

Name: MD RIAD HOSSAIN

Student ID: 2434324

Module: Cloud Engineering principles and practice (ACCA 7027)

Table of content:

1. Introduction

2. Virtualisation in Cloud-Computing

2.1 Principles of Virtualisation

2.2 Example: Virtualisation of Shared Hardware

3.Containerisation in Cloud Environments

3.1 Principles and Benefits of Containerization

4.Container Images and Image Publishing

4.1 Understanding Docker Images

4.2 Example: Pushing an Image to DockerHub

5. Overview of Docker Volumes

5.1 Example: Creating a Docker Volume

6. Concepts of Networking in Docker

6.1 Example: Creating a Docker Network

7 Core Cloud Computing Concepts

7.1 Load Balancing in Cloud Providers

7.2 Scalability in Cloud Providers

7.3 Application Gateways and Their Role

8. why Load Balancing and Scaling are complimentary

9. Benefits of Combining Load Balancing, Scalability, and Application Gateways

10.Drivers Behind the Expansion of Cloud Computing

10.1 Cost Efficiency

10.2 Flexibility

11.Serverless Computing

11.1 Introduction to Serverless Functions

11.2 Examples: AWS Lambda and Azure Functions

12.Conclusion

13.References

Introduction:

Cloud computing is getting one of the most revolutionary technologies in this age which help organizations in terms of scalability, flexibility and cost efficiency. As over the time everything getting digitalize and where the complexity and the demand of user getting increased so the cloud platforms start rely on virtualization, containerization, and automated technologies to make sure the reliability and increase the performance. This concept helps the organization and the businesses to avoid extra infrastructure cost, increase development process and support world-wide available services. In the sense of cloud virtualization take the first step toward resource utilization by allowing multipoles virtual machine to run on shared hardware. Additionally, as software mechanism evolving and demand greater agility, containerization come into the picture as a lightweight and most efficient alternative. Another one is Docker, has very significance in the cloud native application development by packaging system and making their dependencies, plugins into portable, separate units.

This report explores all those technologies and their mechanism in depth and also demonstrate how this technology allow consistency, efficiency and simplified workflows. In this report, practical staffs are integrated to show the essential container technologies such as Dockerfiles, images, networking, and volumes and the examples are given reflect the activities performed in the class and demonstrate how containersation supports real life could operations. The report also examines the depth concepts of load balancing, scalability, and application gateways, analyzing how provider like AWS and Azure implement this kind of features and why they are important in cloud services. In addition this report also analysis the main factors behind cloud adoption cost efficiency, agility, and innovation and highlishts their strength how they contribute to the ongoing evolution of cloud based technology.

2 virtualization in cloud computing :

Virtualisaion is a core concept of modern cloud technology which allow physical resources to be abtacted into multiple independent virtual environments. This abstraction process makes it happen for a single physical server to run several different different virtual machines and allow them its own OS and applications. This is an huge advantage for cloud user to improve resource utilization, allow several environments and reduces costs of operation with hardware centric deployments. Its also forms the based for on demand provisioning, such as characteristic of cloud supported infrastructure by tech giant like AWS, Microsoft Azure, and Google cloud [1].

2.1 principles of virtualization:

Heart of the virtualization is the hypervisor, a soft layer that responsible to locates physical resources and manage multiple VMs. There are two types of hypervisor type-1 work directly with hardware and type-2 work for VMs. Well, hypervisor basically provides abstraction, isolation and dynamic allocation of resources like CPU, STORAGE, NETWORKING and MEMORY. Here isolation is one of the key principle, it's handle each VM independently, means whatever happens with one VM doesn't affect others. Another principle is encapsulation, where VMs are in file state making them easy to moveable and simple to restore. Overall, resource pooling makes efficient sharing of underlying hardware, reducing wastage and enhance utilization rates. Recent studies show that improvements in hypervisor scheduling and CPU virtualization technologies have significantly reduced virtualization overhead, enabling even latency-sensitive and real time workloads to be supported within cloud environments [2].

2.2 Example: Virtualisation of shared hardware

A practical example of visualization of is running multiple application servers on a single physical host because without virtualization every application would need their own dedicated machine, which will be low utilization rates and higher cost but with a hypervisor a single server can be able to run several virtual machine with different different data server in separate environment. Where CPU, RAM, and storage are allocated for each VM to run its OS and other staffs. This shared system savings the cost and energy. Research shows that server consolidation through virtualization can reduce significant reduction of physical hardware, leading to significant reductions in power and cooling costs across data centers [3]. Additionally, virtualization give the opportunity to live migration, where VM can be shifted one device to another without any difficulties. This capability is essential for load balancing, high availability, and maintenance tasks in modern cloud infrastructure [4].

Overall virtualization transforms static physical resources into separate flexible environment. Where is easy to moves and operates.

3. Containerisation in cloud Environments

In modern cloud computing containerization has become a main technology because it offers lightweight, portable and alternative to manual VM. A container is a separate environment of an application with all necessary libraries, plugins and components that ensure it suitable for any kind of working environments. Some recent research highlights that containerization improves deployment speed, resource scalability which make it a fundamental tool for modern cloud infrastructures [5]. Container share the host OS kernel to process the isolation where VM is takeover full guest operating system which make container much higher density per host. containerized application can deploy much faster compare to VM based deployments, which is why tech giant like AWS, Azure, and google cloud have adopted container platforms like kubernetes as a core services.

3.1 principle and benefits of cotainerisation:

The primary principles of cotanerisation include:

Lightweight virtualization:

containers came into picture to provide OS-level virtualisaiton. This concept based on technologies such as linux namespaces and cgroup which allow multitasking without the heavy overhead of full operating systems. virtualization offers superior performance compared to hypervisor based system due to reduced context switching and lower memory consumption.

Portability:

Portability refers that container includes the application with its all dependencies so it can run in any environment that supports the runtime. This eliminates the “works on my machine “ problem and enhances reproducibility in development and deployment workflows.

Scalability :

Containers start within seconds and allow rapid horizontal scaling. Modern orchestration platforms such as docker swarm and kubernetes automatically deploy and scale containers across clusters based on resource metrics.

4. container images and publishing:

In this system creating images, run the container and push them in hub is the central part of how application packaged, distributed and deployed. In this section it will be explain in depth. What is the role of container images, how they created and how to push them, find them in platforms like Docker hub and the significance of it I will include the example after the details explanation with pushing a html web server which based on Nginx.

4.1 understanding container images:

A container image is just a static, read only template that holds the application code, plugins, runtime, environment variables, and configuration all the setup that needed to run the application inside a container. Well when container runs, it uses this images as a blueprint. Images are immutable but container is mutable means container is changeable but one images are created there are no way to change it. Image are responsible to bundles all required plugins, that helps to run on any machine like (local pc, cloud vm, on-prem server) and also be versioned tags, rolled back, or stored as archived, immutability ensures that what was deployed remains unchanged. One more significance feature is instead of installing plugins on target machine it deploys containers built from images it helps regrading reducing setup time and configurations drift. Here is the example

4.2 Example: Pushing an Image to DockerHub

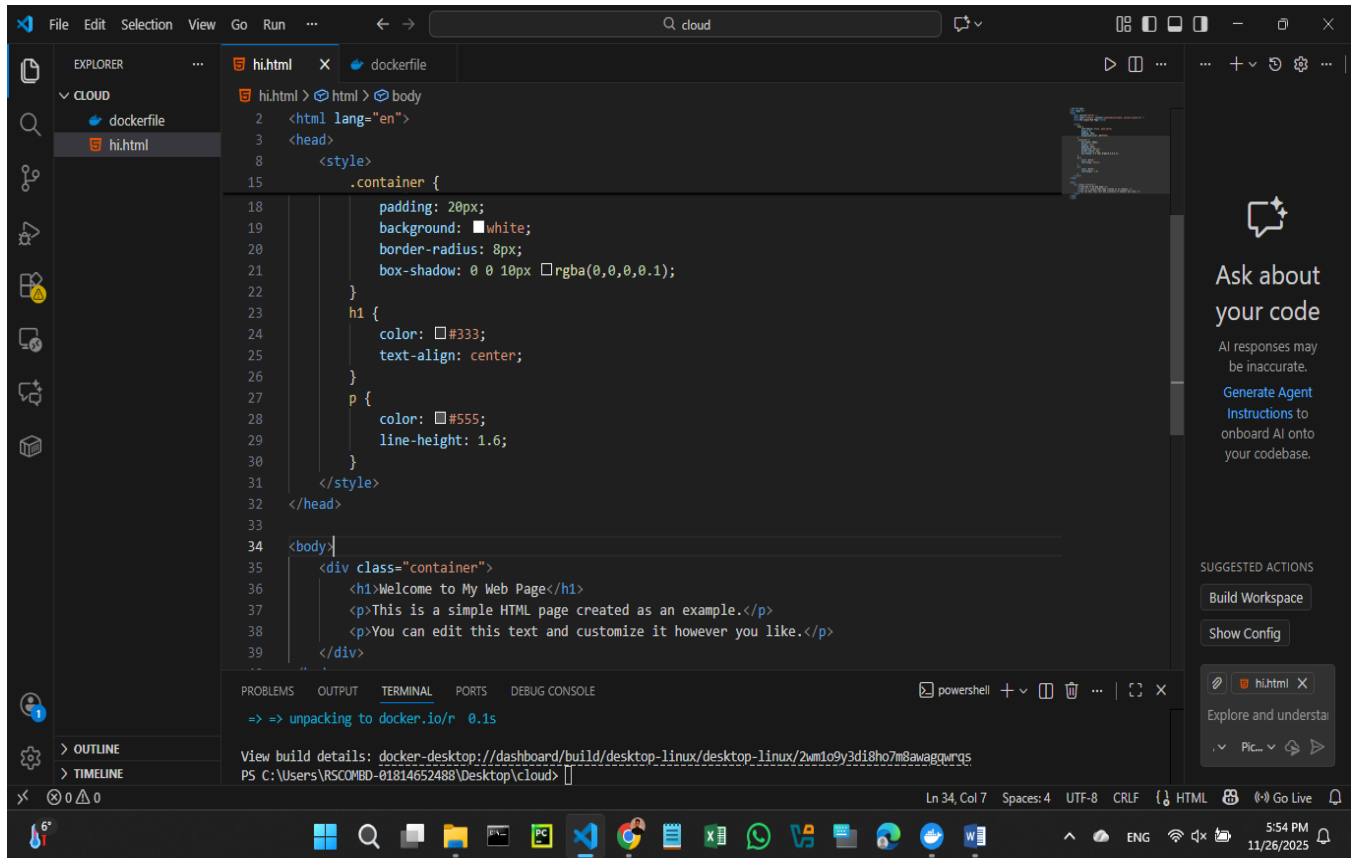


Fig: 1

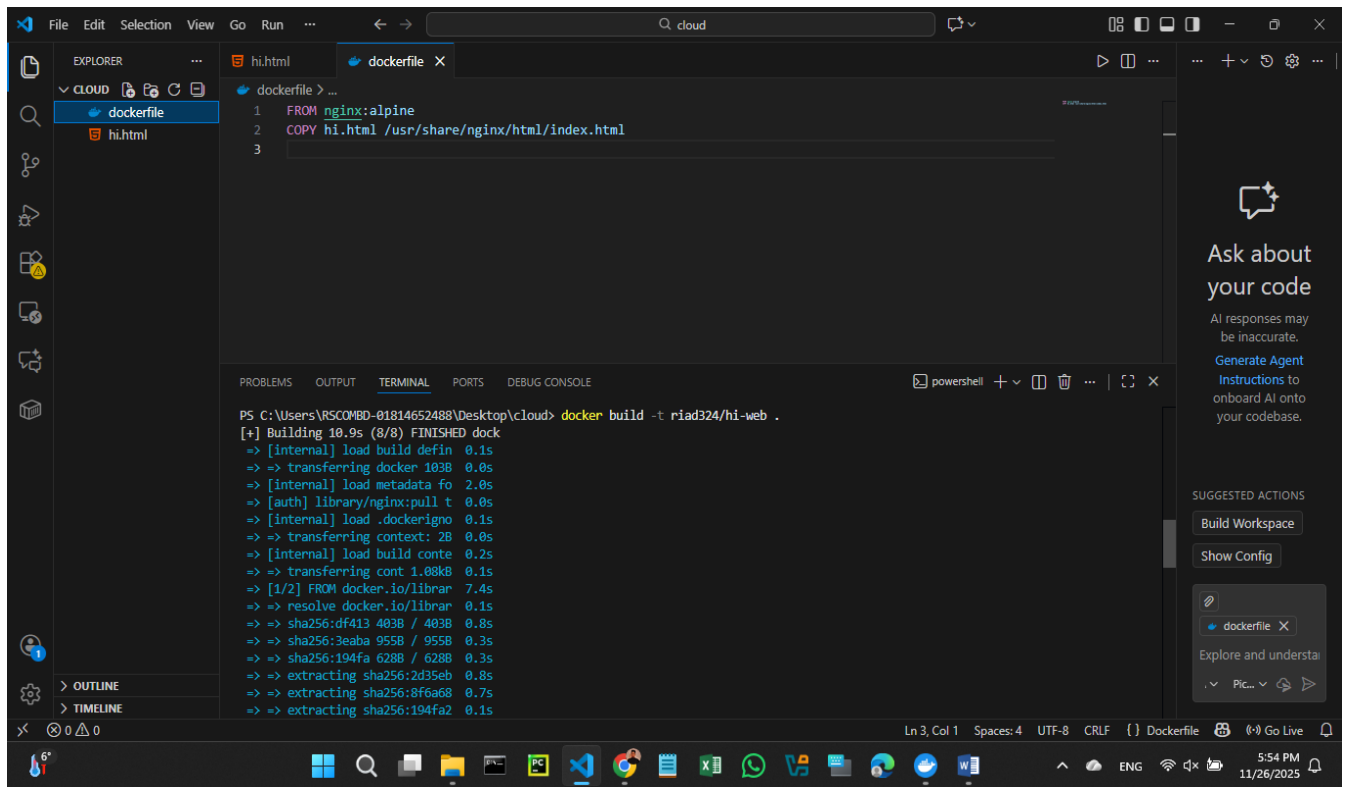


Fig:2

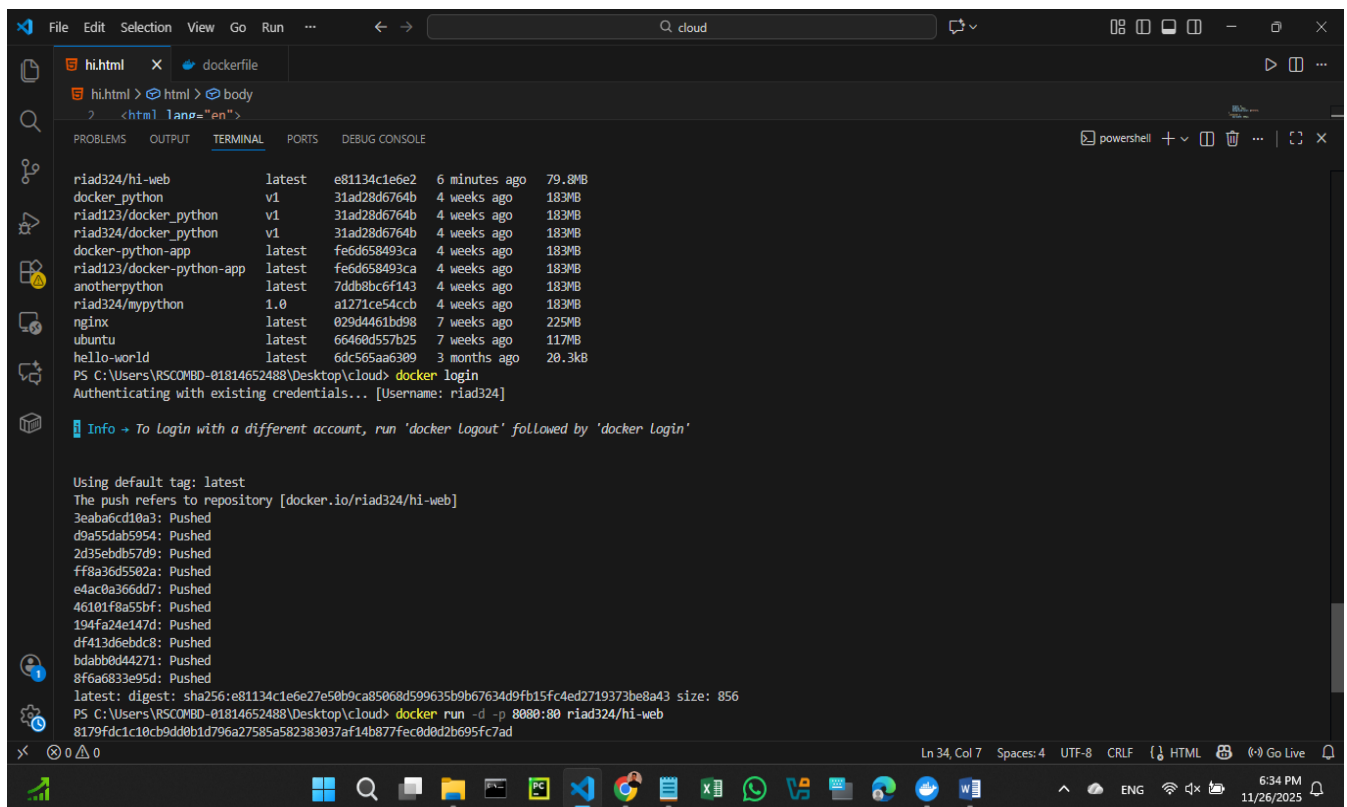
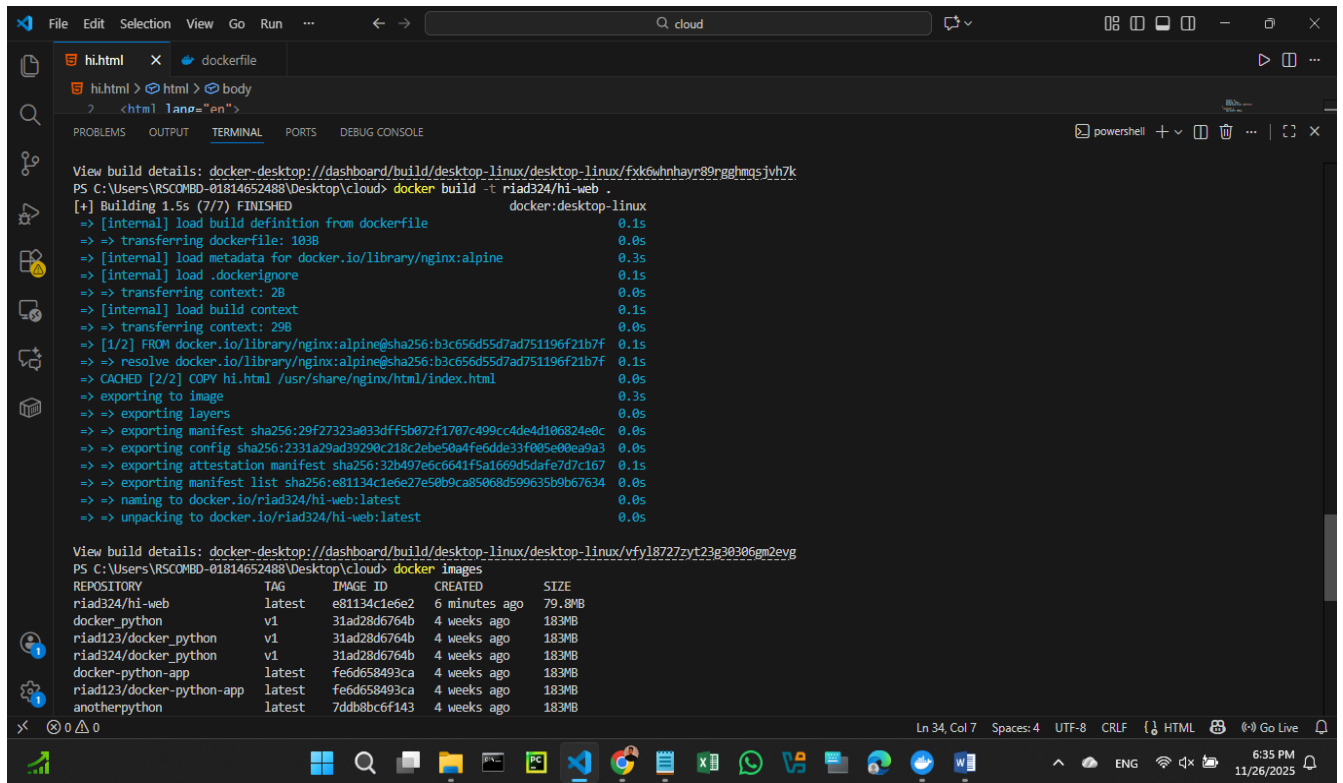


Fig:3



```
View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/fxk6whnhayr89ngghmqsjvh7k
PS C:\Users\RSCOMB-01814652488\Desktop\cloud> docker build -t riad324/hi-web .
[+] Building 1.5s (7/7) FINISHED                                docker:desktop-linux
=> [internal] load build definition from dockerfile             0.1s
=> => transferring dockerfile: 103B                             0.0s
=> [internal] load metadata for docker.io/library/nginx:alpine 0.3s
=> [internal] load .dockerignore                               0.1s
=> => transferring context: 2B                                    0.0s
=> [internal] load build context                                0.1s
=> => transferring context: 29B                                    0.0s
=> [1/2] FROM docker.io/library/nginx:alpine@sha256:b3c65d5d7ad751196f21b7f 0.1s
=> => resolve docker.io/library/nginx:alpine@sha256:b3c65d5d7ad751196f21b7f 0.1s
=> CACHED [2/2] COPY hi.html /usr/share/nginx/html/index.html 0.0s
=> => exporting to image                                         0.3s
=> => exporting layers                                           0.0s
=> => exporting manifest sha256:29f27323a033dff5b072f1707c499cc4de4d106824e0c 0.0s
=> => exporting config sha256:2331a29ad39290c218c2ebe50a4fe6dde33f005e00ea9a3 0.0s
=> => exporting attestation manifest sha256:32b497e6c6641f5a1669d5d4fe7d7c167 0.1s
=> => exporting manifest list sha256:e81134c1e6e27e50b9ca85068d599635b9b67634 0.0s
=> => naming to docker.io/riad324/hi-web:latest                 0.0s
=> => unpacking to docker.io/riad324/hi-web:latest              0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/vfy18727zyt23g30306gm2evg
PS C:\Users\RSCOMB-01814652488\Desktop\cloud> docker images
REPOSITORY          TAG         IMAGE ID      CREATED        SIZE
riad324/hi-web       latest      e81134c1e6e2  6 minutes ago  79.8MB
docker_python        v1         31ad28d6764b  4 weeks ago   183MB
riad123/docker_python v1         31ad28d6764b  4 weeks ago   183MB
riad324/docker_python v1         31ad28d6764b  4 weeks ago   183MB
docker-python-app    latest     fe6d658493ca  4 weeks ago   183MB
riad123/docker-python-app latest     fe6d658493ca  4 weeks ago   183MB
anotherpython        latest     7ddb8bc6f143  4 weeks ago   183MB
```

Fig: 4

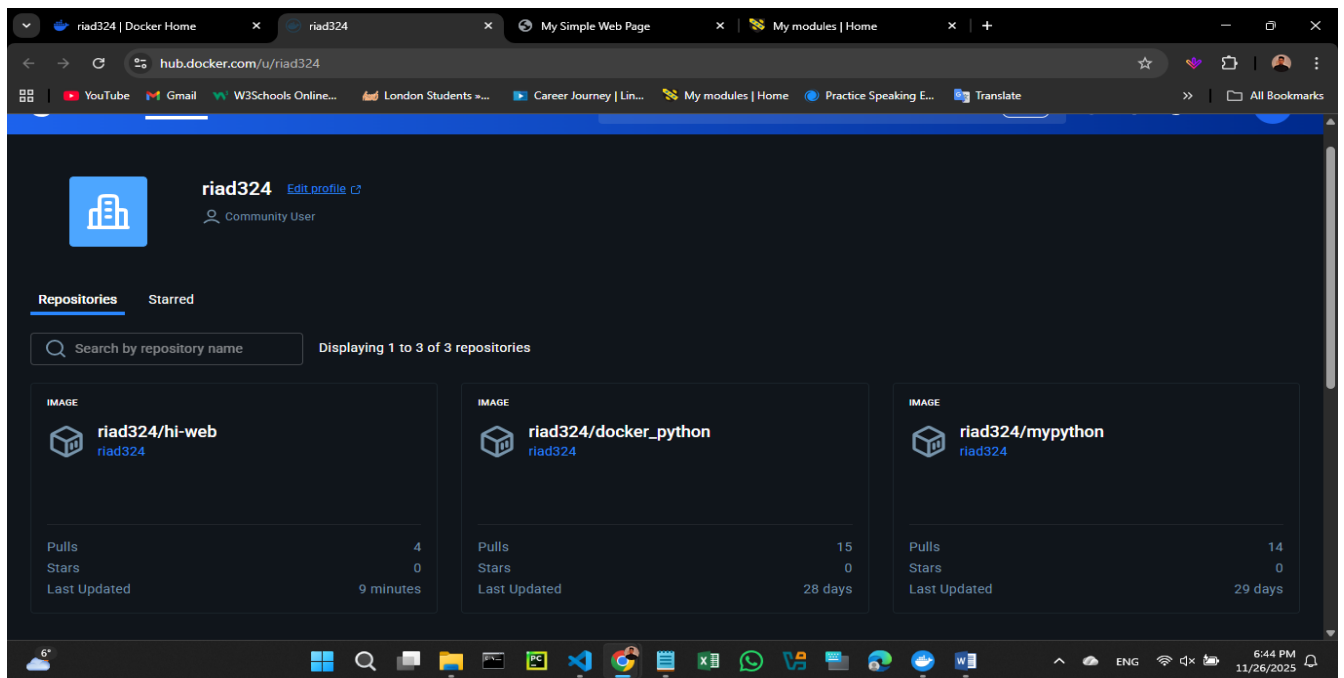


Fig:5

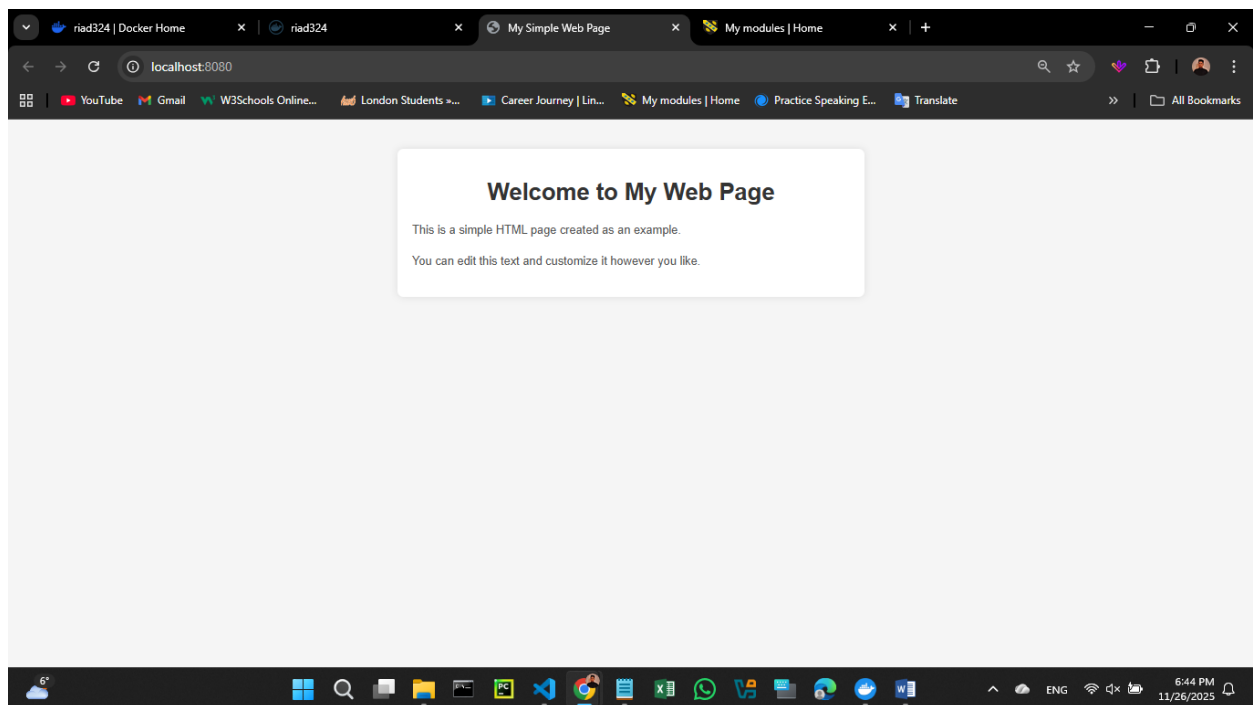


Fig: 6

5 Overview of Docker Volumes

In cloud docker volumes is one of the most major concept. Volumes operate as mounted directories that allow containers to terface with external storage with embedding phase inside the core file of containers. Because volumes are maintained outside the container lifecycle, they prevent data loss and allow more reliable environment. In digital orchestration systems such as kubernetes extend this concept by persistentvolumes and persistentvolumeclaims, providing a space of storage abstraction layer that integrates with cloud storage services. This demonstrates the importance of persistent storage not only at the container runtime but from entire cloud native ecosystem. Furthermore, volumes improve portability. Since the abstracted data from the container, developers can rebuild or update images without affecting of data that existing.

5.1 Example: Creating a Docker Volume

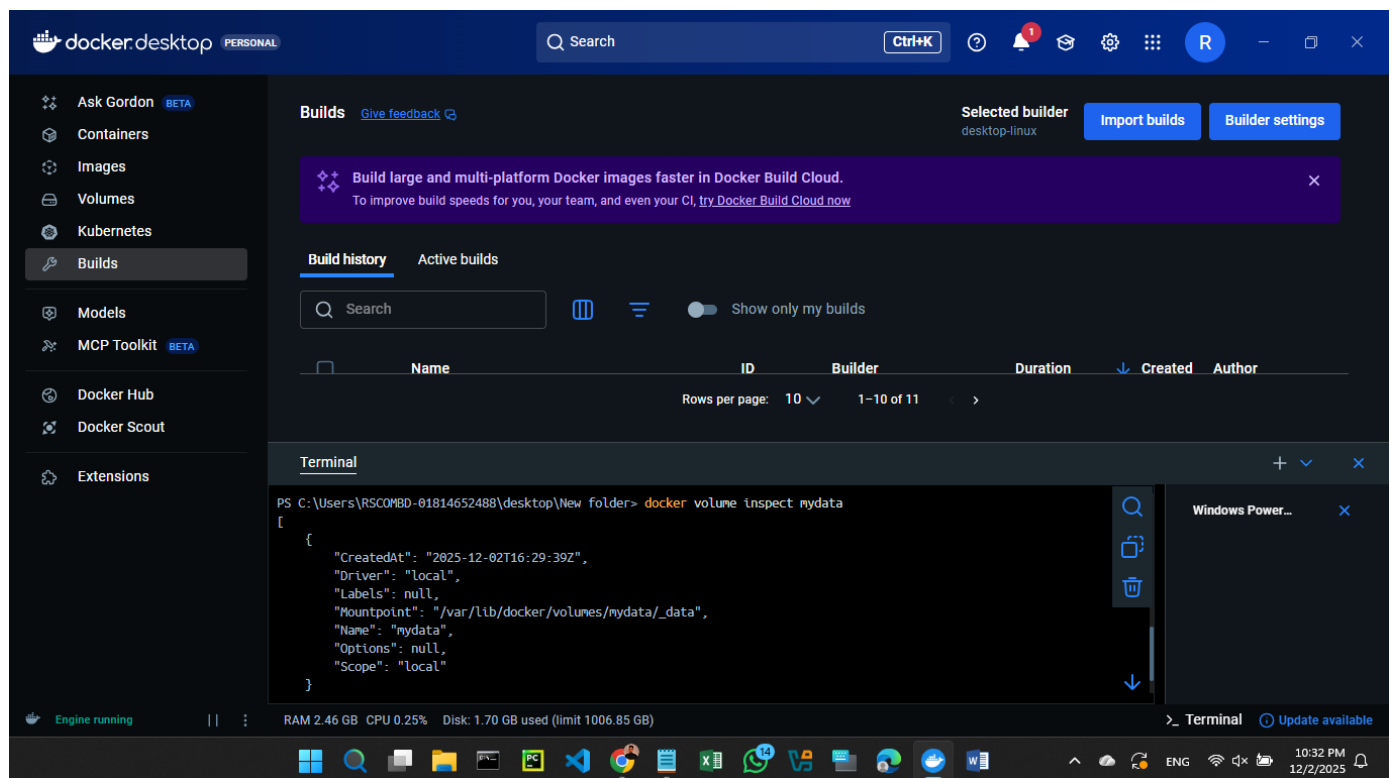


Fig: 7

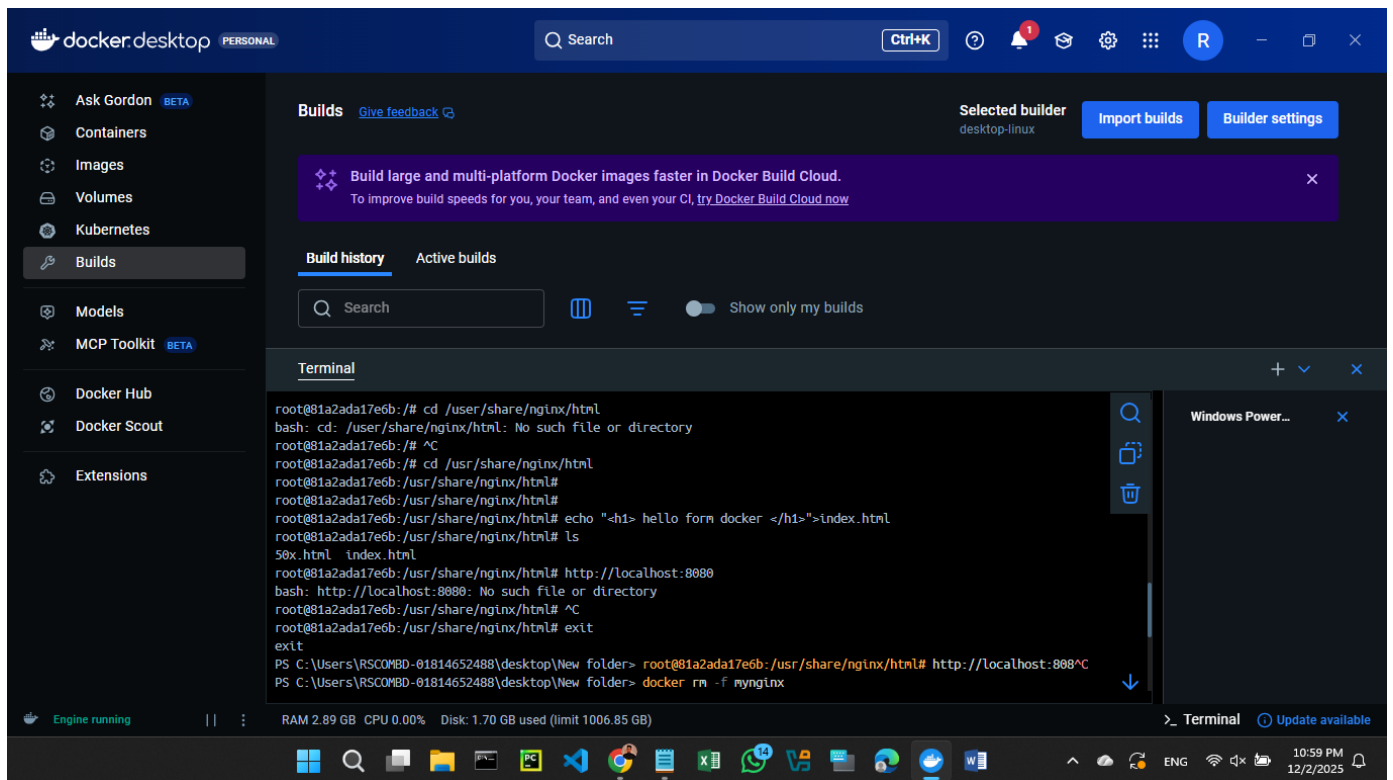


Fig: 8

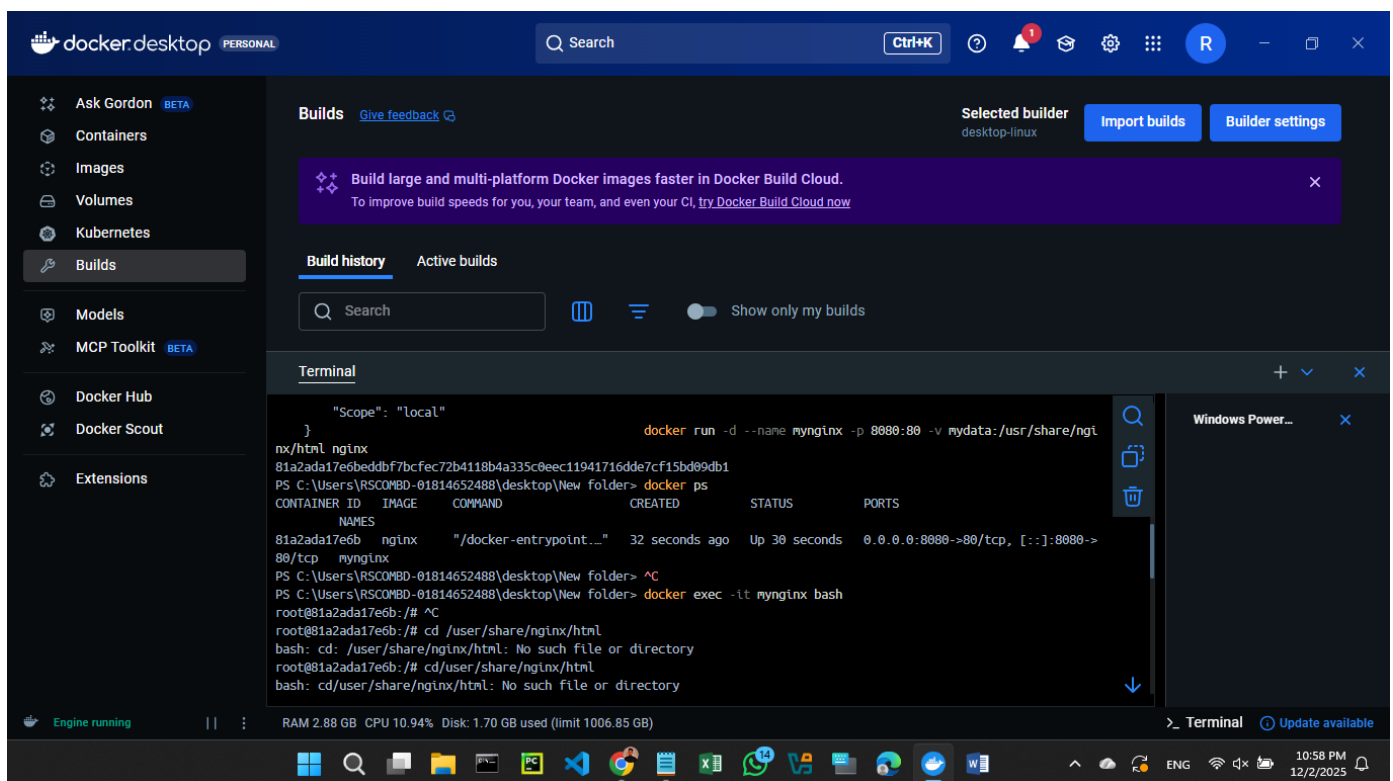


Fig:9

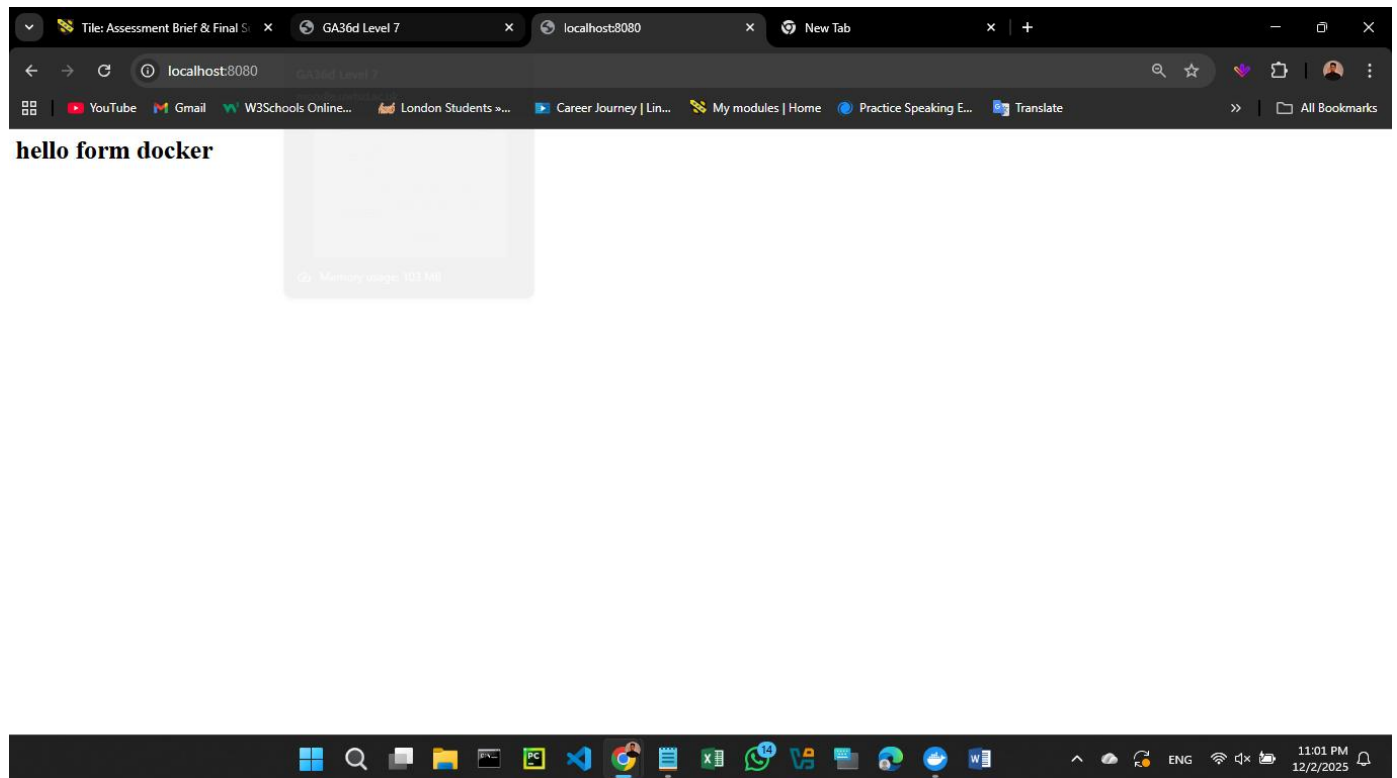


Fig: 10

6. Concepts of Networking in Docker

One of the fundamental component of cloud native application deployment is container networking which allow communication in between containers, host and external systems. Because containers are separated processes, they need a virtual networking layer that allows controlled connectivity while maintaining security patch and performance. Docker offers many built in network drivers – bridge, host, overlay, and macvlan- each designed for different purposes such as single host development, multi host orchestration, or advanced routing scenario. In digital distributed system, container networking is essential for support microservices, work load distribution, and inter container communication. Additionally, container network must implement effective isolation to prevent unauthorized access while still enabling seamless connectivity for dependent service. Docker use SDN principles to create virtual network that container can join. The default network –bridge- assigns each container a private address and allow communications through internal routing. This is suitable for local development and small scale development.

6.2 Example: Creating a Docker Network

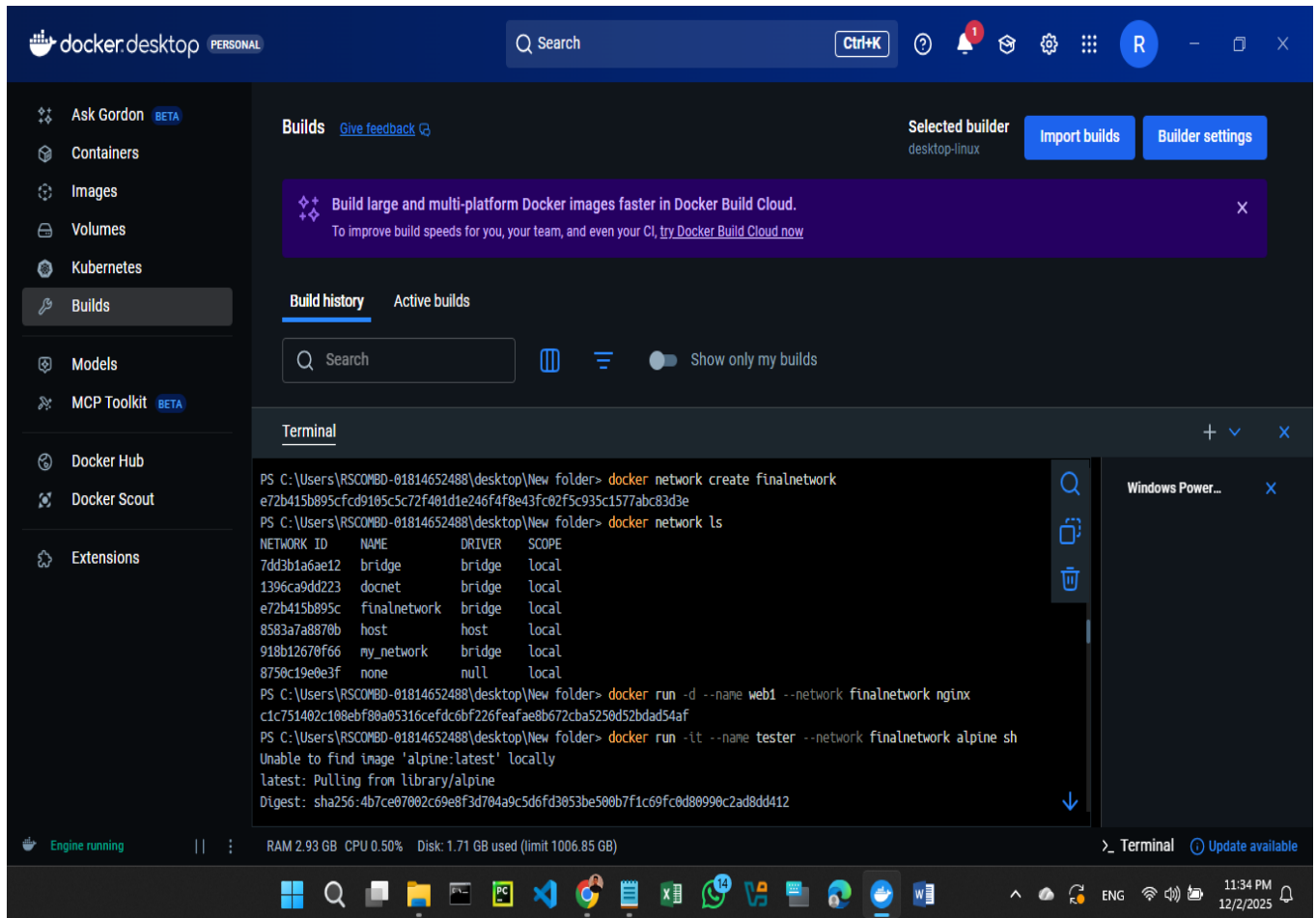
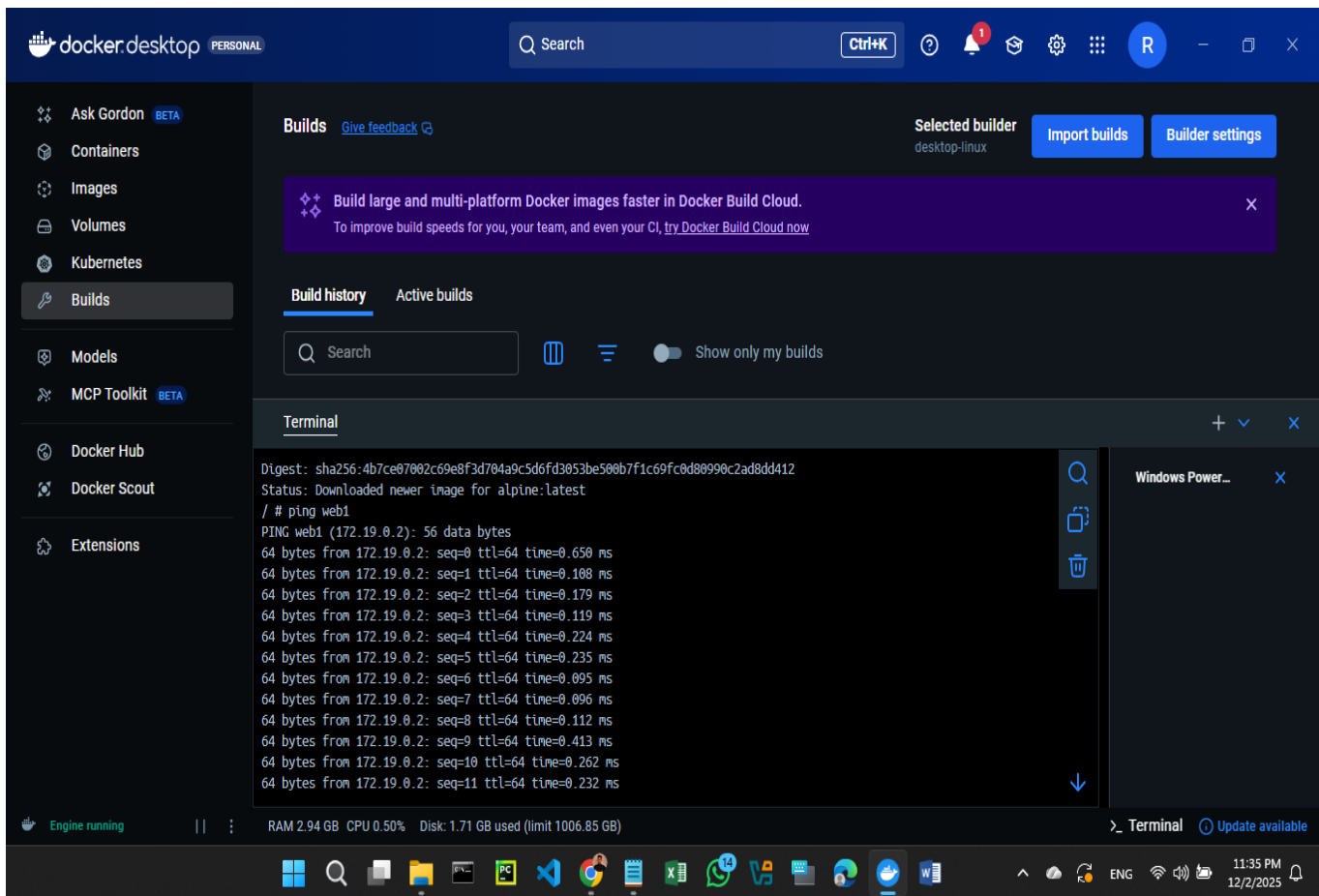


Fig: 11



Fig; 12

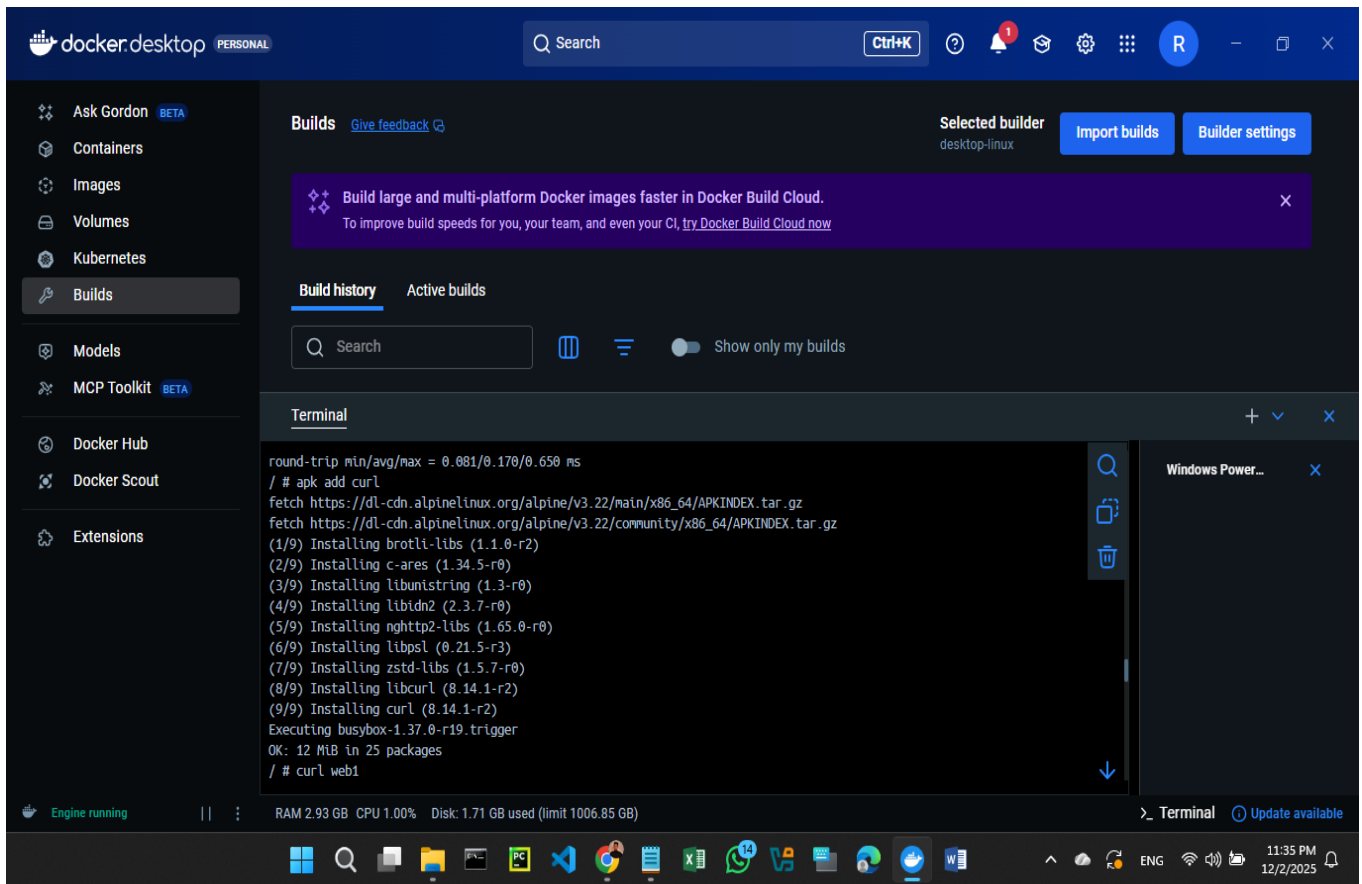


Fig: 13

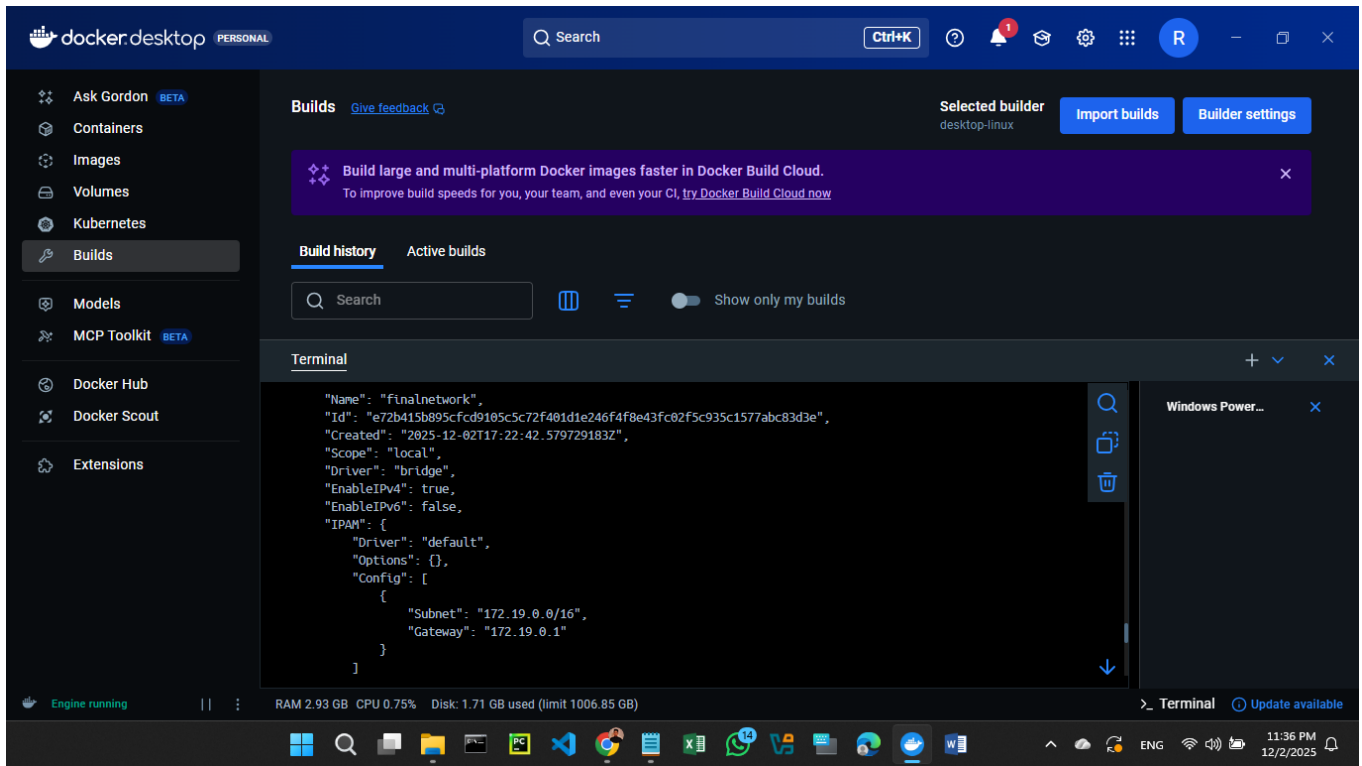


Fig:14

7. Core Cloud Computing Concepts:

In this modern age cloud space rely on three major technologies for high performance such as load balancing, scalability, and application gateways. All these modern tech work together to make sure that app can manage large variations in demand, distribute workloads efficiently, and also manage structured traffic flow. Some major providers like AWS, Azure, and google implement these concepts as fully managed services, making them accessible and reliable for organisations of all sizes.

7.1 Load Balancing in Cloud Providers:

This term refers to the procedure of distributing incoming traffic across many servers to enhance the performance, optimize overload and increase availability. Without load balancing spike in user demand can cause a single server to fail, which will lead to downtime and poor user experience. For understanding there are examples below. AWS ELB distributes traffic across EC2 instances automatically, containers, or Lambda functions. It supports many types of load balancers like (TCP/UDP) and gateway load balancers.

Azure load balancer distributes traffic at layer 4 same as like AWS but application gateway provides layer 7 routing, SSL termination, session and waf capabilities.

Research shows that efficient load balancing significantly improves cloud service reliability and shortens response time mostly in the architecture of distribution where workloads and traffic patterns are unpredictable [6].

7.2 Scalability

This term refers to the capacity of a system to increase or decrease its compute resources based on demand. Cloud providers offer two main types such as:

1. Vertical scaling to increasing the ability of an existing instance (adding more CPU or RAM).
2. Horizontal scaling to adding more instances of the application, mostly used in CND.

the provider implementation AWS used automated system to add or remove EC2 instances based on CPU or custom metrics. Supports predictive scaling using ML. Azure virtual machine scale sets which enables automated scaling of VMs based on rules, schedules, or performance metrics. With Azure load balancer and gateway point out the importance of scalability mechanism in cloud elasticity, which allow companies to optimize resource usage and decrease operational costs.

7.3 Application Gateways

this is a layer 7 load balancer designed to manage, inspect, and route HTTP/HTTPS traffic intelligently. Unlike the traditional load balancers, application gateways understand application level protocols, advance features like SSL-termination, URL base routing, cookiebased session, WAF protection. Here the implementation AWS load balancer allow content based routing, container load distribution and combine with AWS WAF for security policies. Azure gateway offers URL path routing, multi-site hosting, end to end SSL and built in web application firewall protection. This gateways plays most important role in microservice architecture, backend traffic based on URLs services, headers, and API paths as well.

8. why Load balancing and scaling are complementary

Load balancing and scaling are work together to manage application availability and task performance.

A load balancer ensures incoming resuests across multiple targets. So traffic increase beyond the capacity of exiting instaces, a load balancer alone cannot stop the overload. So the traffic balancing but without scaling, servers cloud still overload.

Unlike scaling alone is not enough cause this system response to the capacity of demand, if multiple instaces are added during a traffic surge these instnances are ineffective unless user request cann't properly distributed across them. Without a load balancer, new instances may remain idle while exiting ones become overloaded, leading to poor performace so without load balancer traffic cannot be evenly distributed. They create a dynamic, self-optimism system capable of responding to variable workloads. So integrating these services leads to improved resiliency and reduced latency across cloud deployments.

9. Benefits of Combining Load Balancing, Scalability, and Application Gateways

Let's dive into the benefits when all three used together.

High availability if one instance fails, the load balancer routes traffic to healthy instances. **Elastic performance**, auto scaling come into the picture to add capacity during peak demand and decrease the cost during low usage time duration. **Advanced traffic and security controls**, application gateways filter suspicious traffic, direct requests intelligently and safe backend.

So cloud native applications, especially micro services, depends on this trio to deliver stable, scalable, and smooth secure user experiences. AWS and Azure fully manage these services with minimal cost and allow organizations to focus on development instead of infrastructure management.

10. Drivers Behind the Expansion of Cloud Computing

The rapid growth of cloud services over the time is driven by many compelling factors, including cost efficiency, scalability, accessibility, and operational agility. Most of organizations across all sectors are rapidly shifting workloads to cloud to get the mentioned opportunity and competitive advantages to optimize the expenditures. The major driving forces behind cloud adopting with discuss below with supported by recent research.

10.1 Cost Efficiency

one of most primary motivations for adopting cloud as a platform is cost reduction. In a traditional on premises infrastructure need huge amount of capital for data centres, hardware, and backup system. But cloud computing transforms this system into an operational expenditure approach by pay as you go pricing. Companies only pay for the resources they consume, like compute time storage, or network bandwidth. There are some popular cloud platforms like AWS, AZURE, and GOOGLE Cloud further improve efficiency of cost by offering automated scaling, serverless computing, and optimization tools. According to one study adopting cloud infrastructures reduce their operational and capital costs by 30-50%, primarily due to decrease hardware investments and enhance resource utilization [7]. Additionally, managed services like cloud databases, load balancers, and tools to eliminate the need for dedicated IT support, outcome is additional savings.

10.2 Flexibility

Cloud platforms provides flexibility in many ways like deploying, managing, and scaling applications. Developers will be able to deploy globally within minutes, increasing responsiveness to the needs of business. Flexibility also expand to hybrid and multi-cloud environments, enabling organisations to distribute workload from many cloud providers for redundancy, compliance and performance optimization.

Recent research shows that, cloud environments significantly shorten developments process by offering rapid provisioning, automated infrastructure management, and frictionless deployment pipelines [8].

11. Comparison Between Serverless Functions and Containerisation

Serverless computing also known as functions as a service represents one of the most significant landscape in cloud computing. Unlike manual virtual machines or container-based deployments, functions allow application to run in fully managed environments. This section explore serverless functions, examines AWS lambda and Azure functions, and compares between serverless architecture with containerization, identifying both how they respond to key cloud adoption drivers like cost, efficiency and flexibility.

11.1 Introduction to Serverless Functions

Lets understand the serverless functions mechanism, functions are event driven compute units that helps regrading execute small, stateless pieces of code in response to triggers such as file upoding, time management, handle HTTP requests and database. In the environments of serverless computing developers can uploads their function code, config triggers and let the cloud provider manage the rest. On the other hand in traditional compute models there are no servers for the developer to manage also resources are allocated only hen the function runs and billing is based on execution time, memory usage and the number of requests. This reduce the overhead managing VMs, containers, scaling rules and allowing developers to focus on purely logic.

11.2 Examples: AWS Lambda and Azure Functions

AWS lambda is one of the most widely adopted multi lingual serverless platform that supports python, JS, java and integrates deeply with services like dynamoDB, API gateway, S3 and cloudwatch. Lambda basically automated the scaling to handle huge number of concurrent executions with cold start optimisations introduced in recent years to reduce latency.

Azure function offers also similar types of event driven execution with more strong integration into MS ecosystem including Azure storage, cosmos DB and active directory. It also provides durable function execution which one is supports stateful workflows something manually difficult in serverless environments.

Both platforms offers automatic scaling, built in monitoring system, event based triggers and fine grained billing.

12. Conclusion

Cloud services has fundamentally transformed how environment of applications are developed, deployed, and consumed, provide scalable, flexible, and cost-efficient environment for organisations of all sizes. Well this report investigated the core cloud computing concepts required for modern distributed system, beginning with virtualization- the core technology that abstract physical hardware into flexible, multi-tenant environments. Building on this, containerization was explored as a lightweight and portable alternative that enhances deployment speed, consistency, and proper use utilization of resource. Through practical demonstrations involving Dockerfiles, image construction, networking, volumes, and dockerhub publishing, the report illustrated the complete lifecycle of container based deployment using Nginx web server hosting an HTML page.

The research also figured out necessary cloud native features such as load balancing, scaling, gateways and showing the major cloud providers like AWS, Azure implement these capabilities to ensure availability, task performance, and security as well. The report also demonstrated that while each technology processed value individually, their integration make resilient, elastic and secure cloud infrastructure capable of supporting digital services. The complementary relation between load balancing and scaling was highlighted, emphasizing their combined role in managing workloads.

Additionally the discussion extended to the broader drivers behind cloud adoption, including cost efficiency, managing agility, world wide access and creative acceleration. All these factors keep the flow to propel the rapid adoption of cloud computing from

industries. Finally the report, compared serverless computing with containerization, and outlining their strengths and limitations as well. Serverless platforms provide operational simplicity and cost benefits for event-driven workloads, running applications.

Overall the report demonstrates that effective cloud engineering required the understanding, knowledge not only the technologies but also how they interact to create robust, scalable, and future ready cloud infrastructure.

13. References:

- [1] H. Sturley, A. Fournier, A. Salcedo-Navarro, M. Garcia-Pineda, and J. Segura-Garcia, "Virtualization vs. containerization: A comparative approach for application deployment in the computing continuum focused on the edge," *Future Internet*, vol. 16, no. 11, Art. no. 427, 2024, doi: 10.3390/fi16110427.
- [2] X. Ding and J. Shan, "Diagnosing Virtualization Overhead for Multi-threaded Computation on Multicore Platforms," in **2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom)**, 2015, pp. 226-235, doi:10.1109/CloudCom.2015.102.
- [3] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generation Computer Systems*, vol. 28, no. 5, pp. 755–768, May 2012, doi: 10.1016/j.future.2011.04.019.
- [4] A. Choudhary, M. C. Govil, G. Singh, L. K. Awasthi, and E. S. Pilli, "A critical survey of live virtual machine migration techniques," *Journal of Cloud Computing*, vol. 6, article 23, 2017, doi:10.1186/s13677-017-0092-1.
- [5] W. Pahl, "Containerization and the PaaS cloud," *IEEE Cloud Computing*, vol. 2, no. 3, pp. 24–31, May–June 2015, doi: 10.1109/MCC.2015.51.
- [6] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, June 2009, doi: 10.1016/j.future.2008.12.001.
- [7] A. Khajeh-Hosseini, D. Greenwood, and I. Sommerville, "The cloud adoption toolkit: Supporting cloud adoption decisions in the enterprise," *Software: Practice and Experience*, vol. 42, no. 4, pp. 447–465, Apr. 2012, doi: 10.1002/spe.1072.
- [8] L. Chen, "Continuous delivery: Huge benefits, but challenges too," *IEEE Software*, vol. 32, no. 2, pp. 50–54, Mar.–Apr. 2015, doi: 10.1109/MS.2015.27.

YouTube video link:

<https://youtu.be/7TW9Mi43B5s>