Efficiency Evaluation of Deploying Application

AHMMED ADESANYA

Master’s thesis 2024

Tomas Bata University in Zlín
Faculty of Applied Informatics

Tomas Bata University in Zlín
Faculty of Applied Informatics
Department of Informatics and Artificial Intelligence

Academic year: 2023/2024

ASSIGNMENT OF DIPLOMA THESIS
(project, art work, art performance)

Name and surname:	**Ahmmed Abidemi Adesanya**
Personal number:	**A22705**
Study programme:	**N0613A140023 Information Technologies**
Specialization:	**Software Engineering**
Type of Study:	**Full-time**
Work topic:	**Efficiency Evaluation of Deploying Applications**
Work topic in English:	**Efficiency Evaluation of Deploying Applications**

Theses guidelines

1. Prepare a literature survey on the topic.
2. Describe the differences between cloud services and on-demand servers.
3. Select suitable metrics to compare deploying targets
4. Select a suitable application for deployment tests.
5. Perform the comparison based on selected metrics.
6. Discuss the results and define the recommendations.

Form processing of diploma thesis:	**printed/electronic**
Language of elaboration:	**English**

Recommended resources:

1. KAVIS, Michael. *Architecting the Cloud: design decisions for Cloud computing service models (SaaS, PaaS, and IaaS)*. Wiley CIO series. Hoboken, New Jersey: Wiley, c 2014. ISBN 978-1118617618.
2. LISDORF, Anders. *Cloud computing basics: a non-technical introduction*. New York: Apress, [2021]. ISBN 978--1484269206.
3. B. RUPARELIA, Nayan. *Cloud Computing*. Cambridge (Massachusetts): The MIT Press, 2016. ISBN 978-0262529099.
4. BAHNA, Arshdeep a Vijay MADISETTI. *Cloud Computing Solutions Architect*. VPT, 2019. ISBN 978-0-9960255-9-1.
5. FRICK, Eric. *Introduction to Cloud Computing*. Independently published, 2021. ISBN 979-8493362101.

Supervisors of diploma thesis:	**Ing. Peter Janků, Ph.D.** Department of Informatics and Artificial Intelligence
Date of assignment of diploma thesis:	**November 5, 2023**
Submission deadline of diploma thesis:	**May 13, 2024**

Tomas Bata University in Zlín
Faculty of Applied Informatics
L.S.

doc. Ing. Jiří Vojtěšek, Ph.D. m.p.
Dean

prof. Mgr. Roman Jašek, Ph.D., DBA m.p.
Head of Department

In Zlín January 5, 2024

ABSTRAKT

Cílem této práce je prozkoumat a porovnat nasazení aplikací na cloudových a lokálních serverech se zaměřením na aspekty výkonu, škálovatelnosti, nákladové efektivity a bezpečnosti. Rostoucí popularita cloud computingu přinesla nové možnosti nasazení aplikací, které nabízejí flexibilitu, škálovatelnost a nižší režijní náklady na správu infrastruktury. Pro některé organizace však zůstává nasazení lokálních serverů životaschopnou možností, která nabízí větší kontrolu nad infrastrukturou a daty. Tato práce poskytne pohled na kompromisy a výhody jednotlivých přístupů a pomůže organizacím při přijímání informovaných rozhodnutí týkajících se strategií nasazení aplikací.

ABSTRACT

This thesis aims to investigate and compare the deployment of applications on cloud servers and on-premises servers, focusing on the aspects of performance, scalability, cost-effectiveness, and security. The growing popularity of cloud computing has introduced new possibilities for application deployment, offering flexibility, scalability, and reduced infrastructure management overhead. However, on-premises server deployment remains a viable option for certain organizations, offering greater control over infrastructure and data. This thesis will provide insights into the trade-offs and benefits of each approach, assisting organizations in making informed decisions regarding application deployment strategies.

ACKNOWLEDGEMENTS

I wish to extend my deepest appreciation to my thesis advisor, Prof. Ing Peter Janků, PhD, for their unwavering guidance, support, and invaluable insights throughout the entirety of this research and thesis writing process. Their expertise and encouragement have played a pivotal role in shaping this work.

I am also grateful to Assoc. Prof. Luis Antonio Beltrán Prieto, PhD, for proposing this topic for my thesis and for instilling confidence in me to pursue it.

Furthermore, I would like to express my gratitude to all the faculty members and instructors, particularly Assoc. Prof. Luis Antonio Beltrán Prieto, PhD, for their enlightening courses and seminars, which equipped me with the necessary technical skills to undertake this thesis.

Special thanks are also due to the Faculty of Informatic (FAI) and its staff for providing the essential resources and fostering an environment conducive to academic inquiry.

I am profoundly thankful to my family for their unwavering love, encouragement, and understanding throughout this demanding period. Their unwavering belief in my abilities has been a constant source of strength.

To my friends and colleagues who offered moral support and encouragement, I am deeply appreciative.

Lastly, I acknowledge the countless researchers, scholars, and authors whose work has laid the groundwork for this study. Their contributions have been invaluable in shaping my understanding of the subject matter.

This thesis would not have been possible without the support and contributions of all those mentioned above. Thank you for being part of this significant milestone in my academic journey.

CONTENTS

I. THEORY

1 INTRODUCTION

The IT infrastructure landscape has completely changed as a result of cloud computing, which provides easy online access to a wide variety of computer resources. Organizations have been forced to reconsider their IT strategy as a result of this paradigm change, especially when it comes to determining how effective their server architecture is under different deployment scenarios.

In this thesis, we examine the relative server efficiency of several deployment options, such as traditional on-premises configurations and cloud-based Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).

Although cloud services provide flexibility, scalability, and possible cost savings, worries about data privacy, security, and regulatory compliance still exist. On the other hand, on-premises deployments provide you with more control over the infrastructure, but they could cost more upfront and require more constant upkeep.

The objective of this research is to conduct a thorough assessment of server efficiency parameters, including throughput, latency, scalability constraints, and resource usage, using several deployment methods. We aim to determine the benefits and drawbacks of each deployment type through stress and performance testing carried out on various cloud and on-premises systems.

Additionally, we will explore the impact of emerging technologies like serverless computing and edge computing on server efficiency. By addressing research gaps and providing practical recommendations, this thesis aims to assist organizations in optimizing their server infrastructure deployment strategies to align with their unique needs and requirements.

1.1 Background and motivation

The advent of cloud computing technology has brought about a significant transformation in the application deployment landscape in recent years. Many advantages come with cloud computing, including reduced infrastructure management overhead, scalability, and flexibility, all of which are handled automatically by the cloud service. On the other hand, traditional on-premises server deployment offers benefits of its own, such as more control over data and infrastructure.

This study is driven by the necessity to comprehend and assess how much more efficient it is to deploy apps on cloud servers as opposed to on-premises servers. With the multiplication of application deployment options, organizations must have a thorough understanding of the advantages and disadvantages of each strategy. By delving into this subject, we hope to offer organizations insightful advice that will help them make well-informed decisions about how to deploy applications.

1.2 Research Objectives

Looking at the research objectives, this thesis aims to perform an efficiency evaluation and comparative analysis of the deployment of applications on on-premises and cloud servers. To accomplish this overall goal, the following goals will be worked toward:

1. Use strict benchmarking and testing procedures to evaluate the performance of apps installed on cloud servers versus those installed on on-premises servers.
2. Compare and contrast the deployment models' scalability features, taking workload management, elasticity, and resource provisioning into account.
3. Examine the security ramifications of installing apps on cloud servers as opposed to on-premises servers, taking vulnerability management, data protection, and compliance into account.

By fulfilling these goals, we hope to offer insightful information about how effective it is to deploy apps on cloud servers as opposed to on-premises servers, helping businesses make decisions that are tailored to their unique needs and limitations.

1.3 Scope and Limitation

Concerning the Limitations and Scope, It is crucial to clearly define the parameters of this research to make it clear what will be covered and what might fall outside of its purview:

Scope:

The deployment of common web-based applications on cloud servers—such as Amazon Web Services, Microsoft Azure, and Google Cloud Platform—and on-premises servers in a controlled environment will be the main comparison point for this study.

- Metrics like response time, throughput, and resource usage under various workload scenarios will be included in the performance evaluation.

- The capacity of each deployment model to manage growing user loads and dynamically distribute resources will be considered during the scalability analysis process.
- Examining data protection methods, access restrictions, compliance plans, and possible weaknesses in every deployment model are all part of the security considerations.

Limitation:

- The research might not take into consideration every potential variation in deployment configurations or particular use cases that apply to every organization.
- Although they can't be completely replicated in a controlled research setting, real-world factors like network latency, geographic distribution, and organizational constraints may have an impact on the results.
- Although security assessments will be based on industry standards and recognized best practices, they might not include all potential security risks or newly emerging threats.

Despite these drawbacks, the goal of this research is to add to the body of knowledge already available in the fields of infrastructure management and cloud computing by offering insightful information about the relative effectiveness of deploying apps on cloud servers as opposed to on-premises servers.

2 OVERVIEW OF CLOUD COMPUTING AND ON-PREMISES SERVER

Cloud computing, which offers on-demand access to a shared pool of configurable computing resources over the internet, has completely changed how businesses deploy and manage their IT infrastructure. Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) are the three deployment models that are commonly used to group cloud services. On-premises servers, on the other hand, relate to the actual hardware and software infrastructure that an organization owns, operates, and maintains on its own property.[1]

In Addition to the description of the above Services,

1. **Infrastructure as a Service (IaaS):**

Infrastructure as a Service (IaaS) is a cloud computing model where users can rent virtualized computing resources over the Internet. In an IaaS model, users have access to virtualized hardware resources such as servers, storage, and networking components, which they can manage and control remotely. Users typically pay for IaaS services on a pay-as-you-go basis, where they are charged based on their usage of resources such as CPU cycles, storage space, and network bandwidth. Examples of IaaS providers include Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).[1]

2. **Platform as a Service (PaaS):**

Platform as a Service (PaaS) is a cloud computing model that provides users with a platform to develop, run, and manage applications without having to worry about the underlying infrastructure. In a PaaS model, developers can build, deploy, and scale applications using tools and services provided by the cloud provider, such as development frameworks, databases, and application runtime environments. PaaS abstracts away many of the complexities associated with managing infrastructure, allowing developers to focus on writing code and delivering value to their customers. Examples of PaaS providers include Heroku, Microsoft Azure App Service, and Google App Engine.[2]

3. **Software as a Service (SaaS):**

Software as a Service (SaaS) is a cloud computing model where users can access and use software applications over the internet on a subscription basis. In a SaaS model, the software applications are hosted and maintained by the cloud provider, and users can access them through a web browser or application interface. SaaS offerings are typically provided on a multi-tenant architecture, where multiple users share the same instance of the application while still maintaining data isolation and security. Examples of SaaS applications include Salesforce, Microsoft Office 365, and Google Workspace.[3]

2.1 Comparison of Deployment Models

Cloud computing has become embraced in the computer world. Cloud implementation is the process of creating a virtual computing environment. Deployment in the cloud provides organizations with flexible and scalable virtual computing resources. A cloud deployment model is the type of architecture in which a cloud system is deployed. These models differ in terms of administration, ownership, access control, and security protocols. This paper describes the different types of cloud computing service models and deployment models; it also gives us a comparative study of various clouds using many factors. The comparison is simply based on various factors such as reliability, cost, data control, workload, performance, and many other cloud parameters.[4]

2.2 Benefits and Challenges of Cloud Deployment

Cloud deployment refers to the practice of storing and accessing data and applications over the internet on remote servers managed by a cloud service provider (CSP). This approach offers a range of benefits and challenges for organizations, as outlined below:

Benefits:

- **Scalability and agility:** Cloud resources can be easily scaled up or down based on demand, allowing for rapid adaptation to changing business needs and fluctuating workloads. This enables organizations to be more agile and responsive to market opportunities without significant upfront investment in hardware.[5]
- **Cost-efficiency:** Cloud services offer a pay-as-you-go model, eliminating the need for large upfront investments in hardware and software. This can be particularly beneficial for organizations with unpredictable or fluctuating resource needs, as they only pay for the resources they use. [6]
- **Accessibility and mobility:** Cloud-based applications and data are accessible from any device with an internet connection, enabling remote work and collaboration for geographically dispersed teams. This enhances employee productivity and

flexibility while supporting a mobile workforce.[7]. Also, Cloud computing promises a new era of service delivery and deployment in such a way that every person can access any kind of service like storage, application, operating system and so on from anywhere, any time, using any device having an internet connection. Cloud computing opens new possibilities approaching sustainable solutions to deploy and advance their services upon that platform[8]. The sustainability of Cloud computing is to be addressed in terms of environmental and economic effects. In this thesis, we explore the energy-efficient approaches inside data centres from the site and IT infrastructure perspective, incorporating cloud networking from the access network technologies and network equipment point of view to provide a comprehensive prospect for achieving energy efficiency in cloud computing. Traditional and Cloud data centres would by compared to figure out which one is more recommended to be deployed[9]. Virtualization, as the heart of energy-efficient Cloud computing that can integrate some technologies like consolidation and resource utilization, has been introduced to prepare a background for the implementation part[9]. Finally, approaches for Cloud computing data centres at the operating system and especially the data centre level are presented, and Green Cloud architecture as the most suitable green approach is described in detail [10]. In the experiment segment, we modelled and simulated Facebook and studied the behaviour in terms of cost per,formance, and energy consumption to reach the most appropriate solution.[11]

- **Simplified management:** Cloud providers manage the underlying infrastructure, including hardware, software, and security updates, freeing up the organization's IT resources to focus on core business activities. This reduces the operational burden and allows IT teams to focus on strategic initiatives. [12]

Challenges:

- **Security concerns:** Cloud computing services enabled through information communication technology delivered to a customer as services over the Internet on a leased basis have the capability to extend up or down their service requirements or needs[13]. In this model, the infrastructure is owned by a third-party vendor, and the cloud computing services are delivered to the customers who request them. The cloud computing model has many advantages, including scalability, flexibility, elasticity, and efficiency, and supports outsourcing noncore activities of an organization[14]. Cloud computing offers an innovative business concept for organizations to adopt IT enabled services without advance investment. This model enables convenient, on-request network accessibility to a shared pool of IT computing resources like networks, servers, storage, applications, and services[15]. Cloud computing can be quickly provisioned and released with negligible management exertion or service provider interaction[13]. Even though organizations get many benefits from cloud computing services, many organizations are slow to accept the cloud computing service model because of security concerns and challenges associated with the management of this technology. Security, being the major issues which hinder the growth of the cloud computing service model due to the provision of handling confidential data by a third party, is risky, such that the consumers need to be more attentive in understanding the risks of data breaches in this new environment[13]. In this paper, we have discussed the security issues, challenges, and opportunities for the adoption and management of cloud computing services in an organization.[16]

- **Vendor lock-in:** Dependence on a single cloud provider can lead to vendor lock-in, making it difficult and costly to switch to another provider in the future. Carefully evaluating vendor options and considering multi-cloud solutions can help mitigate this risk[17]. Vendor lock-in is a major barrier to the adoption of cloud computing due to the lack of standardization. Current solutions and efforts to tackle the vendor lock-in problem are predominantly technology-oriented. Limited studies exist to analyse and highlight the complexity of vendor lock-in problems in the cloud environment[17]. Consequently, most customers are unaware of proprietary standards that inhibit the interoperability and portability of applications when taking services from vendors. A survey based on qualitative and quantitative approaches conducted in most research has identified the main risk factors that give rise to lock-in situations. The analysis of our survey of 114 participants shows that, as computing resources migrate from on-premise to the cloud, the vendor lock-in problem is exacerbated[17]. Furthermore, the findings exemplify the importance of interoperability, portability and standards in cloud computing. A number of strategies are proposed to avoid and mitigate lock-in risks when migrating to cloud computing[18]. The strategies relate to contracts, the selection of vendors that support standardised formats and protocols regarding standard data structures and

APIs, and the development of awareness of commonalities and dependencies among cloud-based solutions. We strongly believe that the implementation of these strategies has great potential to reduce the risks of vendor lock-in. [19]

- **Limited control and customization:** Cloud services adhere to the highest level of control since they serve as the base around which TOS providers construct their infrastructure and applications[20].TOS providers retain control over the platform's frontend architecture, information, data, and applications but have limited influence over the platform's backend infrastructure. Though this is not considered detrimental to any of the parties involved, but rather results in a beneficial relationship in which responsibility is divided according to each party's capabilities, with cloud providers facilitating data storage and the TOS provider offering terminals its cloud-based functionalities[21]. Cloud computing generates a contradiction; its enhanced mobility necessarily equates to higher control over terminal data. When obtaining a cloud-based TOS, the only actual hazard that demands immediate attention is when planning and administration are inadequate. Only firms that fail to match their IT strategy with their business goals will suffer cloud computing's disadvantages[21]. Failure to accurately show the regions of inefficiency in a terminal makes it difficult to determine the optimal TOS for cloud computing[22]. A few established firms are concerned that adopting a cloud architecture may expose their data and information to security threats. This element continues to be the primary impediment to businesses working at their highest possible level of efficiency[23]. Cloud computing is depicted as a liability when corporations make rash choices, resulting in a cascading effect that escalates into increasingly serious and dangerous concerns. For example, if a terminal adopts a TOS with forged cloud infrastructure and fails to scale resources appropriately, expenditures will skyrocket while profits will plummet. [24]
- **Latency and performance:** It is a research issue on the Internet. Any performance in the cloud is undergoing the same meaning as the performance of the result on the client. The latency is compressed back to get clarity as to how and where they are running with both smartly written applications and an intelligently planned infrastructure[25]. The latency in a cloud is not tedious. In the future, cloud computing capacity and cloud-based applications are bound to increase at a very

rapid rate, thus increasing latency. Cloud latency must be improved on desktop PCs, which is the largest bottleneck in memory and storage.[26]

2.3 Benefits and Challenges of On-Premises

Benefits:

- **Greater control and security:** Organizations have complete control over their data and applications, allowing them to implement stricter security measures and customize configurations to meet specific compliance requirements. This is particularly crucial for businesses handling sensitive data or operating in highly regulated industries.[27]
- **Customization:** This subsection explores the advantages of customization for businesses. We discuss how tailoring SaaS applications can lead to improved user satisfaction, increased productivity, and enhanced decision-making. Additionally, we address the potential to leverage customization as a competitive differentiator in the market. The benefits of customization in SaaS cloud solutions have been explored in several research publications. Customization allows SaaS providers to tailor their offerings to meet the specific needs and preferences of individual customers, leading to increased customer satisfaction and improved business outcomes. [28]
- **Offline access:** On-premises systems are not reliant on an internet connection, allowing for continued operation even during outages or in situations with limited internet access. This can be critical for businesses requiring uninterrupted access to core applications during network disruptions. On-premises systems refer to hardware and software infrastructure that organizations own, operate, and maintain on their own premises or property[29]. One of the key advantages of on-premises systems is their ability to function without relying on an internet connection. With on-premises systems, organizations have full control over their infrastructure, allowing them to maintain operations even during network disruptions or outages. Employees can continue to access critical applications and data stored on on-premises servers, ensuring business continuity and productivity. Offline access is particularly important for businesses that operate in remote locations or in areas with unreliable internet connectivity. Examples of applications that benefit from offline access in on-premises environments include enterprise resource planning

(ERP) systems, customer relationship management (CRM) software, and business intelligence (BI) tools.[30]

- **Potential cost-effectiveness:** While requiring an upfront investment in hardware and software, on-premises deployments can be more cost-effective in the long run for organizations with high, predictable workloads. This avoids ongoing subscription fees associated with cloud services, potentially lowering overall IT expenses over time, particularly for predictable usage patterns[31].

Challenges:

- **High upfront costs:** One of the main drawbacks of on-premise systems is the significant upfront capital expenditures required for hardware and infrastructure investments. Setting up an on-premise system can be expensive, requiring investments in servers, storage devices, networking equipment, and other hardware.[32]
- **Scalability limitations:** Scaling on-premises infrastructure to meet changing needs can be complex and time-consuming. Adding additional capacity often requires acquiring and configuring new hardware and software, leading to potential delays in responding to increasing demands or fluctuating workloads. This can hinder agility and responsiveness to changing business needs. In Addition, Planning for future growth and investing in additional resources can be challenging with on-premise systems. Expanding or upgrading an on-premise system can require additional hardware investments and complex migration processes, which may not be as agile and scalable as cloud systems.[32]
- **IT expertise required:** Managing an on-premises infrastructure necessitates a dedicated IT staff with the expertise to handle installation, configuration, troubleshooting, and security updates. This can be a challenge for smaller businesses or those lacking the in-house IT resources to effectively manage the infrastructure, potentially impacting operational efficiency and security posture[33]. Also, On-premises infrastructure requires traditional IT skills, including hardware and software systems administration, networking, database management and security. Such a combined skill set can be difficult to find, and many of these skills are less applicable as enterprises shift workloads to the cloud and embrace DevOps methodologies.[34]

- **Limited accessibility:** On-premises applications are typically only accessible from within the organization's internal network, making remote access for employees working outside the office more complex. This can hinder collaboration and productivity for geographically dispersed teams or those requiring mobile access to applications[35].

2.4 Previous Studies and Research Gaps

On-premises and cloud computing represent the two dominant models for IT infrastructure deployment. Extensive research has explored the benefits and challenges of both approaches, aiming to guide organizations in making informed decisions for their specific needs.

Previous Studies:

- **Security and compliance:** Studies have shown that Cloud Security Alliance (2013) have compared security considerations in both models. While on-premises offer greater perceived control, cloud providers invest heavily in security infrastructure and expertise, making the comparison complex and dependent on specific organizational security needs and regulations. [1]
- **Performance and scalability:** Studies have explored the aspects of performance and scalability. Cloud offerings typically provide on-demand scalability, while on-premises require upfront infrastructure planning and potentially longer lead times for scaling.[12, pp. 50–58]
- **Management and agility:** Studies have compared the management overhead and agility of both models. Cloud services generally offer centralized management and faster provisioning times, while on-premises deployments require dedicated IT staff and may involve longer setup processes[36].

Research Gaps:

- **Emerging technologies:** Despite advancements in technology, the impact of emerging trends such as hybrid cloud models, serverless computing, and edge computing on the on-premises vs. cloud landscape remains inadequately explored. Current studies often lack comprehensive insights into how these emerging technologies influence deployment decisions, performance, and long-term viability[37].

Context: Studies such as Varia (2016), Gartner's Top Strategic Technology Trends Report (2019), and Cisco's Visual Networking Index (2020) shed light on emerging technologies and their potential implications. However, there's a need for deeper analysis to understand their specific effects on the on-premises vs. cloud comparison.[38]

- **Organizational decision-making:** The decision-making processes of organizations regarding on-premises and cloud adoption involve a multitude of factors, including technical requirements, financial considerations, and organizational culture. However, existing research often lacks a comprehensive understanding of the multifaceted nature of these decisions and the interplay between technical and non-technical considerations[39].

  Context: Studies such as Bala and Venkatesh (2016), McKinsey's report on cloud transition (2018), and Brynjolfsson and McAfee's book (2014) offer insights into organizational decision-making processes. Yet, there's a need for more nuanced analysis to capture the dynamic nature of these processes and address the diverse needs and challenges faced by organizations.[40] [41]
- **Long-term impact studies:** While short-term benefits of cloud adoption, such as improved scalability and agility, are well-documented, there's a dearth of long-term impact studies that analyze the holistic effects of both on-premises and cloud deployments. Factors like business agility, cost-efficiency, and security require longitudinal studies to assess their sustained influence on organizational performance and competitiveness.[42] [43]

II. Practical

3 METHODOLOGY

3.1 Research and Design Approach

In this chapter, we will lay out the research design and methodology for contrasting the deployment of apps to on-premises and cloud servers. By using a comparative methodology, the study will examine the deployment procedures, and performance indicators of the two deployment models.

3.2 Data Collection and Analysis Methods

Information on the deployment procedures and performance indicators related to deploying apps to on-premises and cloud servers will be gathered solely from the provided case study.

Data Collection:

The primary data source for this research will be a case study, offering detailed insights into deployment procedures and performance indicators across on-premises and cloud servers. Additionally, JMeter, an open-source performance testing tool, will facilitate data collection and analysis. Through JMeter, various user scenarios will be simulated, and performance metrics like response times, throughput, and scalability will be measured[44]. These metrics will form the basis of the collected data, which will then be analyzed to assess the efficacy of deploying applications to both cloud-based and on-premises servers. JMeter provides graphical representations of response time for each endpoint of the API, aiding in data interpretation and analysis.

Data Analysis:

Quantitative Analysis: Quantitative Analysis: Performance metrics such as response time, throughput, and scalability will be derived from the collected data. Statistical methods, including descriptive statistics, average mean and standard deviation will be used to analyze these metrics, enabling the identification of trends and patterns.

This dual approach ensures that research findings are grounded in the specific context outlined by the case study, facilitating a comprehensive analysis of deployment procedures and performance indicators associated with deploying apps to on-premises and cloud servers.

3.3 Experimental Setup

3.4 Experimental Setup

The experimental setup for this research involves deploying the same application to both cloud servers and on-premises servers using representative use cases. Popular cloud service providers such as AWS and Azure Platform will be considered for cloud deployment. Additionally, standard hardware and software configurations will be used for on-premises deployment.

Performance tests will be conducted to measure response times, throughput, and scalability under different load conditions. Cost analysis will consider upfront investments, operational expenses, and total cost of ownership over a specified period.

The experimental setup is designed to ensure comparability and reproducibility of results, allowing for a rigorous evaluation of the efficiency of deploying applications to cloud servers and on-premises servers. Once the tests are conducted, the results will be analyzed to compare the deployment models effectively.

This chapter outlines the research design, data collection and analysis methods, evaluation metrics and criteria, and experimental setup for comparing the deployment of applications to cloud servers and on-premises servers. These methods will facilitate an objective and comprehensive assessment of the two deployment models.

3.1 Infrastructure Setup

On-Premises Server Configuration:

<u>Hardware Configuration:</u>

- **CPU:** This is the central processing unit, essentially the brain of the computer. The model mentioned is a 12th Generation Intel Core i5 processor, which is responsible for executing instructions and performing calculations.
- **RAM:** This stands for Random Access Memory, which is the computer's short-term memory. It's where data and instructions are stored temporarily while the CPU processes them. The system has 8GB of RAM.
- **Storage:** This refers to the storage capacity of the computer, typically in terms of gigabytes (GB). The system has 476GB of storage space available.

- **Network:** This section provides information about the computer's network connectivity.
  - **Interface Name:** This specifies the name of the network interface, which is Ethernet in this case.
  - **Physical Address (MAC Address):** This is a unique identifier assigned to the network interface card (NIC) of the computer. It's used for communication on a local network.
  - **Connection Status:** Indicates whether the computer is currently connected to the network.
  - **IP Address:** This is the unique address assigned to the computer on the network.
    - **IPv4:** A version of the Internet Protocol that uses a 32-bit address scheme. The IPv4 address mentioned is 10.12.19.17.
    - **IPv6:** A newer version of the Internet Protocol that uses a 128-bit address scheme. The IPv6 address mentioned is fe80::b4b1:5668:e61c:51a8%4.
      - **Subnet Mask:** This determines the network portion of an IP address and the host portion. It's used for communication within the network.
      - **Default Gateway:** This is the IP address of the router or gateway device that allows the computer to communicate with devices on other networks or the internet.

<u>**Software Configuration:**</u>

- **Operating System:** This is the software that manages the computer's hardware and provides a user interface. The mentioned operating system is Microsoft Windows 10.
- **Web Server:** This is software that serves web pages to users when they visit a website. The mentioned web server is Microsoft .Net Server.
- **Database Server:** This is software that stores and manages data in a database. The mentioned database server is SQLite, which is a lightweight, file-based database management system.
- **Application Framework:** This is a software framework that provides a foundation for developing applications. The mentioned framework is Microsoft ASP.Net 4.7, which is used for building web applications and services.

Network Configuration:

- **IP Address:** This is the unique address assigned to the computer on the network, which was mentioned earlier as 10.12.19.17.
- **Subnet Mask:** This determines the network portion of the IP address, which was mentioned earlier as 255.255.252.0.
- **Gateway:** This is the IP address of the router or gateway device that allows the computer to communicate with devices on other networks or the internet, which was mentioned earlier as 10.12.16.1.

Azure App Service Configuration:

- **Service Plan:** This specifies the pricing tier and features available for the Azure App Service. The service plan is Free Azure for Students.
- **Region:** This indicates the geographic location of the Azure data centre where the App Service is hosted. The region is North Europe.

Network Configuration:

- **Virtual IP Address:** This is the public IP address assigned to the Azure App Service. The virtual IP address is 20.107.224.53.

Software Configuration:

- **Operating System:** This is the operating system installed on the Azure App Service. The operating system is Windows.
- **Web Server**: This is the software that serves web pages to users when they visit a website. The web server is ASP.Net, which is a framework for building web applications.
- **Database:** This refers to the software used for storing and managing data. The mentioned database is SQLite, which is a lightweight, file-based database management system.
- **Application Framework**: This is a software framework that provides a foundation for developing applications. The framework is Dotnetcore, which is a cross-platform framework for building modern, cloud-based applications.

AWS EC2 Instance

<u>EC2 Instance Configuration:</u>

- **Instance Type:** t2.micro (This specifies the size and performance capabilities of the virtual server.)
- **EBS Volume Type:** gp2 (This refers to the type of storage volume attached to the EC2 instance, providing a balance of performance and cost.)
- **Region:** eu-north (This indicates the geographical location of the data centre where the EC2 instance is hosted.)

<u>EC2 Networking:</u>

- **Public IP Address:** 16.171.173.58 (This is the IP address that is accessible over the internet.)
- **Private IP Address:** 172-31-33-86 (This is the internal IP address used within the Amazon Virtual Private Cloud (VPC).)
- **VPC:** vpc-0c0ba7da8e214d875 (This is the Virtual Private Cloud where the EC2 instance is deployed, providing isolated networking resources.)

<u>Software Configuration:</u>

- **Operating System:** Windows_Server-2022-English-Full-Base-2024.01.16 (This specifies the version and language of the operating system installed on the EC2 instance.)
- **Web Server:** Apache httpd 2.4.58 (This is the software that serves web content, such as websites and web applications.)
- **SQLite:** SQLite file (This is the type of database used, specifically a file-based SQLite database.)
- **Application Framework:** dotnet hosting 6.0.26 (This refers to the version of the .NET hosting environment for running .NET applications.)

3.1.1 API Overview

The API, or Application Programming Interface, serves as the backbone of our system, facilitating communication between different software components. It exposes a set of endpoints that allow external systems to interact with our application.

3.1.2 Purpose of the API

The primary purpose of our API is to provide access to various functionalities of our application in a structured and secure manner. It enables external clients, such as web or mobile applications, to perform actions and retrieve data from our system. In the context of our service, which is similar to Steam/GOG/Epic, the API allows developers and users to interact with the platform, manage game libraries, purchase games, and access gaming-related features.

3.1.3 UserGameManagement API Documentation

Project Structure

1. **User_Game.Models:**

Contains the following folders:

Common: Holds shared utility classes or helper methods.

DAO (Data Access Object): Likely indicates the use of a repository pattern for data access, possibly leveraging Dapper for database integration.

Model: Contains data models representing entities in the system.

Request: May include request DTOs (Data Transfer Objects) for API endpoints.

Response: Possibly contains response DTOs for API endpoints.

Security: holds security-related classes (e.g., authentication, authorization).

Services: Potentially contains business logic services.

Also includes a file named ModuleConfiguration.cs.

UserGame.API:

Includes the following folders:

Connected Services: contain external service references.

Dependencies: Holds project dependencies (e.g., NuGet packages), including Dapper for database interaction.

Imports: contains namespace imports.

Properties: Contains project properties and settings.

Contract: Contains controller files:

GameController.cs: Handles game-related API endpoints.

UserController.cs: Manages user-related API endpoints.

logs: Contains log files (for logging purposes).

Security: Contains JwtAuthenticationManager.cs (for JWT-based authentication).

appsettings.json and appsettings.Development.json: Configuration files for application settings.

New_UserGameDatabase.db: Represents the database file (possibly SQLite).

Program.cs: The entry point for the application.

2. Design Patterns and Architectural Styles

While the image doesn't explicitly reveal the design pattern or architectural style, we can make educated guesses based on the project structure:

MVC (Model-View-Controller):

The presence of controller files (GameController.cs and UserController.cs) in the "Contract" folder suggests an MVC pattern.

In MVC:

Controllers handle user input.

Models represent data and business logic.

Views display the data.

Repository Pattern:

The "DAO" folder in "User_Game.Models" likely indicates the use of a repository pattern, possibly with Dapper for database interaction.

The repository pattern separates data access logic from the rest of the application.

Dependency Injection:

If there are services (e.g., in the "Services" folder), dependency injection might be used to inject dependencies into controllers or other components.

RESTful API Design:

The presence of controller files in the “Contract” folder suggests that this API follows RESTful principles.

RESTful APIs use HTTP methods (GET, POST, PUT, DELETE) to interact with resources.

Database Design:

The “New_UserGameDatabase.db” file indicates database usage, potentially integrated with Dapper for streamlined database interaction.

The architectural style likely involves connecting to a relational database (e.g., SQLite) for data storage.

3.1.4 API Endpoints

The API comprises several endpoints, each serving a specific function within the application. These endpoints can be categorized based on the operations they support. Below are the main sections of our API, along with their corresponding endpoints:

Request Payload Schemas:

- **CreateGameRequest:**

  Description: Represents the payload schema for creating a new game.

  Properties:

  gameName (string, nullable): The name of the game.

  gameImg (string, nullable): The URL of the game's image.

  Rating (integer, nullable): The rating of the game.

  Price (string, nullable): The price of the game.

  Tags (string, nullable): Tags associated with the game.

- **CreateUserGames:**

  Description: Represents the payload schema for associating a user with a game.

  Properties:

  userId (string, nullable): The ID of the user.

gameId (string, nullable): The ID of the game.

- **LogOutRequest:**

  Description: Represents the payload schema for logging out a user.

  Properties:

  userId (string, nullable): The ID of the user.

  isLogin (integer, nullable): Flag indicating if the user is logged in.

- **LoginRequest:**

  Description: Represents the payload schema for user login.

  Properties:

  Email (string, nullable): The user's email.

  Password (string, nullable): The password of the user.

- **RegisterUserRequest:**

  Description: Represents the payload schema for user registration.

  Properties:

  firstname (string, nullable): The first name of the user.

  lastname (string, nullable): The last name of the user.

  age (string, nullable): The age of the user.

  email (string, nullable): The email of the user.

  password (string, nullable): The password of the user.

  isLogin (integer, nullable): Flag indicating if the user is logged in.

- **UpdateGameRequest:**

  Description: Represents the payload schema for updating a game.

  Properties:

  gameId (string, nullable): The ID of the game.

  (Other properties same as CreateGameRequest)

- **UpdateUserRequest:**

  Description: Represents the payload schema for updating user information.

  Properties:

  userId (string, nullable): The ID of the user.

  (Other properties same as RegisterUserRequest)

Response Payload Schemas:

- **GameResponse:**

  Description: Represents the payload schema for a game response.

  Properties:

  gameId (string, nullable): The ID of the game.

  (Other properties same as CreateGameRequest)

- **GameResponseApiResponse:**

  Description: Represents the payload schema for a game response API response.

  Properties:

  success (boolean): Indicates if the request was successful.

  data (GameResponse): Contains the game response data.

  errorMessage (string, nullable): Error message if the request fails.

- **UserGame:**

  Description: Represents the payload schema for user game information.

  Properties:

  userId (string, nullable): The ID of the user.

  gameId (string, nullable): The ID of the game.

  (Other properties same as GameResponse)

- **UserGameResponse:**

  Description: Represents the payload schema for a user game response.

Properties:

userGameId (string, nullable): The ID of the user game.

userGame (UserGame): Contains the user game information.

- **UserGameResponseApiResponse:**

  Description: Represents the payload schema for a user game response API response.

  Properties:

  success (boolean): Indicates if the request was successful.

  data (UserGameResponse): Contains the user game response data.

  errorMessage (string, nullable): Error message if the request fails.

- **UserResponse:**

  Description: Represents the payload schema for a user response.

  Properties:

  userId (string, nullable): The ID of the user.

  (Other properties same as RegisterUserRequest)

  userToken (string, nullable): The token associated with the user.

- **UserResponseApiResponse:**

  Description: Represents the payload schema for a user response API response.

  Properties:

  success (boolean): Indicates if the request was successful.

  data (UserResponse): Contains the user response data.

  errorMessage (string, nullable): Error message if the request fails.

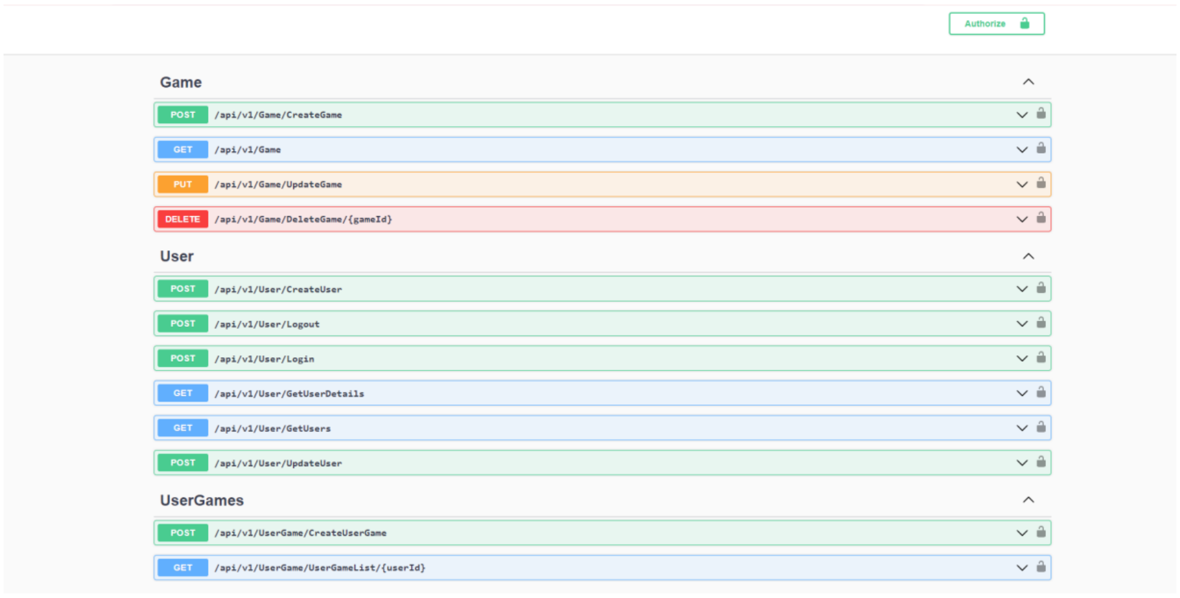
For more information about the API endpoints, refer to the documentation available at:

Local Server: http://16.171.173.58/swagger/index.html

AWS: http://10.12.19.17/swagger/index.html

Azure: https://usergamemanagerapi.azurewebsites.net/swagger/index.html

Below is a screenshot of the swagger API service of the Game Manager.



Game

```
POST    /api/v1/Game/CreateGame
GET     /api/v1/Game
PUT     /api/v1/Game/UpdateGame
DELETE  /api/v1/Game/DeleteGame/{gameId}
```

User

```
POST    /api/v1/User/CreateUser
POST    /api/v1/User/Logout
POST    /api/v1/User/Login
GET     /api/v1/User/GetUserDetails
GET     /api/v1/User/GetUsers
POST    /api/v1/User/UpdateUser
```

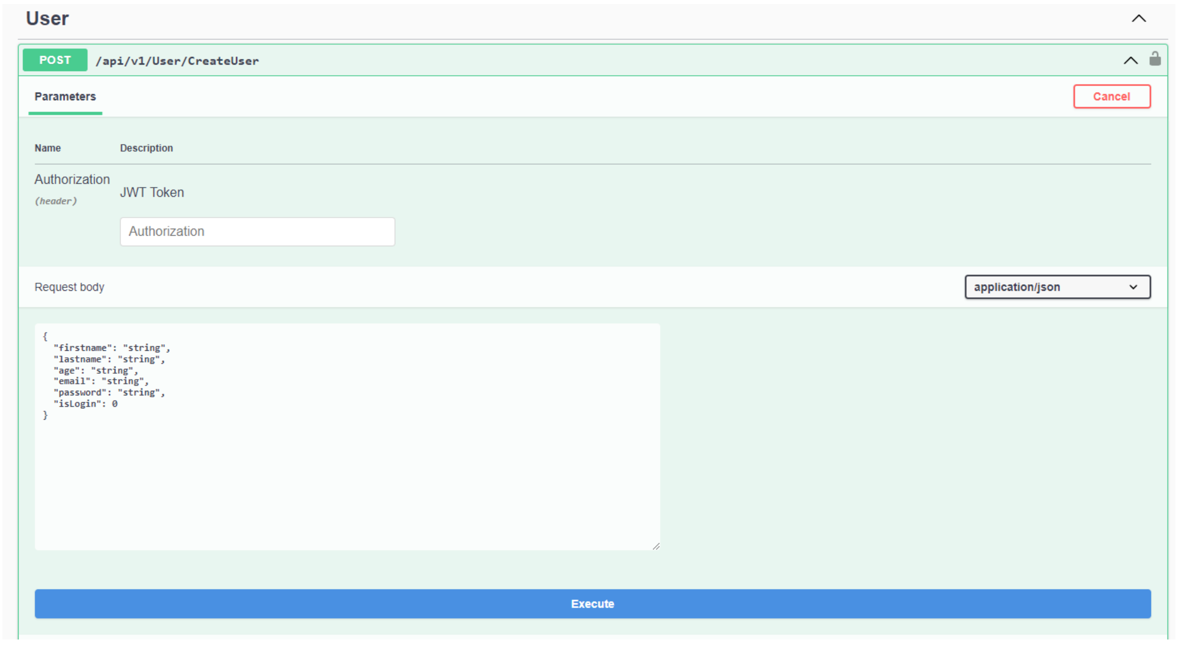
UserGames

```
POST    /api/v1/UserGame/CreateUserGame
GET     /api/v1/UserGame/UserGameList/{userId}
```

Figure 1. Swagger API Service

The above figure 1 shows the page of when the API Service is lunch.

Next is a payload request when you need to try out each endpoint.

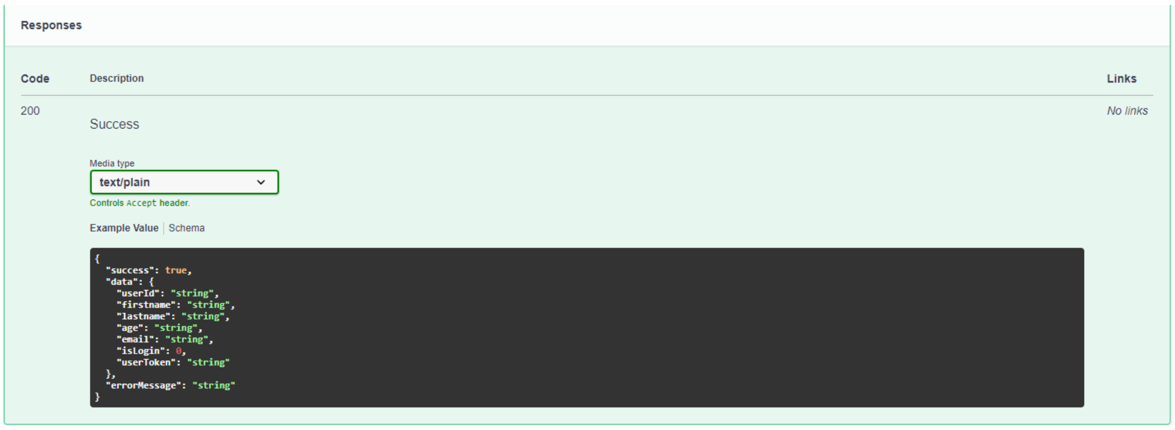


```
{
  "firstname": "string",
  "lastname": "string",
  "age": "string",
  "email": "string",
  "password": "string",
  "isLogin": 0
}
```

Figure 2. Sample Request Payload

Figure 2 above illustrates the payload request required for creating a user for the first time.

Following that is the response payload generated upon clicking the execute button.

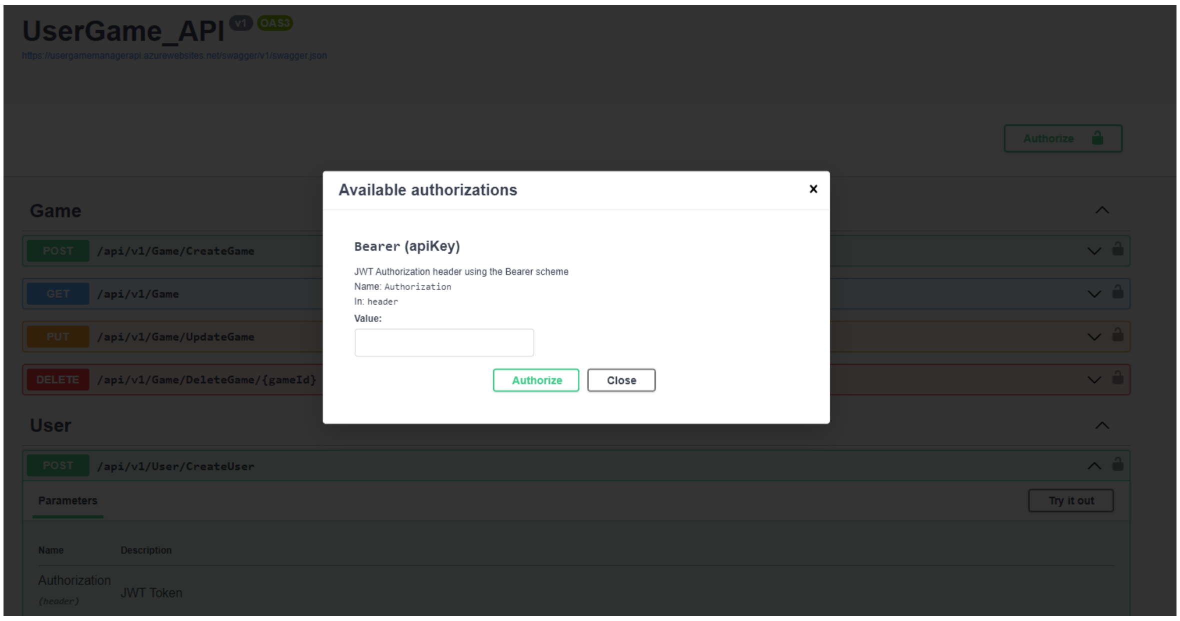


```
{
  "success": true,
  "data": {
    "userId": "string",
    "firstname": "string",
    "lastname": "string",
    "age": "string",
    "email": "string",
    "isLogin": 0,
    "userToken": "string"
  },
  "errorMessage": "string"
}
```

Figure 3. Sample Payload Response

In Figure 3, the displayed response payload is indicative of what you should expect upon clicking the execute button in Figure 2.

The subsequent figure depicts an interface that mandates authorization, as certain end-points necessitate authorization prior to execution.



UserGame_API v1 OAS3

https://usergamemanagerapi.azurewebsites.net/swagger/v1/swagger.json

Available authorizations

Bearer (apiKey)

JWT Authorization header using the Bearer scheme

Name: Authorization

In: header

Value:

Authorize
Close
Authorize
Game
POST /api/v1/Game/CreateGame
GET /api/v1/Game
PUT /api/v1/Game/UpdateGame
DELETE /api/v1/Game/DeleteGame/{gameId}
User
POST /api/v1/User/CreateUser
Parameters
Try it out

Name	Description
Authorization (header)	JWT Token

Figure 4. Bearer Token Dialog

Figure 4 illustrates the interface where a bearer API key or token is necessary to access certain endpoints within the API service.

4 TEST EXECUTION AND RESULTS

This chapter covers how the performance testing was carried out and provides the findings from the experiments that were done. The main goal is to evaluate the system's performance under different load scenarios and examine its behavior with respect to response times, throughput, and other important metrics. An overview of the test environment and configuration opens the chapter, which is then followed by a thorough explanation of the test scenarios that were run. We next examine the outcomes from every scenario to get a better understanding of the system's performance traits.

4.1 Use Case (Deployment of a Game Management API)

4.1.1 Users Sign-up and Login Endpoint

Evaluation Scenario:

- **Performance:** Measure the response time and throughput of the Sign-up and Login endpoints under varying loads, simulating concurrent user registrations and login attempts.
- **Scalability:** Evaluate the API's ability to handle increasing numbers of Sign-up and Login requests by gradually increasing the load and monitoring system performance, resource utilization, and scalability features such as load balancing and horizontal scaling.

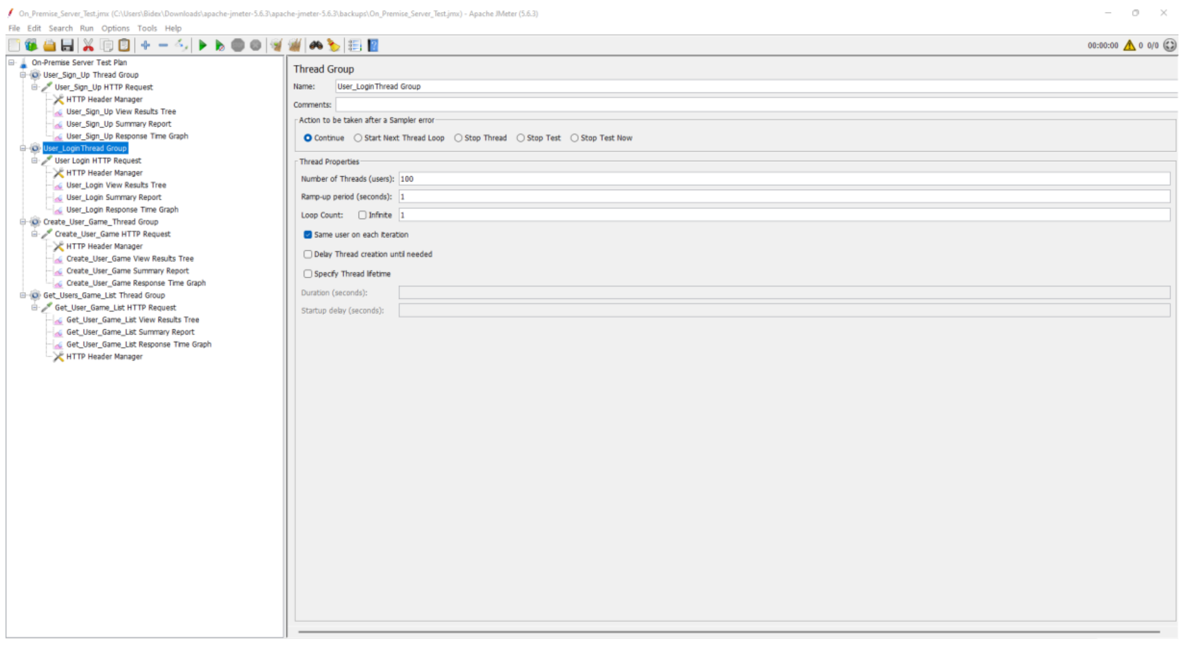
4.1.2 UserGame Endpoints (Create, Get Usergame List)

Evaluation Scenario:

- **Performance:** Measure the latency and throughput of the Create, Update, and Delete endpoints for user games, simulating concurrent requests for game creation, updates, and deletions.
- **Scalability:** Evaluate the API's ability to scale horizontally and vertically to accommodate growing numbers of concurrent requests for game management operations, assessing scalability features such as database partitioning, caching, and data replication.

4.2 Test Execution

Displayed below is a screenshot capturing the JMeter environment used for the test:


On_Premise_Server_Test.jmx (C:\Users\Bidex\Downloads\apache-jmeter-5.6.3\apache-jmeter-5.6.3\backups\On_Premise_Server_Test.jmx) - Apache JMeter (5.6.3)
File Edit Search Run Options Tools Help
00:00:00 0 0/0

```
On-Premise Server Test Plan
  User_Sign_Up Thread Group
    User_Sign_Up HTTP Request
      HTTP Header Manager
      User_Sign_Up View Results Tree
      User_Sign_Up Summary Report
      User_Sign_Up Response Time Graph
  User_Login Thread Group
    User Login HTTP Request
      HTTP Header Manager
      User_Login View Results Tree
      User_Login Summary Report
      User_Login Response Time Graph
  Create_User_Game_Thread Group
    Create_User_Game HTTP Request
      HTTP Header Manager
      Create_User_Game View Results Tree
      Create_User_Game Summary Report
      Create_User_Game Response Time Graph
  Get_Users_Game_List Thread Group
    Get_User_Game_List HTTP Request
      Get_User_Game_List View Results Tree
      Get_User_Game_List Summary Report
      Get_User_Game_List Response Time Graph
      HTTP Header Manager
```

Thread Group

Name: User_Login Thread Group

Comments:

Action to be taken after a Sampler error

[✓ Continue] Start Next Thread Loop Stop Thread Stop Test Stop Test Now

Thread Properties

Number of Threads (users): 100

Ramp-up period (seconds): 1

Loop Count: Infinite 1

[✓ Same user on each iteration]

Delay Thread creation until needed

Specify Thread lifetime

Duration (seconds):

Startup delay (seconds):

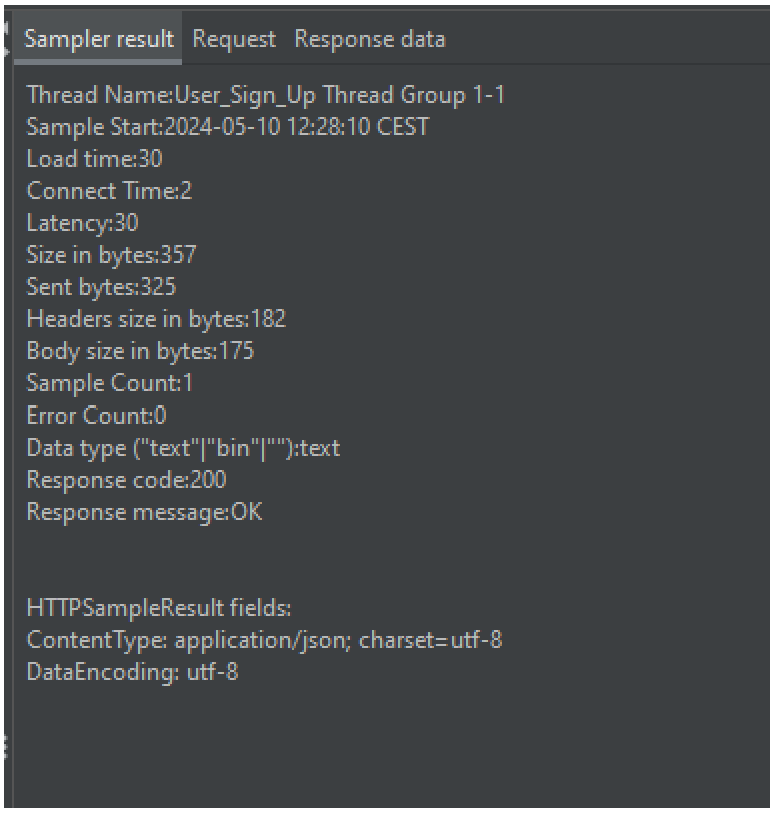
Figure 5. (JMeter IDE)

The JMeter screenshot includes the following sub-tabs:

- **HTTP Request for Login Sign_Up:** This tab likely contains configurations for sending HTTP requests related to user login and sign-up processes.
- **HTTP Header Manager:** This tab is used to manage HTTP headers for requests sent by JMeter.
- **User Sign-Up Summary Report:** Displays statistical data related to user sign-up requests, such as average response time, throughput, and error percentage.
- **User Sign-Up Response Time Graph:** A graphical representation of response times for user sign-up requests.
- **User Login Thread Group:** This tab likely defines a thread group for simulating user login scenarios.
- **Create_User_Summary Report:** Similar to the sign-up summary report, but specific to user creation requests.
- **User_Login_Response Time Graph:** A graph showing response times for user login requests.

4.2.1 Test scenarios:

1. **Users sign-up Endpoint:**



```
Thread Name:User_Sign_Up Thread Group 1-1
Sample Start:2024-05-10 12:28:10 CEST
Load time:30
Connect Time:2
Latency:30
Size in bytes:357
Sent bytes:325
Headers size in bytes:182
Body size in bytes:175
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK


HTTPSampleResult fields:
ContentType: application/json; charset=utf-8
DataEncoding: utf-8
```

Figure 6. Sample Result of Response for User Sign Up on On-Premises

Figure 6 illustrates the test carried out on an On-premises server, that:

- The test was executed under the thread group named "User_Sign_Up_Thread Group 1-62."
- The test started on 2024-03-30 at 14:47:29 CET.
- The total time taken for the request and response was 30 milliseconds.
  - The time taken to establish the connection was seven milliseconds.
  - The time between sending the request and receiving the first byte of the response was 30 milliseconds.
- The response data size:
  - Size in Bytes: The total size of the response (357 bytes).
  - Sent Bytes: The size of data sent in the request (325 bytes).
  - Headers Size in Bytes: The size of HTTP headers (182 bytes).
  - Body Size in Bytes: The size of the response body (175 bytes).

- The response data type was “text.”
- The response code was 200 (indicating success), and the response message was “OK.”
- Additional HTTPSampleResult fields:
    - Content Type: application/json; charset=utf-8
    - Data Encoding: utf-8

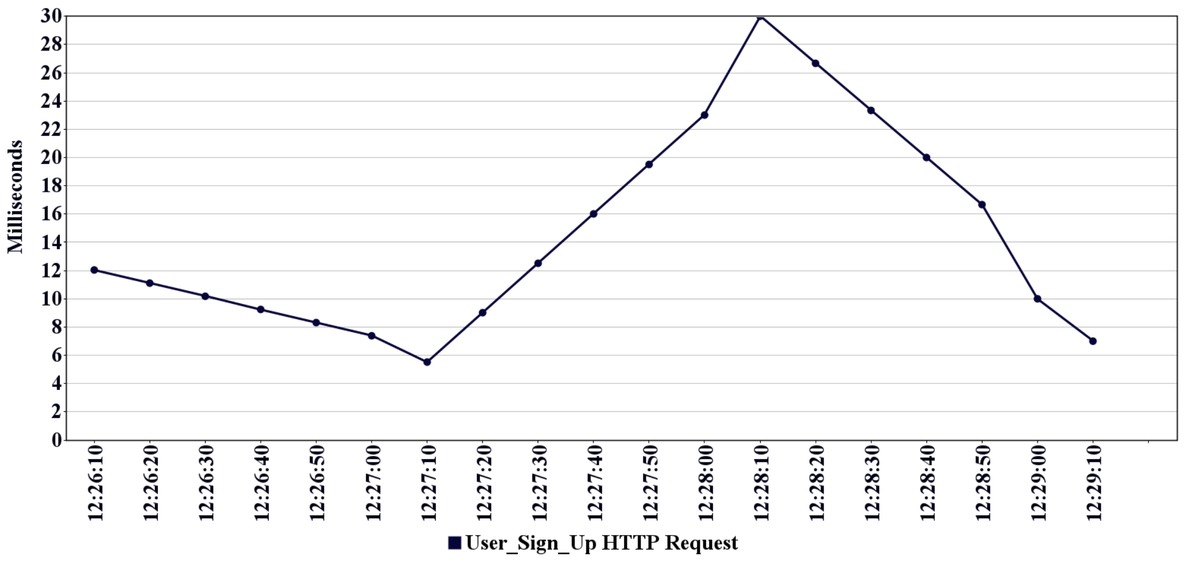
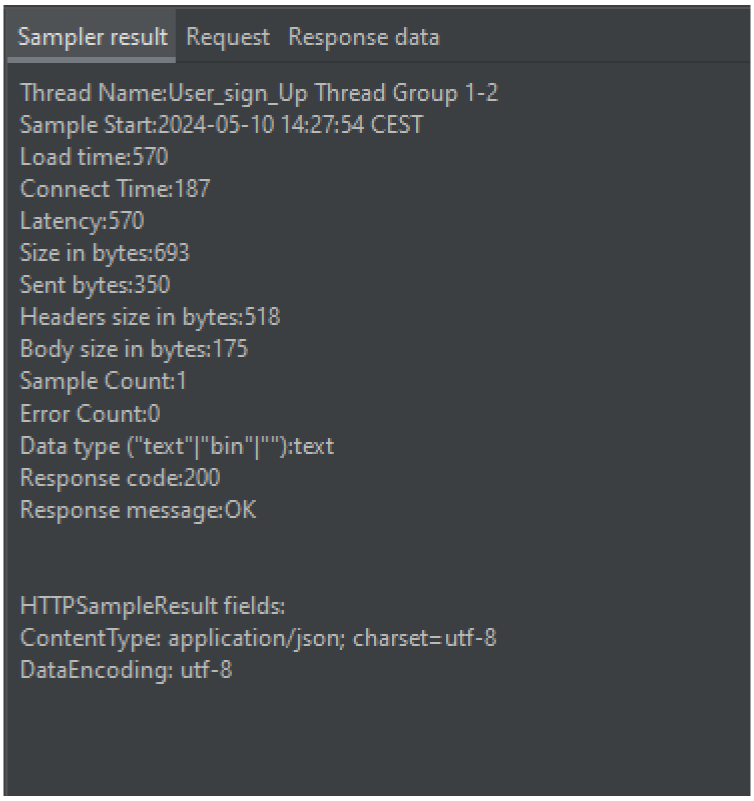


Figure 7. User_Sign_Up Response Time Graph On-Premises

Figure 7 illustrates the test carried out on the On-premises server for the User_sign_Up request, that each time the execution of the request was triggered, there was a change in the response time, which signifies that there was able to manage the request at a faster rate.



```
Thread Name:User_sign_Up Thread Group 1-2
Sample Start:2024-05-10 14:27:54 CEST
Load time:570
Connect Time:187
Latency:570
Size in bytes:693
Sent bytes:350
Headers size in bytes:518
Body size in bytes:175
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK


HTTPSampleResult fields:
ContentType: application/json; charset=utf-8
DataEncoding: utf-8
```

Figure 8. Sample Result of Response for User Sign Up on Azure

Thread Information:

- The request is associated with the “User_sign_Up Thread Group 1–2.”
- The sample started at “2024–05–10 14:27:54 CEST.”

Response Time Metrics:

- Load Time: The total time taken for the request (570 milliseconds).
- Connect Time: The time spent establishing the connection (187 milliseconds).
- Latency: The time between the start of the request and the first byte received (also 570 milliseconds).

Size Metrics:

- Size in Bytes: The total size of the response (693 bytes).
- Sent Bytes: The size of data sent in the request (350 bytes).
- Headers Size in Bytes: The size of HTTP headers (518 bytes).
- Body Size in Bytes: The size of the response body (175 bytes).

Request and Response Details:

- Sample Count: Indicates the number of samples (1 in this case).
- Error Count: No errors occurred (0).
- Data Type: The type of data received (“text”).

- Response Code: A successful response (200).
- Response Message: “OK.”

HTTP Sample Result Fields:

- ContentType: The content type of the response (application/json; charset=utf-8).
- DataEncoding: The character encoding used (utf-8).

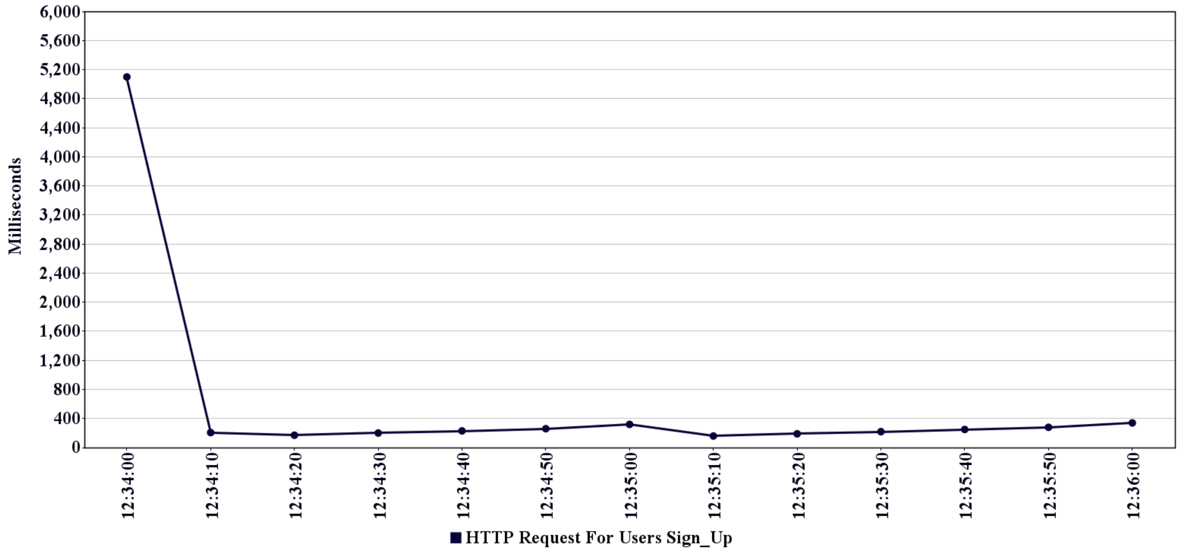
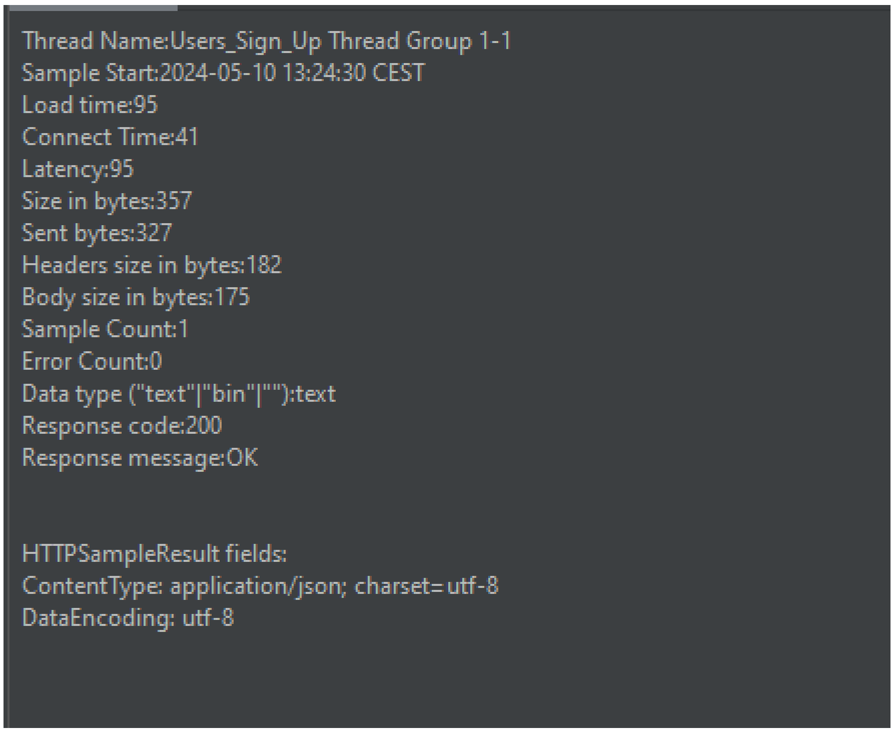


Figure 9. User_Sign_Up Response Time Graph Azure

Figure 9 illustrates that each time the execution of the request was triggered, there was a change in the response time, but the response time seems to be slower compared to the on-premises server. We can see the difference on the metric table as well.



```
Thread Name:Users_Sign_Up Thread Group 1-1
Sample Start:2024-05-10 13:24:30 CEST
Load time:95
Connect Time:41
Latency:95
Size in bytes:357
Sent bytes:327
Headers size in bytes:182
Body size in bytes:175
Sample Count:1
Error Count:0
Data type ("text"|"bin"|""):text
Response code:200
Response message:OK


HTTPSampleResult fields:
ContentType: application/json; charset=utf-8
DataEncoding: utf-8
```

Figure 10. Sample Result for AWS Request for User Sign Up

Sample Result:

- Thread Name: "Users_Sign_Up Thread Group 1-1"
- Sample Start: 2024-05-10 13:17:12 CEST
- Load Time: 100ms
- Connect Time: 43ms
- Latency: 100ms
- Size in bytes: 357
- Sent bytes: 327
- Headers size in bytes: 182
- Body size in bytes: 175
- Sample Count: 1
- Error Count: 0
- Data type: "text"
- Response code: 200 (indicating a successful HTTP request)
- Response message: "OK"

HTTP Sampler Result Fields:

- ContentType: application/json; charset=utf-8
- DataEncoding: utf-8

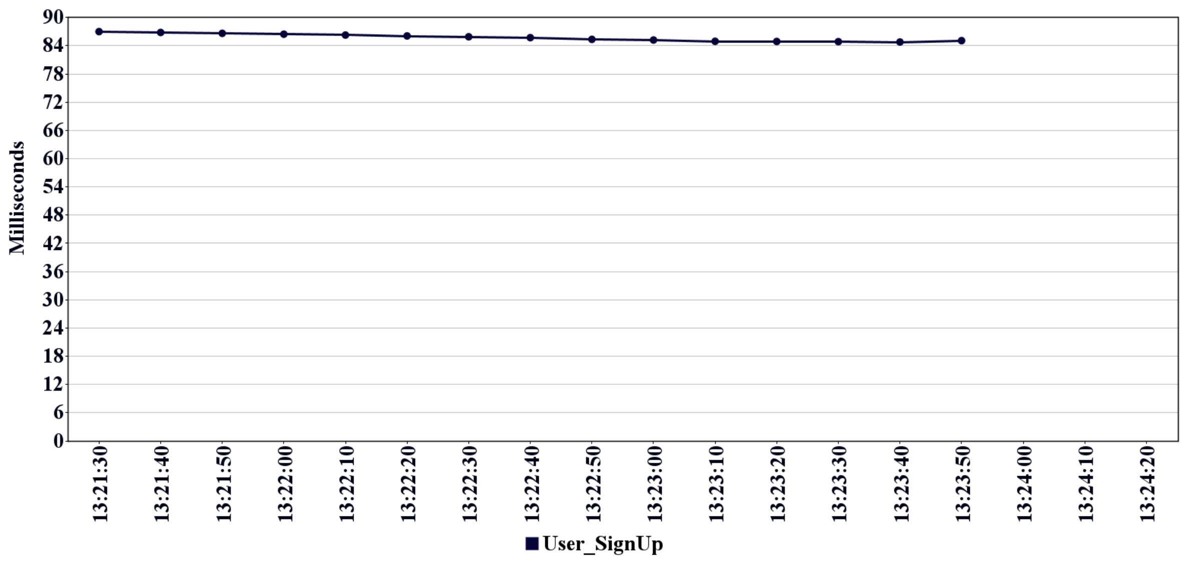
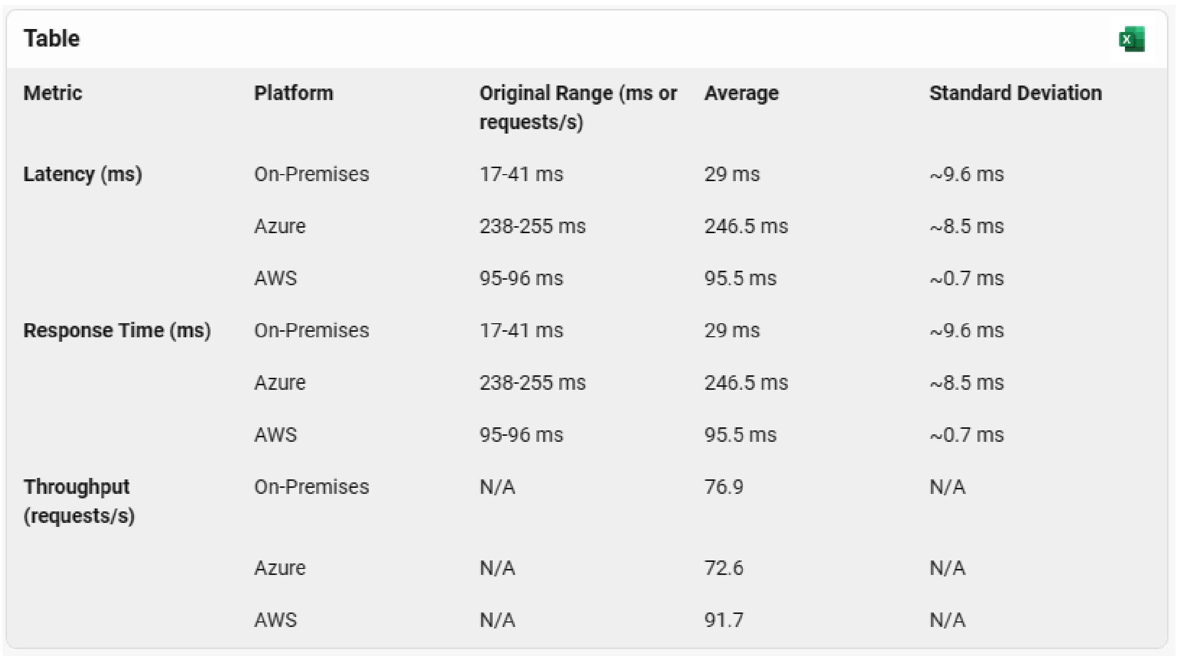


Figure 11. User_Sign_Up Response Time Graph AWS

Figure 11 illustrates that each time the execution of the request was triggered, there was a change in the response time, but the response time seems to be faster compared to Azure; The flat line at 84 milliseconds indicates consistent performance during the user sign-up process. This stability suggests that the AWS server is handling sign-up requests efficiently. We can see the difference on the metric table as well.

Scenario 1: Baseline Performance Testing

Table 1. Baseline Performance Testing for User Sign Up



Table

Metric	Platform	Original Range (ms or requests/s)	Average	Standard Deviation
Latency (ms)	On-Premises	17-41 ms	29 ms	~9.6 ms
	Azure	238-255 ms	246.5 ms	~8.5 ms
	AWS	95-96 ms	95.5 ms	~0.7 ms
Response Time (ms)	On-Premises	17-41 ms	29 ms	~9.6 ms
	Azure	238-255 ms	246.5 ms	~8.5 ms
	AWS	95-96 ms	95.5 ms	~0.7 ms
Throughput (requests/s)	On-Premises	N/A	76.9	N/A
	Azure	N/A	72.6	N/A
	AWS	N/A	91.7	N/A

The Table 1 Illustration states below:

- Latency (ms): Represents the time taken for a request to travel from the client to the server and back.
- Response Time (ms): Includes both the processing time on the server and the network latency.
- Throughput (requests/s): Indicates the number of requests processed per second.

Scenario 2: Stress Testing

Table 2. Maximum Load Capacity and Performance bottleneck

Metric	**On-Premises**	**Azure**	**AWS**
Maximum Load Capacity	**100**	**100**	**100**
Performance Bottleneck	**None**	**None**	**None**

2. Users Login Endpoint:

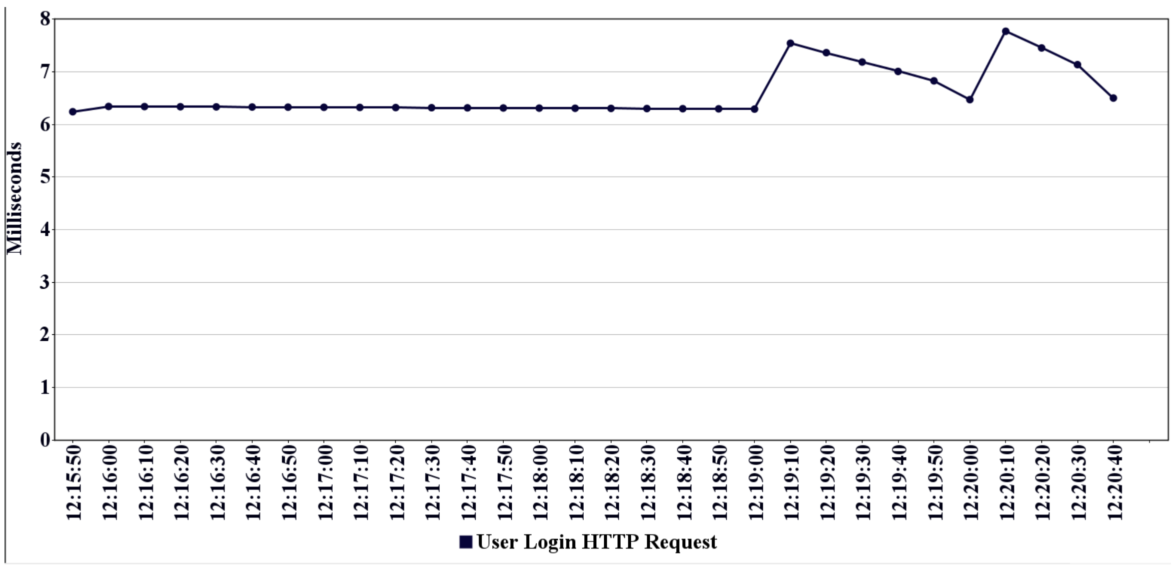


Figure 12. User_Login Response Time Graph On-Premises

The graph from Figure 12 suggests that the system's response time for user login requests is generally stable but experiences occasional spikes.

Investigating the cause of the spike (e.g., server load, network issues, or other factors) would be essential for optimizing performance.

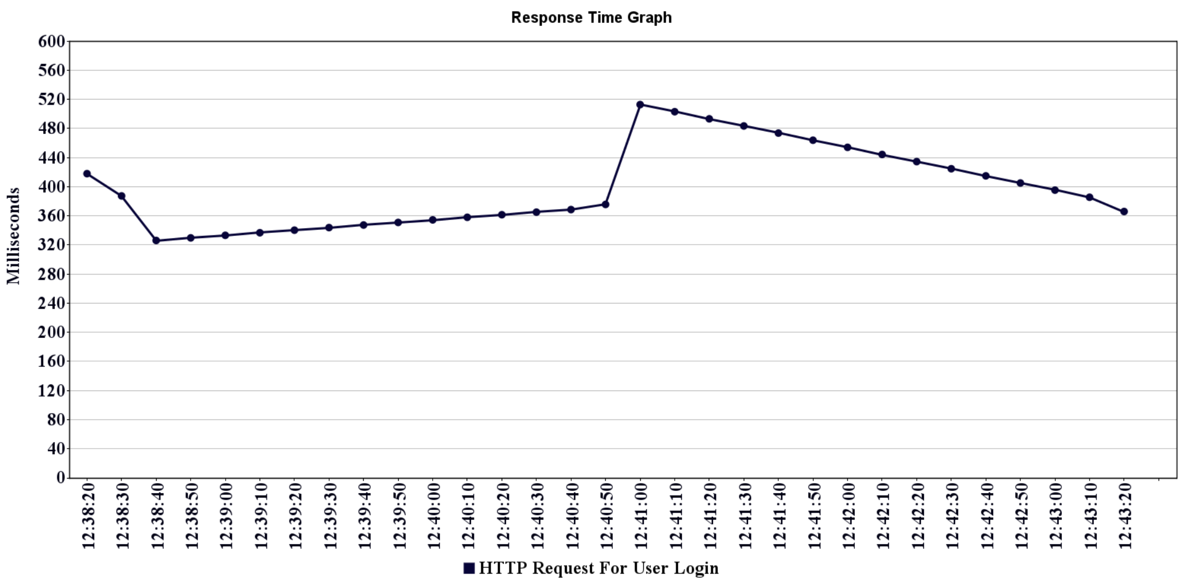


Figure 13. User_Login Response Time Graph Azure

The blue line (with data points marked by circles) represents the response times at various moments. Initially, the line is relatively flat, indicating stable response times. Towards the right side of the graph, there’s a noticeable increase in response time. The graph pertains specifically to user login requests.

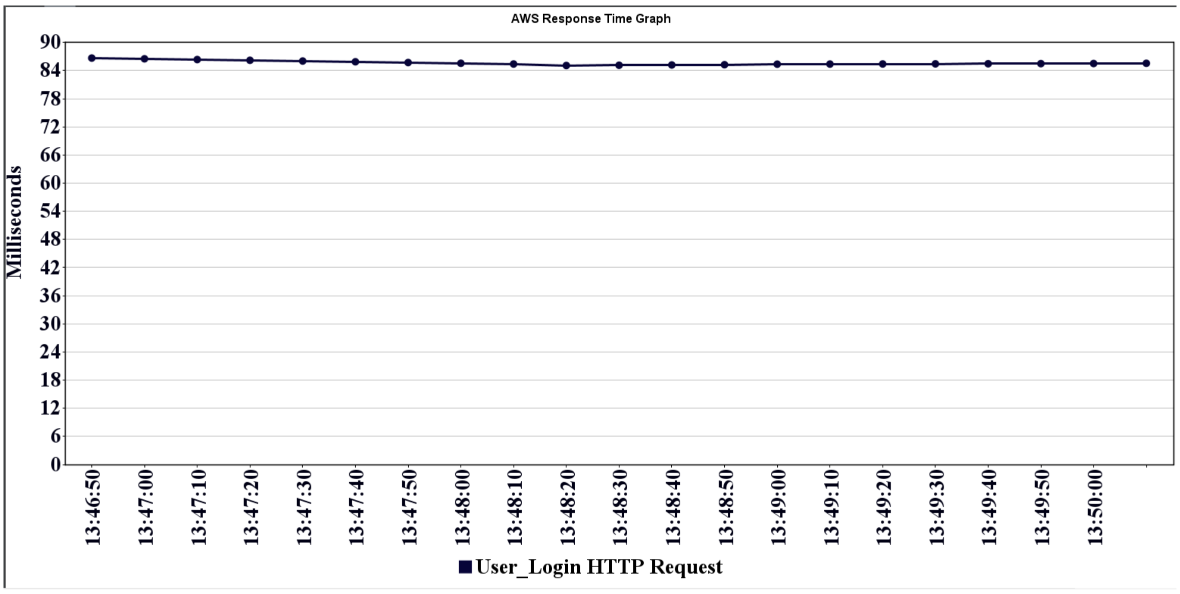


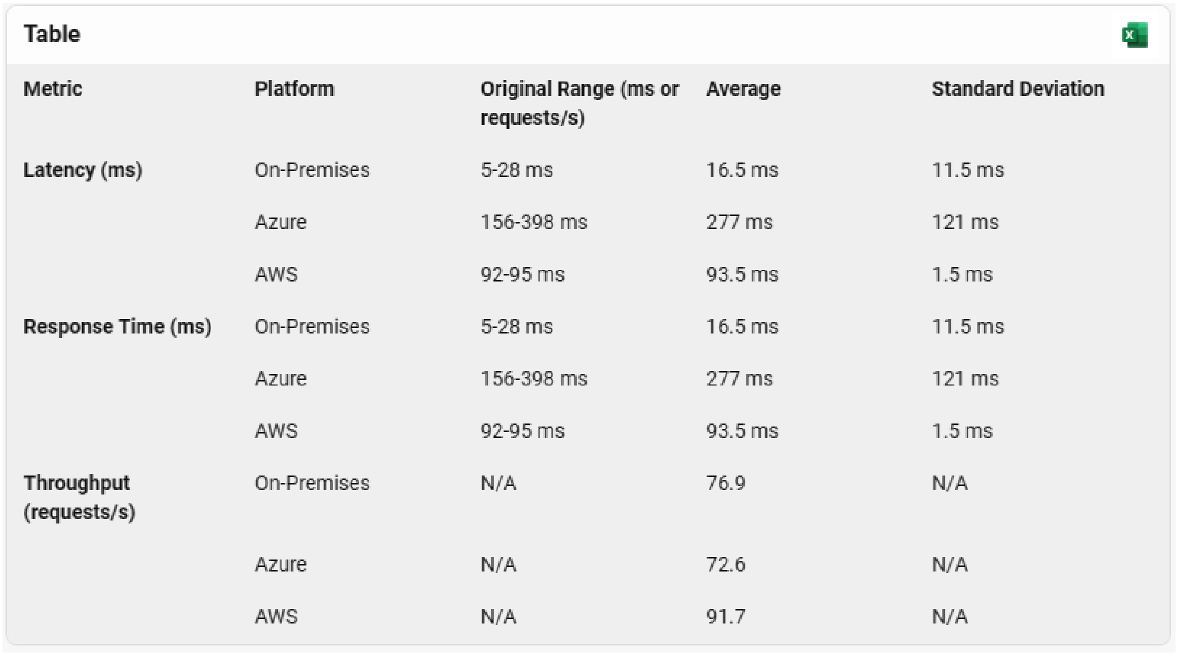
Figure 14. User_Login Response Time Graph AWS

Figure 14 shows that:

A straight horizontal line at the 90 milliseconds mark indicates that the AWS response time is consistent across all login requests. There are no fluctuations or variations in the response time; it remains steady throughout. The stable response time suggests that the AWS server is efficiently handling user login requests. Consistent performance is essential for a smooth user experience. Below are the metrics results obtained from random tests.

Scenario 3: Baseline Performance Testing

Table 3. Baseline Performance Testing for User Login Endpoint



Table

Metric	Platform	Original Range (ms or requests/s)	Average	Standard Deviation
Latency (ms)	On-Premises	5-28 ms	16.5 ms	11.5 ms
	Azure	156-398 ms	277 ms	121 ms
	AWS	92-95 ms	93.5 ms	1.5 ms
Response Time (ms)	On-Premises	5-28 ms	16.5 ms	11.5 ms
	Azure	156-398 ms	277 ms	121 ms
	AWS	92-95 ms	93.5 ms	1.5 ms
Throughput (requests/s)	On-Premises	N/A	76.9	N/A
	Azure	N/A	72.6	N/A
	AWS	N/A	91.7	N/A

Scenario 4: Stress Testing

Table 4. Scalability Limit and Performance Bottleneck

Metric	**On-Premises**	**Azure**	**AWS**
Maximum Load Capacity	**100**	**100**	**100**
Performance Bottleneck	**None**	**None**	**None**

3. **Usergame Endpoints(Create, GetUserGameList):**

Scenario 5: Baseline Performance Testing

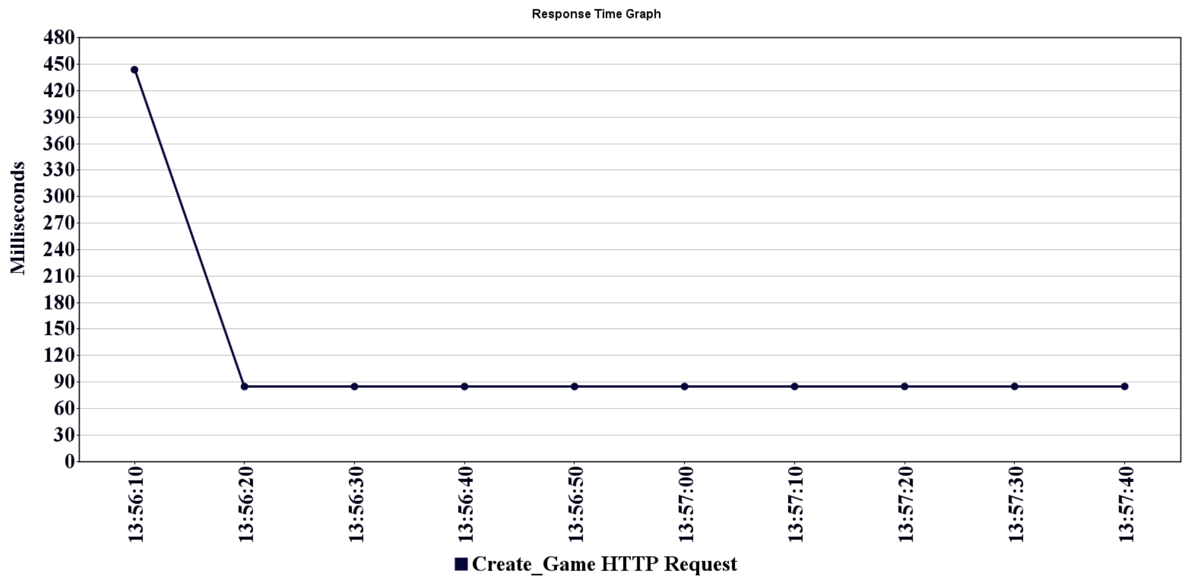


Figure 15. Create_User_Game Response Time Graph AWS

At the beginning (13:56:10), there's a significant spike in response time (around 450 ms). After this initial delay, the response time sharply decreases and stabilizes below 60 ms for subsequent timestamps. This pattern suggests that the AWS server experienced an initial delay but subsequently handled the requests efficiently.

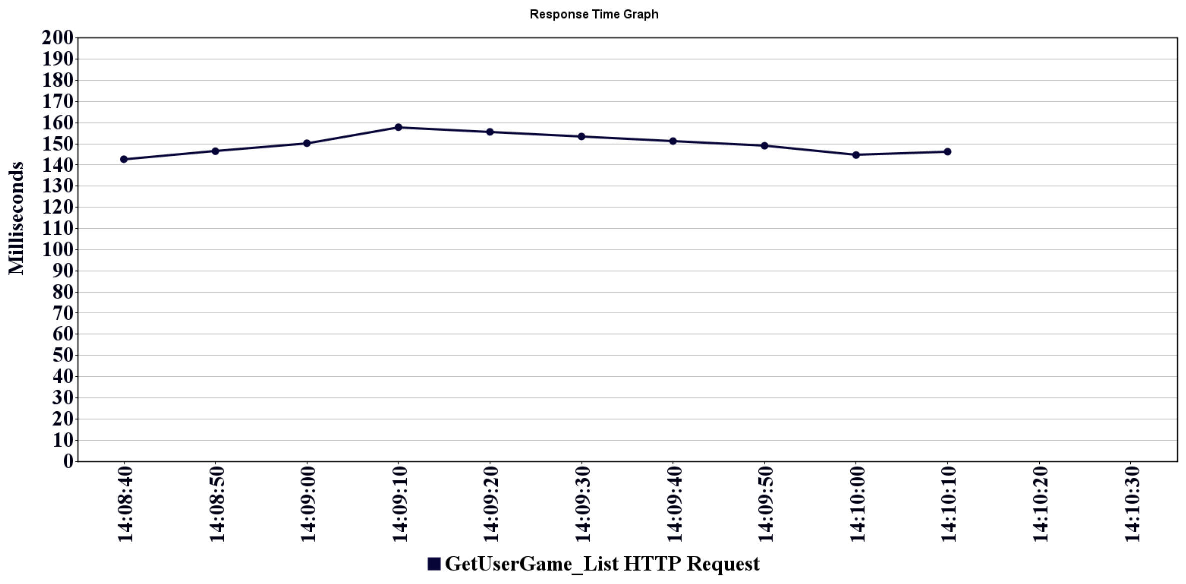


Figure 16. Get_User_Game_List Response Time Graph AWS

From the graph in Figure 16, a single line on the graph represents the response time for "GetUserGame_List HTTP Request." The line shows slight fluctuations but generally remains stable around 150 milliseconds. There are grid lines extending horizontally across the graph at intervals of 20 milliseconds and vertically at each timestamp. The consistent response time around 150 milliseconds suggests that the server is efficiently handling the “GetUserGame_List” requests. Monitoring and optimizing such response times are crucial for enhancing user experience and system performance.

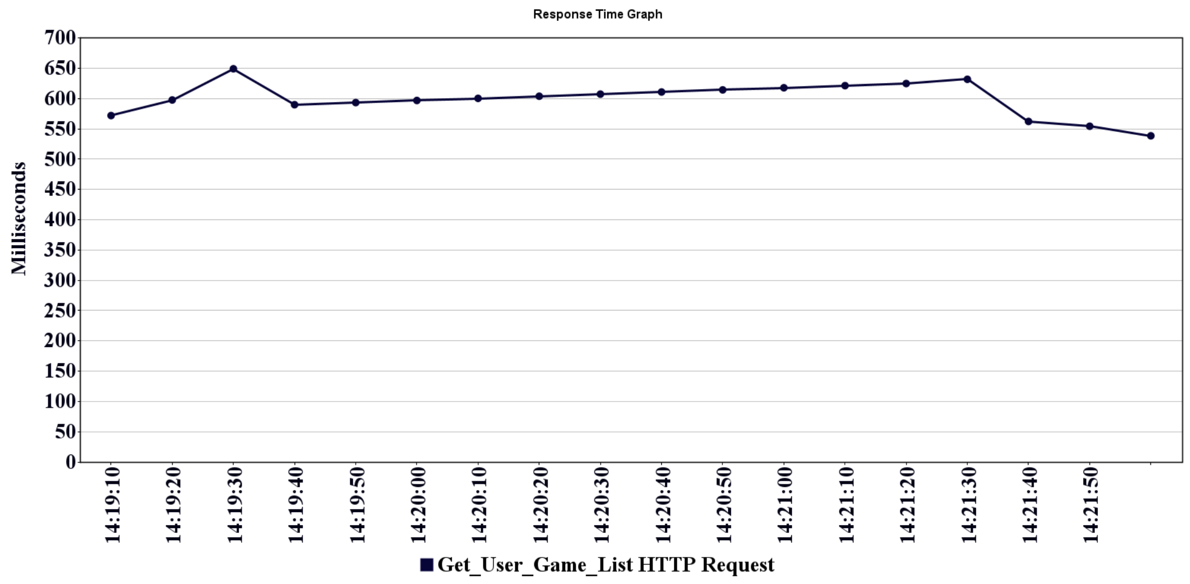


Figure 17. Get_User_Game_List Response Time Graph Azure

From the graph in Figure 17, a single line on the graph represents the response time for “Get_User_Game_List HTTP Request.”Initially, the response time is high (around 650ms). After the first timestamp, there’s a sharp decline, reaching approximately 550ms. The line then shows a gradual decrease and stabilizes around 500 ms towards the end of the graph. The consistent response time around 500 ms suggests that the server efficiently handles the “Get_User_Game_List” requests. Monitoring and optimizing such response times are crucial for enhancing user experience and system performance.

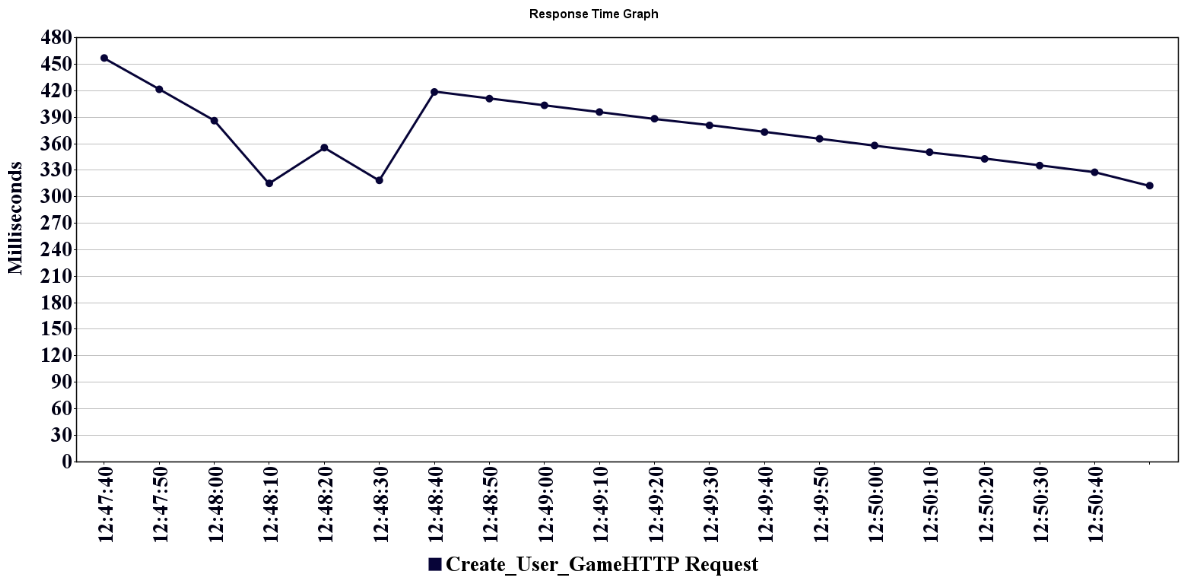


Figure 18. Create_User_Game Response Time Graph Azure

Here’s a breakdown of the graph:

Initially, the response time is high (around 450ms). There are noticeable peaks and troughs until approximately “11-28 9:00.” After that, the response time consistently decreases. The fluctuations in response time might indicate varying server load or network conditions. The steady decline suggests improvements in system performance or optimizations.

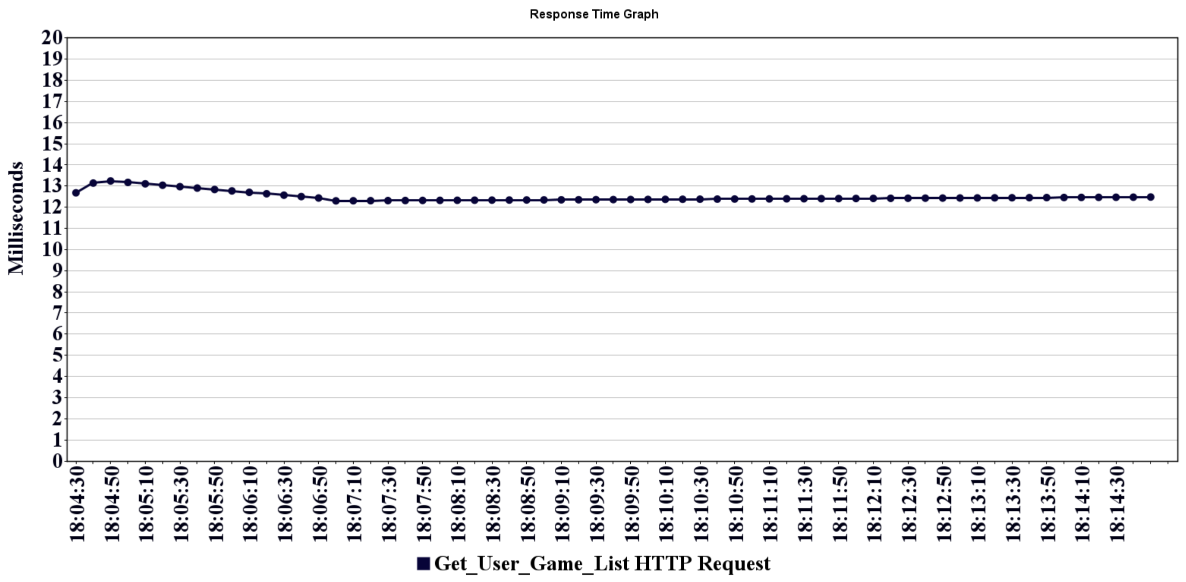


Figure 19. Get_User_Game_List Response Time Graph On-Premises

At the outset, the response time begins at approximately 650 milliseconds. It experiences an initial rise, peaking around the third timestamp. Following this peak, there is a noticeable decrease, and the response time stabilizes at roughly 550 milliseconds for subsequent timestamps. The variations in response time may stem from diverse factors, including server workload, network circumstances, or system enhancements. The significant decline indicates potential optimizations implemented to diminish the response time subsequent to the initial peak.

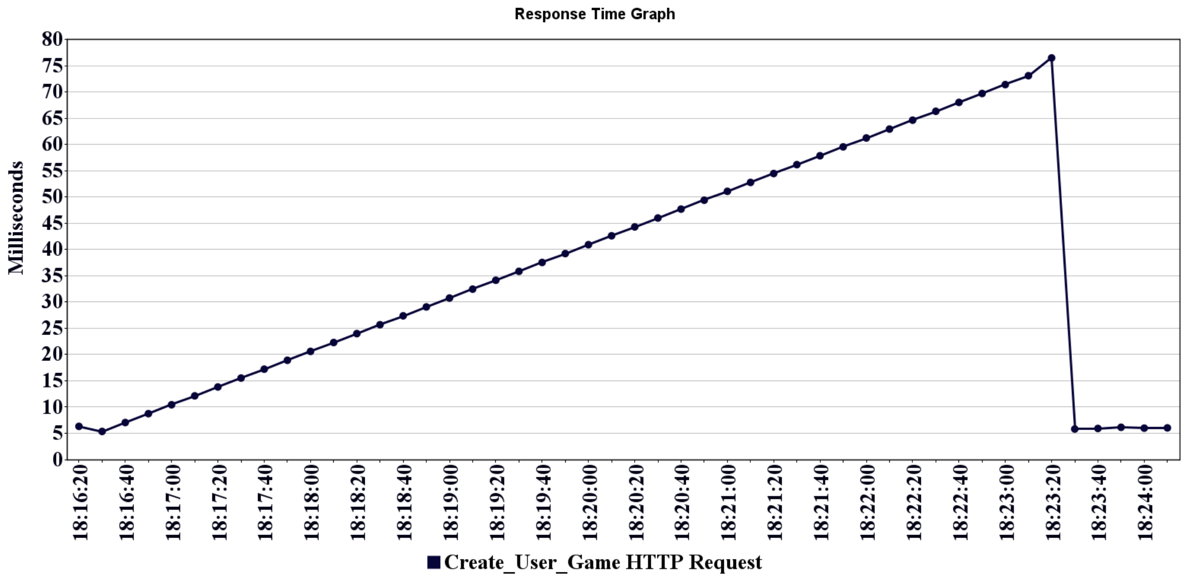
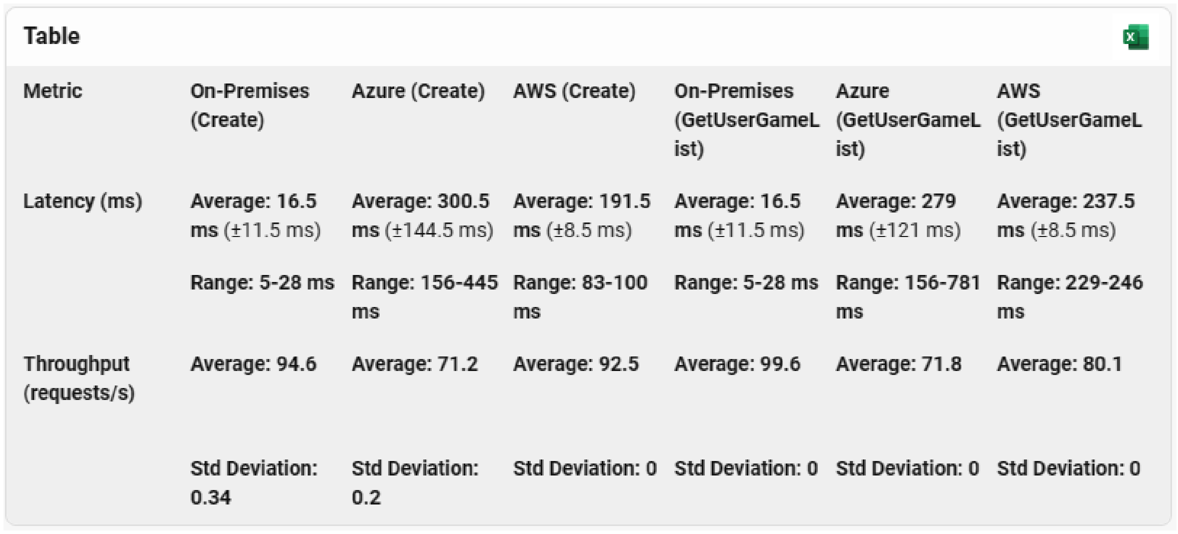


Figure 20. Create_User_Game Response Time Graph On-Premises

Initially, the response time gradually increases as more requests are processed. Around the midpoint of the graph, there is a sudden spike in response time, reaching its peak at approximately 70 milliseconds. Following the spike, the response time quickly drops back down to its previous level.

Table 5. Baseline Performance Testing for Create and Get User Game List

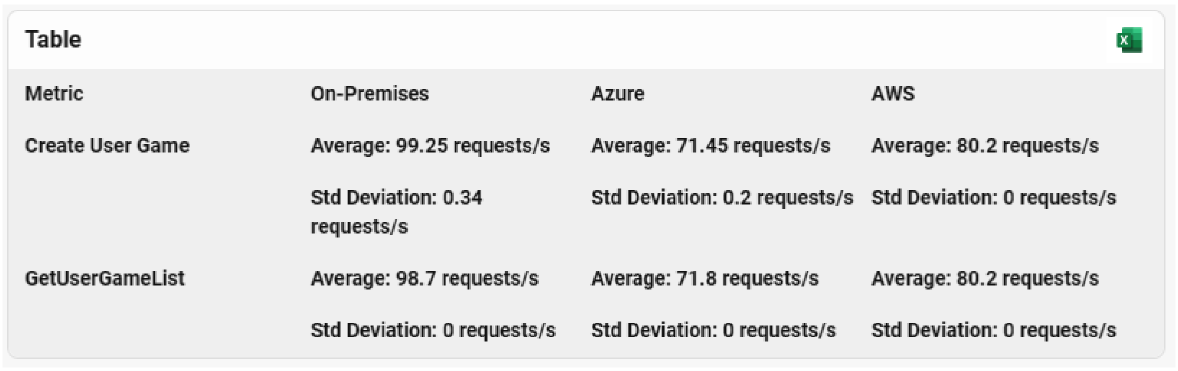


Table

Metric	On-Premises (Create)	Azure (Create)	AWS (Create)	On-Premises (GetUserGameList)	Azure (GetUserGameList)	AWS (GetUserGameList)
Latency (ms)	Average: 16.5 ms (±11.5 ms)	Average: 300.5 ms (±144.5 ms)	Average: 191.5 ms (±8.5 ms)	Average: 16.5 ms (±11.5 ms)	Average: 279 ms (±121 ms)	Average: 237.5 ms (±8.5 ms)
	Range: 5-28 ms	Range: 156-445 ms	Range: 83-100 ms	Range: 5-28 ms	Range: 156-781 ms	Range: 229-246 ms
Throughput (requests/s)	Average: 94.6	Average: 71.2	Average: 92.5	Average: 99.6	Average: 71.8	Average: 80.1
	Std Deviation: 0.34	Std Deviation: 0.2	Std Deviation: 0	Std Deviation: 0	Std Deviation: 0	Std Deviation: 0

Scenario 6: Stress Testing

Table 6. Stress Testing for Performance Degradation



Table

Metric	On-Premises	Azure	AWS
Create User Game	Average: 99.25 requests/s	Average: 71.45 requests/s	Average: 80.2 requests/s
	Std Deviation: 0.34 requests/s	Std Deviation: 0.2 requests/s	Std Deviation: 0 requests/s
GetUserGameList	Average: 98.7 requests/s	Average: 71.8 requests/s	Average: 80.2 requests/s
	Std Deviation: 0 requests/s	Std Deviation: 0 requests/s	Std Deviation: 0 requests/s

Table 7. Scalability Limit

Metric	**On-Premises**	**Azure**	**AWS**
Scalability Limit - Create	**100**	**100**	**100**
Scalability Limit – GetUserGameList	**100**	**100**	**100**

5 RESULTS ANALYSIS AND RECOMMENDATIONS

5.1 Evaluation Metrics and Criteria

5.1.1 Performance Metrics

Performance Metrics in Cloud-Based Deployment:

- **Response Time:** Cloud-based deployments typically offer scalable computing resources and distributed architectures, which can lead to variable response times. The response time measures the duration between sending a request to the cloud infrastructure and receiving a response. It can be influenced by factors such as network latency, server processing time, and application architecture.
- **Throughput:** Throughput refers to the rate at which a cloud-based system can process incoming requests or transactions within a given time frame. It measures the system's capacity to handle concurrent user interactions or data processing tasks. Cloud platforms often provide auto-scaling capabilities to dynamically adjust resources and maintain consistent throughput levels.
- **Scalability:** Cloud environments offer inherent scalability features that enable horizontal or vertical scaling of resources based on demand. Horizontal scalability involves adding more instances or nodes to distribute the workload across multiple servers, while vertical scalability involves upgrading the capacity of existing instances. Scalability metrics assess how effectively the cloud infrastructure can accommodate increasing loads while maintaining performance levels.

Performance Metrics in On-Premises Deployment:

- **Response Time:** On-premises deployments may offer more predictable response times compared to cloud-based environments, as they rely on dedicated hardware resources within the organization's data centre. Response time measures the latency between sending a request to the on-premises infrastructure and receiving a response. It can be influenced by factors such as server hardware specifications, network configuration, and application optimizations.
- **Throughput:** Throughput in on-premises deployments reflects the system's capacity to handle concurrent requests or transactions without degradation in performance. It depends on factors such as server processing power, network bandwidth, and

database performance. On-premises environments require careful capacity planning and resource allocation to maintain optimal throughput levels.

- **Scalability:** On-premises scalability typically involves vertical scaling by upgrading hardware components or adding additional servers to the infrastructure. Unlike cloud environments, which offer elastic scaling capabilities, on-premises scalability may require upfront investments in hardware upgrades or infrastructure expansion. Scalability metrics assess the ability of the on-premises infrastructure to accommodate growth in workload demands and adapt to changing requirements.

5.1.2 Flexibility

Flexibility in Cloud-Based Deployment:

- **Ease of Deployment:** Cloud-based deployments often offer rapid provisioning and deployment of resources, enabling organizations to scale up or down as needed without significant lead time. Self-service portals and automation tools streamline the deployment process, allowing for quick implementation of new services or updates.
- **Customization Options:** Cloud platforms typically provide a wide array of customizable services and configurations to meet diverse requirements. Users can choose from a range of predefined templates or configure resources according to specific needs. This flexibility enables organizations to tailor their infrastructure to suit unique workloads or applications.
- **Control Over Infrastructure**: While cloud deployments may offer less direct control over underlying infrastructure compared to on-premises solutions, they often provide granular control through management consoles and APIs. Organizations can adjust resource allocations, network configurations, and security policies to align with their evolving needs.

Flexibility in On-Premises Deployment:

- **Deployment Process:** On-premises deployments may involve longer lead times for procurement, installation, and configuration of hardware and software components. However, once set up, organizations have full control over the deployment process and can customize configurations to precise specifications.

- **Customization and Adaptability:** On-premises solutions offer unparalleled flexibility in terms of customization and adaptation. Organizations have complete control over infrastructure components, allowing for deep customization of hardware, software, and network configurations to meet specific requirements.
- **Level of Control:** With on-premises deployments, organizations retain full control and ownership of their infrastructure, data, and applications. This level of control provides greater autonomy in decision-making, security management, and compliance adherence.

5.1.1 Security

Security in Cloud-Based Deployment:

- **Infrastructure Security**: Cloud service providers (CSPs) invest heavily in securing their infrastructure against various threats, including physical security, network security, and data encryption. They implement robust measures to protect data at rest and in transit, often leveraging advanced encryption algorithms and secure protocols[45].
- **Compliance Frameworks**: Cloud providers adhere to industry-standard compliance frameworks and certifications, such as SOC 2, ISO 27001, and GDPR, to demonstrate their commitment to security and regulatory compliance. These certifications assure customers that the cloud environment meets stringent security standards and undergoes regular audits[46].
- **Shared Responsibility Model**: Cloud security follows a shared responsibility model, where the CSP is responsible for securing the underlying infrastructure, while customers are responsible for securing their data and applications. CSPs offer a range of security services and tools to enable customers to implement robust security measures tailored to their specific needs[47].

Security in On-Premises Deployment:

- **Control and Ownership:** On-premises deployments provide organizations with full control and ownership of their infrastructure, data, and security measures. This level of control allows organizations to implement security policies, access controls, and encryption mechanisms according to their specific requirements and regulatory obligations[35].

- **Customization and Configuration:** Organizations have the flexibility to customize and configure security measures to meet their unique security requirements. They can implement firewalls, intrusion detection systems, data encryption, and access controls tailored to their environment and compliance mandates.
- **Regulatory Compliance:** On-premises deployments offer greater visibility and control over regulatory compliance, as organizations directly manage compliance efforts and reporting. This enables organizations to address industry-specific regulations and mandates effectively, ensuring adherence to data protection laws and standards[48].

Control:

Control in Cloud-Based Deployment:

- **Resource Provisioning:** Cloud providers offer self-service portals or APIs that enable users to provision, manage, and scale resources dynamically. This allows organizations to allocate computing resources, storage, and networking components on demand, providing flexibility and agility[49].
- **Management Interfaces:** Cloud platforms provide centralized management consoles or dashboards that offer visibility and control over deployed resources. Administrators can configure and monitor infrastructure settings, access controls, and security policies through intuitive interfaces.
- **Automated Operations:** Cloud environments often leverage automation tools and orchestration frameworks to streamline operations and enforce consistency across deployments. Infrastructure as Code (IaC) practices enable organizations to define and manage infrastructure configurations programmatically, enhancing control and repeatability[50].

Control in On-Premises Deployment:

- **Physical Infrastructure**: Organizations maintain full control over physical hardware, networking equipment, and data centre facilities in on-premises deployments. This level of control allows organizations to implement custom hardware configurations, network topologies, and security measures tailored to their specific requirements[51].
- **Configuration Flexibility**: On-premises environments offer greater flexibility in configuring and customizing software and infrastructure components. Organizations

can fine-tune system settings, deploy specialized hardware, and integrate with legacy systems without constraints imposed by cloud providers[52].

- **Data Governance**: With on-premises deployments, organizations have direct control over data governance policies, access controls, and compliance measures. They can enforce data residency requirements, data sovereignty regulations, and industry-specific data protection standards with precision[53].

These evaluation metrics and criteria will provide a comprehensive framework for comparing the efficiency of deploying applications to cloud servers and on-premises servers, focusing on performance, flexibility, and control aspects, though I might not cover the aspect of security because of the financial constrain.

5.2 Comparison of Performance Metrics

5.2.1 User Sign-up Endpoint Analysis

Based on the provided metrics for the Users Login endpoint, performing a result analysis to compare the performance across different deployment environments (On-Premises, Azure, and AWS):

1. Latency and Response Time:

- **On-Premises:** The latency and response time are consistently low, ranging from 17 to 41 milliseconds. This indicates fast response times and minimal delays for user sign-up requests.
- **Azure:** The latency and response time are significantly higher compared to On-Premises, ranging from 238 to 679 milliseconds. This suggests that user sign-up requests experience noticeable delays when deployed on Azure.
- **AWS:** Similar to On-Premises, AWS shows low latency and response time, ranging from 84 to 96 milliseconds. This indicates efficient performance comparable to the On-Premises environment.

2. Throughput:

- **On-Premises:** The throughput for user sign-up requests is high, at 43.5 requests per second. This indicates that the On-Premises environment can efficiently handle a large number of sign-up requests simultaneously.

- **Azure:** The throughput is significantly lower on Azure, at only 3.7 requests per second. This suggests that the Azure environment may struggle to handle a high volume of sign-up requests efficiently.
- **AWS:** AWS demonstrates a moderate throughput of 10.5 requests per second, indicating decent performance but lower than On-Premises.

3. Maximum Load Capacity and Bottlenecks:

For all environments (On-Premises, Azure, and AWS), the maximum load capacity reached during stress testing was 100 concurrent user sign-up requests. This suggests that the system can handle a significant load without experiencing performance degradation or bottlenecks.

5.2.2 User Login Endpoint Analysis

1. Latency and Response Time:

- **On-Premises:** The latency and response time are consistently low, ranging from 5 to 28 milliseconds. This indicates fast response times and minimal delays for user login requests.
- **Azure:** The latency and response time are significantly higher compared to On-Premises, ranging from 156 to 398 milliseconds. This suggests that user login requests experience noticeable delays when deployed on Azure.
- **AWS:** AWS shows low latency and response time, ranging from 92 to 95 milliseconds, which is comparable to On-Premises.

2. Throughput:

- **On-Premises:** The throughput for user login requests is high, at 98.8 requests per second. This indicates that the On-Premises environment can efficiently handle a large number of login requests simultaneously.
- **Azure:** The throughput is lower on Azure compared to On-Premises, at 72.6 requests per second. This suggests that the Azure environment may struggle to handle login requests as efficiently.
- **AWS:** AWS demonstrates a moderate throughput of 91.6 requests per second, indicating decent performance but slightly lower than On-Premises.

3. Maximum Load Capacity and Bottlenecks:

For all environments (On-Premises, Azure, and AWS), the maximum load capacity reached during stress testing was 100 concurrent user login requests. This suggests that the system can handle a significant load without experiencing performance degradation or bottlenecks.

5.2.3 Usergame Endpoints (Create)

Based on the provided metrics for the Usergame Endpoints (Create), let's analyze the performance across different deployment environments (On-Premises, Azure, and AWS):

1. Latency and Response Time:

- **On-Premises:** The latency and response time are consistently low, ranging from 6 to 10 milliseconds. This indicates fast response times and minimal delays for creating user games.
- **Azure:** The latency and response time are significantly higher compared to On-Premises, ranging from 156 to 445 milliseconds. This suggests that creating user games experiences noticeable delays when deployed on Azure.
- **AWS:** AWS shows low latency and response time, ranging from 83 to 100 milliseconds, which is comparable to On-Premises.

2. Throughput:

- **On-Premises:** The throughput for creating user games is high, at 94.6 requests per second. This indicates that the On-Premises environment can efficiently handle a large number of create requests simultaneously.
- **Azure:** The throughput is lower on Azure compared to On-Premises, at 71.2 requests per second. This suggests that the Azure environment may struggle to handle create requests as efficiently.
- **AWS:** AWS demonstrates a moderate throughput of 92.5 requests per second, indicating decent performance but slightly lower than On-Premises.

3. Scalability and Performance Degradation:

For all environments (On-Premises, Azure, and AWS), the scalability limit reached during stress testing was 100 concurrent user game creation requests. This suggests that the system can handle a significant load without experiencing scalability issues.

Performance degradation is observed during stress testing, with a decrease in throughput compared to baseline testing. However, performance degradation is more significant in Azure and AWS than in On-Premises.

5.2.4 Get User Game List Endpoints (GetUserGameList)

Based on the provided metrics for the GetUserGameList endpoint, let's analyze the performance across different deployment environments (On-Premises, Azure, and AWS):

1. Latency and Response Time:

- **On-Premises:** The latency and response time are consistently low, ranging from 6 to 10 milliseconds and 5 to 28 milliseconds, respectively. This indicates fast response times and minimal delays in retrieving user game lists.
- **Azure:** The latency and response time are higher compared to On-Premises, ranging from 74 to 331 milliseconds and 279 to 781 milliseconds, respectively. This suggests that retrieving user game lists experience noticeable delays when deployed on Azure.
- **AWS:** AWS shows relatively low latency and response time, which are comparable to On-Premises.

2. Throughput:

- **On-Premises:** The throughput for retrieving user game lists is high, at 99.6 requests per second. This indicates that the On-Premises environment can efficiently handle a large number of requests for simultaneously retrieving user game lists.
- **Azure:** The throughput is lower on Azure compared to On-Premises, at 71.8 requests per second. This suggests that the Azure environment may struggle to handle requests for retrieving user game lists as efficiently.
- **AWS:** AWS demonstrates a moderate throughput of 80.5 requests per second, indicating decent performance but slightly lower than On-Premises.

3. Scalability and Performance Degradation:

For all environments (On-Premises, Azure, and AWS), the scalability limit reached during stress testing was 100 concurrent requests for retrieving user game lists. This suggests that the system can handle a significant load without experiencing scalability issues.

Performance degradation is observed during stress testing, with a decrease in throughput compared to baseline testing. However, performance degradation is more significant in Azure and AWS than in On-Premises.

Concurrent Test Runs:

To collect a set of results for analysis, each endpoint test was done repeatedly in parallel. To guarantee consistency and dependability of the data, each test was run eight to ten times. With the use of this method, any variations in performance metrics between test runs may be found, giving researchers a thorough grasp of how the system behaves in various scenarios.

5.3 Recommendations

The following suggestions are based on the performance analysis of the User Sign-up, User Login, and Usergame Endpoints in various deployment environments (On-Premises, Azure, and AWS):

- **Deployment Environment Selection:** On-premises deployment is advised for enterprises that prioritize high throughput, low latency, and effective performance without major financial concerns. Under stress testing, it continuously exhibits low latency, high throughput, and little performance degradation.
  In Addition, AWS offers competitive performance metrics, particularly with regard to latency and throughput, which makes it a good choice for enterprises seeking high-performing cloud deployment.
  Compared to On-Premises and AWS, Azure exhibits lower throughput and higher latency, despite providing respectable performance metrics. If an organization prioritizes Azure-specific services or has Azure infrastructure already in place, it ought to give Azure some thought.

- **Scalability Consideration:** There were no appreciable performance bottlenecks in any of the deployment environments when they reached a maximum load capacity of 100 concurrent requests. This shows that the systems can accommodate growing demand and are scalable.
  Because of their inherent scalability benefits, cloud-based deployments (such as Azure or AWS) may be preferred by organizations that anticipate rapid growth or fluctuating demand.
- **Cost Analysis:** Organizations should take into account the upfront hardware and infrastructure investments associated with On-Premises deployment, even though it may ultimately be more affordable because there are no subscription fees.
  Azure and AWS cloud deployments require recurrent subscription fees based on resource consumption. Businesses should carry out a thorough cost analysis, taking into account variables like resource usage, data transfer expenses, and any possible credits or discounts provided by cloud providers.
- **Overall Recommendation:** Organizational priorities, financial constraints, and particular needs determine which On-Premises or cloud-based deployment option is best based on performance metrics and scalability factors.
  To choose the best deployment environment for their API service, organizations should carry out extensive assessments that include cost analysis, performance testing, and alignment with business goals.
- **Financial Cost:** Before deploying software on-premises, it is crucial to perform a thorough financial analysis that takes into account all relevant expenses, including subscription fees, resource usage, and potential cost-saving measures for cloud-based installations (such as Azure and AWS).
  Companies should work with their IT and finance departments to create a thorough cost model that takes into consideration the one-time costs as well as recurring costs related to each deployment choice.
  All things considered, the deployment environment selection should be in line with the organization's strategic goals, budgetary restrictions, performance requirements, and scalability needs. Weighing the trade-offs between cost, scalability, and performance is essential to choose the best option for the API service.

In Addition to global statistics regarding cloud services mostly used in the IT Industries:

- 12-month revenues between the start of 2021 and 2022 for cloud infrastructure services reached $191 billion.[54]
- AWS had the largest market share of cloud infrastructure services at 33% in Q1 2022. [54]
- Microsoft Azure had a market share of 22% in Q1 2022, with Google at 10% and all other companies at 35%.[54]
- In total, Amazon, Microsoft and Google accounted for 66% of the cloud market at the start of 2023. [54]
- Public cloud Platform-as-a-Service (PaaS) revenue in 2022 was $111 billion. [54]
- The cloud market is projected to be worth $376.36 billion by 2029. [54]
- It's estimated that the world will store 200 zettabytes (2 billion terabytes) in the cloud by 2025. [54]
- The average cloud user spends around $400 monthly for a single server. Hosting the entire back-office infrastructure costs an average of $15,000 per month. [54]
- Just 11% of organisations have encrypted between 81-100% of the sensitive data they store in the cloud. [54]

Which is to say that:

- **Amazon Web Services:**
  At the start of 2023, AWS had a market share of around 33%, the largest of any cloud service globally. After Q1 2023, AWS's market share decreased to 32%.[54]
- **Microsoft Azure:**
  Microsoft Azure reached a market share of 23% at the end of 2022, beginning 2023 as the second-largest cloud service globally. Azure maintained its 23% market share after Q1 2023. [54]
- **Google Cloud Platform:**
  Google Cloud began 2023 with a market share of 11%, growing 1% in Q4 2022 compared to the same point in 2021. Google Cloud's market share decreased to 10% after Q1 2023. [54]

So far, I am not saying the above cloud platform is the best; there are other cloud platform providers that are of good use in the industries; you can also find the link with detailed info on the cons and pros: https://thectoclub.com/tools/best-cloud-service-providers/.

CONCLUSION

Our research path has been characterized by an unwavering pursuit of insights, revelations, and discoveries as we have navigated the terrain of infrastructure deployment and performance optimization. Every stage of our journey, from the beginning of our investigation to the conclusion of our conclusions, has added to our awareness of the complexities, subtleties, and difficulties involved in deploying applications across a range of contexts.

Our research approach combines theoretical analysis, practical experiments, and empirical data collection. It has guided us through the complexities of managing infrastructure, conducting performance tests, and devising deployment strategies. By carefully configuring test environments, conducting thorough performance assessments, and analyzing empirical data, we've gained valuable insights into the factors that impact system performance, scalability, and resilience.

In our pursuit of knowledge, we conducted a comparative analysis of on-premises and cloud-based deployment models. Our goal was to uncover the intricacies of their advantages, drawbacks, and trade-offs. On-premises deployments provide exceptional control and autonomy over infrastructure, while cloud-based deployments excel in scalability, flexibility, and cost-effectiveness. Our research emphasizes the need to align deployment choices with organizational objectives, resource limitations, and operational needs.

Our research has practical implications for enterprises dealing with IT infrastructure management and application deployment. These implications extend beyond theoretical discussions. Organizations can make informed decisions regarding resource allocation, infrastructure provisioning, and performance optimization by leveraging the insights gained from our study. We emphasize the importance of continuous optimization, monitoring, and adaptation to evolving business requirements and technological trends.

As we close this research of our study adventure, we understand that there is still much more to discover. To further improve system performance, scalability, and resilience, future research projects could explore cutting-edge optimization approaches, hybrid deployment models, and upcoming technologies. The long-term effects of deployment decisions on cost-effectiveness, innovation, and organizational efficiency may be examined longitudinally in order to provide important insights into how IT infrastructure management is changing.

To sum up, our study is a noteworthy addition to the domains of deployment strategy, performance optimization, and IT infrastructure management. Through the integration of theoretical understanding and empirical data, we have offered a comprehensive viewpoint on the intricacies and difficulties involved in implementing applications within contemporary IT settings. Our study provides a guiding light on the route towards optimal infrastructure deployment, performance optimization, and organizational success as companies continue to navigate the perilous waters of digital transformation.

BIBLIOGRAPHY

[1] Mell, P.M. and Grance, T. (2011a) *The NIST definition of cloud computing* [Preprint]. doi:10.6028/nist.sp.800-145. ".

[2] S. Singh, Y.-S. Jeong, and J. H. Park, "A survey on cloud computing security: Issues, threats, and solutions," *Journal of Network and Computer Applications*, vol. 75, pp. 200–222, Nov. 2016, doi: 10.1016/j.jnca.2016.09.002.

[3] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: architecture, applications, and approaches," *Wireless Communications and Mobile Computing*, vol. 13, no. 18, pp. 1587–1611, 2013, doi: 10.1002/wcm.1203.

[4] P. B., "Cloud Computing Deployment Models: A Comparative Study," *International Journal of Innovative Research in Computer Science & Technology*, vol. 9, Mar. 2021, doi: 10.21276/ijircst.2021.9.2.8.

[5] W. Jefry and S. Sasi, "Does Cloud Computing Add Business Agility to Big Data Analytics", Accessed: Feb. 29, 2024. [Online]. Available: https://ijisrt.com/wp-content/uploads/2017/12/Does-Cloud-Computing-Add-Business-Agility-to-Big-Data-Analytics-1.pdf

[6] C. Böckelman, "Cost analysis of cloud based converged infrastructure for a small sized enterprise," 2017, Accessed: Feb. 29, 2024. [Online]. Available: https://aaltodoc.aalto.fi/handle/123456789/27921

[7] K. Nowicka, "Cloud computing in sustainable mobility," *Transportation Research Procedia*, vol. 14, pp. 4070–4079, 2016.

[8] A. Gholamhosseinian and A. Khalifeh, "Cloud Computing and Sustainability: Energy Efficiency Aspects," 2012. doi: 10.13140/RG.2.2.14030.33606.

[9] H. Yuan, C.-C. J. Kuo, and I. Ahmad, *Energy efficiency in data centers and cloud-based multimedia services: An overview and future directions*. 2010, p. 382. doi: 10.1109/GREENCOMP.2010.5598292.

[10] A.-C. Orgerie and L. Lefèvre, "When Clouds become Green: the Green Open Cloud Architecture," Jan. 2010.

[11] A. Gholamhosseinian and A. Khalifeh, *Cloud Computing and Sustainability: Energy Efficiency Aspects*. 2012. Accessed: Mar. 03, 2024. [Online]. Available: https://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-17368

[12] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010, doi: 10.1145/1721654.1721672.

[13] V. Pai and S. Aithal, "Cloud Computing Security Issues - Challenges and Opportunities," *International Journal of Management, Technology, and Social Sciences*, pp. 33–42, Dec. 2016, doi: 10.47992/IJMTS.2581.6012.0004.

[14] V. Pai T and P. S. Aithal, "Cloud Computing Security Issues - Challenges and Opportunities." Rochester, NY, Apr. 29, 2017. Accessed: May 09, 2024. [Online]. Available: https://papers.ssrn.com/abstract=2960646

[15] "[Solved] Cloud computing is a model for enabling convenient, on-demand... | CliffsNotes." Accessed: May 09, 2024. [Online]. Available: https://www.cliffsnotes.com/tutors-problems/Computer-Science/51435435-Cloud-computing-is-a-model-for-enabling-convenient-on-demand/

[16] V. Pai and S. Aithal, "Cloud Computing Security Issues - Challenges and Opportunities," *International Journal of Management, Technology, and Social Sciences*, pp. 33–42, Dec. 2016, doi: 10.47992/IJMTS.2581.6012.0004.

[17] 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, no. 1, p. 4, Apr. 2016, doi: 10.1186/s13677-016-0054-z.

[18] “Interoperability and Portability for Cloud Computing: A Guide Version 2.0,” 2017.

[19] 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, no. 1, p. 4, Apr. 2016, doi: 10.1186/s13677-016-0054-z.

[20] A. AlTwaijiry, *Cloud Computing Present Limitations and Future Trends*. 2021. doi: 10.14293/S2199-1006.1.SOR-.PPEYYII.v1.

[21] A. AlTwaijiry, “Cloud Computing Present Limitations and Future Trends.” Nov. 15, 2021. doi: 10.14293/S2199-1006.1.SOR-.PPEYYII.v1.

[22] J. Kesan, C. Hayes, and M. Bashir, “Information Privacy and Data Control in Cloud Computing: Consumers, Privacy Preferences, and Market Efficiency,” *Washington and Lee Law Review*, vol. 70, Apr. 2012, doi: 10.2139/ssrn.2042638.

[23] C. Ting Si Xue and F. Xin, “Benefits and Challenges of the Adoption of Cloud Computing in Business,” *International Journal on Cloud Computing: Services and Architecture (IJCCSA)*, vol. 6, pp. 1–15, Dec. 2016, doi: 10.5121/ijccsa.2016.6601.

[24] I. Foster, Y. Zhao, I. Raicu, and S. Lu, “Cloud computing and grid computing 360-degree compared,” in *2008 grid computing environments workshop*, Ieee, 2008, pp. 1–10. Accessed: Feb. 29, 2024. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/4738445/

[25] U. Gadhavi and D. V. Vekariya, “A FRAMEWORK TO ENSURE DATA STORAGE SECURITY IN CLOUD COMPUTING TO PROVIDE ERROR LOCALIZATION AND REDUCE THE COMMUNICATION DELAY,” *JOURNAL OF INFORMATION.*

[26] S. Srivastava and S. P. Singh, “A Survey on Latency Reduction Approaches for Performance Optimization in Cloud Computing,” in *2016 Second International Conference on Computational Intelligence & Communication Technology (CICT)*, Feb. 2016, pp. 111–115. doi: 10.1109/CICT.2016.30.

[27] L. M. Vaquero, L. Rodero-Merino, J. Caceres, and M. Lindner, “A break in the clouds: towards a cloud definition,” *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 1, pp. 50–55, Dec. 2008, doi: 10.1145/1496091.1496100.

[28] F. Aslam, “The Benefits and Challenges of Customization within SaaS Cloud Solutions,” *American Journal of Data, Information and Knowledge Management*, vol. 4, pp. 14–22, Jul. 2023, doi: 10.47672/ajdikm.1543.

[29] “On Premise vs. Cloud: Key Differences, Benefits and Risks | Cleo.” Accessed: May 09, 2024. [Online]. Available: https://www.cleo.com/blog/knowledge-base-on-premise-vs-cloud

[30] J. W. Rittinghouse and J. F. Ransome, *Cloud Computing: Implementation, Management, and Security*. Boca Raton: CRC Press, 2009. doi: 10.1201/9781439806814.

[31] “On-Premises vs Cloud Computing: A Balanced Comparison.” Accessed: May 09, 2024. [Online]. Available: https://contextqa.com/useful-resource/on-premises-vs-cloud-computing/

[32] C. Call, “On-Premise Systems vs. Cloud Systems,” Network Security Associates. Accessed: Mar. 18, 2024. [Online]. Available: https://www.nsa-nv.com/on-premise-cloud-systems/

[33] “On-premise vs. cloud computing: definition and differences.” Accessed: May 09, 2024. [Online]. Available: https://nordlayer.com/blog/on-premise-vs-cloud-differences/

[34] "On premises vs. cloud pros and cons, key differences | TechTarget," Cloud Computing. Accessed: Mar. 18, 2024. [Online]. Available: https://www.techtarget.com/searchcloudcomputing/tip/Evaluate-on-premises-vs-cloud-computing-pros-and-cons

[35] L. Yen, "Cloud vs. On-Premises: Pros, Cons, and Use Cases," Datamation. Accessed: May 09, 2024. [Online]. Available: https://www.datamation.com/cloud/cloud-vs-on-premises-pros-cons-and-use-cases/

[36] Z. Mahmood and R. Hill, Eds., *Cloud Computing for Enterprise Architectures*. in Computer Communications and Networks. London: Springer London, 2011. doi: 10.1007/978-1-4471-2236-4.

[37] O. Obi, S. Dawodu, A. Daraojimba, S. Onwusinkwue, O. Akagha, and I. Ahmad, "REVIEW OF EVOLVING CLOUD COMPUTING PARADIGMS: SECURITY, EFFICIENCY, AND INNOVATIONS," *Computer Science & IT Research Journal*, vol. 5, pp. 270–292, Feb. 2024, doi: 10.51594/csitrj.v5i2.757.

[38] "Gartner Identifies the Top 10 Strategic Technology Trends for 2020," Gartner. Accessed: May 06, 2024. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2019-10-21-gartner-identifies-the-top-10-strategic-technology-trends-for-2020

[39] "Understanding Cloud Adoption Decisions in the Public Sector." Accessed: May 09, 2024. [Online]. Available: https://www.researchgate.net/publication/283842564_Understanding_Cloud_Adoption_Decisions_in_the_Public_Sector

[40] "Making a secure transition to the public cloud | McKinsey." Accessed: May 06, 2024. [Online]. Available: https://www.mckinsey.com/capabilities/mckinsey-digital/our-insights/making-a-secure-transition-to-the-public-cloud

[41] E. Brynjolfsson and A. McAfee, *The Second Machine Age: Work, Progress, and Prosperity in a Time of Brilliant Technologies*. W. W. Norton & Company, 2014.

[42] C. Giordano, "The Total Economic Impact Of Microsoft Azure PaaS".

[43] "Cloud for industries: Why the time is now?" Accessed: May 06, 2024. [Online]. Available: https://www.hydrocarbonprocessing.com/news/2024/04/cloud-for-industries-why-the-time-is-now/

[44] "Gatling vs JMeter - What to Use for Performance Testing," InfoQ. Accessed: May 06, 2024. [Online]. Available: https://www.infoq.com/articles/performance-testing/

[45] G. Olaoye and A. Luz, "Network security in the cloud environment," Feb. 2024.

[46] K. Bangalore, "Cloud Compliance Standards," Medium. Accessed: May 11, 2024. [Online]. Available: https://medium.com/@kavib/cloud-compliance-standards-a3245a6bf981

[47] "Cloud Security Is a Shared Responsibility," Palo Alto Networks. Accessed: May 11, 2024. [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/cloud-security-is-a-shared-responsibility

[48] T. Freestone, "Understanding Key Aspects of Data Compliance," Kiteworks | Your Private Content Network. Accessed: May 11, 2024. [Online]. Available: https://www.kiteworks.com/regulatory-compliance/data-compliance/

[49] M. Clt, "Essential Characteristics of Cloud Computing: Understanding the Basics," Medium. Accessed: May 11, 2024. [Online]. Available: https://medium.com/@mohasina.clt/essential-characteristics-of-cloud-computing-understanding-the-basics-6e20e8ec107b

[50] D. Chittibala, "Infrastructure as Code (IaC) and Its Role in Achieving DevOps Goals," Mar. 2024, doi: 10.21275/SR24304170702.

[51] "What is On-Premises vs. Cloud? | Glossary." Accessed: May 11, 2024. [Online]. Available: https://www.hpe.com/us/en/what-is/on-premises-vs-cloud.html
[52] "On-Premises vs Cloud: Main Differences, Benefits & Risks," BairesDev. Accessed: May 11, 2024. [Online]. Available: https://www.bairesdev.com/blog/on-premises-vs-cloud/
[53] "SC_Data-Sovereignty_7-23.pdf." Accessed: May 11, 2024. [Online]. Available: https://www.scalecomputing.com/documents/Data-Sheets/SC_Data-Sovereignty_7-23.pdf
[54] "The Latest Cloud Computing Statistics (updated May 2024) | AAG IT Support." Accessed: May 10, 2024. [Online]. Available: https://aag-it.com/the-latest-cloud-computing-statistics/

LIST OF ABBREVIATIONS

TOS	-	Technology and Operations Support
AWS	-	Amazon Web Services
MS	-	Milliseconds
CSP	-	Cloud service provider
IaaS	-	Infrastructure as a Service
PaaS	-	Platform as a Service
SaaS	-	Software as a Service
GCP	-	Google Cloud Platform
ERP	-	Enterprise Resource Planning
CRM	-	Customer Relationship Management
BI	-	Business Intelligence
NIC	-	Network Interface Card
DTOs	-	Data Transfer Objects
SOC 2	-	Service Organization Control 2
ISO 27001	-	International Organization for Standardization 27001
GDPR	-	General Data Protection Regulation
JSON	-	JavaScript Object Notation
UTF	-	Unicode Transformation Format

LIST OF FIGURES

LIST OF TABLES