT product do not need direct access to the hardware. This means that the cloud providers provide their services to anyone in the world, while compared to private cloud its developers need direct access to the hardware. Companies that operate as a global business with the help of cloud computing are able to be operating as if they were working from the same office [7]. In most cases, the cloud providers have services where you can choose exactly where your application will be running from. Meaning that if you prefer your application to be running in, for example, Europe rather than in the United States (US), you have the option to do so. This provides its customers with a lot of flexibility to fit their specific use cases.

With the help of cloud providers, companies are able to make sure that their applications become more secure [8]. Cloud computing provides advanced security measures to protect its customer's data. In a greater extent of cases in private computing, it can be really hard to make sure that the company's data being saved is secure enough. By using cloud providers, all data stored is encrypted to the latest standards.

Another important part of cloud computing is its impact on the environment [9]. Cloud computing companies often in a lot of cases try to place their computing facilities in the most practical way possible. In the latest year, it has grown that large IT companies are placing their data facilities in the Northern parts of Sweden, where they are utilizing the power of clean energy. For companies, this means that they can strive for more environmentally friendly computing where all resources in the computing capabilities are always using their computing to their need. This means that cloud providers can be much more effective with energy usage.

Most cloud providers today focus on offering high availability and reliability to their customers. The cloud provider does this by building a redundancy system meaning that if certain computer instances would fail, it can just switch to another instance, which results in that it wouldn't cause a

decrease in availability for its customers [10].

2.1.1 Cloud service models

Cloud providers provide different types of cloud models and are composed of the following models [11]; Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Function as a Service (FaaS), and Software as a Service (SaaS). All of these different types of services in most cases are combined by a business to fit a specific need of a customer.

2.1.1.1 Infrastructure as a Service (IaaS)

IaaS is the most popular model [12] where it provides virtualized computing resources like servers, storage, and networking. In a lot of cases, infrastructure as a service is that it provides a resource on-demand service where all instances of an application live on multiple computing instances. This means that if the application needs more scalability, it will automatically process it and create new instances. The most popular providers of this service are AWS, Microsoft Azure, and GCP.

2.1.1.2 Platform as a Service (PaaS)

PaaS describes the computing model [13] that provides a platform and environment for developers to build and deploy their desired applications. PaaS, therefore, offers a complete development and deployment infrastructure where its customers are free in more detail and ability to adjust and change the infrastructure as fitted. This means that if the customer desires they can specify operating systems, databases, middleware, and deployment instruments.

2.1.1.3 Software as a Service (SaaS)

SaaS is a computing model [14] that specifies the process of delivering software applications over the internet. The models help the process of hosting and managing applications through mobile apps and websites (web applications). This means that users can access the software without needing to install or maintain the software on their own devices. Examples of these types of services are Canvas, Google Workspace, and Microsoft Office 365.

2.1.1.4 Function as a Service (FaaS)

FaaS is a cloud computing model [15] that offers a platform for executing functions on-demand without the requirement of managing servers. In this model, the developers are writing functions that perform very specific tasks which are triggered by an event such as a system event or a user action. These types of actions are usually run as an instance and therefore have an extremely good chance to automatically scale to the specific demand. Examples of these types of models are called AWS Lambda, GCP Functions, and Microsoft Azure Functions.

2.1.2 The Cloud Computing Revolution: Past, Present, and Future

The concept of cloud computing dates back to the 1960s with regard to utility computing. A model where the computing resources are provided as a metered system. John McCarthy, an American computer scientist also known as one of the founders of AI, had a vision that was stated at a celebration in 1961 "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility ... The computer utility could become the basis of a new and important industry" [16, Page 1]. Since then, the concept of cloud computing has only grown.

2.1.2.1 Past

In the 1990s, the idea of virtualizing computing emerged [17]. This concept allowed a system to run multiple operating systems, which increases the efficiency of the computing resources. This concept is the foundation of cloud computing today, where it is possible to run numerous different instances of the same machine. In the early 2000s, AWS launched and is considered to be the first IaaS platform on the market. These services marked the beginning of the modern cloud-computing era. Later in the late 2000s, Google introduced the first PaaS platform that allowed developers to build and run the application entirely in the cloud without having to stress about the underlying infrastructure. Just a year later the company Salesforce launched Salesforce Customer relationship management (CRM) which is considered to be the first SaaS application.

2.1.2.2 Present

The adoption of cloud computing is on the rise and more businesses are moving and started using the cloud as a core of their business. According to a report by Grand View Research, the global cloud computing market was valued at USD 483.98 billion in 2022 and is expected to expand at a Compund Annual Growth Rate (CAGR) of 14.1% from 2023 to 2030 [18]. Another report by Fortune Business Insights [19] states that the global cloud computing market size was valued at USD 405.65 billion in 2021 and is projected to grow to USD 1,712.44 billion by 2029, at a CAGR of 19.9% during the forecast period.

Small and medium-sized businesses have been quickly adapting to the cloud computing world due to its affordability and easy-to-use systems [20]. Larger enterprises have also started moving their services to the cloud but at a much slower pace due to legacy systems and security concerns on important data. The potential of the future trends av development in cloud computing technology can only grow.

2.1.2.3 Future

Cloud providers have in recent years been integrating Artificial Intelligence (AI) and Machine Learning (ML) in cloud computing to help businesses to automate and optimize their operations. Because the industry of cloud computing has grown [21], it has provided the providers with a lot of capital to keep expanding their business. With the usage of these assets, cloud computing providers are trying to improve and accomplish three different things. The first one is edge computing, which involves the process of storing and managing data closer to the source of the data. This can help their customers to provide lesser latency and improve the performance of their applications. This has also played an important role when the European Union (EU)'s strong laws [22] on data storage. The laws are regarding the usage and storage of user data within the EU. It states that no user data can be stored outside the EU.

The second thing is cloud security [8]. An important concern for a lot of bigger companies is the security of cloud providers. This is a really concerning subject, especially for companies that see their data as top secret and can't under any circumstances be accessed by other parties. The last thing is hybrid cloud environments. The cloud providers want to help companies by helping and providing so that the businesses can both use a public and private cloud solution. This means that the business is able to have control over its data

where they see fit.

2.2 Pricing Models of Cloud Computing

Among the different cloud computing models, the most popular pricing models are on-demand, reserved, spot, and hybrid [23]. Each of these models has its own perks and is designed in the best way possible to fit the customer's needs.

2.2.1 On-demand Model

The on-demand pricing model [23] allows the customer to pay exactly for the resources that were consumed. In this model, the user is not required to pay upfront or make any commitment or reservation. Among the different pricing models, this has become one of the most popular ones. The pricing is based on the number of resources consumed such as computing power in seconds, storage usage, data transfer, and other services.

The advantage of using this type of pricing model is that it provides flexibility and cost-effectiveness for unpredictable workloads. It allows businesses to easily scale up exactly based on the usage and then pay for the usage. This is especially helpful when businesses have it hard to predict what the upcoming usage of their services is. The disadvantage of the on-demand pricing model is that in the long run on steady workloads where the usage is predictable, it is much more expensive than the other services.

2.2.2 Reserved Model

The reserved pricing model [23] is a pricing model where the customer reserves the cloud resource in advance for a specified period of time. This period can range anywhere from days to even in some instances multiple years. Because this pricing model requires commitment, it in many cases offers considerable discounts because the customers will often pay upfront. The advantages of the reserved pricing model are that it saves a lot of money for steady workloads and predictable costs. The disadvantages are often less flexibility to adjust the usage of the computing and the potential for unused reserved computing power, which can result in a waste of money.

2.2.3 Spot Model

The spot pricing model [23] allows the customer to create a bid on unused cloud computing resources that are usually available at a lower price than the on-demand rates. This means that the pricing of the spot model is usually really volatile based on supply and demand. This means that the advantages of it are that its customers have a potential for cost saving during the off-peak period. This can easily be a good pricing model if certain customers do not need to run their computing power when the demand is high. The disadvantage of this type of pricing model is that it lacks predictability, which can result in a loss of access to resources.

2.2.4 Hybrid Model

The last pricing model is hybrid pricing [23]. This model combines different pricing models to optimize cost-effectiveness and flexibility based on the workload requirements of a customer. This means that this pricing model will for example associate variable workload tasks with on-demand pricing and reserved pricing for steady workloads. But the disadvantages of using this type of pricing model are that in many cases the complexity of it will grow and the potential for additional management overhead could increase.

2.3 Deployment Efforts of Cloud Computing

The deployment of applications on cloud providers is a process that has been made easier and easier over the years [24]. The providers in most cases offer environments for containers where practically any software is able to run on any system. But even so, in most cases, there are a lot of things that developers need to do for specific providers.

This means that for businesses to migrate existing applications and data to the cloud could require a lot of effort and could be costly for a company. Even in some cases, there is a total lack of compatibility where some legacy applications might not be compatible with cloud environments, making the migrations difficult. There are also data security concerns where sensitive data on the cloud could pose a security risk such as a data break and/or data loss. Ensuring that the process of securely transferring data and storing it in the cloud could be a tedious process.

Cloud platforms often are a service where the provider is a lock-system [25], meaning that it becomes extremely hard and expensive to move to another

provider in the future. Businesses need to carefully evaluate the different cloud providers and their services to avoid being locked in with cloud providers. The providers often charge a lot to move data out of their platform, which makes it costly and in some cases a really complex issue. Migrating to the cloud is a process that needs to be well planned and well executed, where businesses need to have a thought-out plan of exactly how they are going to use cloud providers' services.

Setting up automatic deployment for cloud providers [24] has made it possible for developers to have continuous integration of development. With the help of cloud providers, deploying the latest version of the software has never been easier and in most companies, the deployment of the software happens multiple times a day. Cloud computing services have made it really easy with already pre-built software so that developers can easily maintain and deploy their applications with ease.

2.4 Private Computing

Private computing refers to the usage of dedicated hardware and software resources within a business's own infrastructure [4]. This means that the organization accesses its own resources using private networks in most cases rather than over the internet through third-party cloud providers. Private computing has been the standard until cloud computing standard becoming popular.

Private computing involves the process of building and maintaining the IT infrastructure to host a company application, data, and services with a secured network [26]. Private computing remains popular due to its ability to provide data privacy and security. A lot of companies can under no circumstances have their data accessed by a third party, and by building and maintaining their own infrastructure they can make sure that it is possible.

An important part of private computing and the reason why it is still popular today is due to the fact that organizations are freely able to customize their infrastructure to their need. Compared to cloud computing, there are some limitations to really specific solutions that are not possible. This means that the company is able to customize its solution for its specific need.

In the long run, it has been proven that private computing is a lot more cost-effective, but due to the higher upfront costs and development/deployment [27] efforts are still a tough choice for businesses. There are two pricing types for private computing; Capital Expenditures (CAPEX) and Operating Expenses (OPEX). CAPEX are the costs that a company has for building,

maintaining, and improving fixed assets. These types of expenses are usually a one-time expense that is made to create long-term value for a business. Then there are the OPEX. The OPEX are what the company has to pay day-to-day to run the infrastructure. This can have anything from rent, utilities, and employee salaries. This means that later in this report when private computing is compared to cloud computing, both CAPEX and OPEX will be added and compared.

Chapter 3

Methods

The method for conducting this thesis is going to consist of multiple parts. The authors want the best conclusion possible to solve the problem of choosing between private versus public and different public cloud Services. To get the best conclusion, there has to be a thorough research process and data collection. This chapter is going to explain the research process and paradigm, data collection and the methods for analyzing the collected data, and also the design and development of the ticket system. There is also going to be a brief conclusion of all the sections at the end.

3.1 Research Process

The research process for this study is designed to collect both practical and theoretical insights. Practical insights will be gathered through the development and deployment of a ticket system to different public cloud providers. Theoretical insights will be obtained through a qualitative and quantitative literature study.

To elaborate, practical insights will involve collecting pricing and deployment effort data from different public cloud providers by developing and deploying a ticket system. This approach will enable the study to collect real-time data from public cloud providers and provide a more accurate representation of the differences between private and public cloud computing pricing and deployment efforts.

Theoretical insights will be collected by conducting a comprehensive literature review of previous research on the topic. The review will be both qualitative and quantitative in nature and will draw from a variety of sources to ensure a comprehensive analysis. A range of databases and search engines

will be used, with specific keywords identified to ensure relevant research is included. Both practical and theoretical insights will be analyzed and synthesized to form the best possible conclusions.

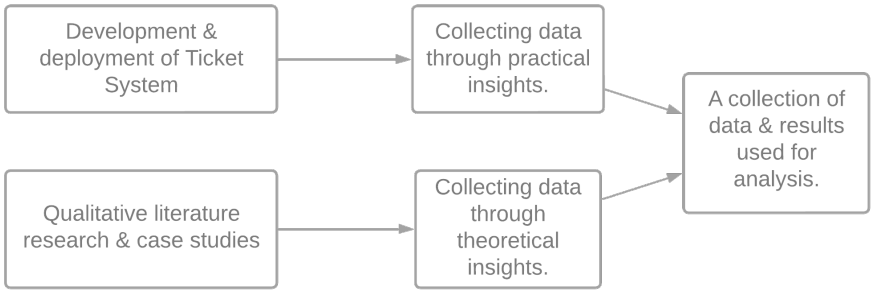

Figure 3.1: Research Process

The ethical considerations for this study include obtaining informed consent from participants, maintaining their anonymity and confidentiality, and ensuring that the data collected is accurate and representative. Limitations of the study include potential biases in the literature review and data collection, which will be addressed through rigorous analysis and interpretation of the data.

3.2 Research Paradigm

This study employs a mixed-methods research design, which combines both qualitative and quantitative research methods. The rationale for this design is to obtain a comprehensive understanding of private and public cloud computing pricing and deployment efforts.

The use of both qualitative and quantitative methods allows for a triangulation of data, whereby the findings from one method are used to confirm or refute the findings from the other method. The ticket system data collected through the quantitative approach will provide empirical evidence on pricing and deployment effort differences, while the literature review will provide insights into the underlying factors that contribute to these differences.

The advantages of the mixed-methods design include the ability to collect both qualitative and quantitative data, which enhances the validity and reliability of the study. The use of both methods allows for a more comprehensive analysis of the topic, which can lead to more nuanced

findings. Additionally, this design is well-suited to exploring complex research questions that require multiple perspectives.

However, there are some disadvantages to this design. One potential disadvantage is that it may require a larger sample size and more resources to conduct than a single-method design. Another challenge is the potential for methodological conflicts, where the qualitative and quantitative methods may produce different or conflicting results. Finally, the integration of the findings from both methods can be challenging and time-consuming.

Overall, the mixed-methods research design was selected for this study as it provides a more comprehensive understanding of private versus public cloud computing pricing and deployment efforts than a single-method design. The advantages of this design outweigh the potential disadvantages, and the integration of both methods will lead to a more robust and nuanced analysis of the topic.

3.3 Case Study

The authors will conduct a case study to compare the deployment efforts both using private cloud and different public cloud providers using the ticket system developed for *amaceit AB*. The case study is aimed with the goal of giving deeper practical insights into the subject, which is beneficial for the overall comparison and conclusion. There might although be some negative outcomes from the case study, which might be that the results are not generalized to a larger group of individuals, organizations, or businesses and can lead to bias.

3.4 Data Collection Methods

In order to compare the pricing and deployment effort of the different cloud providers as well as private computing, an essential part is data collection. The data that we are going to use will be collected from different sources such as the cloud providers' websites, documentation for different use cases, and then the actual cost after the case study.

The collected data will not only be the current but also the previous data from the different providers of both deployment efforts and pricing. This will help to build an understanding of identifying trends and patterns. The data collection process will include basic information about the different cloud providers as well as information on the different services that they provide.

3.4.1 Data sources

The main data sources for the cloud providers will be the official websites of the different companies. This will provide accurate and up-to-date information on the different services. In many cases, the providers will also provide documentation and use cases where they give estimates in both terms of deployment and pricing. For example, in the service Elemental Media package [28], in the pricing section they first mention the basic pricing information. They also provide three different use cases on how the service is integrated with other applications and its pricing of it. This will be able to provide a better picture of the real costs of using the service.

Other sources that will be used are the technical documentation of the cloud providers. The technical documentation will help with deploying the different applications but also help to give a better understanding of what exactly the different services do. The documentation will especially be important when the different services of the different cloud providers will be compared because in some situations the exact services will not exist.

Two other sources that will be used are white papers and research papers. For each of the different services that will be compared, a white paper containing information about the service from the different providers. The white paper will be able to provide an accurate and in-depth analysis of the service. Industry reports containing information on the trends of private computing and cloud computing will be used to get information on exactly how the market is changing and adapting to cloud computing.

Private computing will be harder to get an exact estimation of. The different data that will be needed to estimate the deployment and pricing of private computing will include the building and setting up of the infrastructure, as well as its maintenance of it.

3.4.2 Data variables

The first main information that will be gathered is the different cloud providers that will be compared. That information includes name, location, and type of provider. After the larger competitors have been identified, the next step is to identify the different services that they offer. This includes services like storage, computing power, security measures, and other features that they offer.

From each service, both pricing and estimation of time and effort required to deploy an application need to be extracted. These will serve as an estimate and later will be used to compare against the real cost and effort of deployment. During the case study, the time tracking for deploying the application will be

tracked. There will also be a level for the deployment efforts when deploying the application on each cloud provider, which will be measured on a scale of 1-100.

A ranking of each of the compared cloud providers will be made based on various factors such as the difficulty of deployment, documentation, customer support, and overall satisfaction. Any other relevant data variables that are collected during the case study will be recorded and tracked, such as performance metrics.

3.4.3 Data collection procedures

The procedure of collecting the data will be identifying the largest cloud providers, which will be the basis of comparison. The next step is to identify based on the case study (the ticket system) what exactly the services of the different cloud providers will be compared, including identifying how the data points can be collected (pricing, deployment effort). Collection of basic information about each service provided by each cloud provider, such as their specific use case of them.

The data that will be collected through the case study will be collected by measuring the time required to deploy the application on each cloud service. An evaluation of the difficulty of deployment on a scale of 1-100. Ranking the service by other measurements, such as the documentation and support provided.

The measurements of private computing such as the price and development effort will be gathered by simulating and estimating the requirement of deploying the application in private infrastructure.

3.5 Data Analysis Methods

The purpose of this section is to describe the methods used to analyze the data collected on private versus public cloud computing in terms of pricing and deployment efforts. The data includes both pricing and deployment effort metrics, which will be collected over the span of this thesis. The analysis of the metrics will be focused on comparing the different cloud providers as well as comparing them to private computing. In order to perform the best analysis, several different analysis methods will be used.

3.5.1 Statistical analysis

The first analysis method which will be used is statistical analysis. Descriptive statistics will be used to summarize the pricing and deployment data, as well as inferential statistics to measure against the hypotheses data and identify differences between private and cloud providers. The tools to make these types of analyses will be Microsoft Excel [29] and will also serve as data storage of the data collected.

The offered service by the different providers constantly changes in terms of deployment and pricing. This means that there will be some data distribution and normality assumptions. The goal is to try to access historical pricing and deployment effort to both understand and check for normality.

The statistical analysis will include data cleaning and preprocessing procedures where invalid and missing data need to be handled. For certain services, it could be extremely hard to get a unit pricing and deployment of the service. The variance of information could make it hard to compare the different services against each other.

The presentation of the result from the statistical analysis will be in the form of tables and graphs to describe the history of both pricing and deployment effort. These graphs will play a key part in the discussion of the findings in the statistical analysis.

3.5.2 Thematic analysis

The second analysis method which will be used is thematic analysis. The focus of this analysis is to understand the common themes and patterns that emerge from the qualitative data collected during the case study. The qualitative data includes the measurement of data through the case study by comparing the deployment effort and time for the different cloud providers.

One of the main themes that the analysis will contain from the qualitative data is the time and effort required to deploy applications to various cloud providers. The data revealed from it will show that a few providers will have easy-to-use interfaces and streamlined deployment processes, while others required significant expertise and resources to complete the same task.

Another theme that will exist in the qualitative data is the compatibility and integration with existing systems. Many of the challenges in integrating new cloud services with existing infrastructure are the large need of needing to rework and configure existing systems to become compatible with cloud providers.

The results of the thematic analysis will be presented in the form of a detailed report that describes the key themes and patterns that emerged from the qualitative data. The findings from the thematic analysis will be discussed in relation to the quantitative data collected and analyzed.

3.5.3 Comparison of private and public cloud computing pricing

In this analysis, an overview of the distinct pricing models by the different providers will be the basis of the comparison. When the different pricing models are compared between the providers, a lot of factors affecting the pricing will be discussed. These factors play a significant part when comparing private computing against cloud computing.

The analysis will also be centered around the cost-effectiveness of private and public cloud computing for different business models and business sizes. The different pricing models have trade-offs between private and public computing. Case studies or examples of businesses that have switched between private to public computing or vice versa will also be used to understand the costs of doing it.

3.5.4 Comparison of deployment efforts

The ticket system that will be developed will require services that the cloud providers offer. The deployment efforts will then be used to compare the different platforms based on the specific case study of deploying the application. The different metrics that will be involved in this process will be time, complexity, and ease of use for the different services.

The analysis of the results and the identification of any significant differences in the application's deployment among the providers offer valuable insights into their attributes. The difference between the providers in terms of deployment could be everything from infrastructure, tools, and services that they provide.

3.6 Ticket System Design and Development

The purpose of the ticket system is going to both be a good comparison application for the different cloud providers and also provide the company *amaceit AB* with a useful tool. The context of the ticket system is that the

company *amaceit AB* both internally and externally will use it to keep track of everything that happens with different projects.

3.6.1 Introduction to the ticket system and its Purpose

The ticket system will serve as a project information page both internally and externally by the company *amaceit AB*. The ticket system will have features like creating projects and associating tickets with them. The tickets will have information about what they are about and contain information relevant to the ticket. There will also be functionalities such as chat to keep up to date with the specific tickets.

amaceit AB will use this ticket system to keep track of everything that needs to be done with the project. This includes projects for external clients. The system is meant to serve as a platform to access the current state of the projects and a place to gather all the information.

The ticket system will have two main user categories, internal users and external users. The internal users can manage and view all projects and tickets, while the external users can only manage their own project tickets. This will help the company to keep track that someone internally is dealing with the specific ticket.

The ticket system will also include functionalities such as analysis, which will be a report on the tickets. The report will include information and analysis of all the tickets, for example, how many have been resolved, who resolved the most, etc.

3.6.2 Selection of services

The reason for developing the ticket system is also because it requires a large variety of services provided by the cloud platforms. The application will require services for database hosting, file storage, endpoint gateway, and computing power.

The different cloud providers offer different solutions for these different services. This means that to get the best possible analysis of the deployment effort and pricing, the most similar services will be selected.

3.6.3 Development of the ticket system

The development of the ticket system contains a variety of steps. The first step is to choose what programming languages and frameworks that will

be involved in the ticket system. This process includes the verification that the terminologies and frameworks are compatible with the different cloud providers.

The first step includes the development of the user interface and User Experience (UX) design. The UX design will be a back-and-forth process with the manager at *amaceit AB* to understand the requirements of the ticket system.

The second step is the development of the back-end logic and functionalities of the system, such as the database and data/file management. This step also includes the logic of handling the Application Programming Interface (API) gateway for requests.

The next to last step of the development of the ticket system is the process of making the service compatible with the different cloud providers. Each provider in most cases offers a toolset/framework for working towards their services.

The last step is the verification stage, which is the process of testing the application on different cloud providers. The verification also involves verifying that the ticket system fulfilled all of its requirements.

3.7 Limitations of the Study

This section will highlight and acknowledge that every study has its limitations and potential weaknesses. In the field of cloud computing, the service and providers are constantly changing. The conclusion that the report will come might only be accurate and relevant right now, and might not be useful in a few years.

3.7.1 Scope limitation

This study will only focus on a small sample size of the cloud providers and businesses that use cloud computing services. This means that the sample size represented in this thesis may not represent the entire population of different businesses.

Only a handful of different services will be compared, and therefore a lot of services that the cloud providers offer will not be tested or compared. The missed services could play an extremely important part in the selection of cloud providers for a business.

The thesis will only compare the three major cloud providers, which results in a lot of smaller/medium size cloud providers that will not be compared or

taken into account in this study.

This study focuses solely on the pricing and deployment efforts of the different cloud providers. Other crucial factors such as security and reliability will not be highly valued in the study. These factors due in the real world play an important part when selecting cloud providers for businesses.

3.7.2 Data limitation

The different cloud providers investigated in this report frequently change and update their services which means that both pricing and deployment efforts might change. Due to the frequent change, it is important to provide a history of data to get a clearer picture of the different service that is provided. This type of data might not be easily accessible and missing, which can result in data anomalies and analysis not being accurate.

3.7.3 Methodological limitations

The deployment effort part of the study has one big limitation/drawback. The drawback is the existing knowledge of the authors of this thesis. Because both authors already have an extent of comprehension of using different cloud providers and services, results that both deployments in terms of ease of use and time might not be accurate due to bias towards it.

3.7.4 External limitations

This study was conducted in a real-world setting where external factors change frequently such as the current market desire, technological advancement, and policy changes. The different cloud providers that are compared in this report might also frequently change pricing and which services they offer. New services and or improved services can make a significant change in both the pricing and deployment of the application.

3.8 Conclusion

The research methodology used in this study was a case study approach. The data collection methods included literature research by studying previous papers on the subject and collecting data from different sources such as cloud providers' websites. Data collection will also be done through a case study of the deployment of a ticket system to different cloud providers, where the

authors will set a level of difficulty measured on a scale of 1-100. The data analysis methods included statistical analysis, a comparison of private and public cloud computing pricing, and a comparison of deployment efforts.

The recommendation for future research in this area is to conduct more case studies to determine the best practices for pricing and deployment of private versus public cloud computing services.

Chapter 4

Exploring Cloud Resources: A Case Study on Developing, Deploying, and Analyzing Cloud Services

This chapter will conclude the details about the methods, materials, and tools used to achieve the thesis goals and objectives. To accomplish this, a ticket system was developed which is used as a case study to compare the different cloud platforms against each other. The high-level overview of what was performed in the research:

- Conducted a thorough review of existing literature and research on cloud computing services and their pricing and deployment models.
- Collected and analyzed data on the pricing and deployment efforts of three major cloud computing providers: AWS, Azure, and GCP.
- Developed a custom ticket system for comparing the deployment effort of each provider.
- Compiled and analyzed the data collected during our study, identifying key similarities and differences between the pricing and deployment models of each provider.
- Discussed the challenges and obstacles encountered during our project, including issues related to security, compliance, migration, and integration.

4.1 Data Collection

The first step of the research project was data collection. This step includes the process of gathering data from the different cloud providers and estimations of hardware costs for private cloud. This data was then gathered to be used to compare and make an analysis against the actual cost and towards each other.

4.1.1 Public Cloud

- Research on the pricing models and deployment efforts of major public cloud providers such as AWS, Azure, and GCP.
- Gathering information on the different services offered by these public cloud providers, such as virtual machines, storage, networking, and security.
- Analyzing the pricing structures of the different services, including on-demand pricing, reserved instances, and spot instances.
- Collecting data on the deployment effort required for each service, including setting up and configuring the service, deploying applications, and managing the service over time.
- Comparing the pricing and deployment effort of each service across different public cloud providers to identify the most cost-effective and efficient option for businesses.

For public cloud, the first data collection that was made was for the different pricing models for the different services that were going to be used which were a storage system, backend deployment, database, and an authentication service. The reason for selecting these specific services has to do with them being among the most popular services that cloud providers provide. By using these services a good basis can be built in understanding their differences when comparing them against each other.

This different service will be compared both against private cloud but also against the actual cost from the ticket system. The data collected was then logged in an Excel document to provide an overview of the different providers. The collection of other services not mentioned above such as virtual machines, networking, and security services was also collected and logged but not as thoroughly.

4.1.2 Private Cloud

The collection of data for private cloud was a bit harder. As discussed in 2.4 estimating private computing for a service like a ticket system can be CAPEX but not as many OPEX. But the gathering of data is estimated to be centered around hardware that should be able to handle 100x the estimated traffic of the ticket system to allow expansion. The process of gathering this data was done by looking up the latest server hardware that fulfills the traffic requirements.

4.2 Platform Comparison

The platform comparison was conducted through quantitative research. The authors decided to include the top three public cloud service providers in the quantitative research. The top three public cloud service providers were AWS, Microsoft Azure, and GCP [30]. The platform comparison consisted of two parts, i.e. investigating the pricing and estimating the deployment efforts for respective public cloud service platforms. By choosing to evaluate the top three public cloud platform services, which have a broad supply of services and efficient pricing models, the research will show relevant data for a broader audience of users, other than choosing for example a less popular cloud service platform that offers niched services. The authors compared the different pricing models and pricing levels through thorough quantitative research and by deploying the ticket system to different public cloud providers. A deeper description of the development and deployment of the ticket system will be mentioned later in this chapter. Different services from respective cloud providers were put in a Microsoft Excel sheet. The services were evaluated in the following fields.

- Which platform is being evaluated?
- Which service is being evaluated?
- Is there a free version of the service available?
- What price models are available?
- How much does it cost?
- Is there an automatic distribution setup?
- Are there any extra resources or services needed?

- How much documentation is needed?
- Is the resource or service well-documented?
- What is the deployment efforts rating, on a scale of 0-100?

4.3 Ticket System Development

In order to apply the theoretical literature study that compares different cloud platforms, a case study was conducted to test deployment efforts in a practical scenario and gain insights into the costs associated with hosting the same product on multiple platforms.

The case study is subjected to the deployment of a ticket system. This means that the first step of performing the case study, the development part of the ticket system, was needed. The ticket system will also be used by the host company *amaceit AB* and therefore had requirements that needed to be fulfilled.

4.3.1 Project Requirements

To understand the requirements of the ticket system, an initial interview with the host company *amaceit AB* was performed to understand the specific requirements and use case scenarios. By doing this, the authors could get a basic understanding of what the use case was and could take the next step in the development of the product.

After the requirements were established, a design of the product was developed in the tool Figma [31]. Designing the application in Figma allowed the authors to get a specific prototype of the product, which was used to test the product against users. By doing this the authors were able to get feedback on the application which then was used to improve the tool further.

4.3.2 Architecture

The authors chose to divide the application into a frontend application and a backend API that both communicate with the cloud service provider's authentication service. For example, *amaceit AB* has an Active Directory (AD) at Microsoft that contains all of their internal and external users. By setting up a specific authentication endpoint, the endpoint can check if the user trying to authenticate is in the AD. This way, only the internal and external users at *amaceit AB* get access to the ticket system with their Microsoft credentials.

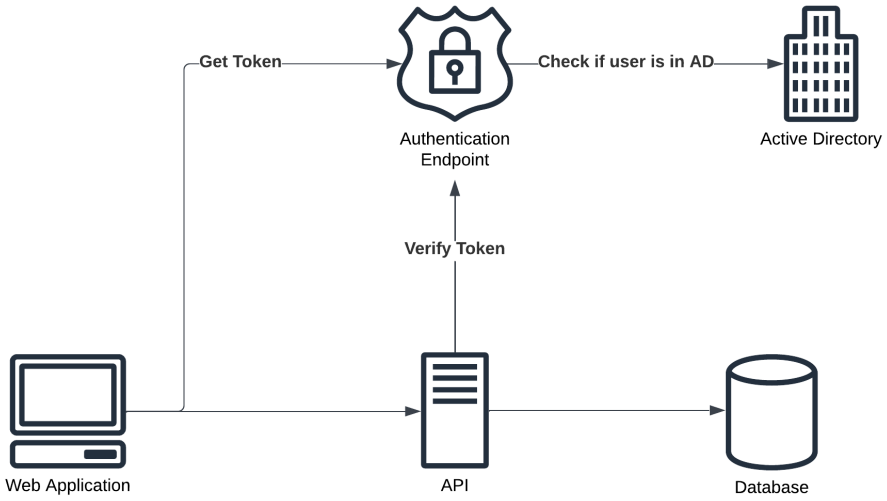

Figure 4.1: General Ticket System Architecture

4.3.3 Frontend Development

In the section on frontend development of the Ticket System, it is important to mention the technology stack used, which includes React [32] and Redux [33]. React is a popular library for building user interfaces, while Redux is a state container for managing application states. These tools were chosen because they offer a powerful combination for building complex web applications with ease and because both authors have good knowledge of developing with these tools.

The frontend designs then furthermore were developed after the Figma designs. By following this procedure, the authors could follow the prototype design to develop the application and achieve the desired result. The authors were able to develop an application that was a near replica of the prototype, besides some components that changes because the requirements of the application changed. The last step of the development of the frontend application included a testing round of users with the host company. This served as a final review of the application and that it achieved the desired requirements from the host company.

4.3.4 Backend Development

The backend was built with the respective cloud provider's serverless service. The backend consisted of a Representational State Transfer (REST) API, the respective cloud provider's authentication service, and a database. The REST API was built respectively on each platform, with Python and SQL Server as a database management system. With this consistency, the authors had to change as little as possible in the code to deploy the application on all the cloud platforms. The backend application also needs to be well-documented when the product is handed over to *amaceit AB*, so they can continue to develop the ticket system.

4.4 Ticket System Deployment

The ticket system was developed as a case study to compare different cloud platforms in terms of their pricing and deployment efforts. The general approach of development was to first create a generic version of the program that could be deployed on all cloud platforms, and then perform individual development for each cloud provider. This allowed for a systematic comparison of the deployment effort required by each provider.

In order to deploy the ticket system, the authors made a deliberate choice to assess the performance of the three public cloud service providers: AWS, Microsoft Azure, and GCP. To manage the deployment process GitHub actions were used, which allowed for seamless integration and effortless updates. GitHub actions and continuous integration for the specific case study might be a bit of an overkill but the goal is to measure that the application would receive a lot of updates. The reason for being overkill is that the system does not need continuous integration because it was only required to be deployed once.

4.4.1 AWS

The first provider that the authors deployed the application to was AWS. Amazon offers a straight forwards way for both a free and paid subscription which allowed the authors to ease set up the account to start deploying the application.

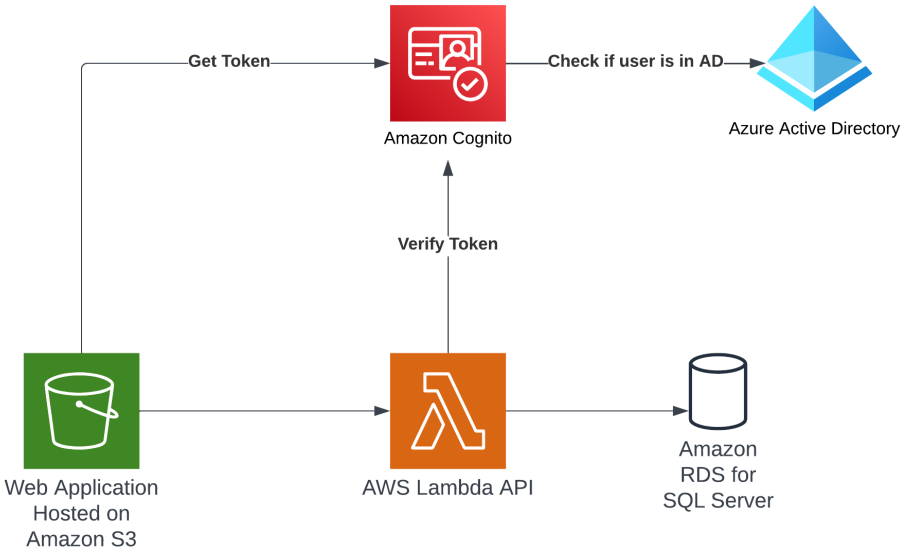

Figure 4.2: Ticket System architecture when hosted on AWS.

4.4.1.1 User Authentication

AWS offers an authentication service called Amazon Cognito [34] which allows a straightforward way to handle users and authentication for the application. The authentication service offers a way to both authenticate users to use the application and can also be used to authenticate users to access the backend system of the application. Having an all-in-one system like this allowed ease of setup for the other services without needing to create multiple systems for managing the user in the system.

When configuring the authentication, AWS uses something called a user pool. A user pool represents a group of users. For our use case, we needed to create two different pools, one for internal users at the host company *amaceit AB* and a pool for external users. When creating the user pool you get a lot of customization to fit your desired situation, for example, which parameters a user enters when registering (email, name, company...). After the steps of configuring the service, the user pool is then assigned to a specific AWS Identity and Access Management (IAM) [35] to specify what they are allowed to do.

4.4.1.2 API - Serverless Computing

The solution for serverless computing that AWS offers is called Lambda [36]. AWS lambda has become over the years one of AWS's best solutions for cloud computing for being cheap and really easy to use. The steps for configuring a lambda include two steps, creating an AWS IAM group for the lambda and creating the actual lambda.

The IAM group specifies exactly what the lambda is allowed to do, for example, which resources it is allowed to access on AWS. The creation of the lambda was even easier where the only configuration that was made was specifying the lambdas name and runtime (python for this situation). Lambda provides a service that allows testing of a request against the lambda function to ensure its proper functionality works as intended. The lambda works by defining a handler that receives two parameters, event, and context 4.1. The event contains the request that the user has sent to the lambda.

Listing 4.1: AWS Lambda Python Handler

```
def handler(event, context):
    return {
        "statusCode": 200,
        "body": json.dumps({"data": "Hello From Lambda"})
    }
```

4.4.1.3 Hosting the website

The last necessary step of deploying the project is hosting the website files. AWS offers a storage solution called S3 [37]. AWS s3 storage solution offers a really simple way of storing information in the cloud. By simply creating a storage bucket and uploading the frontend files, we can then use our website provider to point toward those files.

4.4.1.4 Workflows Deployment

The workflow deployment via GitHub actions for both deploying the website and the backend API was simple. Doing this allows Continuous Integration (CI) to continually release new versions of the software. The documentation on setting up deployment via GitHub actions was straightforward and didn't require a lot of configurations. In listing 4.2 it can be seen the exact configuration for deploying a lambda.

Listing 4.2: AWS Lambda Deployment

```
on:
  push:
    branches:
      - main
jobs:
  build-deploy:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v2
      - uses: actions/setup-python@v2
      - uses: aws-actions/setup-sam@v1
      - uses: aws-actions/configure-aws-credentials@v1
        with:
          aws-access-key-id: ${{ secrets.ID }}
          aws-secret-access-key: ${{ secrets.SECRET }}
          aws-region: eu-west-1
      - run: sam build --use-container
      - run: sam deploy --no-confirm-changeset
             --no-fail-on-empty-changeset --stack-name
             sample-endpoint --s3-bucket
```

Uploading the latest version of the website was a bit more difficult. There wasn't any direct way provided by AWS to update the S3 object via GitHub actions, but there were a lot of 3rd party integrations to achieve this result. The one that was used is called "s3-upload-action" [38]. It has nearly the same structure as updating the Lambdas as seen in listing 4.3.

Listing 4.3: AWS S3 Deployment

```
on:
  push:
    branches:
      - main
jobs:
  upload:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@master
      - uses: shallwefootball/s3-upload-action@master
        with:
          aws_key_id: ${{ secrets.ID }}
```

```
aws_secret_access_key: ${{ secrets.SECRET}}
aws_bucket: ${{ secrets.BUCKET }}
source_dir: 'dirname'
```

4.4.1.5 Storing data

The authors chose to use Amazon RDS for SQL Server as a solution for storing data. Amazon RDS for SQL Server was easy to set up and was automatically configured to fit the ticket system's needs. Amazon RDS provides a service for backing up files regularly [39]. The backup of the database was turned on by default. These backups were kept for a configurable number of days. The authors did not use any Object-Reltaional Mapping (ORM) for the backend, so the database had to be set up manually by importing a SQL file that contained all the tables and constraints. This could be done directly in the built-in query editor.

Listing 4.4: The SQL code for creating the table 'User'.

```
CREATE TABLE [User] (
    Id VARCHAR(36) PRIMARY KEY DEFAULT NEWID(),
    Role VARCHAR(10) DEFAULT 'USER',
    CompanyId VARCHAR(36),
    LastLogin DATETIME NOT NULL DEFAULT GETDATE(),
    Created DATETIME NOT NULL DEFAULT GETDATE(),
    Email VARCHAR(500) NOT NULL,
    Name VARCHAR(500) NOT NULL,
    FOREIGN KEY (CompanyId) REFERENCES [Company](Id)
);
```

4.4.2 Microsoft Azure

The second cloud platform that the authors evaluated was Microsoft Azure. *amaceit AB*'s IT infrastructure is based on Microsoft's products and therefore the ticket system was going to be hosted on Azure when the product was handed over. Microsoft offers all accounts a free trial with 200$ to be spent in 30 days and over 40 popular services to use during the first 12 months [40].

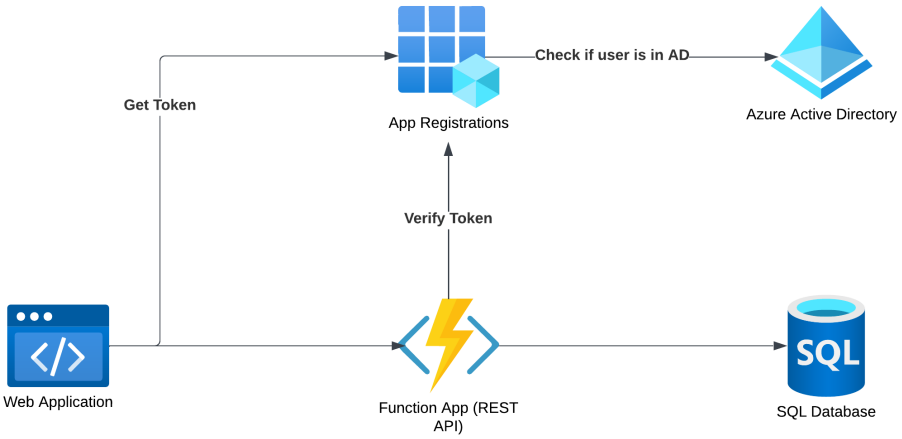

Figure 4.3: Ticket System architecture when hosted on Azure.

4.4.2.1 User authentication

The authors used Azure App Registrations, which is an authentication service, to authenticate users to the ticket system. Azure App Registrations supports authentication through Azure AD, which means that all the users in *amaceit AB* AD can be authenticated to the ticket system application [41]. It also means that admins for the AD can add external users, for instance, a client user, that can then access the ticket system through the App Registration service.

4.4.2.2 API - Azure Functions

Azure Functions is Azure's resource for building serverless applications. The Functions service provides the developer with creating several functionalities, such as API endpoints called 'HTTPTriggers' or 'TimerTriggers' that can execute functions when called by a timer. Since Visual Studio Code is developed by Microsoft, there are many built-in extensions for Azure and it is possible to manage all of the subscription resources directly in Visual Studio Code. Creating a new trigger for the API was done automatically with the Visual Studio Azure Functions extension. Configuring Cross-Origin Resource Sharing (CORS) was easily done by instead of adding CORS headers for each Hypertext Transfer Protocol (HTTP) response in the triggers, the Azure Functions app CORS could be modified easily in the Azure Portal. The Azure Functions API was automatically deployed to Aure with a GitHub Actions workflow. The workflow was created automatically when setting up the Azure

Functions app.

Listing 4.5: Azure Functions Deployment

```
on:
  push:
    branches:
      - main

env:
  AZURE_FUNCTIONAPP_PACKAGE_PATH: '.'
  PYTHON_VERSION: '3.9'

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Setup Python version
        uses: actions/setup-python@v1
        with:
          python-version: ${{ env.PYTHON_VERSION }}

      - name: Create and start virtual environment
        run: |
          python -m venv venv
          source venv/bin/activate

      - name: Install dependencies
        run: pip install -r requirements.txt

      - name: Upload artifact for deployment job
        uses: actions/upload-artifact@v2
        with:
          name: python-app
          path: |
            .
            !venv/
```

```
  deploy:
    runs-on: ubuntu-latest
    needs: build
    environment:
      name: 'Production'
      url: ${{ steps.deploy-to-function.outputs.webapp-url }}

    steps:
      - name: Download artifact from build job
        uses: actions/download-artifact@v2
        with:
          name: python-app
          path: .

      - name: 'Deploy to Azure Functions'
        uses: Azure/functions-action@v1
        id: deploy-to-function
        with:
          app-name: 'amaceit-ticket-system-api'
          package: ${{ env.AZURE_FUNCTIONAPP_PACKAGE_PATH }}
          publish-profile: ${{ secrets.AZUREAPPSERVICE }}
```

4.4.2.3 Hosting the website

The web application was hosted on Azure with Azure Static Web Apps which is a free resource (during the 12-month free trial period) that can host static websites built with HyperText Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript [42]. Each time the web application was updated on the main branch of the repository the web app was deployed on Azure. The GitHub Actions workflow file was created automatically when setting up the resource in Azure.

Listing 4.6: Azure Static Web Apps Deployment

```
on:
  push:
    branches:
      - main
  pull_request:
```

```
    types: [opened, synchronize, reopened, closed]
    branches:
      - main

jobs:
  build_and_deploy_job:
    if: github.event_name == 'push' ||
        (github.event_name == 'pull_request' &&
        github.event.action != 'closed')
    runs-on: ubuntu-latest
    name: Build and Deploy Job
    steps:
      - uses: actions/checkout@v3
        with:
          submodules: true
      - name: Build And Deploy
        id: builddeploy
        uses: Azure/static-web-apps-deploy@v1
        env:
          CI: false
        with:
          azure_static_web_apps_api_token: ${{ secrets.TOKEN }}
          repo_token: ${{ secrets.GITHUB_TOKEN }}
          action: "upload"

  close_pull_request_job:
    if: github.event_name == 'pull_request' &&
        github.event.action == 'closed'
    runs-on: ubuntu-latest
    name: Close Pull Request Job
    steps:
      - name: Close Pull Request
        id: closepullrequest
        uses: Azure/static-web-apps-deploy@v1
        with:
          azure_static_web_apps_api_token: ${{ secrets.TOKEN }}
          action: "close"
```

4.4.2.4 Storing data

Storing data was solved by using Azure SQL Database which uses SQL Server as Database Management System (DBMS). Azure SQL The file containing all tables and constraints for the ticket system could be added easily in the built-in query editor.

4.4.3 Google Cloud

The third cloud platform the authors evaluated and deployed the application to was the GCP. GCP offers both a free trial and a free tier with many free usage limit resources and 300$ of credits to spend.

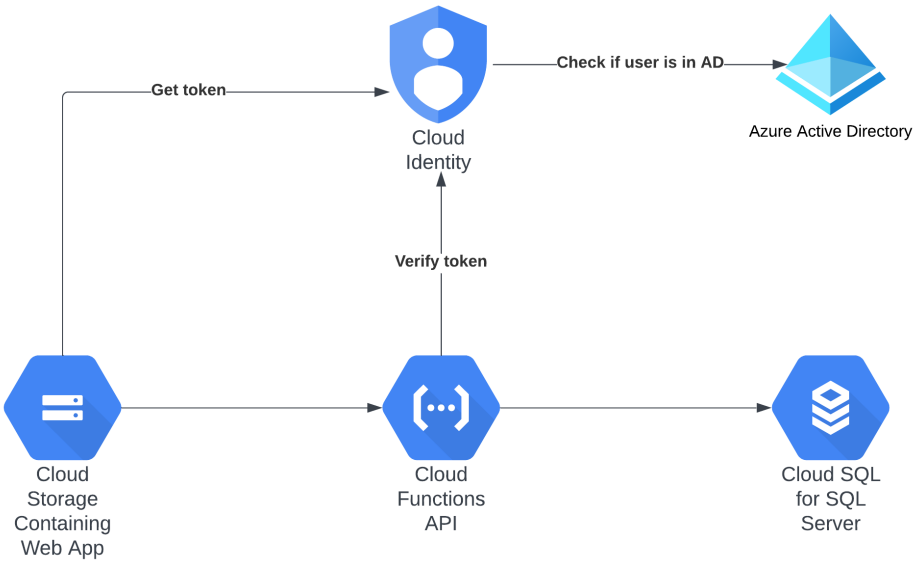

Figure 4.4: Ticket System architecture when hosted on GCP.

4.4.3.1 User authentication

GCP offers user authentication for businesses and organizations through Cloud Identity resource [43]. Since *amaceit AB* is a Microsoft-based company, Cloud Identity was a great choice for user authentication due to its ability to federate Google Cloud with AD. This means that GCP services, for example, the Cloud Functions API, could be authenticated with Cloud Identity that only accepts internal and external users from Azure AD.

4.4.3.2 API - Cloud Functions

Cloud Functions is a serverless (FaaS) type of product that Google Clouds provides [44]. Cloud Functions is very similar to Azure Functions in that it makes the development much easier for the developer since the developer only needs to write small parts of code that listen to certain events. These events are also called triggers, just as in Azure. Cloud Functions has an automatic distribution setup that works both for GitHub and Cloud Source Repositories that lowered the deployment efforts of the API.

Listing 4.7: Cloud Functions Deployment

```
jobs:
  job_id:
    runs-on: 'ubuntu-latest'
    permissions:
      contents: 'read'
      id-token: 'write'

    steps:
    - uses: 'actions/checkout@v3'

    - id: 'auth'
      uses: 'google-github-actions/auth@v1'
      with:
        workload_identity_provider:
            ${{ secrets.WORKLOAD_IDENTITY_PROVIDER }}
        service_account:
            ${{ secrets.SERVICE_ACCOUNT }}

    - id: 'deploy'
      uses: 'google-github-actions/deploy-cloud-functions@v1'
      with:
        name: 'my-function'
        runtime: 'python3.9'
```

4.4.3.3 Hosting the website

The web application was hosted on Cloud Storage, which is a GCP service for storing unstructured data. Cloud Storage is very similar in a way to AWS s3, which are both storage services that can be used to host static web applications.

Listing 4.8: Cloud Storage Deployment

```
jobs:
  job_id:
    permissions:
      contents: 'read'
      id-token: 'write'

    steps:
    - name: Build And Deploy
        id: builddeploy
        run: |
            npm install
            npm run build

    - id: 'auth'
      uses: 'google-github-actions/auth@v1'
      with:
        workload_identity_provider:
            ${{ secrets.WORKLOAD_IDENTITY_PROVIDER }}
        service_account:
            ${{ secrets.SERVICE_ACCOUNT }}

    - id: 'upload-file'
      uses: 'google-github-actions/upload-cloud-storage@v1'
      with:
        path: './'
        destination: 'bucket/'

    - id: 'uploaded-files'
      uses: 'foo/bar@main'
      env:
        file: '${{ steps.upload-file.outputs.uploaded }}'
```

4.4.3.4 Storing data

GCP offers Cloud SQL for SQL Server if users want to migrate and run SQL Server on GCP. Just like AWS and Azure, GCP provides automated configurations for the SQL Server. Cloud SQL has, just like AWS and Azure, automated backups enabled as default with seven automated backups [45].

The Internet Protocol (IP) address for both the local and the Cloud Functions app was registered for access to the database. Cloud SQL does not have a built-in query editor in the browser and the developer could connect either through SQL Server Management Studio or by using the built-in Cloud Shell in the browser. Both ways of connecting to the database and executing queries needed authentication with username and password. Creating all of the tables and constraints for the database was done in the Cloud Shell.

4.4.4 Managing Serverless Applications with Serverless

In the section 4.4 it has been mentioned the process and services used to deploy the application for the different providers. The specific step of using serverless computing to manage the backend endpoints has been one of the main parts of deploying the service. To ease CI a framework called Serverless [46] has become popular. Serverless provides developers with an easy way of deploying to AWS or other providers by simply running one single command. By doing this it is not required to configure anything on the cloud provider manually but declare everything in a serverless.yml file.

Using this structure, a developer can in a more ordered and structured fashion configure and manage a serverless infrastructure. The configuration can be anything from configuring the specific files a serverless instance should contain to the resources it is allowed to access. Serverless also provides a way to deploy a service to multiple cloud providers at once (in many cases, not a useful option) by simply running one command. In listing 4.9 it can be seen how simply an endpoint can be configured using Serverless.

Listing 4.9: Serverless Deployment

```
service: ticket-system

frameworkVersion: '3'

provider:
  name: aws
  runtime: python3.9
  region: eu-west-1
  stage: main
  iamManagedPolicies:
    - "arn:aws:iam::aws:policy/AmazonS3FullAccess"
    - "arn:aws:iam::aws:policy/AmazonCognitoPowerUser"
```

```
functions:
  sampleEndpoint:
    handler: sample-endpoint/handler.lambda_handler
    description: Sample API endpoint
    memorySize: 128
    events:
      - http:
          path: sample-endpoint
          method: post
          cors: true
          authorizer:
            name: cognito-authorizer
            arn: ...
```

4.5 Measure of Deployment

The measuring of deployment was made by measuring the difficulty of deploying the ticket system to the different providers and the time it took. The difficulty was measured using different parameters such as the amount of documentation, required work, and difficulty of performing the deployment. It was possible to use these different parameters against all three different providers to measure the deployment effort of it.

An Excel document was then used to collect all of the measurements for each of the services. This means while deploying the application against the services the time it took was estimated and the deployment steps (e.g., use of documentation) were logged to estimate the deployment effort. After the service was deployed the logged information was then gathered, summarized, and lastly rated on a scale from 0 to 100.

Chapter 5

Results and Analysis

This chapter presents the results and analysis of the different cloud computing services and presents the challenges and solutions related to deploying applications on the different cloud providers. The objectives of the case study were to evaluate the performance, availability, scalability, pricing, and deployment effort of private, public, and hybrid cloud computing services, and to identify the challenges of doing so.

In order to obtain an understanding of the effort and pricing of the different cloud platforms, a mixed-methods research design was utilized for this study. This involved collecting and analyzing both quantitative and qualitative data from various sources. Case studies were used to provide real-world examples of the challenges and opportunities associated with cloud computing. The case study has given a practical example of trying to migrate and deploy an existing application on different cloud platforms.

The data collected from these various sources were analyzed using several different techniques. Descriptive statistics are used to summarize the pricing and deployment data, as well as inferential statistics to measure against the hypotheses data and identify differences between private and cloud providers. Thematic analysis was also used to analyze the data collected from the case studies, identifying key themes and patterns from deploying to different providers. By using a combination of these different methods, a more complete picture of the research was obtained.

5.1 Overview of Pricing and Pricing Models

This section covers an overview of the different pricing of the different cloud computing services. Each cloud computing company's services are

represented in different pricing models.

The pricing models offered by cloud service providers are extremely important in determining the cost-effectiveness and flexibility of adopting cloud computing services. In this section, we provide an in-depth overview of the pricing structure in cloud computing, focusing on the various pricing models available to businesses and the different services cloud computing company offers.

5.1.1 Instance Computing

Instance computing has become one of the most important and popular services that cloud providers provide. The selection of the right instance type plays a crucial role in optimizing performance, scalability, and cost-effectiveness. Different cloud service providers offer a wide range of instance computing options to meet any specific requirement a customer has.

5.1.1.1 Amazon AWS

AWS's instance computing, EC2, is offered as on-demand, spot instance, and dedicated pricing models [47]. The AWS instance computing offers a wide range of different instances based on your needs. To date, AWS provides a total of 565 different instance types in the region of eu-west-1. The two most popular instance types are called t3.micro and m5.large.

The t3.micro instance is part of the T3 family of instances, which is designed for general-purpose workloads and offers a balance of computing power, memory, and network resources. The specific instance t3.micro offers 2 vCPU, 1GiB of memory, and Network up to 5 Gigabit as on-demand pricing for $0.0104 per hour. Running the instance constantly for a year the cost comes down to $91.104.

The m5.large instance is part of the M5 family of instances, which is optimized for a balance of compute power, memory, and network resources. The specific instance m5.large offers 2 vCPU, 8GiB of memory, and Networking up to 10 Gigabit as on-demand pricing for $0.096. Running the instance constantly for a year the cost comes down to $840.96.

The EC2 instances are also offered as dedicated hosting which offers savings compared to the on-demand pricing. The dedicated hosting model offers 65 different instance types. Both the T3 and M5 family is offered at a discounted rate of 37%.

The third model the EC2 instances are offered in is spot pricing. Spot pricing is a scheme that certain instances cheaper when they are unused by

other customers. Therefore the spot pricing constantly changes from time to time. As of writing this, the t3.micro had a saving of 59.67% and m3.large 45% which can be seen in figure 5.1.

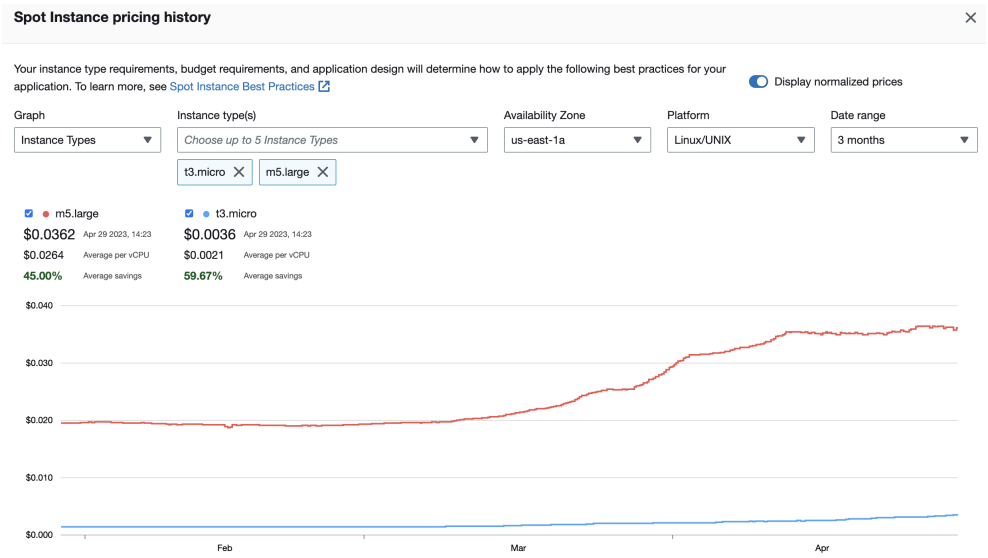

Figure 5.1: Spot Pricing History Graph

5.1.1.2 Microsoft Azure

Microsoft Azure's instance computing, virtual machine, is offered as on-demand (Pay as you go), spot instance, reserved instance, and saving plan [48]. Like AWS instance, computing offers a wide range of different instance families to fit a customer's needs. To date, Azure offers a total of 83 different families and each family contains 5-10 different instance options. The most popular options are the D2 v3 and F4s_v2.

The D2 v3 instance is part of the Dv3 family. The family is designed for general purposes and provides a balance of CPU-to-memory ratio and is suitable for a wide range of workloads. The specific instance D2 v3 offers 2 vCPU, 8GiB of memory, and 50 GiB of temporary storage as on-demand pricing for $0.0973 per hour. Running the instance constantly for a year the cost comes down to $852.348.

The F4s_v2 instance is part of the Fs_v2 family. The family is designed for compute-intensive workloads that require high CPU performance. The specific instance F4s_v2 offers 4 vCPU, 8 GiB of memory, and 16 GiB of temporary storage as on-demand pricing for $0.199 per hour. Running the instance constantly for a year the cost comes down to $1743.24.

The virtual machine instances are also offered as dedicated hosting which offers savings compared to the on-demand pricing. The dedicated hosting model is offered for all of the different instance types as a 1 and 3-year plan. Both the Fs_v2 and Dv3 family is offered at a discounted rate of 13% for 1 year and 31% for 3 years.

The third model the virtual machine instances are offered in is spot pricing. Spot pricing is a scheme that certain instances cheaper when they are unused by other customers. Therefore the spot pricing constantly changes from time to time. As of writing these are offered at a discounted price of 61%.

5.1.1.3 Google GCP

GCP's Compute Engine [49], equivalent to Azure's virtual machine, offers flexible pricing models including on-demand (pay-as-you-go), committed use discounts, and reserved instances. GCP Compute Engine provides a wide range of instance families to fit all customer needs. Currently, GCP offers numerous instance families, each comprising multiple instance options. The popular options include the N2, N1, E2, and E2 shared-core families.

For example, the N2 family offers a balanced CPU-to-memory ratio suitable for various workloads [50]. The N2 standard instance, such as the N2 standard-2 instance, provides 2 vCPUs, and 8 GB of memory at an on-demand price of $0.112452 per hour. Running this instance continuously for a year would cost approximately $985.07952.

On the other hand, the E2 family is designed for cost-effective workloads with high-performance requirements. The E2 standard instance, like the E2 standard-4 instance, offers 4 vCPUs, and 16 GB of memory, at an on-demand price of $0.15518 per hour. The yearly cost for running this instance constantly would amount to around $1359.3768.

Compute Engine also offers committed use contracts, which provide significant discounts for 1-year and 3-year commitments. Additionally, GCP's spot instances, offer significantly lower prices compared to regular instances which in many cases offer up to 6x cheaper prices In table 5.1 and 5.2 it can be seen the discounted prices for both the predefined N2 and E2 family instance.

Table 5.1: GCP N2 Instance Pricing

Item	On-demand price (USD)	Spot price (USD)	1 year commitment	3 year commitment
Predefined vCPUs	$0.036602 / vCPU hour	$0.00591 / vCPU hour	$0.023059 / vCPU hour	$0.016471 / vCPU hour
Predefined Memory	$0.004906 / GB hour	$0.000792 / GB hour	$0.003091 / GB hour	$0.002208 / GB hour

Table 5.2: GCP G2 Instance Pricing

Item	On-demand price	Spot price	1-year commitment)	3-year commitment
Predefined vCPUs	$0.025255 / vCPU hour	$0.007955 / vCPU hour	$0.015911 / vCPU hour	$0.011365 / vCPU hour
Predefined Memory	$0.003385 / GB hour	$0.001066 / GB hour	$0.002133 / GB hour	$0.001523 / GB hour

5.1.2 Serverless Computing

Serverless computing is a set of services in cloud computing that allows developers to focus solely on writing and deploying code without the need to manage the underlying infrastructure. The selection of the right serverless computing type plays a crucial role in cost-effectiveness as well as performance-wise. Each of the cloud providers discussed in this report offers multiple serverless computing services to fit their customer's needs.

5.1.2.1 Amazon AWS

The two primary serverless computing services offered by AWS are Lambda [36] and Fargate [51]. Lambda is the more popular option which offers a cheap and quick solution for serverless computing while Fargate costs more but allows for more configuration.

Lambdas has two different pricing categories, duration and request amount. The duration is priced by the amount of GiB of ram times the duration of the computing [52]. The request pricing is a flat fee for every million requests sent to lambda which can be seen in table 5.3. The cost of the duration is divided into three different categories based on the amount of computing performed per month. The more computing that has been done the cheaper it becomes per second. For the use case of using Lambdas as an HTTP endpoint (API), the average runtime of the lambda is around 250ms.

Table 5.3: AWS Lambda Pricing

Architecture	Duration	Price per GB-second	Price per 1M requests
x86	First 6 Billion GB-hour / month	$0.06	$0.20
Arm	First 7.5 Billion GB-hour / month	$0.048	$0.20

Fargate allows for more customization regarding the specific computing power used in the serverless request [53]. This means that a customer is allowed to specify the storage amount, Virtual Processor (vCPU)s, and ram. Therefore the pricing of the Fargate is specified on those three parameters. Storage is priced at $0.000133 per GiB per hour (excluding the 30GiB of free

storage). The vCPUs are priced at $0.0144309 per vCPU per hour. Lastly, the ram is priced at $0.0015838 per GiB per hour. It is important to note that there are a few constraints where there requires a minimum ram/vCPUs towards each other. A few of these constraints can be seen in table 5.4.

Table 5.4: AWS Fargate Constraints

CPU	Memory Values
0.25 vCPU	0.5 GB, 1 GB, and 2 GB
1 vCPU	Min. 2 GB and Max. 8 GB, in 1 GB increments
4 vCPU	Min. 8 GB and Max. 30 GB, in 1 GB increments
16 vCPU	Min. 32 GB and Max. 120 GB, in 8 GB increments

5.1.2.2 Microsoft Azure

The main serverless computing service offered by Microsoft Azure is Azure Function [54]. Azure Functions is the more popular option of the serverless solutions that Azure offers. Azure Functions is offered both as a normal and premium plan. The premium plan allows for more configuration.

Azure Functions Premium has two different pricing categories, memory duration, and vCPU duration which can be seen in table 5.6. The memory duration is priced by the amount of GiB of ram times the duration of the computing and the vCPU duration is priced by the amount of vCPUs times the duration of the computing. [55]. The Azure Functions normal plan has two different pricing categories, execution time and request amount. The request pricing is a flat fee for every million requests sent to the lambda at a fee of $0.2. The second pricing category, execution time, is priced at $0.0576 per GiB memory per hour which can be seen in table 5.5. For the use case of using Azure Functions as an HTTP endpoint (API), the average runtime of the Azure Function is around 250ms.

Table 5.5: Azure Function Basic Pricing

Meter	Free Grant (Per Month)	Pay as you go
Execution Time	400,000 GB-hours	$0.000016/GB-hour
Total Executions	1 million executions	$0.20 per million executions

Table 5.6: Azure Function Premium Pricing

Meter	Plan	Pay as you go	1 year savings plan	3 year savings plan
vCPU duration	vCPU	$0.173 vCPU/hour	$0.14359 vCPU/hour (17% savings)	$0.14359 vCPU/hour (17% savings)
Memory duration	Memory	$0.0123 GB/hour	$0.010209 GB/hour (17% savings)	$0.010209 GB/hour (17% savings)

5.1.2.3 Google GCP

The main serverless computing service offered by GCP is Cloud Functions [56]. Cloud Functions is the more popular option of the serverless solutions that Google GCP offers.

GCP Cloud Functions has three different pricing categories, memory duration, vCPU duration and lastly requests amount. The request pricing is a flat fee for every million requests sent to the lambda at a fee of $0.4 [57] where the first 2 million requests are free every month. In table 5.7 it can be seen the different instances that cloud functions offer and the pricing of each function. For the use case of using Cloud Functions as an HTTP endpoint (API), the average runtime of the instance is around 250ms.

Table 5.7: GCP Cloud Functions Pricing

Memory	vCPU	Price/hours (Tier 1 Price)	Price/hour (Tier 2 Price)
128MB	0.083 vCPU	$0.008316	$0.011664
512MB	0.333 vCPU	$0.0333	$0.04662
2048MB	1 vCPU	$0.1044	$0.14616
8192MB	2 vCPU	$0.2448	$0.34272
32768MB	8 vCPU	$9.792	$13.7088

5.1.3 Data Storage

In cloud computing, data storage is a fundamental aspect that plays a crucial role in any digital infrastructure. As businesses generate vast amounts of data, efficient and cost-effective storage solutions become extremely important.

5.1.3.1 Amazon AWS

AWS offers one option for storage which is called S3 [37] which is offered in the pricing model as on-demand. The pricing for the on-demand is divided into the categories of storage size, request & data retrieval, and data transfer. The S3 service is then also divided into different categories of storage types. The storage type is divided into different categories depending on the retrieval

time of data. This means that they offer a lower price to store data but the cost of the retrieval time of the data could be high.

The different storage types are divided into the categories S3 standard (General purpose storage for any type of data, typically used for frequently accessed data), S3 Intelligent - Tiering (Automatic cost savings for data with unknown or changing access patterns), S3 Standard - Infrequent Access (For long-lived but infrequently accessed data that needs millisecond access), S3 One Zone - Infrequent Access (For re-creatable infrequently accessed data that needs millisecond access), S3 Glacier Instant Retrieval (For long-lived archive data accessed once a quarter with instant retrieval in milliseconds), S3 Glacier Flexible Retrieval - (For long-term backups and archives with retrieval option from 1 minute to 12 hours) and lastly S3 Glacier Deep Archive (For long-term data archiving that is accessed once or twice in a year and can be restored within 12 hours). Despite the cost savings, the cheaper options have a significant drawback because in many cases, there is a minimum time the data needs to be stored. Because of this additional costs could exist. All of the specific pricing can be seen in table 5.8.

Table 5.8: AWS S3 Pricing

Storage Type	**Storage Tier**	**Price per GB**
S3 Standard	First 50 TB / Month	$0.023
S3 Intelligent - Tiering	Monitoring and Automation First 50 TB / Month	$0.0025 per 1,000 objects $0.023
S3 Standard - Infrequent Access	All Storage / Month S3 One Zone - Infrequent Access	$0.0125 $0.01
S3 Glacier Instant Retrieval	All Storage / Month	$0.004
S3 Glacier Flexible Retrieval (Formerly S3 Glacier)	All Storage / Month	$0.0036
S3 Glacier Deep Archive	All Storage / Month	$0.00099

The request and data retrieval pricing category is reversed for the cheaper options of the different S3 types. This means that the cheaper the S3 storage is (longer retrieval time), the more expensive it becomes to retrieve it which can be seen in table 5.9.

Table 5.9: AWS Request & Data Retrieval Pricing

Data	**PUT, COPY, POST, LIST requests**	**GET, SELECT, and all other requests**	**Lifecycle Transition requests into**	**Data Retrieval requests**	**Data retrievals (per GB)**
S3 Standard	$0.005	$0.0004	N/A	N/A	N/A
S3 Standard - Infrequent Access	$0.01	$0.001	$0.01	N/A	$0.01
S3 Glacier Deep Archive	$0.05	$0.0004	$0.05	N/A	N/A
Standard	N/A	N/A	N/A	$0.10	$0.02
Bulk	N/A	N/A	N/A	$0.025	$0.0025

The last pricing category for AWS S3 is the transfer rate. The transfer rate is divided into transfer IN and transfer OUT. The transfer in rate is $0 while

the transfer rate out is a bit more. This has to do with the lock-in model which makes it harder to move from a cloud platform but easy to move to it. The rates are represented in table 5.10.

Table 5.10: AWS Transfer Pricing

Data Transfer	Transfer Type	Price
Data Transfer IN To Amazon S3 From Internet	All data transfer in	$0.00/GB
Data Transfer OUT From Amazon S3 To Internet	First 10 TB / Month	$0.09/GB

5.1.3.2 Microsoft Azure

Microsoft Azure offers two option for storage which is called Azure Blob Storage [58] and Azure Files [59]. Both services are offered in the pricing model as on-demand. The pricing for the on-demand is divided into the categories of storage size, request & data retrieval, data transfer, and storage plan (standard, premium). The storage service is then also divided into different categories of storage types. The storage type is divided into different categories depending on the retrieval time of data. This means that they offer a lower price to store data but the cost of the retrieval time of the data could be high.

Azure Blob Storage is offered in 5 different plans and each plan has its benefits and drawbacks. The plans that are included are premium (high-performance tier for mission-critical workloads with low latency requirements), hot (frequently accessed data with low latency and optimized for active workloads), cool (data accessed less frequently but still requires quick access when needed, suitable for long-term backup and secondary copies), cold (data rarely accessed but requires long-term retention, optimized for archival and compliance scenarios) and lastly archive (most cost-effective tier for data with minimal access requirements, ideal for long-term retention and regulatory compliance). Even for the cost savings the cheaper options have one major drawback which is that in many cases the data needs to be stored for a minimum amount of time [60]. Because of this additional costs could exist. All of the specific pricing can be seen in table 5.11.

Table 5.11: Azure Blob Storage Pricing

Data Storage	Price Tier	Premium	Hot	Cool	Cold	Archive
First 50 TB / month	Price per GB	$0.15	$0.018	$0.01	$0.0036	$0.00099
Next 450 TB / month	Price per GB	$0.15	$0.0173	$0.01	$0.0036	$0.00099
Over 500 TB / month	Price per GB	$0.15	$0.0166	$0.01	$0.0036	$0.00099

The actual storage pricing is also available as a saving plan for both 1 and 3-year terms. The pricing plan for the reserved pricing for the service Azure Blob Storage can be seen in table 5.12.

Table 5.12: Azure Blob Storage Pricing Reserved

Reserved	1-year reserved			3-year reserved		
Capacity	Hot	Cool	Archive	Hot	Cool	Archive
100 TB / month	$1,545	$840	$91	$1,244	$676	$84
1 PB / month	$15,050	$8,179	$883	$11,963	$6,502	$810

The second important pricing category of storage is the operations and data transfer cost. These expenses are different for the different storage types. It is much more expensive to read and write to the archive storage type than the cheapest option storage type, hot which can be seen in table 5.13. It is important to note that Azure charges a bandwidth cost of $0.01 per GB transferred to the internet.

Table 5.13: Azure Blob Operation/Data Transfer Cost

Operations and data transfer	Premium	Hot	Cool	Cold	Archive
Write operations (per 10,000)	$0.0228	$0.065	$0.13	$0.234	$0.13
Read operations (per 10,000)	$0.0019	$0.005	$0.013	$0.13	$6.50
Iterative Read Operations (per 10,0003	N/A	$0.005	$0.013	$0.13	$6.50
Iterative Write Operations (100's)	N/A	$0.065	$0.13	$0.234	$0.13
Data Retrieval (per GB)	Free	Free	$0.01	$0.03	$0.02
Data Write (per GB)	Free	Free	Free	Free	Free
Index (GB/month)	N/A	$0.026	N/A	N/A	N/A
All other Operations (per 10,000), except Delete	$0.0019	$0.005	$0.005	$0.0052	$0.005

Azure Files is offered in 4 different plans and each plan has its benefits and drawbacks. The plans that are included are premium (provides high-performance, low-latency file shares optimized for transactional workloads with demanding transactions and throughput requirements), transaction optimized (offers optimized performance for transactional workloads with

a balance between performance and cost), hot (optimized for frequently accessed file shares with low latency and quick data access), and lastly cool (designed for infrequently accessed file shares with lower storage costs, suitable for data with longer retention periods and less frequent access). Even for the cost savings the cheaper options have one major drawback which is that in many cases the data needs to be stored for a minimum amount of time [61]. Because of this additional costs could exist. All of the specific pricing can be seen in table 5.14.

Table 5.14: Azure Files Storage Pricing

Storage Type	Premium	Transaction optimized	Hot	Cool
Data at-rest (GiB/month)	$0.16 per provisioned GiB	$0.06 per used GiB	$0.0255 per used GiB	$0.015 per used GiB
Snapshots (GiB/month)	$0.136 per used GiB	$0.06 per used GiB	$0.0255 per used GiB	$0.015 per used GiB
Metadata at-rest (GiB/month)	Included	Included	$0.027	$0.027

The actual storage pricing is also available as a saving plan for both 1 and 3-year terms. The pricing plan for the reserved pricing for the service file storage can be seen in table 5.15.

Table 5.15: Azure Files Reserve Storage Pricing

Reserved Capacity	1-year reserved			3-year reserved		
	Premium	Hot	Cool	Premium	Hot	Cool
10 TiB / month	$1,343.50	$214.09	$125.92	$1,081.34	$172.34	$101.39
100 TiB / month	$12,779.50	$2,036.75	$1,198.09	$10,485.75	$1,671.17	$983.03

The second important pricing category of storage is the operations and data transfer cost. These expenses are different for the different storage types. It is much more expensive to read and write to the Cool storage type than the cheapest option storage type, Premium/Transaction Optimized which can be seen in table 5.16.

Table 5.16: Azure Files Operation/Data Transfer Cost

Transactions and Data Transfer	Premium	Transaction optimized	Hot	Cool
Write transactions (per 10,000)	Included	$0.015	$0.065	$0.13
List transactions (per 10,000)	Included	$0.015	$0.065	$0.065
Read transactions (per 10,000)	Included	$0.0015	$0.0052	$0.013
All other operations (per 10,000)	Included	$0.0015	$0.0052	$0.0052
Data retrieval (per GiB)	N/A	N/A	N/A	$0.01

5.1.3.3 Google GCP

GCP offers one option for storage which is called Cloud Storage [62] which is offered in the pricing model as on-demand. The pricing for the on-demand is divided into the categories of storage size, request & data retrieval, and data transfer. The Cloud Storage service is then also divided into different categories of storage types. The storage type is divided into different categories depending on the retrieval time of data. This means that they offer a lower price to store data but the cost of the retrieval time of the data could be high.

The different storage types are divided into the categories [63] Standard Storage (designed for frequently accessed data that requires high availability and low latency), Nearline Storage (suitable for data that is accessed less frequently but requires relatively fast retrieval when needed), Durable Reduced Availability (DRA) Storage (a cost-effective storage option with lower availability compared to standard storage), Coldline Storage (is optimized for long-term archival storage with infrequent access) and lastly Archive Storage (is the most cost-effective storage option offered by GCP. It is designed for long-term retention and regulatory compliance). Even for the cost savings the cheaper options have one major drawback which is that in many cases the data needs to be stored for a minimum amount of time. Because of this additional costs could exist. All of the specific pricing can be seen in table 5.17.

Table 5.17: GCP Cloud Storage Pricing

Location	Standard storage (GB/Month)	Nearline storage (GB/Month)	Coldline storage (per GB/Month)	Archive storage (GB/Monthh)
USA	$0.020	$0.010	$0.004	$0.0012
Europe	$0.023	$0.013	$0.006	$0.0025
Asia	$0.020	$0.010	$0.005	$0.0015

The request and data retrieval pricing category is reversed for the cheaper options of the different Cloud Storage types. This means that the cheaper the storage is (longer retrieval time), the more expensive it becomes to retrieve it which can be seen in table 5.18.

Table 5.18: GCP Request & Data Retrieval Pricing

Storage Class	Price per 1,000 operations
Standard storage	$0.005
Nearline storage	$0.01
Coldline storage	$0.02
Archive storage	$0.05

The last pricing category for GCP Cloud Storage is the transfer rate. The transfer rate is divided into transfer IN and transfer OUT. The transfer in rate is $0 while the transfer rate out is a bit more. This has to do with the lock-in model which makes it harder to move from a cloud platform but easy to move to it. The rates are represented in table 5.19. It is important to note that GCP charges a bandwidth cost of $0.01 per GB transferred to the internet.

Table 5.19: GCP Transfer Pricing

Storage Type	Price per GB
Standard storage	$0 per GB
Nearline storage	$0.01 per GB
Coldline storage	$0.02 per GB
Archive storage	$0.05 per GB

5.1.4 Database

The field of database management plays a crucial role in modern information systems, enabling applications and services to efficiently store, organize, and retrieve vast amounts of data. Cloud providers offer a wide range of different database solutions to specific customers' needs.

5.1.4.1 Amazon AWS

AWS offers both NoSQL and SQL database options. The NoSQL's most popular solution is the service called DynamoDB [64] which is a serverless on-demand pricing service. The SQL option (RDS [65]) offers an on-demand pricing service. Because DynamoDB is a serverless solution the pricing is based on both the amount of read and writes made but also the amount of

storage used. This is a bit different from the SQL option which offers only one pricing category of an hourly rate for the database being active.

The DynamoDB option as discussed is AWS's own solution of a NoSQL database which uses a primary key sort key solution. Using this type of solution requires an extremely small amount of computing power to retrieve items (if done correctly) from the database. This means when a read or write operation is performed the database capacity is divided into units. The pricing of units can be seen in table 5.20. The second pricing category of DynamoDB is the storage amount. The pricing per GiB comes down to $0.28 where the first 25GiB is free of charge.

Table 5.20: AWS DynamoDB Write/Read Pricing

On-Demand Throughput Type	**Price**
DynamoDB Standard table class	
Write Request Units (WRU)	$1.25 per million write request units
Read Request Units (RRU)	$0.25 per million read request units
DynamoDB Standard-Infrequent Access	
Write Request Units (WRU)	$1.56 per million write request units
Read Request Units (RRU)	$0.31 per million read request units

The relational databases offer on-demand pricing for different types of instance types. This means that if the RDS requires more capacity for reading/writing a customer is able to choose a more powerful instance type. The pricing of the different instances is between $0.021 - $12.48 [66]. The RDS also charges for the actual data storage which is priced at $0.115 per GB-month. The SQL database is also offered as a reserved instance where AWS offers two agreements, a 1-year, and a 3-year. The 1-year agreement offers 32% in savings while the 3-year offers 53%.

5.1.4.2 Microsoft Azure

Microsoft Azure offers both NoSQL and SQL database options. The NoSQL's most popular solution is the service called CosmosDB [67] which is a serverless on-demand pricing service. The SQL option [68] offers an on-demand pricing service. Because CosmosDB is a serverless solution the pricing is based on both the amount of read and writes made but also the amount of storage used. This is a bit different from the SQL option which offers only one pricing category of an hourly rate for the database being active.

The CosmosDB option as discussed is Microsoft Azure's own solution of a NoSQL database which uses a primary key sort key solution. Using this type of solution requires an extremely small amount of computing power to retrieve items (if done correctly) from the database. This means when a read or write operation is performed the database capacity is divided into units [69]. The pricing of units can be seen in table 5.21. The second pricing category of CosmosDB is the storage amount which can be seen in table 5.22.

Table 5.21: Azure Cosmos Write/Read Pricing

Account Type	Total RU/s per hour	Price per 100 RU/s
Single-region write account	100 RU/s x 1.5 x 1 region	$0.008/hour
Single-region write account with data distributed across multiple regions	100 RU/s x 1.5 x N regions	$0.008/hour
Multi-region write account distributed across multiple regions	100 RU/s x N regions	$0.016/hour

Table 5.22: Azure Cosmos Storage Pricing

Consumed Storage	Total GB	Price
Transactional (row-oriented)	1 GB x N regions	$0.25/month
Analytical (column-oriented)	1 GB x N regions	$0.03/month

The relational databases offer on-demand pricing for different types of instance types. This means that if the RDS requires more capacity for reading/writing a customer is able to choose a more powerful instance type. The pricing for the standard instance type is $0.5218 per vCPU per hour [70] which can be seen in table 5.23. The service also charges for the storage amount which is priced at $0.12 per GB-month. The SQL database is also offered as a reserved instance where Azure offers two agreements, a 1 year and a 3 year. The 1-year agreement offers 21% in savings while the 3-year offers 33%.

Table 5.23: Azure RDS Basic Pricing Layout

Minimum vCores	Maximum vCores	Minimum Memory (GB)	Maximum Memory (GB)	Price
0.5	80	2.02	240	$0.5218/vCore-hour

5.1.4.3 Google GCP

GCP offers both NoSQL and SQL database options. The NoSQL's most popular solution is the service called Cloud Firestore [71] which is a serverless on-demand pricing service. The SQL option called Cloud SQL [72] offers

an on-demand pricing service. Because Firestore is a serverless solution the pricing is based on both the amount of read and writes made but also the amount of storage used. This is a bit different from the SQL option which offers only one pricing category of an hourly rate for the database being active.

The Firestore option as discussed is GCP's own solution for a NoSQL database [73]. The service does provide a free quota every single day and all the operations performed beyond are priced at their rates. The pricing is set as per the document read/write/stored which can be seen in table 5.24.

Table 5.24: GCP Firebase Write/Read Pricing

Service	Free quota per day	Price beyond the free quota (per unit)	Price unit
Document Reads	50,000	$0.0345	per 100,000 documents
Document Writes	20,000	$0.1042	per 100,000 documents
Document Deletes	20,000	$0.0115	per 100,000 documents
Stored Data	1 GiB storage	$0.1725	GiB/Month

The relational databases offer on-demand pricing for different types of instance types. This means that if the RDS requires more capacity for reading/writing a customer is able to choose a more powerful instance type. The service charges $0.108 per GB-month in terms of data storage. The SQL database is also offered as a reserved instance where GCP offers two agreements, a 1-year, and a 3-year [74]. All the pricing can be seen in the table 5.25.

Table 5.25: GCP RDS Pricing

Resource	1-year commitment (USD)	3-year commitment (USD)	Price per Unit
vCPUs	$0.0479	$0.03592	Per vCPU
Memory	$0.0081	$0.00607	Per GB

5.2 Evaluation of Deployment Efforts

The evaluation of deployment efforts for hosting a static web application, serverless FaaS product serving as a REST API, authentication, and data storage on a SQL Server database across various cloud providers is based on the following data.

- Is there an automatic distribution setup?
- Is there any extra resource or services needed?

- How much documentation is needed?
- Is the resource or service well-documented?
- What is the deployment efforts rating, on a scale of 0-100?

5.2.1 Web application

The first field of evaluation of deployment efforts was made on hosting a static web application for the ticket system. A static web application service does only need to support HTML, CSS, and JavaScript. That is why some Cloud providers have storage that users can access through HTTP as a solution instead of services specifically made for static web applications. Sometimes these storage solutions do not support Hypertext Transfer Protocol Secure (HTTPS) directly and therefore require an additional step to secure the traffic.

5.2.1.1 AWS - S3

AWS S3 did not provide users with an automated distribution setup. AWS S3 does not have any tools for uploading files via GitHub actions and users need to use third-party actions, for example 'Upload S3' [38]. Having reliable public tools for automated distribution would lower the deployment efforts. By default, AWS S3 does only support HTTP and not HTTPS. HTTPS is very important for security, especially for the ticket system. To get HTTPS for AWS S3, developers need to implement Amazon CloudFront, which adds more deployment efforts and increases the cost of hosting a static web application. The amount of documentation needed was slightly higher due to the implementation of CloudFront for HTTPS and Amazon provides an 1830-page user guide document [75].

5.2.1.2 Azure - Static Web App

Azure's Static Web App resource did provide users with automated distribution setups for repositories on GitHub and Azure DevOps. There were no additional services needed for the automated distribution and securing of traffic with HTTPS. Since the setup for the resource was very thought out there was no documentation needed in order to deploy the web application. There are also quickstart guides for popular technologies, such as React, available depending on which technology the web app is built with [76]. Azure provides 66 documents with how-to guides, tutorials, and samples with many

tutorials for specific technologies and integrating the web application with Azure Functions [77].

5.2.1.3 GCP - Cloud Storage

GCP Storage had no automated distribution setups. However, GCP Storage does support automated distribution with repositories on GitHub and Cloud Source Repositories. There are official GitHub actions from Google available for uploading files [78]. GCP Storage serves multiple purposes besides hosting static web applications. Therefore, there are 121 user guides in the documentation to help developers with any use-case [79]. GCP Storage does not enable HTTPS connections by default and the developer must implement an external HTTPS load balancer for securing connections [80].

5.2.2 API

All of the implementations of serverless APIs on respective cloud platform was very similar and therefore, the results of deployment efforts did not vary that much. Evaluating the deployment efforts of the serverless API came down to comparing how much documentation was needed and available, and if there were any automatic distribution setups available.

5.2.2.1 AWS - Lambda

AWS Lambda is a serverless computing service provided by AWS. It simplifies the deployment of code and allows developers to run their applications or functions without provisioning or managing servers. AWS Lambda provides a web-based console called the AWS Management Console. This console allows its customers to create, configure, and manage their Lambda functions.

AWS Lambda also integrates with AWS CodePipeline, a fully managed continuous delivery service. With CodePipeline, developers can automate the build, test, and deployment process of their Lambda functions.

Furthermore, AWS Lambda supports integration with popular development and deployment tools such as AWS CloudFormation. CloudFormation is a service that enables infrastructure as code. This tool provides a template and configuration to define and deploy Lambda functions along with their associated resources, reducing deployment effort and ensuring consistent deployments. But an AWS lambda can simply be created and deployed by running the command in the listing 5.1.

Listing 5.1: AWS Lambda CLI command for deployment of a serverless function

```
aws lambda create-function --function-name my-function \
--zip-file function.zip --handler index.handler
        --runtime python3.8 \
--role arn:aws:iam::123456789012:role/lambda-ex
```

5.2.2.2 Azure - Azure Functions

Azure Functions has many functionalities to lower deployment efforts. For example, adding and writing functions can be done directly in the Azure Portal and provides the developer with a real-time console for debugging, and testing of functions can be done in the same view. This testing feature completely eliminates the need for third-party API-testing tools such as Postman.

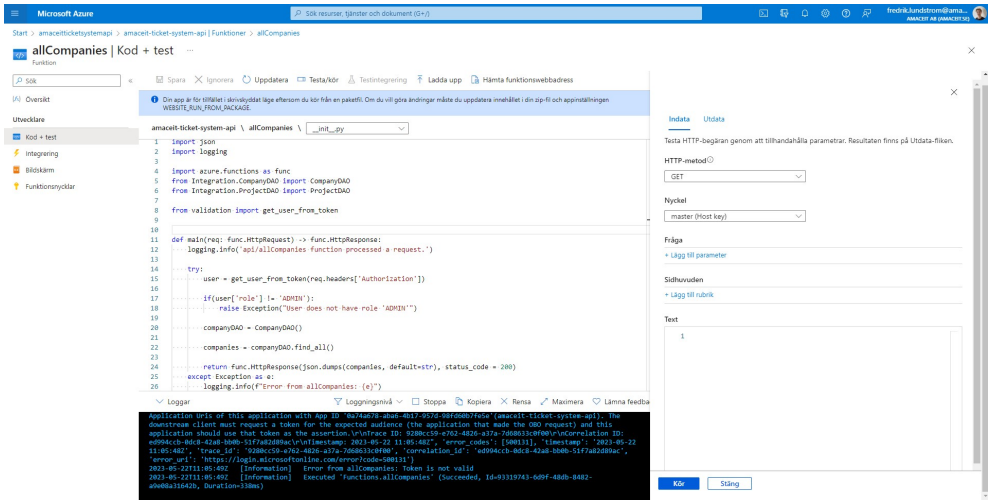

Figure 5.2: View, editing, testing, and debugging of functions can be done directly in the Azure Portal.

Azure Functions provides developers with an automatic distribution setup to ease the workload. This could be configured in the 'Distribution Center' and worked with GitHub and Azure DevOps. Azure Functions could also be distributed with SSL File Transfer Protocol (FTPS). Azure Functions also facilitated the CORS header option under a tab called 'CORS' where developers could add allowed origins. There was a lot of documentation required to set up an environment locally and several errors occurred for

GitHub actions. Microsoft provides 225 user guides on different programming languages, migration, and integration for Azure Functions [81]. Deploying new functions to Azure Functions can be done in the web browser, Azure Functions extension for Visual Studio Code, or as a command in Azure Functions Core Tools.

Listing 5.2: Azure Functions Core Tools command for deploying a new HTTP endpoint function.

```
func new --template "Http Trigger" --name <FUNCTION_NAME>
```

5.2.2.3 GCP - Cloud Functions

GCP Functions provided the authors with an automatic distribution setup. The supported version control hosting services for the setup was GitHub and Cloud Source Repositories. Functions can be edited and executed directly in the web browser which is a good option if developers do not need to use any version control tool like git. Deploying new functions to Cloud Functions can be done with a Cloud Shell command.

Listing 5.3: Cloud Shell command for deploying a new HTTP endpoint function.

```
gcloud functions deploy <FUNCTION_NAME> --trigger-http
```

5.2.3 Authentication

The evaluation of authentication services provided by the respective cloud providers was done with having in mind that the information about the *amaceit AB* organization and its user was kept in a Microsoft Active Directory. This means that both AWS's and GCP's authentication services have to federate the Active Directory from Azure and therefore implementing authentication for AWS and GCP requires an additional step. This step will not be counted towards the deployment efforts. Other than the additional step for AWS and GCP there are many similarities for all of the respective cloud provider's user authentication services.

5.2.3.1 AWS - Amazon Cognito

AWS provided user authentication with Amazon Cognito. Connecting Amazon Cognito to AWS Lambda was easy. The authors could specify for

each route if it was supposed to be secured, choose authorization from the Amazon Cognito resource, and what scope. No user pools had to be created for Amazon Cognito since the pools were federated from the *amaceit AB* Active Directory.

5.2.3.2 Azure - App Registrations

Azure provides user authentication with App Registrations. App Registrations were more complex to implement than with Amazon Cognito since Azure Functions does not have the ability to secure routes directly and there was much more coding that was needed. App Registrations does although provide developers with a Quick Start guide that creates ready-to-use applications and that uses authentication with that specific App Registration. Quick Start supports creating applications for web applications, mobile and desktop applications, single-page applications, and daemon applications.

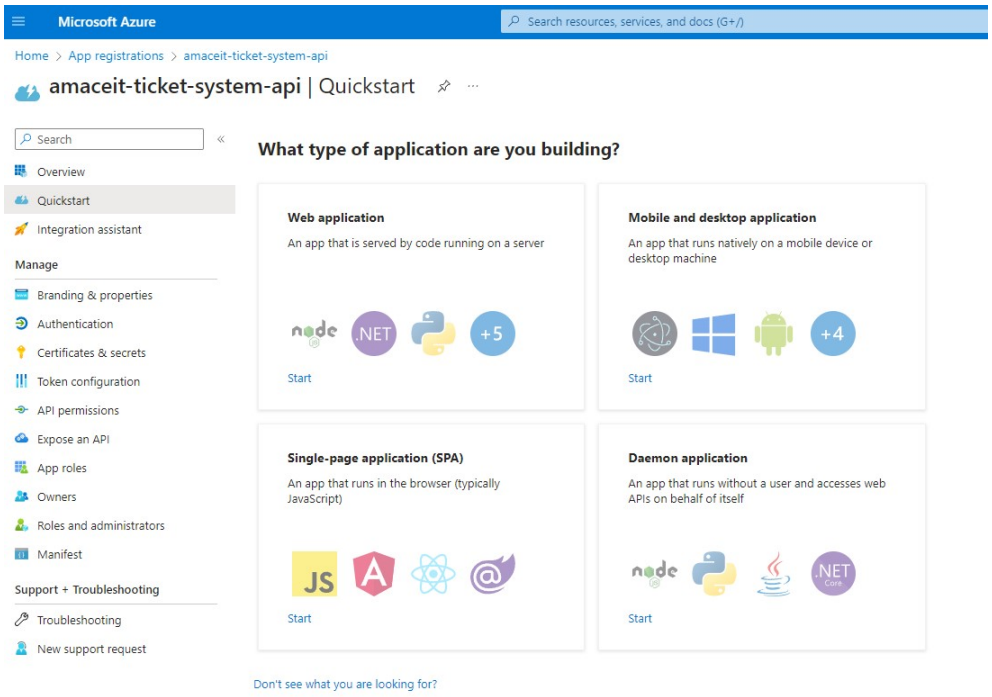

Figure 5.3: Different Quick Start options for App Registrations.

5.2.3.3 GCP - Cloud Identity

GCP provides user authentication with Cloud Identity that can federate AD. This service was very different from AWS Cognito and Azure App

Registrations and the Cloud Functions app had to be configured a lot more by specifying members and roles in the code of the application. There was not much documentation done on how to implement Cloud Identity with Cloud Functions which made the deployment efforts much higher in order to get user authentication working.

5.2.4 Database

All of the respective databases, i.e. Amazon RDS for SQL Server, Google Cloud SQL for SQL Server, and Azure SQL Database, had almost no deployment efforts at all. The reason for not having much deployment effort is due to all cloud providers providing a step-by-step solution for setting it up. Because the same database solution was used across the different providers the same setup/structure could be used. Google Cloud SQL for SQL Server ended up with some more deployment efforts made since Google Cloud SQL does not have any built-in Query Editor.

5.3 Case Study: Ticket System

The ticket System serves as a practical example to evaluate and compare different cloud services in terms of deployment and pricing. The development of the ticket system requires a variety of cloud services such as database hosting, file storage, API endpoint, and computing power such as serverless computing.

By implementing the ticket system on various cloud platforms, the case study aims to analyze the effort and pricing associated with each provider's services. The selection of services is crucial in order to ensure a fair comparison. Similar services from different cloud providers are chosen to obtain the best possible analysis.

5.3.1 Theoretical Workload

In the chapter, the case study will be tested against a theoretical workload to estimate the cost of each different provider. The workload will be divided into the services of serverless computing, data storage, database, and authentication service. The workload that will be used for estimating the cost of different services can be summarized:

1. 1000 active daily users

2. Average application use time: 30 minutes
    (a) Performs 250 API calls
    (b) Performs 1000 database calls
3. Each user generates 36KB/per day of database data
4. Each user generates 10MB/per day of file storage
5. Each user access 50MB/per day of file data

The application will be estimated to have already one year's worth of data stored by users and the accumulating of new data will not be counted. It is important to note that the average lambda run time that is used is 300ms. This means that the total application of cloud usage over a year will be as follows:

1. Data Stored
    (a) Database Storage (year): $36KB365 * 1000 = 13.14GB$
    (b) File Storage (year): $10MB * 365 * 1000 = 365TB$
    (c) File Storage Access (year): $50MB(5files)$*365*1000 $= 1825TB$
2. Application Usage
    (a) API calls (year): $250 * 365 * 1000 = 91,250,000calls/year$
    (b) Serverless run time (year): $91,250,000 * 0.3s = 27375000s/year$
    (c) Database calls (minute): $(1000 * 1000)/(8h * 60min) = 2083calls/min$

5.3.2 Serverless

The serverless part of the application that will be affected by the theoretical workload is the API calls and the runtime of the API calls.

5.3.2.1 Amazon AWS

For the estimated serverless calls and runtime, the pricing of the service for AWS will be their standard rate at $0.2 per million requests and $0.0000166667 per second of runtime. With the theoretical workload of 91,250,000 API calls per year and 27375000s of execution time pear year, the pricing comes down to:

Requests:

$$\$0.2 * 91.250000 \approx \$20$$

Runtime:

$$\$0.0000166667 * 27375000s \approx \$460$$

5.3.2.2 Microsoft Azure

For the estimated serverless calls and runtime, the pricing of the service for Microsoft Azure will be their standard rate at $0.2 per million requests and $0.000016 per second of runtime. With the theoretical workload of 91,250,000 API calls per year and 27375000s of execution time pear year, the pricing comes down to:

Requests:

$$\$0.2 * 91.250000 \approx \$20$$

Runtime:

$$\$0.000016 * 27375000s \approx \$440$$

5.3.2.3 Google GCP

For the estimated serverless calls and runtime, the pricing of the service for GCP will be their standard rate at $0.2 per million requests and $0.0000165 per second of runtime. With the theoretical workload of 91,250,000 API calls per year and 27375000s of execution time pear year, the pricing comes down to:

Requests:

$$\$0.4 * 91.250000 \approx \$40$$

Runtime:

$$\$0.0000165 * 27375000s \approx \$450$$

5.3.2.4 Comparison

As seen by running the ticket system with the theoretical workload of 250 API calls for 1000 users per day over a year it can be seen that the pricing difference between the different cloud providers is minimal 5.4.

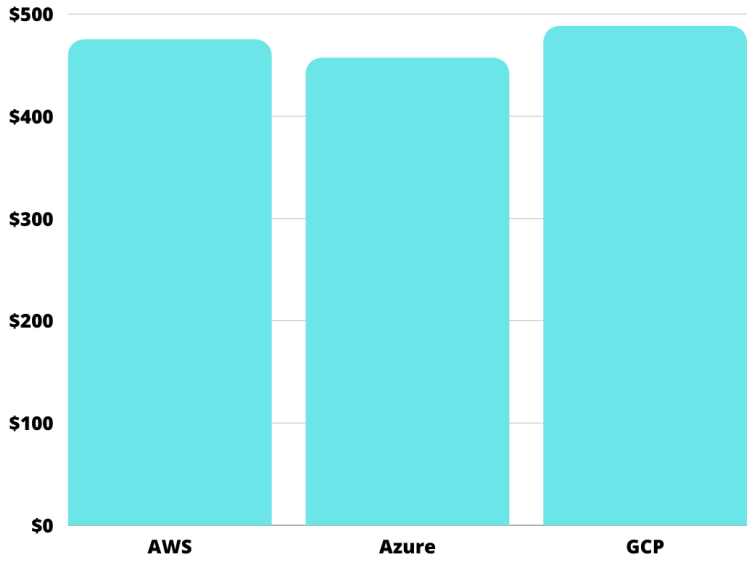

Figure 5.4: Case Study Serverless Comparison

5.3.3 File Storage

The file-storing part of the application that will be affected by the theoretical workload is both the generated and accessed data in terms of operations and the data stored over the period of the application. The formula for storage pricing is derived by: Price GB/month * 12 months * total storage. The access fee is derived from the formula: *Price Access GB * Accessed Data/year*.

5.3.3.1 Amazon AWS

The estimation of file storage involves two different categories, existing data, and accessing/uploading data. The files will be using the storage category of S3 Standard because of the need for fast access. The S3 standard is priced at $0.026 per GB. Uploading data to file storage is free but the access of data is priced at $0.05 per GB.

Storage Pricing:

$$\$0.026 * 12 * 365 * 1000 \approx \$114000$$

Storage Access:

$$\$0.05 * 1825 * 1000 \approx \$91000$$

$$\$0.0004 * 5 * 365 * 1000 \approx \$90$$

5.3.3.2 Microsoft Azure

The estimation of file storage involves two different categories, existing data, and accessing/uploading data. The files will be using the storage category of Hot because of the need for fast access. The Hot is priced at $0.018 per GB. Uploading data is free but accessing it is $0.0228 per 10000 reads and $0.01 per GB transferred.

Storage Pricing:

$$\$0.018 * 12 * 365 * 1000 \approx \$79000$$

Storage Access:

$$\$0.01 * 1825 * 1000 \approx \$18000$$

$$\$0.0228/10000 * 5 * 365 * 10000 \approx \$40$$

5.3.3.3 Google GCP

The estimation of file storage involves two different categories, existing data, and accessing/uploading data. The files will be using the storage category of Standard because of the need for fast access. The Standard is priced at $0.026 per GB. Uploading data is free but accessing it is $0.005 per 1000 reads and $0.01 per GB transferred.

Storage Pricing:

$$\$0.026 * 12 * 365 * 1000 \approx \$114000$$

Storage Access:

$$\$0.01 * 1825 * 1000 \approx \$18000$$

$$\$0.005/1000 * 5 * 365 * 1000 \approx \$10$$

5.3.3.4 Comparison

As seen by running the ticket system with the theoretical workload over a year it can be seen that the pricing difference between the different providers is

quite big. This has to do with both the cost of storing the data over the period and most importantly the difference in accessing data. It can be seen that AWS charges a lot more to transfer data out of their service 5.5.

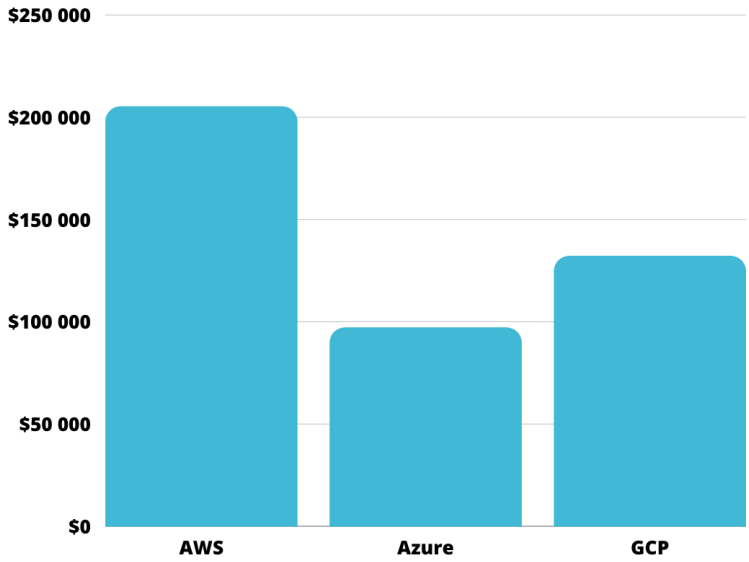

Figure 5.5: Case Study File Storage Comparison

5.3.4 Database

The database part of the application that will be affected by the theoretical workload is the operations performed on the database and the data stored. Because the application doesn't require a powerful instance to perform the operations one of the cheaper options can be used. The formula for database storage pricing is derived by: price per GB/month * 12 months * storage in GB. The formula for database instance runtime pricing is derived by: price per hour * 24 hours * 365 days.

5.3.4.1 Amazon AWS

For the estimation of the database, the service involves the runtime of the database server and the actual storage of the data. The storage of data is priced at $0.115 per GB-month and the server runtime pricing is $0.342 per hour (db.m5.xlarge).

Database Storage:

$$\$0.115 * 12 * 13.14GB \approx \$20$$

Database Instance runtime:

$$\$0.342 * 24 * 365 \approx \$3000$$

5.3.4.2 Microsoft Azure

For the estimation of the database, the service involves the runtime of the database server and the actual storage of the data. The storage of data is priced at $0.12 per GB-month and the server runtime pricing is $0.51 per hour (2 vCPU, 10.2GB memory).

Database Storage:

$$\$0.12 * 12 * 13.14GB \approx \$20$$

Database Instance runtime:

$$\$0.51 * 24 * 365 \approx \$4500$$

5.3.4.3 Google GCP

For the estimation of the database, the service involves the runtime of the database server and the actual storage of the data. The storage of data is priced at $0.108 per GB-month and the server runtime pricing is $0.1832 per hour (2 vCPU, 10GB memory).

Database Storage:

$$\$0.108 * 12 * 13.14GB \approx \$20$$

Database Instance runtime:

$$\$0.1832 * 24 * 365 \approx \$1500$$

5.3.4.4 Comparison

As seen by running the ticket system with the theoretical workload over a year it can be seen that the pricing difference between the different providers in terms of database costs is quite big as well. In figure 5.6 it can be seen that Azure ends up costing the most whereas GCP costs are not even half of Azures.

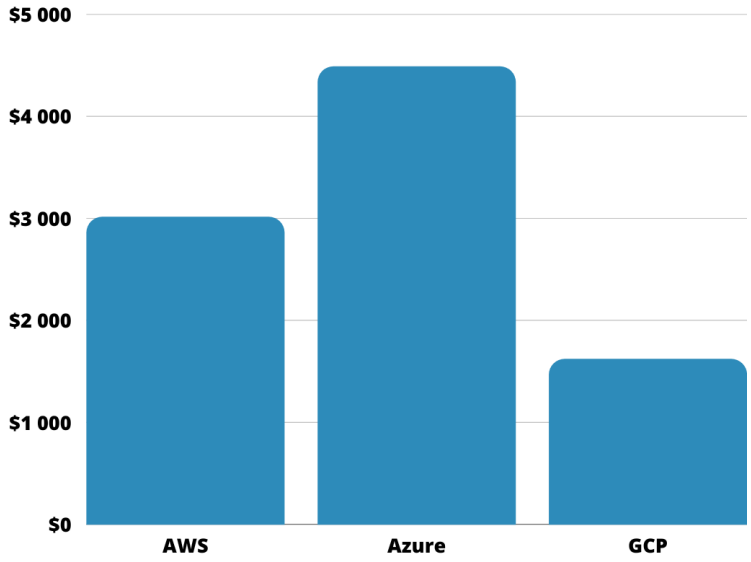

Figure 5.6: Case Study Database Comparison

5.4 Evaluation of Cloud Services

The analysis of various cloud providers yielded interesting findings that show distinct offerings for each platform. In terms of pricing, a detailed comparison revealed significant variations among providers, highlighting the importance of careful evaluation to ensure cost-effectiveness. By examining the pricing structures and models, businesses can make informed decisions that align their specific use case with the best provider to get the best services and pricing.

Another crucial aspect explored in the discussion is the deployment effort required for different cloud providers. This examination considered factors such as setup complexity, resource allocation, and integration processes. The findings revealed significant differences in the ease and efficiency of deploying different providers, showing the variety of levels of complexity associated with configuration and deployment processes. These variations underscore the challenges that organizations may encounter when setting up their cloud infrastructure with different providers.

5.4.1 Pricing Comparison of Different Providers

All of the different providers provide hundreds of different services to fit every single business's needs. The four most popular service categories of cloud computing that will be compared are instance computing, serverless

computing, data storage, and database services. These services are in many cases the least required services needed to deploy an application or product.

5.4.1.1 Instance Computing

Instance computing is one of the most popular options when choosing a cloud service. Instance computing allows an individual, business, or organization to run an application or product on a cloud provider with ease. The three most popular providers provide different instance categories to fit specific use cases. The pricing of these instances is nearly the same across the different providers. But it is important to note that the dedicated pricing plans Google GCP was significantly cheaper than the other providers providing a discount of around 60% 5.1.1.3. In contrast, AWS offers a 37% discount 5.1.1.1 and Microsoft Azure offers a 31% discount 5.1.1.2.

A few of these different families provided by the different cloud providers are general-purpose, compute-optimized, memory-optimized, and accelerator optimized. Each family offers different combinations of CPU, memory, and storage, allowing users to choose the configuration that best aligns with their specific requirements. By providing families with various configurations and optimizations, GCP, Microsoft Azure, and AWS enable their customers to select the most suitable instance type for their specific computing use case.

5.4.1.2 Serverless Computing

The general layout of serverless computing was nearly the same for the different providers. All providers with their basic plan charged a flat fee for every million requests where AWS and Microsoft Azure charged $0.2 and Google GCP charged $0.4. The providers then also charged a rate for memory usage per second. This means that a customer is able to control the amount of memory the serverless instance has access to.

A big difference between the different providers was their option to control the amount of vCPU. AWS has no option at all to control the amount of vCPU the instance has access to while Azure with their premium plan allows a customer to specify the amount of vCPU as well.

GCP offers only seven different configurations of vCPU with a defined amount of memory. In contrast, this is a bit different from AWS and Azure which allows the customer to modify the memory in smaller increments.

The pricing difference for the actual computation of the basic product comes down to that AWS charges $0.06 for every GB-hour, Azure charges

$0.0576 for every GB-hour, and lastly, GCP charges $0.008316 for every GB-hour. In this category, we can see that GCP charges the smallest amount per GB-hour of computation.

As discussed the different cloud providers do utilize other services than just their basic serverless computing service. Like AWS offers a service called Fargate which allows for more configuration to it like computing power. But because Fargate does not represent the basic serverless function structure because the initial load time of a Fargate is around 30-40 seconds [51] which makes it hard to use it as an API endpoint or as similar services.

5.4.1.3 Data Storage

Every single provider provides multiple options for storage. Their main service for storage then provides different categories for how often the data stored is accessed. The expensive storage category has a much faster retrieval time of the data and lowered access cost while the cheaper category has a slower retrieval time and increase in access cost.

Their fastest service which is in many cases the most used storage category is priced at $0.026 per GB-month for AWS, $0.018 per GB-month for Azure, and lastly $0.026 per GB-month for GCP. in terms of storage cost for their fast options is similar pricing but Azure is a bit cheaper than the other options.

The different service providers also charge based on operations and data retrieval. But a major difference between the providers is their fee for GB of access for their popular fastest option. GCP and Microsoft Azure charge a bandwidth of $0.01 per GB transferred out to the internet while Amazon AWS charges $0.09 - $0.05 per GB transferred. This can play a big part in the cost for a business to host their data.

5.4.1.4 Database

The provider's solution for a SQL database charges based on the data stored and the server computations time. The cloud providers have multiple options for different computation machines that could be used to fit a customer's needs. It is important to note that a more cost-effective solution in many cases is to use the cloud provider serverless option for databases. Using this type of service instead of paying a server that is always running a fee for reading/writing units is used instead.

AWS charges $0.021 per hour for their cheapest SQL server and a price of $0.115 per GB-month of storage. Microsoft Azure charges $0.5218 per hour for their standard SQL server and a price of $0.12 per GB-month of

storage. Lastly, Google GCP charges $0.0126 per hour for their cheapest SQL solution and a price of $0.108 per GB-month of storage. This shows that GCP's cheapest SQL solution is much cheaper than AWS and Azure.

5.4.2 General Deployment Effort of Different Providers

The overall score for deployment efforts shows that AWS had the lowest total deployment efforts when deploying the ticket system and GCP had the highest. GCP ended up with the highest score due to the high development efforts in implementing user authentication.

Table 5.26: Deployment Effort results for all services on respective cloud platforms. A lower score is better.

	Amazon AWS	**Google GCP**	**Microsoft Azure**
Hosting static web application	25	25	0
Serverless API	30	50	50
Authentication	10	50	25
Database	0	5	0
Total	65	130	75

5.4.3 Case Study: Pricing Differences

The case study has provided a real-world example that can be analyzed based on pricing differences between the three different cloud providers. The case study focuses on three key areas of cloud computing: serverless computing, data storage, and database usage. These areas represent fundamental components that organizations consider when selecting a cloud provider. Examining the pricing models and offerings of AWS, GCP, and Microsoft Azure in these areas will help to show the actual cost of running a service or application on the cloud. The case study focuses on the theoretical workload as mentioned in 5.3.1.

5.4.3.1 Serverless

As earlier discussed the cost of serverless computing among the different cloud providers is nearly the same. Running the serverless computing structure with the theoretical workload over a period of one year can be seen in table 5.27. It can be seen that the difference between the different providers is extremely small but Microsoft Azure comes down to be the cheapest option of them all.

Table 5.27: Case Study Serverless Computing Comparisons

	Amazon AWS	Google GCP	Microsoft Azure
Request	$20	$40	$20
Runtime	$460	$450	$440
Total	≈ $480	≈ $490	≈ $460

5.4.3.2 File Storage

The file storage as discussed will consist of the most difference between the providers. This is due to AWS charging an access fee for their fast storage option. Running the theoretical workload on each provider's fastest storage option it comes down to Microsoft Azure being the cheapest option by large margins which can be seen in table 5.28. Due to AWS charging a lot more for the bandwidth fee out to the internet, it becomes the expense option out of them all.

Table 5.28: Case Study Storage Comparisons

	Amazon AWS	Google GCP	Microsoft Azure
Storage Pricing	$114000	$114000	$79000
Storage Access Operation	$90	$40	$10
Bandwidth Fee	$91000	$18000	$18000
Total	≈ $205000	≈ $132000	≈ $97000

5.4.3.3 Database

The database storage solution for the different providers is charged based on the SQL server runtime and the storage cost. As discussed in 5.4.1.4 Microsoft Azure charged the most for their database instance. As seen in table 5.29 GCP comes down to be the cheapest solution by large margins.

Table 5.29: Case Study Database Comparisons

	Amazon AWS	Google GCP	Microsoft Azure
Database Storage	$20	$20	$20
Database Runtime	$3000	$1500	$4500
Total	≈ $3000	≈ $1500	≈ $4500

5.4.3.4 Total Costs: AWS vs. Azure vs. GCP

In the case study theoretical workload, it can be seen that the biggest pricing category is the storage in both terms of accessing and storing data. This means that because of this Microsoft Azure comes down to be the cheapest option out of the three different providers which can be seen in table 5.7. As mentioned AWS becomes the most expensive option out of the three options due to its access fees for transferring data OUT to the internet being five to nine times more expensive.

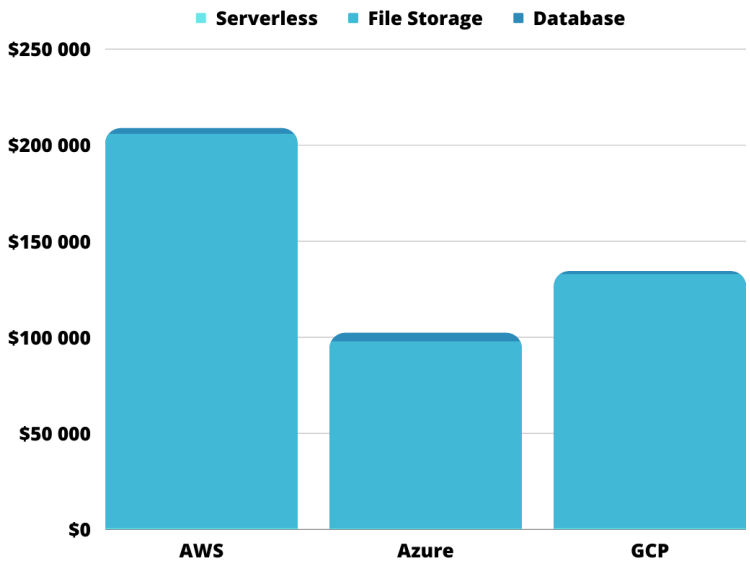

Figure 5.7: Case Study Theoretical Workload Total Pricing

Chapter 6

Conclusions and Future work

Cloud computing has risen over the past decades, revolutionizing the way businesses and individuals leverage computing resources. This chapter aims to provide an overview of aspects related to cloud computing, including its importance, pricing models offered by cloud providers, deployment efforts, and future implications.

Cloud computing has gained significant attention due to its numerous advantages and benefits. Organizations can now access computing resources on demand, enabling scalability and flexibility. The ability to scale infrastructure has become of critical importance for businesses across various industries.

Understanding cloud providers' pricing models is crucial for businesses seeking to optimize costs and make informed decisions. Choosing the correct cloud provider becomes an extremely important discussion due to the lock-in model that each provider has. By comparing pricing structures, storage options, and data transfer costs, organizations can effectively manage their expenses in the cloud.

Deploying applications and services in the cloud requires planning and effort. Different cloud platforms offer varying deployment processes and tools. By understanding the deployment efforts required for various cloud providers, businesses can make decisions about the best possible provider for their needs. Examining the deployment processes, ease of use, and automation tools available results in providing insights into achieving efficient deployment lines cloud deployments.

6.1 Summary of Findings

Cloud computing providers offer numerous services to fit every industry business's needs. When comparing the cloud providers there might not be any big differences at the surface but when you compare the services, deployment effort, and pricing against each other there are a few differences that can make an incredibly big difference.

6.1.1 Pricing

All the cloud providers offer the same pricing models which are on-demand, subscription-based, spot-pricing, and reserved instances. There are quite a few differences in which services are offered in the different services where for example GCP and Microsoft Azure offered more pricing models for their serverless solutions while AWS only offered one option. The pricing disparity between these services was not significant for specific services like serverless computing. While certain services there was quite a huge pricing difference for example with the transfer rate with storage.

6.1.2 Deployment Effort

The deployment effort for the different services varied significantly, with some requiring considerably more time, resources, and complexity compared to others. Certain services had a lot more configuration steps, extensive customization, and integration with existing systems, leading to a substantial deployment effort. On the other hand, certain services may have offered streamlined deployment processes, automated provisioning, and simplified management interfaces, resulting in a comparatively lower deployment effort. But an interesting thing was that the process of deploying for the different services didn't have a big difference in the fundamentals, but had a huge difference in the required configurations. Meaning that even if a developer has the knowledge of deploying for one cloud provider, they still require a lot of knowledge to deploy to another provider.

6.1.3 Other Findings

There was also a big difference in the amount of services that the different providers provided. Microsoft Azure was the provider that offered the most amount of services with a total of a bit more than 300 while AWS had a bit

more than 200 and lastly, GCP had a bit more than 100 services 5.1. This can play a huge part when deciding which cloud provider to choose. The type of service that the different cloud providers provide plays a big part when a business needs to improve and extend its application or service. Especially this becomes hard with the lock-in factor, which in the future can be a big problem if the cloud provider that is being used is missing an important cloud service that is needed.

It is also important to note that the existing documentation of the providers' varied a lot. A provider like AWS offered a lot of specific case examples for different scenarios, which made the development of the case study really easy. In contrast, GCP provided in many cases just one example or just general documentation for the different services, which then required the usage of external documentation and help.

6.2 Importance of Cloud Computing

Cloud computing has become a revolutionary technology that has transformed the way businesses operate and leverage their IT infrastructure. With its flexible and scalable structure, cloud computing has become a vital component of modern-day enterprises, enabling them to enhance productivity, streamline operations, and achieve significant cost savings. This section explores the importance of cloud computing and highlights its numerous benefits for businesses across various industries.

6.2.1 The Crucial Role of Cloud Computing

Cloud computing plays a crucial role in today's technological landscape, offering services to individuals, businesses, and organizations. One of its most significant contributions lies in providing cost-effective alternatives and scalable solutions with flexibility. Moreover, cloud providers provide accessibility and collaboration, leading to improved efficiency in service and application development. With its benefits, cloud computing has become a vital resource in driving innovation and meeting the evolving needs of modern enterprises.

6.2.1.1 Cost savings

Cloud computing offers cost-effective solutions for businesses by eliminating the need for large early investments in hardware and infrastructure. It allows

businesses to pay for the resources they use, making it a more affordable option compared to traditional private cloud systems.

6.2.1.2 Scalability and flexibility

Cloud computing provides businesses with the ability to scale their resources up or down based on demand (on demand). This scalability ensures that businesses can use resources based on their specific need. Especially with private cloud it in many cases becomes hard to handle peak traffic wherever the usage of cloud services the service can scale based on the need.

6.2.1.3 Accessibility and collaboration

Cloud computing allows developers to access their applications and data from anywhere, at any time. This means that it is possible for developers to collaborate on the same application even though they might be located in different parts of the globe. With cloud computing, there is no need to manually manage infrastructure, making it effortlessly accessible from anywhere, thereby streamlining development processes.

6.2.1.4 Improved efficiency

Cloud computing streamlines IT processes and reduces the time and effort required for infrastructure management. This has made it possible for businesses and organizations to develop and release products, services, and applications in a much faster way. By also using cloud services and utilizing a SaaS 2.1.1.3 structure, the users of the products can seamlessly get updates.

6.2.1.5 Enhanced data security

Cloud service providers invest heavily in security measures and protocols to protect their customers' data. They employ advanced encryption techniques, regular backups, and robust access controls, ensuring that data is secure and protected from unauthorized access or loss. This is an essential part, especially when managing your own infrastructure, there could be bigger security risks of unknown bugs and issues. By using a cloud provider, there will be a big company behind it that invest in both money and manpower to make sure that the security of their products is good.

6.2.1.6 Business Continuity and disaster recovery

Cloud computing offers built-in redundancy and backup systems, which improves business continuity and disaster recovery capabilities. In the event of a hardware failure or natural disaster, businesses can quickly restore their data and applications from backups stored in the cloud. Because of the help of multiple facilities located all around the globe, if a location's infrastructure would fail, a customer would be able to switch to another facility's location.

6.2.2 Evolving Landscape of Cloud Computing

The evolving landscape of cloud computing reflects the nature of technology and its continuous evolution to meet the increase in demands. As cloud computing gains popularity and becomes increasingly integrated into various sectors, it becomes crucial to understand every single part that the different popular cloud providers provide.

6.2.2.1 Optimization of pricing models

Further investigation can be done to analyze and optimize the pricing models offered by cloud service providers. This research can focus on identifying pricing strategies that align with different business scenarios and offer the best value for money. For a business when integrating cloud services into their products, it is extremely important to do an in-depth analysis of the application usage to understand what services and pricing models should be used. Doing this will result in the best cost-optimal solution that still can fulfill the requirement of scalability.

6.2.2.2 Security and privacy advancements

As cloud computing continues to evolve, there is a constant need for improving security measures and addressing privacy concerns. It is important to understand what services and solutions the cloud providers provide in terms of security and privacy. Even though a cloud provider offers the industry's best security and privacy, does that mean a company's crucial and private data can be trusted in the hands of another company?

6.2.2.3 Hybrid cloud and multi-cloud strategies

The adoption of hybrid cloud and multi-cloud architectures is increasing among businesses. Further research can be conducted to explore the

challenges, benefits, and best practices for implementing and managing hybrid and multi-cloud environments effectively. This can especially play an important aspect for businesses that require service from multiple different cloud providers. While a business might favor a specific cloud provider, it might be missing out on other cloud providers' services. Utilizing multiple cloud providers could in some aspects provide cost-effectiveness by selecting the cheapest service from all available providers. But using a multi-cloud infrastructure will increase the complexity of a system, but could it still be worth it?

6.3 Private vs Public Cloud

Despite the rapid development of public cloud services, private cloud still plays an important role, and it will probably continue to be like that for a while. Choosing between private and public clouds comes down to specific needs since they are good at different things such as security, cost, customization, performance, availability, etc.

6.3.1 Cost, Performance & Availability

A big selling point for public cloud providers is having the best-performing product with as little downtime as possible. For example, despite its name, a serverless application is distributed across many servers to ensure the best performance and availability possible. Achieving performance with a private cloud could be very costly since performance requires suitable hardware and bandwidth, and availability requires that somehow the server has as little downtime as possible. This could be done by, for example, building the backend with microservices and distributing the microservices across multiple servers.

6.3.2 Use Case Considerations

Considering the use case, i.e. the ticket system, choosing the usage of public cloud was the most suitable approach. There was no need for the customizability of private cloud since the ticket system did not need any customer service. Public cloud solved the problem of maintaining a complicated API with unnecessary amounts of code. Instead, the serverless API approach was used, and the authors were able to develop the application much faster, which lowers the cost of developing and acquiring, and

maintaining a server. *amaceit AB* also had an AD at Microsoft, which could be used to authenticate users to the ticket system. If the ticket system had been developed purely for private cloud, *amaceit AB* would have had to build its own authentication service, which could not only be very costly but could also add security risks. The data for the ticket system was not of the sensible kind that it had to be stored on a private network database server, and could, by all means, be stored on a public cloud database service. The ticket system could although be implemented with hybrid cloud and using public cloud services for the authentication and having their own private server for hosting the API.

6.4 Cloud Providers Pricing

Cloud provider pricing is an important factor for organizations considering using cloud services. Cloud providers offer various pricing models to fit different businesses' requirements. A few of these pricing models include on-demand pricing, reserved instances, spot instances, and dedicated plans. Each pricing model has its advantages and considerations, such as flexibility, cost predictability, and trade-offs between upfront payment and usage.

While cloud providers offer a range of services, such as compute instances, storage, and databases, the pricing for these services often shows minimal variation among the different providers. As discussed in 5.1 it can be seen that for example in serverless computing the price was pretty much the same across the providers. But certain services like storage varied a lot for the different providers where Azure ended up being the cheapest. This is why it is important for a business to compare the providers against each other to find the best one for the specific use case. The overall competitive landscape drives providers to offer similar pricing structures to attract customers. However, there may be slight variations in pricing for specific services or regions due to factors like infrastructure costs, market demand, and economies of scale.

It is essential to consider real-world scenarios and use cases when analyzing cloud provider pricing. In the specific case study that was made, AWS had the highest cost due to the expense of extracting files from the storage. This is because AWS charges a bandwidth fee for transferring data out to the internet of 5x to 9x the amount that Azure and GCP do. This implied that the service ended up costing around double the amount for the case study if it would have used AWS. This demonstrates that pricing comparisons should not be limited to headline rates but should consider the specific requirements and usage patterns of the applications or workloads under consideration.

Spot instances and reserved instances are pricing options offered by cloud

providers to provide additional cost savings for users. Spot pricing allows users to bid on unused compute capacity, enabling them to access resources at significantly reduced rates. When comparing the different services it is important to note that the spot pricing savings were different among the different providers where certain providers offered a lot more in savings. Reserved instances involve committing to a specific instance type and duration, offering discounted rates compared to on-demand instances. In this context, AWS offers up to 32% savings for reserved instances, while GCP and Microsoft Azure can provide up to 55% savings. Note that these discounts are not true across all services where in some cases AWS offers more discounts in their reserve pricing models than others. These varying discount levels make it essential for users to evaluate their workload characteristics and select the most cost-effective options for their specific use case.

6.5 Cloud Providers Deployment Effort

The deployment efforts for applications to AWS, Microsoft Azure, and GCP did vary to some degree. The deployment efforts were measured in if there was any automatic distribution setup provided when deploying services if there were any extra services or resources needed, how much documentation was needed, if the service was well-documented, and any extra comments from the authors. This study found out that AWS had the lowest deployment efforts needed for the ticket system case study. This was due to the well-documented serverless Lambda service, which was used for the API, and Amazon Cognito, which was used for authentication. Unlike Azure and GCP, connecting the user authentication for the API with Amazon Cognito reduced the total deployment effort score. Azure scored with the second-highest development efforts required to implement the ticket system. Both AWS and Azure fitted the needs of the ticket system very well. GCP scored with the highest total development efforts needed to implement the ticket system. The high score depended on the deployment efforts of hosting a static web application, which is stored on a storage service that does not directly support HTTPS and needs an extra service. The high score also depended on that Cloud Identity was hard to implement for user authentication, and it did not for example come with the sample application code that Azure provided. Cloud Identity did not have the easy connectivity that Azure had between App Registrations and Azure Functions, and AWS had between AWS Lambda and Amazon Cognito.

6.6 Future Implications

The trends of cloud computing suggest that companies will utilize more hybrid and multi-cloud computing in the future. It can be companies that are currently using private cloud solutions and are moving towards better performing, more available, and cost-effective solutions like serverless computing. There are also new emerging technologies emerging like edge computing and AI that will benefit public cloud even more. With the proliferation of connected devices and the need for real-time data processing, edge computing will become a critical component of cloud computing. Edge computing enables data processing and storage closer to the source, reducing latency and improving responsiveness. By pushing computation to the edge of the network, cloud services can be extended to remote locations and resource-constrained devices, enabling applications like autonomous vehicles, smart cities, and industrial automation.

Cloud computing will also become increasingly intertwined with AI technologies. Cloud platforms will provide powerful AI capabilities, including machine learning, natural language processing, and computer vision, as ready-to-use services. This integration will democratize AI, enabling businesses of all sizes to leverage advanced analytics, predictive models, and automation tools, resulting in smarter decision-making, improved customer experiences, and enhanced operational efficiency.

The future of cloud computing will emphasize sustainability and energy efficiency. Cloud providers will invest in renewable energy sources, optimize data center cooling systems, and implement energy-efficient hardware to reduce their carbon footprint. Additionally, advanced resource allocation and workload management techniques will be employed to maximize resource utilization and minimize energy consumption, resulting in greener and more environmentally friendly cloud infrastructures.

6.7 Future Work

The thesis has shown various aspects of cloud computing, revealing valuable insights into pricing models and deployment strategies. However, there are still several categories for further exploration and improvement within the field of research. This section presents an overview of the future work that can be done to extend the knowledge and address the limitations of the current research in this thesis.

Future work in cloud computing holds immense potential for enhancing the cost-efficiency, scalability, security, and overall performance of cloud-based solutions. By focusing on the following areas of investigation, researchers can contribute to the advancement and evolution of cloud computing technologies as we help companies find the correct solution based on their specific needs.

1. **Explore additional cloud computing service providers:** Investigate and compare pricing models and deployment efforts this thesis has only covered the three major cloud providers AWS, Microsoft Azure, and GCP. There is a huge market for cloud computing and even the smaller cloud providers can provide services and pricing competition against the bigger ones. This means especially for smaller to medium-sized companies that there might be better providers out there to fit their specific needs.

2. **Evaluate the performance and cost-effectiveness of hybrid cloud solutions:** Analyze the practice and benefits of adopting hybrid cloud architectures that combine private and public cloud services. This type of structure can really help companies that value their data and in the current situation, can't have another company accessing it. A hybrid cloud solution is also a possible solution for smaller companies due to the minimized IT infrastructure startup expenses.

3. **Assess the security and privacy implications of cloud computing:** Investigate the potential risks and vulnerabilities associated with cloud computing services and propose strategies to mitigate security threats and protect sensitive data. This is absolutely something that further needs to be investigated so that companies can get a full picture of the risks of having private data hosted on a cloud provider's services.

4. **Investigate the challenges of cloud migration and vendor lock-in:** Analyze the difficulties and risks involved in migrating existing applications and data to cloud platforms, and propose strategies to minimize vendor lock-in and ensure seamless transitions. This plays a big importance for companies that in an early stage start committing to a cloud provider. In many cases, this could result in unnecessary costs and missed services from other providers.

5. **Fault Tolerance and Disaster Recovery:** Compare different cloud providers against private computing in terms of fault tolerance and disaster recovery. Especially comparing different providers could give a

business insight for the specific provider to choose. This can play a big part when choosing a less popular cloud provider instead of one of the more popular ones.

References

[1] M. Corporation, "Form 10-Q for the quarterly period ended December 31, 2022," Dec. 2022. [Online]. Available: https://www.sec.gov/Archives/edgar/data/789019/000156459023000733/msft-10q_20221231.htm [Page 1.]

[2] "Computer operating systems market share 2012-2023." [Online]. Available: https://www.statista.com/statistics/268237/global-market-share-held-by-operating-systems-since-2009/ [Page 1.]

[3] S. Zhang, S. Zhang, X. Chen, and X. Huo, "Cloud computing research and development trend," in *2010 Second International Conference on Future Networks*, 2010. doi: 10.1109/ICFN.2010.58 pp. 93–97. [Page 5.]

[4] S. Goyal, "Public vs private vs hybrid vs community-cloud computing: a critical review," *International Journal of Computer Network and Information Security*, vol. 6, no. 3, pp. 20–29, 2014. [Pages 5 and 12.]

[5] A. Aljabre, "Cloud computing for increased business value," *International Journal of Business and social science*, vol. 3, no. 1, 2012. [Page 5.]

[6] S. Lehrig, H. Eikerling, and S. Becker, "Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics," in *Proceedings of the 11th international ACM SIGSOFT conference on quality of software architectures*, 2015, pp. 83–92. [Page 6.]

[7] H. Saini, A. Upadhyaya, and M. K. Khandelwal, "Benefits of cloud computing for business enterprises: A review," in *Proceedings of International Conference on Advancements in Computing & Management (ICACM)*, 2019. [Page 6.]

[8] S. Carlin and K. Curran, "Cloud computing security," in *Pervasive and Ubiquitous Technology Innovations for Ambient Intelligence Environments*. IGI Global, 2013, pp. 12–17. [Pages 6 and 9.]

[9] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, and K. Pentikousis, "Energy-efficient cloud computing," *The computer journal*, vol. 53, no. 7, pp. 1045–1051, 2010. [Page 6.]

[10] E. Bauer and R. Adams, *Reliability and availability of cloud computing*. John Wiley & Sons, 2012. [Page 7.]

[11] E. Gorelik, "Cloud computing models," Ph.D. dissertation, Massachusetts Institute of Technology, 2013. [Page 7.]

[12] S. Bhardwaj, L. Jain, and S. Jain, "Cloud computing: A study of infrastructure as a service (iaas)," *International Journal of engineering and information Technology*, vol. 2, no. 1, pp. 60–63, 2010. [Page 7.]

[13] C. Pahl, "Containerization and the paas cloud," *IEEE Cloud Computing*, vol. 2, no. 3, pp. 24–31, 2015. [Page 7.]

[14] S. Satyanarayana, "Cloud computing: Saas," *Computer Sciences and Telecommunications*, no. 4, pp. 76–79, 2012. [Page 7.]

[15] A. P. Rajan, "A review on serverless architectures-function as a service (faas) in cloud computing," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 18, no. 1, pp. 530–537, 2020. [Page 8.]

[16] A. J. Younge, G. von Laszewski, L. Wang, S. Lopez-Alarcon, and W. Carithers, "Efficient resource management for cloud computing environments," in *International Conference on Green Computing*, 2010. doi: 10.1109/GREENCOMP.2010.5598294 pp. 357–364. [Page 8.]

[17] J. Surbiryala and C. Rong, "Cloud computing: History and overview," in *2019 IEEE Cloud Summit*. IEEE, 2019, pp. 1–7. [Page 8.]

[18] "Cloud computing market size, share; trends report, 2030." [Online]. Available: https://www.grandviewresearch.com/industry-analysis/cloud-computing-industry [Page 9.]

[19] "Cloud computing market size, share; covid-19 impact analysis, by type (public cloud, private cloud, and hybrid cloud), by service (infrastructure

as a service (iaas), platform as a service (paas), and software as a service (saas)), by industry (bfsi, it and telecommunications, government, consumer goods and retail, healthcare, manufacturing, and others), and regional forecast, 2022-2029." [Online]. Available: https://www.fortunebusinessinsights.com/cloud-computing-market-102697 [Page 9.]

[20] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, and A. Ghalsasi, "Cloud computing—the business perspective," *Decision support systems*, vol. 51, no. 1, pp. 176–189, 2011. [Page 9.]

[21] B. Gupta, P. Mittal, and T. Mufti, "A review on amazon web service (aws), microsoft azure & google cloud platform (gcp) services," in *Proceedings of the 2nd International Conference on ICT for Digital, Smart, and Sustainable Development, ICIDSSD 2020, 27-28 February 2020, Jamia Hamdard, New Delhi, India*, 2021. [Page 9.]

[22] "Storing and processing data in europe: The free flow of non-personal data." [Online]. Available: https://europa.eu/youreurope/business/running-business/developing-business/free-flow-non-personal-data/index_en.htm [Page 9.]

[23] P. Samimi and A. Patel, "Review of pricing models for grid & cloud computing," in *2011 IEEE Symposium on Computers & Informatics*, 2011. doi: 10.1109/ISCI.2011.5958990 pp. 634–639. [Pages 10 and 11.]

[24] L. Savu, "Cloud computing: Deployment models, delivery models, risks and research challenges," in *2011 International Conference on Computer and Management (CAMAN)*. IEEE, 2011, pp. 1–4. [Pages 11 and 12.]

[25] J. Opara-Martins, R. Sahandi, and F. Tian, "Critical analysis of vendor lock-in and its impact on cloud computing migration: a business perspective," *Journal of Cloud Computing*, vol. 5, pp. 1–18, 2016. [Page 11.]

[26] J. Duan, P. Faker, A. Fesak, and T. Stuart, "Benefits and drawbacks of cloud-based versus traditional erp systems," *Proceedings of the 2012-13 course on Advanced Resource Planning*, 2013. [Page 12.]

[27] P.-F. Hsu, S. Ray, and Y.-Y. Li-Hsieh, "Examining cloud computing adoption intention, pricing mechanism, and deployment model," *International Journal of Information Management*, vol. 34, no. 4, 2014. [Page 12.]

[28] "Just-in-time video packaging – aws elemental mediapackage – amazon web ..." [Online]. Available: https://aws.amazon.com/mediapackage/ [Page 18.]

[29] Microsoft, "Microsoft excel spreadsheet software: Microsoft 365." [Online]. Available: https://www.microsoft.com/en-us/microsoft-365/excel [Page 20.]

[30] M. Zhang, "Top 10 Cloud Service Providers Globally in 2023," Jan. 2023. [Online]. Available: https://dgtlinfra.com/top-10-cloud-service-providers-2022/ [Page 29.]

[31] "About Figma, the collaborative interface design tool. — figma.com," https://www.figma.com/about/, [Accessed 23-Apr-2023]. [Page 30.]

[32] "React — react.dev," https://react.dev/, [Accessed 23-Apr-2023]. [Page 31.]

[33] "React Redux | React Redux — react-redux.js.org," https://react-redux.js.org/, [Accessed 23-Apr-2023]. [Page 31.]

[34] "Authentication Service - Customer IAM (CIAM) - Amazon Cognito - AWS — aws.amazon.com," https://aws.amazon.com/cognito/, [Accessed 23-Apr-2023]. [Page 33.]

[35] "Access Management - AWS Identity and Access Management (IAM) - AWS — aws.amazon.com," https://aws.amazon.com/iam/, [Accessed 23-Apr-2023]. [Page 33.]

[36] "Serverless Computing - AWS Lambda - Amazon Web Services — aws.amazon.com," https://aws.amazon.com/lambda/, [Accessed 23-Apr-2023]. [Pages 34 and 51.]

[37] "Cloud Object Storage – Amazon S3 – Amazon Web Services — aws.amazon.com," https://aws.amazon.com/s3/, [Accessed 24-Apr-2023]. [Pages 34 and 53.]

[38] "Upload S3 - GitHub Marketplace — github.com," https://github.com/marketplace/actions/upload-s3, [Accessed 24-Apr-2023]. [Pages 35 and 63.]

[39] "Amazon RDS Backup & Restore | Cloud Relational Database | Amazon Web Services." [Online]. Available: https://aws.amazon.com/rds/features/backup/ [Page 36.]

[40] “Create Your Azure Free Account Today | Microsoft Azure.” [Online]. Available: https://azure.microsoft.com/en-us/free/ [Page 36.]

[41] cephalin, “Konfigurera Azure AD-autentisering - Azure App Service,” Mar. 2023. [Online]. Available: https://learn.microsoft.com/sv-se/azure/app-service/configure-authentication-provider-aad [Page 37.]

[42] “Azure Static Web Apps – App Service | Microsoft Azure.” [Online]. Available: https://azure.microsoft.com/en-us/products/app-service/static [Page 39.]

[43] “Cloud Identity.” [Online]. Available: https://cloud.google.com/identity [Page 41.]

[44] “Cloud Functions.” [Online]. Available: https://cloud.google.com/functions [Page 42.]

[45] “About Cloud SQL backups | Cloud SQL for MySQL.” [Online]. Available: https://cloud.google.com/sql/docs/mysql/backup-recovery/backups [Page 43.]

[46] “Serverless: Develop & Monitor Apps On AWS Lambda — serverless.com,” https://www.serverless.com/, [Accessed 24-Apr-2023]. [Page 44.]

[47] “EC2 On-Demand Instance Pricing – Amazon Web Services — aws.amazon.com,” https://aws.amazon.com/ec2/pricing/on-demand/, [Accessed 29-Apr-2023]. [Page 48.]

[48] “Pricing - Windows Virtual Machines | Microsoft Azure — azure.microsoft.com,” https://azure.microsoft.com/en-us/pricing/details/virtual-machines/windows/, [Accessed 07-May-2023]. [Page 49.]

[49] “Compute Engine: Virtual Machines (VMs) | Google Cloud — cloud.google.com,” https://cloud.google.com/compute, [Accessed 07-May-2023]. [Page 50.]

[50] “Pricing | Compute Engine: Virtual Machines (VMs) | Google Cloud — cloud.google.com,” https://cloud.google.com/compute/all-pricing, [Accessed 07-May-2023]. [Page 50.]

[51] "Serverless Compute Engine–AWS Fargate–Amazon Web Services — aws.amazon.com," https://aws.amazon.com/fargate, [Accessed 29-Apr-2023]. [Pages 51 and 77.]

[52] "Serverless Computing – AWS Lambda Pricing – Amazon Web Services — aws.amazon.com," https://aws.amazon.com/lambda/pricing/, [Accessed 29-Apr-2023]. [Page 51.]

[53] "Serverless Compute Engine–AWS Fargate Pricing–Amazon Web Services — aws.amazon.com," https://aws.amazon.com/fargate/pricing, [Accessed 29-Apr-2023]. [Page 51.]

[54] "Azure Functions – Serverless Functions in Computing | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/products/functions/, [Accessed 07-May-2023]. [Page 52.]

[55] "Pricing - Functions | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/functions/, [Accessed 07-May-2023]. [Page 52.]

[56] "Cloud Functions | Google Cloud — cloud.google.com," https://cloud.google.com/functions, [Accessed 07-May-2023]. [Page 53.]

[57] "Pricing | Cloud Functions | Google Cloud — cloud.google.com," https://cloud.google.com/functions/pricing, [Accessed 07-May-2023]. [Page 53.]

[58] "Azure Storage Blobs Pricing | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/storage/blobs/, [Accessed 07-May-2023]. [Page 55.]

[59] "Azure Files - Managed File Shares and Storage | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/products/storage/files, [Accessed 07-May-2023]. [Page 55.]

[60] "Azure Storage Blobs Pricing | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/storage/blobs/#purchase-options, [Accessed 07-May-2023]. [Page 55.]

[61] "Azure Files Pricing | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/storage/files/, [Accessed 07-May-2023]. [Page 57.]

[62] "Cloud Storage | Google Cloud — cloud.google.com," https://cloud.google.com/storage, [Accessed 07-May-2023]. [Page 58.]

[63] "Pricing | Cloud Storage | Google Cloud — cloud.google.com," https://cloud.google.com/storage/pricing, [Accessed 07-May-2023]. [Page 58.]

[64] "Fast NoSQL Key-Value Database – Amazon DynamoDB – Amazon Web Services — aws.amazon.com," https://aws.amazon.com/dynamodb/, [Accessed 29-Apr-2023]. [Page 59.]

[65] "Fully Managed Relational Database - Amazon RDS - Amazon Web Services — aws.amazon.com," https://aws.amazon.com/rds, [Accessed 29-Apr-2023]. [Page 59.]

[66] "Amazon RDS for MySQL Pricing – Amazon Web Services — aws.amazon.com," https://aws.amazon.com/rds/mysql/pricing/, [Accessed 30-Apr-2023]. [Page 60.]

[67] "Azure Cosmos DB - NoSQL and Relational Database | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/products/cosmos-db/, [Accessed 07-May-2023]. [Page 60.]

[68] "Azure SQL Database – Managed Cloud Database Service | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/products/azure-sql/database/, [Accessed 07-May-2023]. [Page 60.]

[69] "Pricing - Azure Cosmos DB | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/cosmos-db/autoscale-provisioned/, [Accessed 07-May-2023]. [Page 61.]

[70] "Pricing - Azure SQL Database Single Database | Microsoft Azure — azure.microsoft.com," https://azure.microsoft.com/en-us/pricing/details/azure-sql-database/single/, [Accessed 07-May-2023]. [Page 61.]

[71] "Firestore: NoSQL document database | Google Cloud — cloud.google.com," https://cloud.google.com/firestore, [Accessed 07-May-2023]. [Page 61.]

[72] "Cloud SQL for PostgreSQL, MySQL, and SQL Server | Cloud SQL: Relational Database Service | Google Cloud — cloud.google.com," https://cloud.google.com/sql, [Accessed 07-May-2023]. [Page 61.]

[73] "Pricing | Firestore | Google Cloud — cloud.google.com," https://cloud.google.com/firestore/pricing, [Accessed 07-May-2023]. [Page 62.]

[74] "Pricing | Cloud SQL: Relational Database Service | Google Cloud — cloud.google.com," https://cloud.google.com/sql/pricing, [Accessed 07-May-2023]. [Page 62.]

[75] "What is Amazon S3? - Amazon Simple Storage Service — docs.aws.amazon.com," https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html, [Accessed 29-May-2023]. [Page 63.]

[76] "Azure Static Web Apps – App Service | Microsoft Azure." [Online]. Available: https://azure.microsoft.com/en-us/products/app-service/static [Page 63.]

[77] craigshoemaker, "Azure Static Web Apps documentation." [Online]. Available: https://learn.microsoft.com/en-us/azure/static-web-apps/ [Page 64.]

[78] "upload-cloud-storage," May 2023, original-date: 2020-10-31T01:09:32Z. [Online]. Available: https://github.com/google-github-actions/upload-cloud-storage [Page 64.]

[79] "Product overview of Cloud Storage | Google Cloud." [Online]. Available: https://cloud.google.com/storage/docs/introduction [Page 64.]

[80] "Host a static website | Cloud Storage | Google Cloud." [Online]. Available: https://cloud.google.com/storage/docs/hosting-static-website [Page 64.]

[81] ggailey777, "Azure Functions documentation." [Online]. Available: https://learn.microsoft.com/en-us/azure/azure-functions/ [Page 66.]

TRITA-EECS-EX- 2023:0000

€€€€ For DIVA €€€€

{
"Author1": { "Last name": "Lundström",
"First name": "Fredrik",
"Local User Id": "u100001",
"E-mail": "frlundst@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
}
},
"Author2": { "Last name": "Kristiansson",
"First name": "Casper",
"Local User Id": "u100002",
"E-mail": "casperkr@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
}
},
"Cycle": "1",
"Course code": "II142X",
"Credits": "15.0",
"Degree1": {"Educational program": "Degree Programme in Computer Engineering"
,"programcode": "TIDAB"
,"Degree": "Bachelors degree"
,"subjectArea": "Technology"
},
"Title": {
"Main title": "Cloud Computing Pricing and Deployment Efforts",
"Subtitle": "Navigating Cloud Computing Pricing and Deployment Efforts: Exploring the Public-Private Landscape",
"Language": "eng" },
"Alternative title": {
"Main title": "Prissättning och Implementeringsinsatser för Molntjänster",
"Subtitle": "Att Navigera Molntjänsters Prissättning och Implementeringsinsatser: Utforska det Offentlig-Privata Landskapet",
"Language": "swe"
},
"Supervisor1": { "Last name": "Montelius",
"First name": "Johan",
"Local User Id": "u100003",
"E-mail": "johanmon@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "Computer Science" }
},
"Supervisor2": { "Last name": "Kjörk",
"First name": "Mattias",
"E-mail": "mattias.kjork@amaceit.se",
},
"Examiner1": { "Last name": "Gauraha",
"First name": "Niharika",
"Local User Id": "u100004",
"E-mail": "niharika@kth.se",
"organisation": {"L1": "School of Electrical Engineering and Computer Science",
"L2": "Computer Science" }
},
"Cooperation": { "Partner_name": "amaceit AB"},
"National Subject Categories": "102, 10201, 10202, 10205, 10206, 10299, 20206",
"Other information": {"Year": "2023", "Number of pages": "1,100"},
"Copyrightleft": "copyright",
"Series": { "Title of series": "TRITA-EECS-EX" , "No. in series": "2023:0000" },
"Opponents": { "Name": "Kalle Elmdahl, Hampus Nilsson"},
"Presentation": { "Date": "2022-06-02 14:00"
,"Language":"eng"
,"Room": "Grimeton"
,"Address": "Isafjordsgatan 22 (Kistagången 16)"
,"City": "Stockholm" },
"Number of lang instances": "2",
"Abstract[eng]": €€€€

The expanding adoption of cloud computing services by businesses has transformed IT infrastructure and data management in the computing space. Cloud computing offers advantages such as availability, scalability, and cost-effectiveness, making it a favored choice for businesses of all sizes. The aim of this thesis is to compare private and public cloud computing services in terms of pricing and implementation effort as well as comparing the cloud providers to each other. The top three cloud providers that will be examined are Google GCP, Microsoft Azure, and Amazon AWS. The study examines different pricing models and evaluates their effectiveness in different business scenarios. In addition, the thesis also discusses the challenges associated with building and maintaining private infrastructure and the deployment of applications to cloud computing service are examined. The research methodology involves data collection, analysis, and a case study of developing and deploying a ticketing system application on different cloud platforms. The ticket system helps to provide a realistic example and investigation of the cloud providers. The findings will help companies make informed decisions regarding the selection of the most appropriate cloud computing service based on pricing models and implementation efforts. The thesis provides valuable information on private and public cloud computing and recommends appropriate pricing models for different scenarios. This study adds to existing knowledge by analyzing current pricing models and deployment concepts in cloud computing. The thesis does not propose new solutions but follows a structured format compiling information on private, and public cloud computing and a comprehensive review of cloud computing pricing models and marketing efforts. €€€€,

"Keywords[eng]": €€€€
Cloud computing, Private cloud, Public cloud, Cloud computing services, Cost-effectiveness, Implementation effort, Google GCP, Microsoft Azure, Amazon AWS, Pricing models, Cloud adoption, Cloud cost management, Cloud migration, Instance computing, Serverless computing, Data storage €€€€,
"Abstract[swe]": €€€€
Den växande adoptionen av molntjänster inom företag har förändrat IT-infrastrukturen och datahanteringen inom datorområdet. Molntjänster erbjuder fördelar såsom tillgänglighet, skalbarhet och kostnadseffektivitet, vilket gör det till ett populärt val för företag i alla storlekar. Syftet med denna avhandling är att jämföra privata och offentliga molntjänster med avseende på prissättning och implementeringsinsatser samt att jämföra molnleverantörerna med varandra. De tre främsta molnleverantörerna som kommer att undersökas är Google GCP, Microsoft Azure och Amazon AWS. Studien undersöker olika prismodeller och utvärderar deras effektivitet i olika affärsscenarier. Dessutom diskuterar avhandlingen också utmaningarna med att bygga och underhålla privat infrastruktur samt implementeringen av applikationer till molntjänster. Forskningsmetodologin omfattar datainsamling, analys och en fallstudie av utveckling och implementering av ett support system på olika molnplattformar. Supportsystemet hjälper till att ge ett realistiskt exempel och undersökning av molnleverantörerna. Resultaten kommer att hjälpa företag att fatta informerade beslut när det gäller valet av lämpligaste molntjänst baserat på prismodeller och implementeringsinsatser. Avhandlingen tillhandahåller värdefull information om privat och offentlig molntjänst och rekommenderar lämpliga prismodeller för olika scenarier. Denna studie bidrar till befintlig kunskap genom att analysera nuvarande prismodeller och implementeringskoncept inom molntjänster. Avhandlingen föreslår inga nya lösningar, men följer en strukturerad format genom att sammanställa information om privat och offentlig molntjänst samt en omfattande översikt av prismodeller och marknadsinsatser inom molntjänster. €€€€,
"Keywords[swe]": €€€€
Molntjänster, Privat moln, Offentligt moln, Kostnadsjämförelse, Kostnadseffektivitet, Google GCP, Microsoft Azure, Amazon AWS, Molninförande, Molnkostnadshantering, Molnmigration, Instance computing, Serverless computing, Dataförvaring €€€€,
}

acronyms.tex

```
\setabbreviationstyle[acronym]{long-short}

\newacronym{IT}{IT}{Information Technology}
\newacronym{GCP}{GCP}{Google Cloud Platform}
\newacronym{AWS}{AWS}{Amazon Web Services}
\newacronym{US}{US}{United States}
\newacronym{IaaS}{IaaS}{Infrastructure as a Service}
\newacronym{PaaS}{PaaS}{Platform as a Service}
\newacronym{FaaS}{FaaS}{Function as a Service}
\newacronym{SaaS}{SaaS}{Software as a Service}
\newacronym{CRM}{CRM}{Customer relationship management}
\newacronym{CAGR}{CAGR}{Compund Annual Growth Rate}
\newacronym{AI}{AI}{Artificial Intelligence}
\newacronym{ML}{ML}{Machine Learning}
\newacronym{EU}{EU}{European Union}
\newacronym{CAPEX}{CAPEX}{Capital Expenditures}
\newacronym{OPEX}{OPEX}{Operating Expenses}
\newacronym{UX}{UX}{User Experience}
\newacronym{API}{API}{Application Programming Interface}
\newacronym{AD}{AD}{Active Directory}
\newacronym{REST}{REST}{Representational State Transfer}
\newacronym{IAM}{IAM}{Identity and Access Management}
\newacronym{CI}{CI}{Continuous Integration}
\newacronym{ORM}{ORM}{Object-Reltaional Mapping}
\newacronym{CORS}{CORS}{Cross-Origin Resource Sharing}
\newacronym{HTTP}{HTTP}{Hypertext Transfer Protocol}
\newacronym{CSS}{CSS}{Cascading Style Sheets}
\newacronym{DBMS}{DBMS}{Database Management System}
\newacronym{IP}{IP}{Internet Protocol}
\newacronym{vCPU}{vCPU}{Virtual Processor}
\newacronym{HTTPS}{HTTPS}{Hypertext Transfer Protocol Secure}
\newacronym{FTPS}{FTPS}{SSL File Transfer Protocol}
\newacronym{HTML}{HTML}{HyperText Markup Language}
```