Original Article

Estimating the Deployment Time for Containerized Application Using Novel Microsoft Azure Based Docker Container over Google Cloud Platform Based Docker Container

Niranjan S[1], Saravanan M.S[2]

[1]Research Scholar, Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, Tamil Nadu. India, Pincode: 602105.
[2]Project Guide, Corresponding Author, Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Saveetha University, Chennai, Tamilnadu. India, Pincode: 602105.

Abstract

Aim: The aim of the study is to estimate the deployment time of Containerized application using Novel Microsoft Azure based Docker Container over Google cloud platform based docker container.
Materials and Methods: Sample groups that are considered in the project can be classified into two each has 20 samples, one for Google cloud platform and other for Novel Microsoft Azure based docker container, which are tested using 0.80 for G-power to determine the
sample size and for t-test analysis. 20 applications have been used in each group for estimating deployment time. **Results:** The Novel Microsoft Azure based Docker container with efficient accuracy of 77%, which by far seems to be better than the Google cloud platform based docker container which gives around 70.00%. As a result, the study group has a statistically negligible difference. The significance is around 0.0477 ($p<0.05$) and therefore there is a statistical insignificant difference among the study group.
Conclusion: The deployment time of Containerized application in the Novel Microsoft Azure based Docker container is better compared to Google cloud platform based docker container.

Keywords: Novel Microsoft azure based docker container, Google cloud platform based docker container, Deployment time, Docker container, Containerized application, Docker image.

DOI: 10.47750/pnr.2022.13.S04.189

INTRODUCTION

The adoption of virtualization technology has risen rapidly in recent years. As a result, there is a greater need for efficient and secure virtualization solutions (Sioshansi 2014). Virtualization technologies such as container-based virtualization and hypervisor-based virtualization have recently arrived on the market (Ganesan, Skaria, and Vos 2020). However, due to the difficulties and risks associated with virtualization, containerization has gained popularity among many consumers and companies; among all container suppliers, Docker has been the most common way to deploy software applications or services in recent years (Arundel and Domingus 2019). In comparison to the respective virtualizations, Docker containers were used to create a more feature-rich infrastructure, which prompted a thorough examination (Turnbull 2014). Docker containers, according to research, have a significant market share (Juell 2020). A Docker image is a small, standalone software package that contains everything needed to run a programme, including code, runtime, system tools, system libraries, and settings(Mouat 2015). The procedure for creating unique Docker images of an ASP.NET Core application and submitting them to a private Azure Container Registry repository (ACR)(Dumpleton 2018). These images will be utilized by Azure DevOps to deploy the Containerized application to Docker containers in the Azure App Service (Linux).When developing cloud-native apps, the Docker Azure Integration allows developers to use native Docker commands to run applications on Azure Container Instances (ACI) (Klaffenbach et al. 2019; Arundel and Domingus 2019). Developers may also utilize the Docker CLI to: Easily log into Azure, thanks to the interface between Docker and Microsoft developer tools.Many tools are available on Google Cloud Platform for interacting with and operating Docker-based containerized workloads, including managed Kubernetes and serverless container execution.The first is Cloud Run, which is very simple and straightforward to use and will be enough for the majority of users. Containers are treated as "services," with a set amount of RAM and one (or two) CPU

cores assigned to them(Denmark et al., n.d.). You can give the containers their own domains and run apps on various ports.

There are around 75 IEEE papers and 81 google scholar papers have been published over the past 5 years. The most cited article is "An Empirical Analysis of the Docker Container Ecosystem on GitHub".Containers enable the packaging of an application, along with its dependencies and execution environment, into a standardized, self-contained container that can be used for software development and deployment on any platform. Docker containers have become the de-facto standard format in the software development world due to their quick adoption (Cito and Gall 2016). Following the concept of Infrastructure-as-Code, the contents of a Docker container are declaratively described in a Dockerfile, which stores instructions to attain a specific infrastructure state (IaC). Dockerfiles in source code repositories may thus enable the execution of computer code in an isolated and speedy environment with a single command.Deploying container images can be excruciatingly sluggish, taking several minutes depending on the size of the image and the state of the network (Mouat 2015). However, in circumstances like a fog-assisted augmented reality Containerized application where end users are mobile and new containers must be dynamically constructed when a user reaches a new geographical area, such delays are unacceptable (Mouat 2015; De Paolis and Bourdot 2020). As a result, reducing deployment times as much as feasible is critical to provide a positive user experience.

Our institution is passionate about high quality evidence based research and has excelled in various fields (Parakh et al. 2020; Pham et al. 2021; Perumal, Antony, and Muthuramalingam 2021; Sathiyamoorthi et al. 2021; Devarajan et al. 2021; Dhanraj and Rajeshkumar 2021; Uganya, Radhika, and Vijayaraj 2021; Tesfaye Jule et al. 2021; Nandhini, Ezhilarasan, and Rajeshkumar 2020; Kamath et al. 2020).In the existing research they didn't identify the deployment speed of the applications and efficient platform for deploying applications. The main aim of our project is to deploy some set of applications on a Novel Microsoft Azure based Docker container and Google Cloud platform based docker container to calculate the deployment time of an application in seconds.Decreasing the deployment time of an application makes the application more efficient.

MATERIALS AND METHODS

The research work was performed in the Cloud Computing Laboratory, Department of Computer Science and Engineering, Saveetha School of Engineering, SIMATS (Saveetha Institute of Medical and Technical Sciences). The proposed work contains two groups. Group 1 is taken as a Novel Microsoft Azure based Docker container and group 2 as Google cloud platform based docker container. The Google cloud platform based docker container and Novel Microsoft Azure based Docker container were executed and evaluated a different number of times with a sample size of 20. The minimum power analysis for G-Power calculation is fixed at 0.8 and the maximum accepted error is fixed at 0.05 . Same set of 20 Containerized applications are used to calculate the deployment time of application in each platform to get the accuracy of each platform.

Testing setup for this proposed system used a Vmware workstation. Vmware workstation is used to create a guest OS. Hardware configuration for this proposed system is Intel core i5 8th gen processor and requires 4GB random access memory and 256GB Solid state drive used. The configuration of the system is the Windows 10 operating system.

Procedure for deploying application on microsoft azure based docker container

Step-1: Create Azure account

To gain access to the Microsoft Azure management portal, you must have an Azure subscription and a Microsoft account associated with that subscription. The subscription provides a way to control the access to and the use of the Azure subscribed service.

Step-2: Create Container Registry

You create a registry by using either the Azure portal or the Azure CLI acr create command.

- az acr create --name myregistry --resource-group mygroup --sku standard --admin-enabled true

Use the CLI to upload the Docker file and additional files that make up your image instead of developing it yourself and uploading it to Container Registry. The image will then be built for you by Container Registry.

- az acr build --file Dockerfile --registry myregistry --image myimage

Step-3:Create Docker image and upload it to Azure Container Registry

- git clone https://github.com/MicrosoftDocs/mslearn-deploy-run-container-app-service.git
- az acr build --registry <container_registry_name> --image webimage

Step-4:Create web app from a Docker image

An Azure-based web app's hosting environment is provided by Azure App Service. App Service can be configured to retrieve the web app's image from an Azure Container Registry repository.

Step-5:Update the image and deploy app
The following command shows how to create a task called build webapp.

- az acr task create --registry <container_registry_name> --name buildwebapp --image webimage --context https://github.com/MicrosoftDocs/mslearn-deploy-run-container-app-service.git --file Dockerfile --git-access-token <access_token>

Algorithm for deploying application using Microsoft azure based docker image container is given below:

Input: Application A to deploy
A=state_application
C=Container Registry(*C)
R=ResourceGroup(*G)
D=Dockerfile(D)
I=DockerImage(*I)
Create Application(*A) //Function to Create application
Create Application(*A)=new Application(*A)
ResourceGroup(*G) //Function to Create resource group
Create ResourceGroup(*G)=new ResourceGroup(*G)
Create Dockerfile(*D) //Function to create dockerfile
Create Dockerfile(*D)=new Dockerfile(*D)
Create ContainerRegistry(*C) //Function to create Container Registry
Create ContainerRegistry(*C)=new ContainerRegistry(*C)
Create DockerImage(*I) //Function to Create DockerImage
Create DockerImage(*I)=new DockerImage(*I)
if(Enable Docker(A) == true)
Create ContainerRegistry(C)
Create Dockerfile(D)
Configuration of(R,C,D,I)
Run DockerImage(I)
Run Application(A)
End if
Else
"Error while enabling docker"
End Else

Output:
Containerized Application Deployment
Deployment time
Availability

Procedure for deploying application on Google cloud platform based docker container

Step-1: Create Google cloud account

The Google Cloud Free Program comprises the following:Free Tier: All Google Cloud customers can use select Google Cloud products like Compute Engine, Cloud Storage, and BigQuery—free of charge, within specified monthly usage limits.

Step-2:Create simple application

We store our code in a directory called gcp-api(call this anything you like) under the name app.py.

Step-3:Create Docker file

The Dockerfile is a template for creating containers. It instructs Docker on how to rearrange our scripts and files to create a self-contained application.

Step-4:Building the Docker Image

- $docker build -t gcp-api

Deploy with Google Cloud Platform

- $gcloud auth login

This command opens our web browser and allows us to log in to Google Cloud as usual. We configure Docker to used our GCP credentials with:

- $gcloud auth configure-docker

Step-5:Build Container Registry

Before we can use GCR (or any other GCP services), we need to create a project. We can do this by navigating to the project selector page in the GCP console and clicking Create Project.

Finally, we can upload our container to GCR by submitting it to Cloud Build — the GCP service that builds Docker containers.

- gcloud builds submit --tag gcr.io/[PROJECT-ID]/gcp-api

Step-6:Cloud Run

Now we have our Docker container ready; we can deploy it with Cloud Run. We will see the deployment status in the Cloud Run console, which should take no longer than a few minutes.Once complete, we will see a green tick next to our deployment name and our deployment URL next to that.

Algorithm for deploying application using Google Cloud Platform based docker container is given below:

Input: Application A to deploy
A=state_application
D=Dockerfile(D)
C=Container Registry(*C)
I=DockerImage(*I)
G=Connect GoogleCloud

```
Create Application(*A)              //Function to Create application
        Create Application(*A)=new Application(*A)
Create Dockerfile(*D)               //Function to create dockerfile
          Create Dockerfile(*D)=new Dockerfile(*D)
Create ContainerRegistry(*C)        //Function to create Container Registry
        Create ContainerRegistry(*C)=new ContainerRegistry(*C)
Create DockerImage(*I)              //Function to Create DockerImage
        Create DockerImage(*I)=new DockerImage(*I)
if( Enable Docker(A)  == true)
Create Dockerfile(D)
Create ContainerRegistry(C)
Configuration of(R,D,C,I)
Connect(D,I)
Run DockerImage(I)
CloudRun(A)
  End if
```

Else
"Error while enabling docker"
End Else

Output:
Application Deployment
Deployment time
Availability

Statistical Analysis

Statistical software used in the study is IBM SPSS version 26. The independent sample T-test calculation for analysing equal variance, standard error, and levene's test are evaluated. Attributes like platform, Deployment Time, accuracy, availability are dependent variables. Independent sample T-test has been carried out for evaluating the accuracy (Scholl, Swanson, and Fernandez 2016) .

RESULTS

In this proposed system it was observed that the Novel Microsoft Azure based Docker container appears to be better than the Google cloud platform based docker container. Novel Microsoft Azure based Docker container enables the continuous delivery and deployment of large, complex applications. Table 1 represents deployment time and accuracy. Table 2 shows the statistical calculation such as mean, standard deviation and standard error mean for Novel Microsoft Azure based Docker container and Google cloud platform based docker image respectively. The mean, standard deviation and standard error mean for the Novel Microsoft Azure based Docker container are 69.20,6.723,3.007 respectively. The mean, standard deviation and standard error mean for Google cloud platform based docker containers are 64.20,5.975,2.672 respectively.

It is inferred that the mean accuracy for T-test is 69 which is greater than the mean accuracy of comparison architecture which is 64.20. Moreover, the mean accuracy value of the Novel Microsoft Azure based Docker container is around 69.20 which seems to be superior to the Google cloud platform based docker container. In Table 3, it was observed that the Levens test for equality of variance and its significance for the Novel Microsoft Azure based Docker container is 0.025 and 0.877 respectively and standard error difference and confidence interval are lower than Google cloud platform based docker container. Fig. 1 represents the architecture of the proposed system. Mean accuracy and mean loss graph is depicted in Fig. 2. Microsoft Azure based Docker container seems to appear better for deploying applications.

DISCUSSION

The proposed Novel Microsoft Azure based Docker container provides better deployment time compared to Google cloud platform based docker container.Empirical study on how to quickly generate and deploy code on virtualized infrastructure has gotten greater attention as modern Web engineering ideas like continuous deployment and cloud computing have become more relevant (Grance, U. s. Department of Commerce, and Patt-Corner 2012).Containers allow programmes to share an operating system (and, where suitable, binaries and libraries), resulting in deployments that are substantially smaller than hypervisor deployments, allowing for the storage of hundreds of containers on a single physical host (versus a strictly limited number of VMs) (Arundel and Domingus 2019). Because containers use the host OS, restarting or rebooting a container does not result in the host OS being restarted or rebooted(Arundel and Domingus 2019; Ifrah 2020). Base images are used to construct Docker containers. A Docker image might be as simple as the OS essentials or as complex as a pre-built application stack ready to run (Azraq et al. 2017).
Each Dockerfile is a script that uses a series of commands (instructions) and arguments to conduct activities on a base image in order to produce (or form) a new image (Kane and Matthias 2018). They're used to organize deployment artifacts and streamline the entire deployment process.To enhance speed and manage application clustering, software developers prefer PaaS, which will employ a container if one is available for its runtime. If this is not the case, the PaaS will run a container on a virtual machine (Dumpleton 2018). As a result, containers are becoming more common as PaaS grows in popularity.Cloud computing advancements necessitate ongoing examination of the cloud's performance across a wide range of applications. When opposed to Amazon Web Services, Microsoft Azure has lesser bandwidth, which implies that applications can benefit from a faster network and more RAM at a lower cost.

The launch of Google's new Cloud Platform prompted us to submit nine Proof of Concepts (PoC) aimed at demonstrating and testing the platform's capabilities in the context of scientifically-driven activities and requirements.Google Cloud Platform provides a variety of data security, storage, serving, and analysis capabilities. These cloud services create a secure cloud perimeter for data, allowing it to undergo various operations and transformations without ever leaving the cloud ecosystem (Dumpleton 2018; Zhu et al. 2014). Cloud Platform is a collection of modular cloud-based services that can be used to create anything from simple websites to complex multitier web-based apps.When using Docker to create images, each action (that is, each command run, such as apt-get install) creates a new layer on top of the previous one (Turnbull 2014). Using Dockerfiles, commands can be run manually or automatically.

CONCLUSION

The Novel Microsoft Azure based Docker container provides the high availability for application with better deployment time and maintenance compared to Google cloud platform based docker container deployment technique.The deployment time for Novel Microsoft Azure based Docker container is 36,45,39,56,33 (in sec) which is lowest compared to the Google cloud platform based docker container.

DECLARATIONS

Conflict of Interests

No conflict of interest in this manuscript.

Author Contribution

Author SN was involved in data collection, data analysis, manuscript writing. Author SMS was involved in conceptualization, guidance and critical review of manuscript.

Acknowledgments

The authors would like to express their gratitude towards Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences (Formerly known as Saveetha University) for providing the necessary infrastructure to carry out this work successfully.

Funding

We thank the following organizations for providing financial support that enabled us to complete the study.

1. Renaissance Technologies, Chennai.
2. Saveetha University
2. Saveetha Institute of Medical and Technical Sciences.
3. Saveetha School of Engineering.

REFERENCES

1. Arundel, John, and Justin Domingus. 2019. Cloud Native DevOps with Kubernetes: Building, Deploying, and Scaling Modern Applications in the Cloud. "O'Reilly Media, Inc."
2. Azraq, Ahmed, Hala A. Aziz, Uzma Siddiqui, and I. B. M. Redbooks. 2017. Essentials of Application Development on IBM Cloud. IBM Redbooks.
3. Cito, Jürgen, and Harald C. Gall. 2016. "Using Docker Containers to Improve Reproducibility in Software Engineering Research." Proceedings of the 38th International Conference on Software Engineering Companion. https://doi.org/10.1145/2889160.2891057.
4. Denmark, Scott, Andrew Zahrt, William Darrow, Brennan Rose, and Jeremy Henle. n.d. "Computational Methods for Training Set Selection and Error Assessment Applied to Catalyst Design: Guidelines for Deciding Which Reactions to Run First and Which to Run Next." https://doi.org/10.26434/chemrxiv.13239422.v1.
5. De Paolis, Lucio Tommaso, and Patrick Bourdot. 2020. Augmented Reality, Virtual Reality, and Computer Graphics: 7th International Conference, AVR 2020, Lecce, Italy, September 7–10, 2020, Proceedings, Part I. Springer Nature.
6. Devarajan, Yuvarajan, Beemkumar Nagappan, Gautam Choubey, Suresh Vellaiyan, and Kulmani Mehar. 2021. "Renewable Pathway and Twin Fueling Approach on Ignition Analysis of a Dual-Fuelled Compression Ignition Engine." Energy & Fuels: An American Chemical Society Journal 35 (12): 9930–36.
7. Dhanraj, Ganapathy, and Shanmugam Rajeshkumar. 2021. "Anticariogenic Effect of Selenium Nanoparticles Synthesized Using Brassica Oleracea." Journal of Nanomaterials 2021 (July). https://doi.org/10.1155/2021/8115585.
8. Dumpleton, Graham. 2018. Deploying to OpenShift: A Guide for Busy Developers. "O'Reilly Media, Inc."
9. Ganesan, Kamesh, Rithin Skaria, and Frederik Vos. 2020. Hands-On Linux Administration on Azure: Develop, Maintain, and Automate Applications on the Azure Cloud Platform, 2nd Edition. Packt Publishing Ltd.
10. Grance, Tim, U. s. Department of Commerce, and Robert Patt-Corner. 2012. Cloud Computing Synopsis and Recommendations.

Createspace Independent Publishing Platform.
11. Ifrah, Shimon. 2020. "Deploy Docker Container Host on Azure Virtual Machine." Getting Started with Containers in Azure. https://doi.org/10.1007/978-1-4842-5753-1_5.
12. Juell, Kathleen. 2020. From Containers to Kubernetes with Node.js. DigitalOcean.
13. Kamath, S. Manjunath, K. Sridhar, D. Jaison, V. Gopinath, B. K. Mohamed Ibrahim, Nilkantha Gupta, A. Sundaram, P. Sivaperumal, S. Padmapriya, and S. Shantanu Patil. 2020. "Fabrication of Tri-Layered Electrospun Polycaprolactone Mats with Improved Sustained Drug Release Profile." Scientific Reports 10 (1): 18179.
14. Kane, Sean P., and Karl Matthias. 2018. Docker: Up & Running: Shipping Reliable Containers in Production. "O'Reilly Media, Inc."
15. Klaffenbach, Florian, Oliver Michalski, Markus Klein, Mohamed Wali, Namit Tanasseri, and Rahul Rai. 2019. Implementing Azure: Putting Modern DevOps to Use: Transform Your Software Deployment Process with Microsoft Azure. Packt Publishing Ltd.
16. Mouat, Adrian. 2015. Using Docker: Developing and Deploying Software with Containers. "O'Reilly Media, Inc."
17. Nandhini, Joseph T., Devaraj Ezhilarasan, and Shanmugam Rajeshkumar. 2020. "An Ecofriendly Synthesized Gold Nanoparticles Induces Cytotoxicity via Apoptosis in HepG2 Cells." Environmental Toxicology, August. https://doi.org/10.1002/tox.23007.
18. Parakh, Mayank K., Shriraam Ulaganambi, Nisha Ashifa, Reshma Premkumar, and Amit L. Jain. 2020. "Oral Potentially Malignant Disorders: Clinical Diagnosis and Current Screening Aids: A Narrative Review." European Journal of Cancer Prevention: The Official Journal of the European Cancer Prevention Organisation 29 (1): 65–72.
19. Perumal, Karthikeyan, Joseph Antony, and Subagunasekar Muthuramalingam. 2021. "Heavy Metal Pollutants and Their Spatial Distribution in Surface Sediments from Thondi Coast, Palk Bay, South India." Environmental Sciences Europe 33 (1). https://doi.org/10.1186/s12302-021-00501-2.
20. Pham, Quoc Hoa, Supat Chupradit, Gunawan Widjaja, Muataz S. Alhassan, Rustem Magizov, Yasser Fakri Mustafa, Aravindhan Surendar, Amirzhan Kassenov, Zeinab Arzehgar, and Wanich Suksatan. 2021. "The Effects of Ni or Nb Additions on the Relaxation Behavior of Zr55Cu35Al10 Metallic Glass." Materials Today Communications 29 (December): 102909.
21. Sathiyamoorthi, Ramalingam, Gomathinayakam Sankaranarayanan, Dinesh Babu Munuswamy, and Yuvarajan Devarajan. 2021. "Experimental Study of Spray Analysis for Palmarosa Biodiesel-diesel Blends in a Constant Volume Chamber." Environmental Progress & Sustainable Energy 40 (6). https://doi.org/10.1002/ep.13696.
22. Scholl, Boris, Trent Swanson, and Daniel Fernandez. 2016. Microservices with Docker on Microsoft Azure (includes Content Update Program). Addison-Wesley Professional.
23. Sioshansi, Fereidoon P. 2014. Distributed Generation and Its Implications for the Utility Industry. Academic Press.
24. Tesfaye Jule, Leta, Krishnaraj Ramaswamy, Nagaraj Nagaprasad, Vigneshwaran Shanmugam, and Venkataraman Vignesh. 2021. "Design and Analysis of Serial Drilled Hole in Composite Material." Materials Today: Proceedings 45 (January): 5759–63.
25. Turnbull, James. 2014. The Docker Book: Containerization Is the New Virtualization. James Turnbull.
26. Uganya, G., Radhika, and N. Vijayaraj. 2021. "A Survey on Internet of Things: Applications, Recent Issues, Attacks, and Security Mechanisms." Journal of Circuits Systems and Computers 30 (05): 2130006.
27. Zhu, Wei-Dong, Manav Gupta, Ven Kumar, Sujatha Perepa, Arvind Sathi, Craig Statchuk, and I. B. M. Redbooks. 2014. Building Big Data and Analytics Solutions in the Cloud. IBM Redbooks.

TABLES AND FIGURES

Table 1. Application deployment time and accuracy for Novel Microsoft Azure based Docker container and Google cloud platform based docker container.

ITERATION NO (n)	Novel Microsoft Azure based Docker Container		Google cloud platform based docker container	
	Time (in Sec)	Accuracy(%)	Time (in Sec)	Accuracy(%)
1	36	77	43	69
2	45	68	54	63
3	39	69	46	64
4	56	73	61	70

5	33	59	39	55

Table 2. Group statistical analysis of Novel Microsoft Azure based Docker container with mean value of 69.20 and Google cloud platform based docker container with mean value of 64.20 and similarly the results of Standard Deviation and Standard Error Mean are given.

T-Test:

Group Statistics

GROUP		**N**	**Mean**	**STD Deviation**	**STD Error mean**
ACCURACY	**Novel Microsoft Azure based Docker Container**	5	69.20	6.723	3.007
	Google cloud platform based docker container	5	64.20	5.975	2.672

Table 3. Independent Sample T-test Results with confidence interval of 95% and level of significance greater than 0.05 (Novel Microsoft Azure based Docker Container deployment seems to appear better for the deployment of containerized application).

		Levene's Test for Equality of Variance		**Levene's Test for Equality of Variance**						
	Equal Variances	**F**	**Sig.**	**t**	**df**	**Sig.(2-tailed)**	**Mean Difference**	**Std. Error Differ -ence**	**95% Confidence Interval of the Difference**	
									Lower	**Upper**
ACCURACY	**Assumed**	.025	.877	1.243	8	.249	5.000	4.022	-4.276	14.276
	Not Assumed			1.243	7.891	.250	5.000	4.022	-4.298	14.298

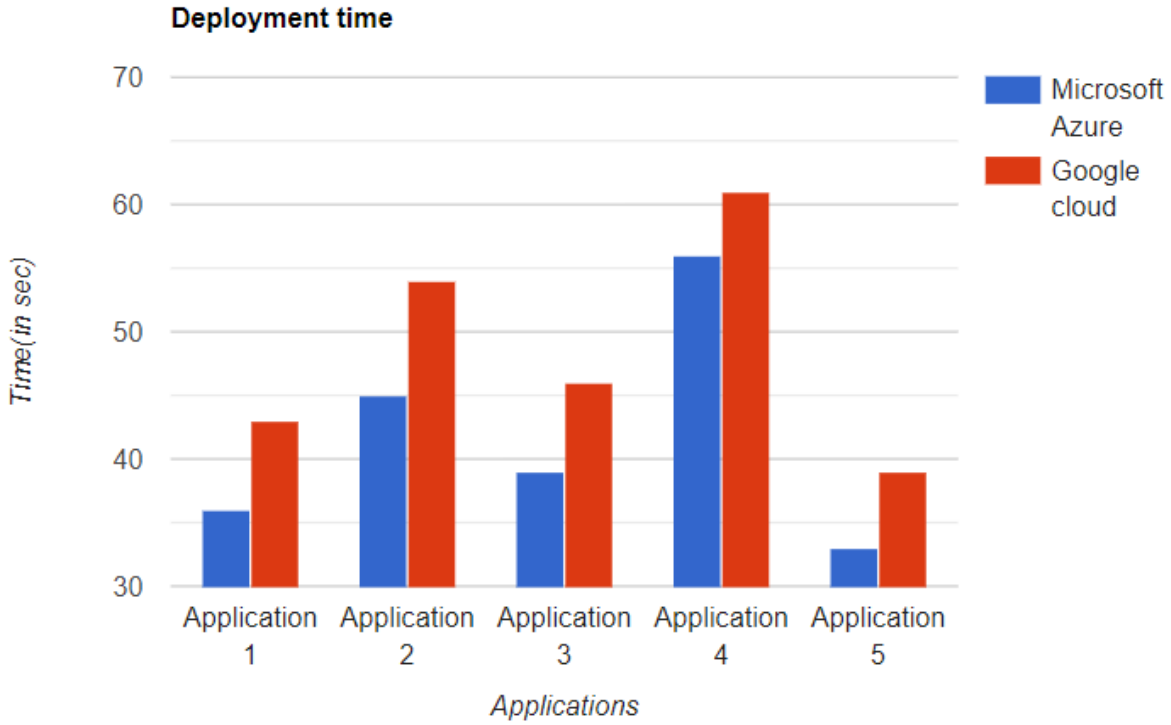

Fig. 1. Comparing the deployment time between Novel Microsoft Azure based Docker container and Google cloud platform based docker container using the Bar graph.

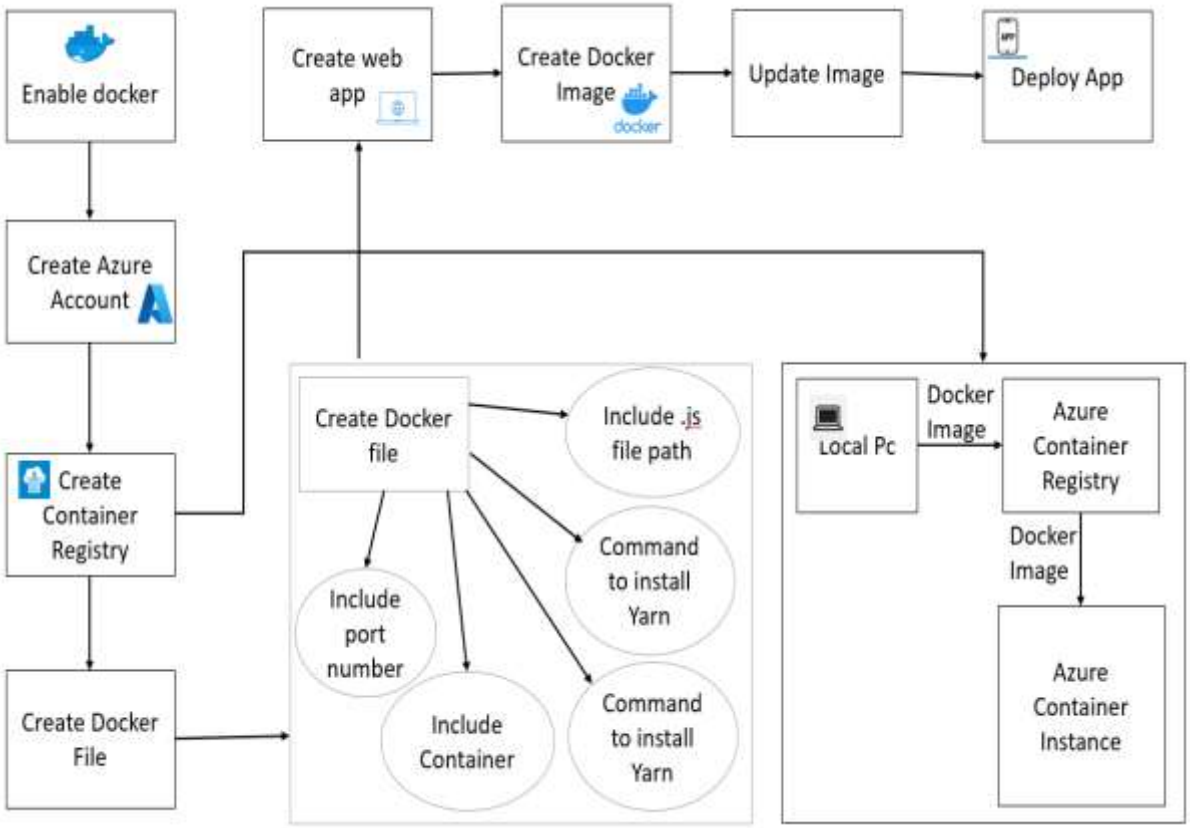

Fig. 2. Architecture for Deploying a Containerized application in Docker using different platforms. Docker, Dockerfile, ContainerRegistry are the important components in the architecture.

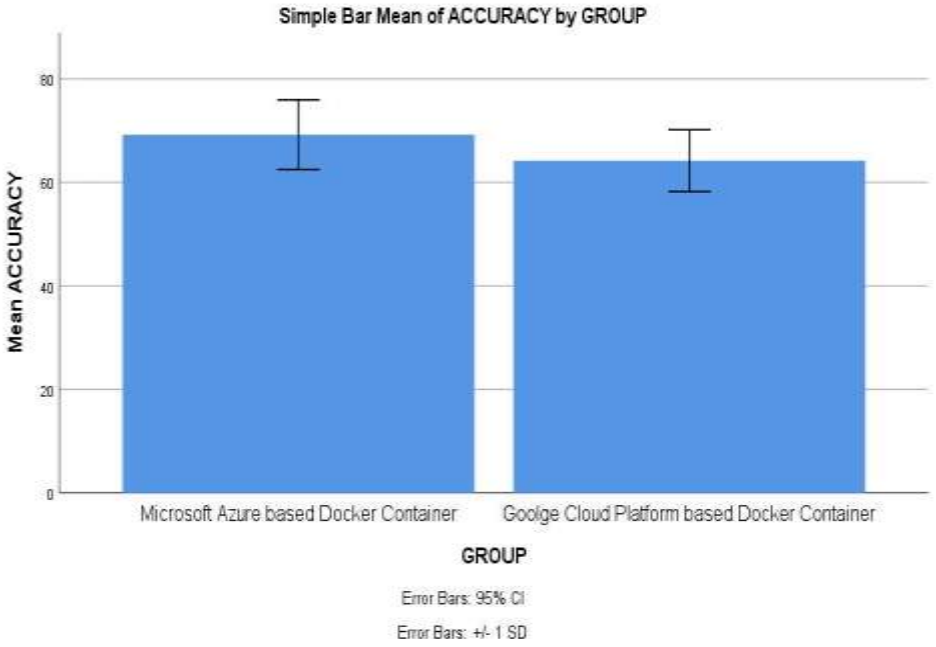

Fig. 3. Bar graph analysis of Novel Microsoft Azure based Docker Container and Google cloud platform based docker container. Graphical representation shows the mean Accuracy of 69.20% and 64.20% for the proposed platform (Microsoft Azure) and Google cloud respectively. X-axis : MZ vs GC, Y-axis : Mean Accuracy ± 1 SD.