

# **TITLE: “PEGA on Google cloud platform”**

**A Report Submitted in Partial Fulfillment of the Requirements for the  
6<sup>th</sup> Semester B.Tech. Summer Internship  
(CS 497)**

**By  
Ishika Raj (2012050)**

**Under the Guidance of  
Prashant Kumar  
(Asst. Manager, OPTUM)**



**Computer Science and Engineering Department  
NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR  
Assam**

**Duration (06/06/2023-05/08/2023)**



**To Whom It May Concern**

**Date: 10-Aug-2023**

**Name: ISHIKA RAJ**

**Unique ID: 002053137**

This is to certify that **ISHIKA RAJ D/o SRI RAJAN** has worked with **Optum Global Solutions (India) Pvt. Ltd** as an Intern with Technology division from **6-Jun-2023** to **5-Aug-2023**. Her conduct during the aforesaid period was found to be satisfactory.

We wish her all the best in her future endeavor.

**For Optum Global Solutions (India) Pvt. Ltd**

A handwritten signature in black ink, appearing to read "Ashish Garg", with a horizontal line extending from the end of the signature.

---


**Ashish Garg**

**Senior Director - Recruitment**

# DECLARATION

## **Title: PEGA on Google cloud platform**

I declare that the presented work represents largely my own ideas and work in my own words. Where others' ideas or words have been included, I have adequately cited and listed in the reference materials. I have adhered to all principles of academic honesty and integrity.

<b>Name</b>	<b>Scholar ID</b>	<b>Scanned Signature</b>
Ishika Raj	2012050	

Department of Computer Science and Engineering

**National Institute of Technology Silchar, Assam**

# ACKNOWLEDGEMENT

I would like to express my sincere gratitude to all those who contributed to the successful completion of the DevOps project involving the migration of the PEGA application to Google Cloud. This project marked a significant milestone in our pursuit of enhanced flexibility and choice for our users.

First and foremost, I extend our heartfelt thanks to the entire team for their dedication, hard work, and collaborative spirit. Each team member played a crucial role in the planning, execution, and seamless transition of the application, showcasing exceptional technical expertise and problem-solving skills.

I would like to extend my appreciation to Prashant Kumar Asst. Manager, whose guidance and support were invaluable throughout the project. His insights and encouragement fuelled our motivation and paved the way for a successful outcome.

Additionally, we would like to express our gratitude to OPTUM for providing the necessary resources and creating an environment that fosters innovation and growth. I acknowledge the broader community of professionals and experts whose knowledge-sharing and collaboration in the field of DevOps have been a constant source of inspiration.

Last but not the least, I also extend my gratitude and appreciation to the Computer Science and Engineering Department of National Institute of Technology Silchar for giving us an opportunity to work on this project as an intern.

Thank you all for the contributions and commitment to excellence.

**Sincerely,**

**Ishika Raj (2012050)**

# **ABSTRACT**

This DevOps initiative orchestrates the migration of the PEGA application from on-premise servers (Azure and HCC) to Google Cloud, providing users with the flexibility to choose their hosting platform. Under Prashant Kumar the project executed meticulous planning and testing for a seamless transition. Supported by Optum UGH, this collaborative effort signifies a commitment to adaptability, user-centricity, and innovation within the DevOps landscape.

**Name: Ishika Raj**

**Scholar ID: 2012050**

## **LIST OF ABBREVIATIONS**

VM	Virtual Machines
K8	Kubernetes
GCP	Google Cloud platform
HCC	HealthCare Cloud (Optum Cloud)

## **LIST OF TABLES**

### **TABLE:**

1. Cost Comparison of the different cloud services for PEGA.....	34
--	----

## **LIST OF FIGURES**

### **FIGURE:**

1. GCP architecture.....	13
2. PEGA application on google cloud platform.....	22
3. Logged in application on google cloud platform.....	22
4. Custom Dashboard.....	23

# TABLE OF CONTENT

Declaration.....	iii
Acknowledgement.....	iv
Abstract.....	v
List of Abbreviations.....	vi
List of Tables.....	vii
List of figures.....	vii
1. Introduction.....	10
1.1 Motivation.....	10
1.2 Problem Statement.....	10
1.3 Objective.....	10
2. Proposed Work.....	11
2.1 What is Pega.....	11
2.2 About Google Cloud.....	11
2.3 GCP Architecture.....	13
2.4 Tools and Technologies.....	14
2.4.1 Dockers.....	14
2.4.2 Terraform.....	15
2.4.2.1 The different Infrastructure as Code (IaC).....	16
2.4.3 Kubernetes.....	17
2.4.4 Helm.....	18
2.4.5 Jenkins.....	19
2.4.6 Github Actions.....	20
2.4.7 Rally Board.....	20
2.4.8 Pega.....	20
2.4.9 Pega SRS.....	20
2.4.10 Elasticsearch.....	21
2.4.11 Fluentd.....	21
2.4.12 Hazelcast.....	21
2.5 Dashboard.....	23
3. Results.....	24
4. Conclusion.....	25



# **1. INTRODUCTION**

## **1.1 Motivation**

This intern was engaged in the deployment of the PEGA application, a vendor application designed to transition from the use of Virtual Machines (VM) to cloud infrastructure. The primary motivation behind this endeavor is to render the team proficient in cloud technologies, providing them the autonomy to select their preferred cloud platform. This transformative project is driven by the overarching goal of enhancing efficiency and adaptability within the organization.

## **1.2 Problem Statement**

The current technological landscape within our organization relies on Virtual Machines (VM) for the deployment of the PEGA application. However, this approach poses challenges in terms of scalability, flexibility, and resource optimization. The need for a more agile and efficient solution has led to the initiation of the PEGA Deployment Project. The project aims to address the limitations of the existing VM-based deployment model by transitioning to a cloud infrastructure. This shift is motivated by the desire to enhance scalability, streamline operations, and provide our internal PEGA team with the freedom to choose and leverage diverse cloud platforms. The problem at hand is to seamlessly migrate from VM to cloud deployment, ensuring a smooth transition that maximizes efficiency, minimizes downtime, and aligns with the organization's long-term technological goals."

## **1.3 Objective**

The primary objective of the PEGA Deployment Project is to transition from the current Virtual Machine (VM)-based deployment model to a cloud infrastructure, thereby addressing limitations in scalability, flexibility, and resource optimization. This strategic initiative aims to enhance operational efficiency and empower the internal PEGA team with the freedom to choose and leverage diverse cloud platforms. The project seeks to achieve a seamless migration process, minimizing downtime and disruptions while maximizing the benefits of cloud deployment

## **2. PROPOSED WORK**

### **2.1 WHAT IS PEGA**

1. Pega is a low code platform for AI-powered decisioning and workflow automation.
2. Pega is mostly used for building internal tools.
3. Since it is mostly a no code or low code tool, employees with less development experience can build applications for their needs.
4. It can be deployed on any platform and is valuable to Optum due to its comprehensive BPM and CRM capabilities.

### **2.2 ABOUT GOOGLE CLOUD.**

Google Cloud is a robust and versatile cloud computing platform provided by Google, offering a comprehensive suite of services for building, deploying, and scaling applications. With a focus on infrastructure, platform, and application services, Google Cloud provides solutions for a wide range of computing needs. Compute Engine allows users to deploy virtual machines, while App Engine offers a fully managed platform for application deployment. Kubernetes Engine provides a managed Kubernetes service for containerized applications. Google Cloud also excels in storage and databases with services like Cloud Storage, Cloud SQL, and Cloud Bigtable. It emphasizes networking and security through features like Virtual Private Cloud (VPC) and Cloud Load Balancing. Developer tools like Cloud Build and Cloud Source Repositories streamline the development process. Stackdriver offers monitoring and diagnostics, ensuring optimal performance.

## WHY GOOGLE CLOUD?

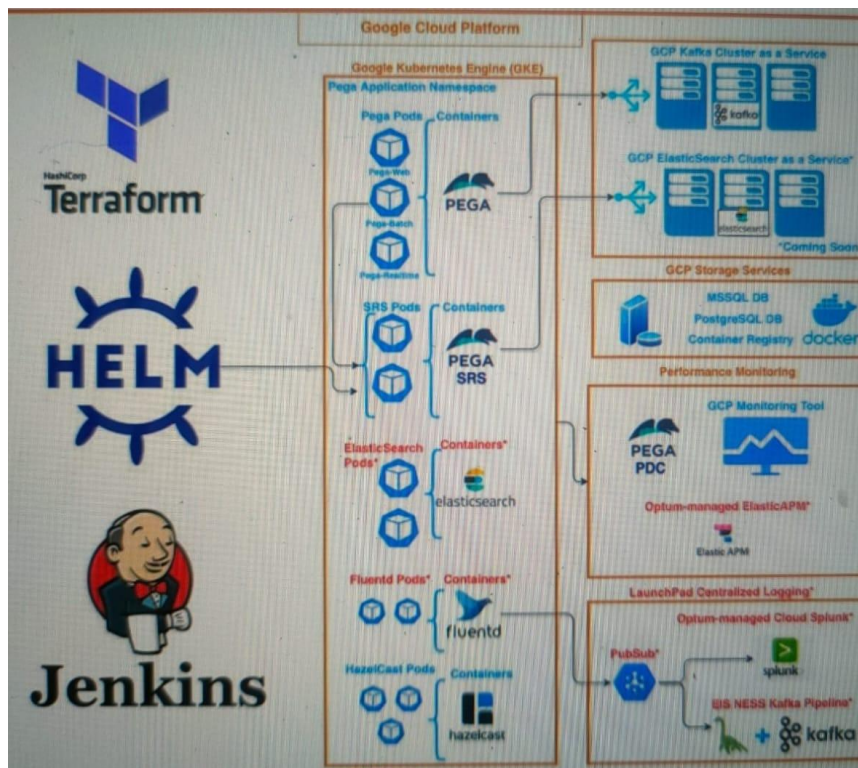
Choosing Google Cloud for PEGA deployment can be influenced by several factors, Here are some reasons why Google Cloud might be chosen for deploying PEGA applications:

1. Scalability and Performance:
  - a. Google Cloud grows and shrinks easily based on what's needed, making it great for PEGA apps that have changing workloads.
2. Global Network and Low Latency:
  - a. Google Cloud is everywhere, so PEGA apps can be close to users worldwide, making them faster and more responsive.
3. Integration with GCP Services:
  - a. Google Cloud has extra tools that can make PEGA apps even better, like storing data in Cloud Storage, analyzing it with BigQuery, and more.
4. Security and Compliance:
  - a. Google Cloud takes security seriously, with strong protections and rules to follow.
5. Cost Optimization:
  - a. Google Cloud has flexible pricing, letting organizations pay for what they use. This helps manage costs effectively.
6. Managed Services and DevOps Support:
  - a. Google Cloud has services that do a lot of the hard work for the app, making it easier to deploy and manage PEGA apps. This fits well with modern DevOps practices.
7. Open Source and Multi-Cloud Capabilities:
  - a. Google Cloud likes open-source tools and works well with other clouds.

Choosing Google Cloud for PEGA deployment is about getting a flexible, fast, and secure platform with extra tools and innovation options. It's like having a high-tech playground for the PEGA application.

## 2.3 GCP ARCHITECTURE:

Google Cloud Platform (GCP) boasts a sophisticated architecture that forms the backbone of its cloud services. At its core are virtual machines (VMs) managed by Compute Engine, serving as digital workhorses for diverse tasks. GCP's Storage offerings, especially Cloud Storage, act as organized repositories for data, ensuring accessibility and scalability. The network infrastructure, including Virtual Private Clouds (VPCs), facilitates seamless communication among resources globally. Specialized tools like App Engine simplify app development, Cloud SQL manages databases efficiently, and BigQuery accelerates data analysis. Security measures, such as Identity and Access Management (IAM) and robust encryption, fortify the digital playground.



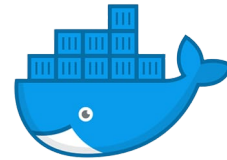
**Fig1: The GCP architecture that had been used in OPTUM.**

It's important to note that the specific use and integration of these technologies in Optum's systems would depend on the organization's unique requirements, goals, and technology stack. Each of these tools plays a role in enhancing efficiency, data management, and system performance in the context of healthcare and business operations. The details about each of them is discussed in the upcoming section.

## 2.4 TOOLS AND TECHNOLOGIES:

### 2.4.1 DOCKERS:

Docker is a platform that enables the development, deployment, and running of applications in containers. Containers are lightweight, standalone, and executable software packages that include everything needed to run a piece of software, including the code, runtime, libraries, and system tools. Docker provides a consistent and reproducible environment, ensuring that an application runs the same way across different environments, from development to testing to production.



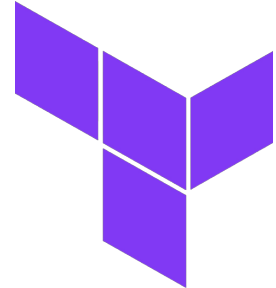
Docker plays a crucial role in the PEGA Deployment Project, facilitating the transition from Virtual Machines (VM) to a cloud infrastructure. Here's how Docker is important for the project:

1. Containerization: Docker packages PEGA and dependencies into lightweight, portable containers for consistent deployment across environments.
2. Consistency: Ensures uniform deployment across varied environments, eradicating "it works on my machine" issues.
3. Isolation and Security: Containers run independently, bolstering security by limiting vulnerabilities' impact and enabling resource constraints.
4. Scalability: Facilitates seamless scaling of PEGA application with automated orchestration, crucial for dynamic cloud workloads.
5. Resource Efficiency: Lightweight containers optimize resource usage, a key advantage in cost-effective cloud deployments.
6. Portability: Highly portable containers simplify migration between on-premises VMs and diverse cloud platforms.
7. Version Control and Rollbacks: Supports versioning for tracking changes and enables quick rollbacks, minimizing downtime and ensuring reliability.

In essence, Docker's containerization streamlines deployment, enhances security, and aligns with cloud-native principles, making it indispensable in transitioning from VM to cloud infrastructure.

## 2.4.2 TERRAFORM :

Terraform is an open-source Infrastructure as Code (IaC) tool developed by HashiCorp. It allows users to define and provision infrastructure using a declarative configuration language. Terraform enables the creation, modification, and versioning of infrastructure in a safe and efficient manner, making it a valuable tool for managing the cloud infrastructure in the PEGA Deployment Project. Terraform is used to define the cloud infrastructure, networking, security groups, and any other resources required for the deployment. It ensures that the infrastructure is provisioned consistently across different environments, aligning with the project's goal of transitioning from Virtual Machines to a cloud infrastructure. Here's how Terraform is used in the project:



1. Infrastructure as Code (IaC):
  - a. Terraform uses declarative language to define the desired state of the infrastructure. This includes specifying the cloud resources, networking components, security configurations, and other elements needed for the PEGA application's deployment.
2. Cloud-agnostic Configuration:
  - a. Terraform supports multiple cloud providers, including AWS, Azure, Google Cloud, and others. This allows the project to be cloud-agnostic, meaning the same Terraform configurations can be used to provision infrastructure across different cloud platforms.
3. Modularity and Reusability:
  - a. Terraform configurations can be modularized, allowing the creation of reusable modules for common infrastructure patterns. This promotes consistency across different environments and simplifies the management of complex infrastructures.
4. Scalability and Elasticity:
  - a. With Terraform, it's easy to scale the infrastructure up or down based on demand. This is particularly important in cloud environments where resource needs can change dynamically. Terraform allows for the definition of scalable and elastic infrastructure configurations.

### 2.4.2.1 The different Infrastructure as Code (IaC):

IaC be implemented for various components in the PEGA Deployment Project , these IaC components leverage Terraform to define, provision, and manage critical infrastructure elements some of the IaC implemented in this projects are:

1. VPC Network:
  - a. Objective: Create a Virtual Private Cloud (VPC) network to isolate and organize resources.
  - b. Terraform Code:
    - i. Defines VPC parameters such as name and CIDR block.
    - ii. Uses the AWS provider to provision the VPC.
    - iii. Ensures consistency in network configuration across environments
2. GKE Cluster (Google Kubernetes Engine):
  - a. Objective: Provision a Google Kubernetes Engine (GKE) cluster for container orchestration.
  - b. Terraform Code:
    - i. Specifies cluster name and node pool configurations.
    - ii. Utilizes the Google Cloud provider to create the GKE cluster.
    - iii. Enables scalability and efficient management of containerized applications.
3. Artifact Repository:
  - a. Objective: Establish an artifact repository for storing and versioning application artifacts.
  - b. Terraform Code:
    - i. Defines the repository name and sets access controls.
    - ii. Uses the AWS S3 bucket to create a private artifact repository.
    - iii. Supports versioning for artifacts and facilitates artifact lifecycle management.
4. PostgreSQL Database:
  - a. Objective: Set up a PostgreSQL database to support the PEGA application.
  - b. Terraform Code:
    - i. Specifies database name, user credentials, and other configurations.
    - ii. Utilizes the AWS RDS service to provision a PostgreSQL database.
    - iii. Ensures consistency and repeatability in database creation.
5. Venafi (Certificate Management):
  - a. Objective: Integrate with Venafi for automated certificate management.
  - b. Terraform Code:
    - i. Configures Venafi URL and API token for authentication.
    - ii. Uses the Venafi provider to request and manage SSL certificates.
    - iii. Automates certificate issuance and renewal for secure communication.

## 2.4.3 KUBERNETES

Kubernetes (often abbreviated as K8s) is an open-source container orchestration platform that automates the deployment, scaling, and management of containerized applications. It provides a robust and flexible framework for orchestrating containers, enabling efficient management of microservices and containerized workloads. In the PEGA Deployment Project, Kubernetes is likely used to enhance the deployment and scalability of the PEGA application. Here's how Kubernetes is typically utilized in such a project:



1. Container Orchestration:
  - a. Role in Project: Kubernetes orchestrates the deployment and management of Docker containers that encapsulate the PEGA application and its dependencies.
  - b. How it Works: Kubernetes automates tasks such as starting, stopping, scaling, and updating containers, ensuring that the PEGA application runs reliably and efficiently.
2. Scaling and Load Balancing:
  - a. Role in Project: Kubernetes allows dynamic scaling of the PEGA application by adjusting the number of running containers based on demand.
  - b. How it Works: Through features like Horizontal Pod Autoscaling, Kubernetes can automatically scale the number of PEGA application instances to handle varying workloads. Load balancing ensures even distribution of traffic across these instances.
3. Service Discovery and Networking:
  - a. Role in Project: Kubernetes simplifies service discovery and communication between different components of the PEGA application.
  - b. How it Works: Kubernetes Services abstract away the underlying network details, providing a consistent way for containers to communicate with each other. This is crucial for microservices architecture.
4. Rolling Updates and Rollbacks:
  - a. Role in Project: Kubernetes facilitates seamless updates and rollbacks of the PEGA application without causing downtime.
  - b. How it Works: Rolling updates allow new container versions to be gradually deployed while old versions are phased out, ensuring continuous availability. If issues arise, Kubernetes supports easy rollbacks to previous versions.



## 2.4.4 HELM:

Helm is a package manager for Kubernetes applications, allowing users to define, install, and upgrade even the most complex Kubernetes applications. It simplifies the deployment and management of applications on Kubernetes clusters by providing a standardized way to package, version, and deploy applications, along with their dependencies. In the PEGA Deployment Project, Helm can play a significant role in streamlining the deployment and management of the PEGA application within a Kubernetes environment. Here's how Helm helps in this project:



1. The Helm chart is a versioned package that contains all the necessary information for deploying and managing the PEGA application on a Kubernetes cluster. This simplifies the distribution and sharing of applications.
2. Users can roll out new versions of the PEGA application using Helm, and if issues arise, Helm supports easy rollbacks to previous versions. This ensures reliable application updates and maintenance.
3. Helm charts can specify dependencies on other charts, ensuring that all necessary components are deployed together. This simplifies the handling of complex application architectures.
4. Helm charts can be stored in repositories, either public or private, making it easy to share and distribute charts across teams or organizations. This is beneficial for maintaining a central repository of applications.
5. Users in the PEGA Deployment Project can leverage existing Helm charts from the Helm Hub or other repositories, saving time and effort in creating and maintaining deployment configurations.

## 2.4.5 JENKINS

Jenkins is an open-source automation server that facilitates the continuous integration and continuous delivery (CI/CD) of software projects. It helps automate various stages of the software development lifecycle, from building and testing to deploying and monitoring applications. In the context of the PEGA Deployment Project, Jenkins plays a crucial role in streamlining and automating the development and deployment processes. Here's how Jenkins is typically used and how it contributes to the project.

various Jenkins jobs are created to automate key tasks in the software delivery and infrastructure provisioning process. Here are descriptions of the specific Jenkins jobs that had been created:



**Jenkins**

### 1. Jenkins Job: Pushing and Pulling Images into Artifact Repository:

- Automate the process of pushing Docker images to the artifact repository after a successful build.
- Automate the process of pulling Docker images from the artifact repository during deployment.

### 2. Jenkins Job: Creating Terraform Infrastructure

- Automate the creation of infrastructure components defined in Terraform scripts.
- Support the dynamic provisioning of resources such as VPC, network, and other cloud services.

### 3. Jenkins Job: Deploying Terraform Infrastructure

- Automate the deployment of the PEGA application and related components onto the provisioned infrastructure.

## 2.4.6 GITHUB ACTIONS:

GitHub Actions has been integral to the success of our PEGA Deployment Project, streamlining our software development and deployment processes. Its automation capabilities, consistency, and collaborative features have streamlined the CI/CD pipeline, contributing to the project's efficiency and reliability. Some of the work includes:

1. Continuous Integration (CI):
  - a. GitHub Actions automates building and testing the PEGA application with each code update, ensuring code quality early in the development cycle.
2. Docker Image Management:
  - a. Automated workflows push new Docker images to our repository after successful builds and pull them for deployment, ensuring we use the right versions.
3. Terraform Infrastructure Setup:
  - a. Workflows handle the provisioning of essential infrastructure using Terraform, such as VPC networks and clusters required for the PEGA application.
4. Kubernetes Deployment:
  - a. GitHub Actions deploys the PEGA application to Kubernetes clusters. Workflows manage tasks like fetching code, setting variables, and executing deployment scripts.

## 2.4.7 RALLY BOARD

Rally, now known as Broadcom Rally, is a popular Agile project management tool designed to help teams plan, track, and manage their work efficiently. The tool provides a collaborative platform for Agile and DevOps teams, offering features to support iterative development and continuous improvement.

**2.4.8 PEGA:** Pega is a leading platform for business process management (BPM) and customer relationship management (CRM). It offers tools for building and management applications with a focus on optimizing business processes and improving customer experience.



**2.4.9 PEGA SRS (System Requirements Specification):** PEGA SRS refers to the System Requirements Specification documentation for PEGA applications. It outlines the detailed requirements, functionalities, and specifications that the system must meet to fulfill business objectives. This document serves as a blueprint for the development and deployment of PEGA applications.

**2.4.10 Elasticsearch:** Elasticsearch is a powerful and scalable search and an analytics engine. In the context of Optum, Elasticsearch may be used for indexing and searching large volumes of healthcare and business data, providing efficient and real-time access to information.



**2.4.11 Fluentd:** Fluentd is an open-source data collector that allows the unified logging layer. It simplifies the collection and transportation of log data from various sources to different destinations. In Optum, Fluentd could play a role in aggregating and managing logs from diverse applications and systems for centralized analysis.



**2.4.12 Hazelcast:** Hazelcast is an in-memory data grid and computing platform. It enables the distributed storage and processing of data across multiple nodes. In an Optum setting, Hazelcast might be used to enhance performance and scalability by distributing and managing data in-memory across a network of connected nodes.

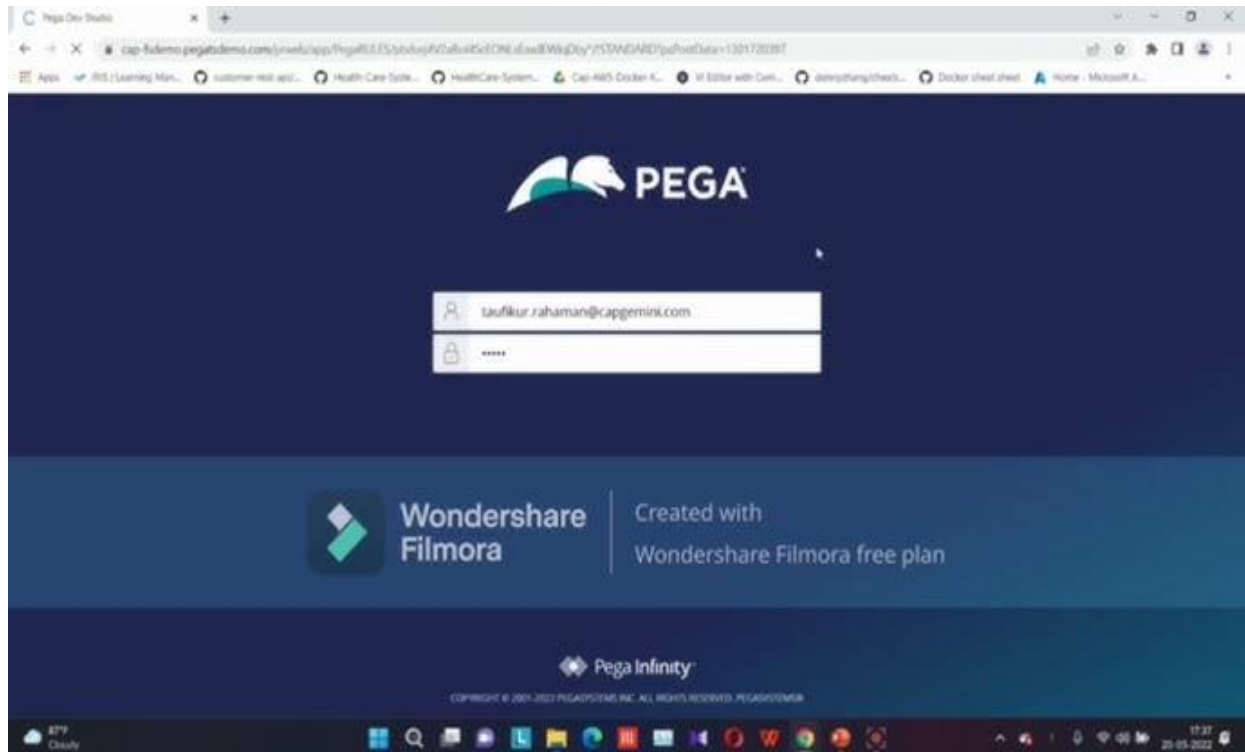


Fig 3: PEGA APPLICATION ON GOOGLE CLOUD:

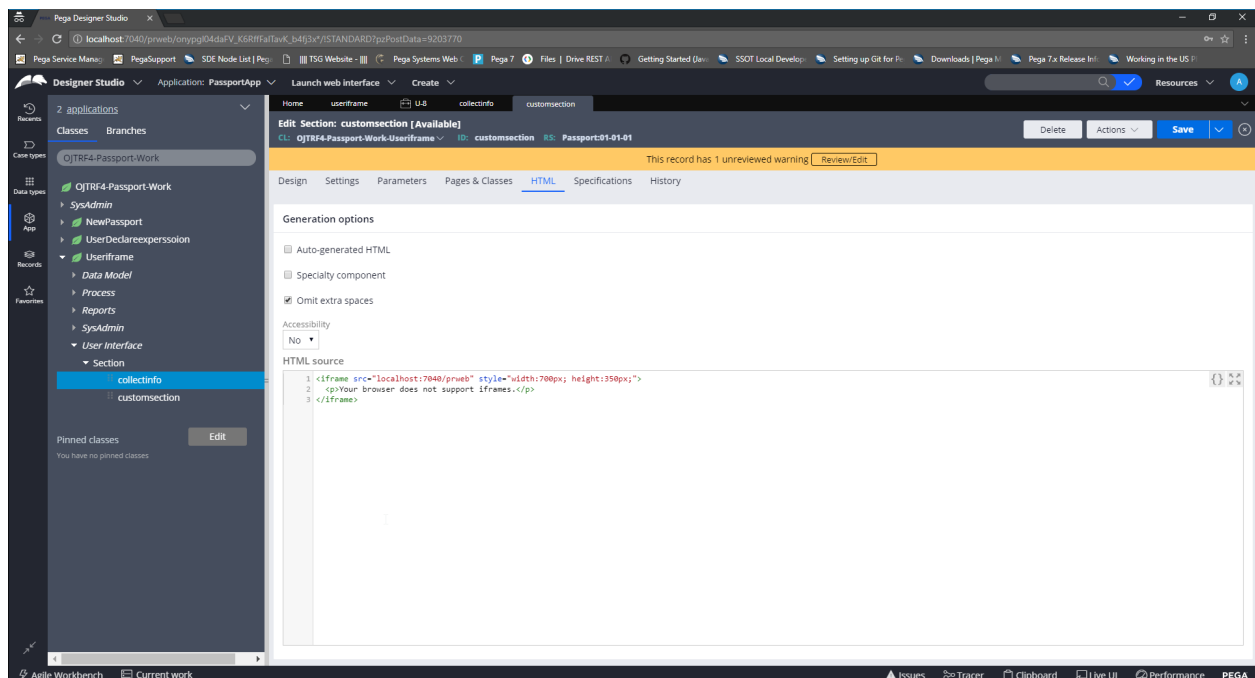


Fig 3: LOGGED IN PEGA APPLICATION ON GOOGLE CLOUD:

## 2.5 DASHBOARD

Once the deployment is completed , the next step involves the maintenance and monitoring of the infrastructure created , for this we use a dashboard.It combines the various widgets like charts , matrices , logs to provide a comprehensive view of the resources status.GCP allows us to created custom dashboards apart from the already existing ones. Some widgets were created which have the feature to add alerts , that enables to quickly notify the issue and address them before they become critical

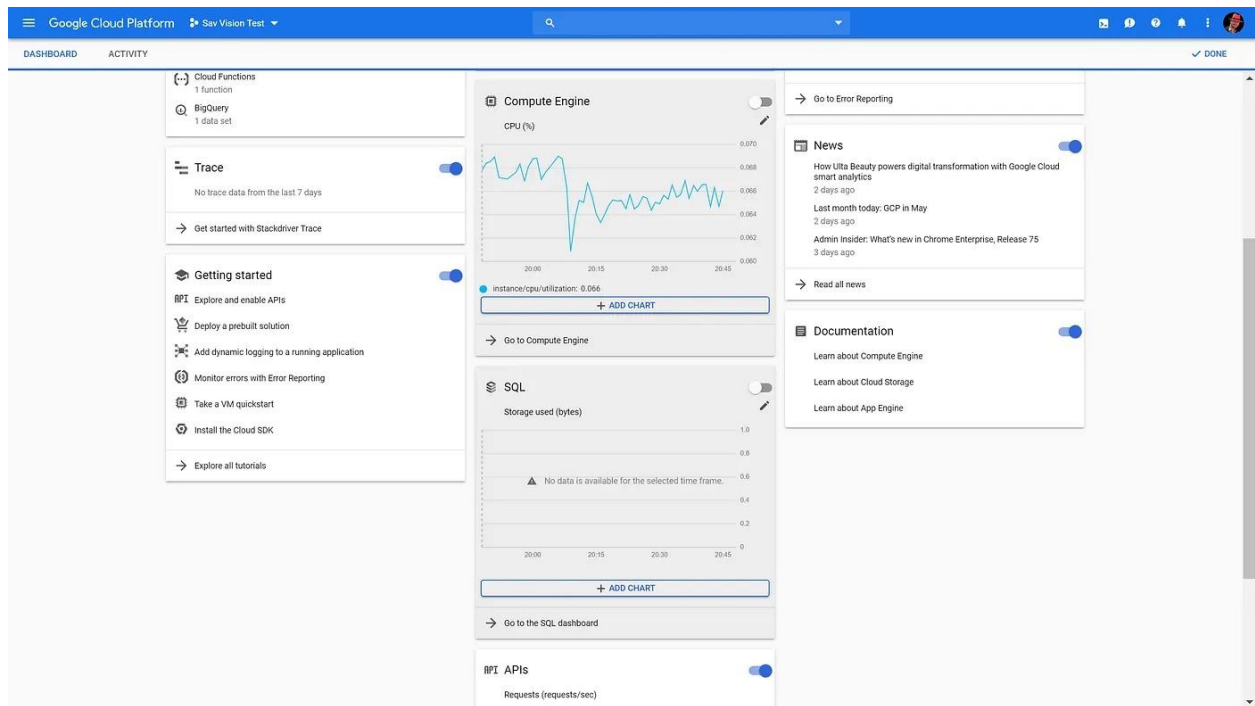


Fig 4: CUSTOM DASHBOARD IN GCP

### 3. RESULTS

#### **.FINANCIAL ASPECT OF GOOGLE CLOUD:**

The financial aspect of using Google Cloud Platform (GCP) involves understanding the pricing model, optimizing costs, and planning for budget management. Here's an overview of key financial considerations for using GCP:

1. GCP typically follows a pay-as-you-go pricing model, where you pay for the resources and services you use. This flexibility is beneficial for scaling resources based on demand.
2. GCP offers sustained use discounts for certain resources. If you continuously use a virtual machine (VM) for a month, you automatically get a discount.
3. Set up budget alerts to receive notifications when costs reach predefined thresholds, helping to avoid unexpected expenses.

CLOUD PROVIDER	GCP	HCC	AZURE
INFRASTRUCTURE COST			
Monthly	\$1656	\$3168	\$1850
Yearly	\$19872	\$38016	\$22200
TOTAL COST (using usage)			
Monthly	X	\$3061	\$3817
Yearly	X	\$43932	\$45604

\*All the costs are calculated in this table .

\* The total usage cost is yet to be calculated after it had been successfully deployed and used for a considerable amount of time , as for now we have only worked for the infrastructure creation thus , after a year of usage and monitoring of the application we would be able to predict the total price on GCP , which is expected to be cheaper than HCC and Azure

## 4. CONCLUSION

During the internship , I was exposed to many new areas of how things actually work in an enterprise and how theoretical knowledge on some technology stacks are practically employed in use for business around the world.

I was also able to work on my team working skills , Everytime a task was assigned to us on the rally board I had to first gain some insights of that particular domain and later implement it within some prescribed time frame, so the challenges were the best motivation for our work throughout the journey.

Besides, working on the Clouds gave me the opportunity to dive deep into the domain of Devops to gain knowledge, and learn about most of the famous clouds like Azure , AWS etc.

I got to interact with the working professionals all around the Globe, and we were fortunate enough to be a part of the Global team .My works were praised by the senior managers, developers, mentors , as well as the technical head of the department.

All in all it was a thrilling and a great learning experience to get some insight into the corporate world and having worked for such a valuable project which had its lows and highs eventually at different circumstances.

Thankyou

ISHIKA RAJ

Scholar ID:2012050