**CONTAINER ORCHESTRATION SYSTEM UTILIZING KUBERNETES.**

**SCOPE:**

To Orchestrate an application and a database using Kubernetes

**PURPOSE:**

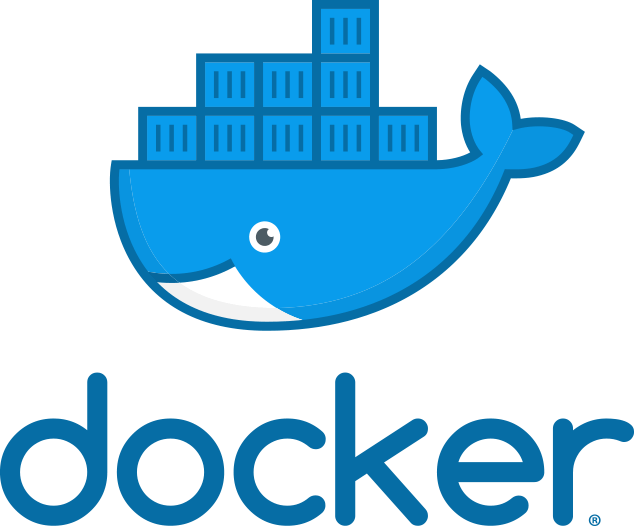
Deployment , Scaling and Management of containerized applications across

a distributed system.

**TOOLS AND TECHNOLOGIES USED:**

1. **DOCKER:**

Docker is an open-source platform for building, shipping, and running containerized applications**.**



Docker has become a popular tool for application development and deployment, as it enables developers to package their applications into containers and deploy them to any platform that supports Docker, from local development environments to public clouds. It also provides a wide range of integrations and extensions, allowing users to customize and extend the platform as needed.

1. **GITHUB:**

GitHub is a web-based platform used for version control and collaboration on software development projects. It provides a central repository for code and allows developers to work together on projects, track changes to code over time, and collaborate on new features or bug fixes.



1. **JIRA:**

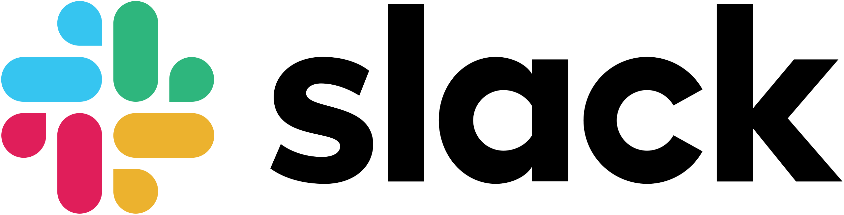
Jira is a software product developed by Atlassian that provides issue and project tracking, bug tracking, and project management capabilities.



It is a popular tool for Agile software development and is used by many software development teams to plan, track, and release software products.

1. **SLACK:**

Slack is a cloud-based collaboration platform that provides chat and communication tools for teams. It is designed to improve communication and collaboration within teams and across organizations, and provides a central location for team communication.



1. **PROMETHEUS:**

Prometheus is an open-source monitoring and alerting system that is widely used in the Kubernetes ecosystem for monitoring and alerting on containerized applications.

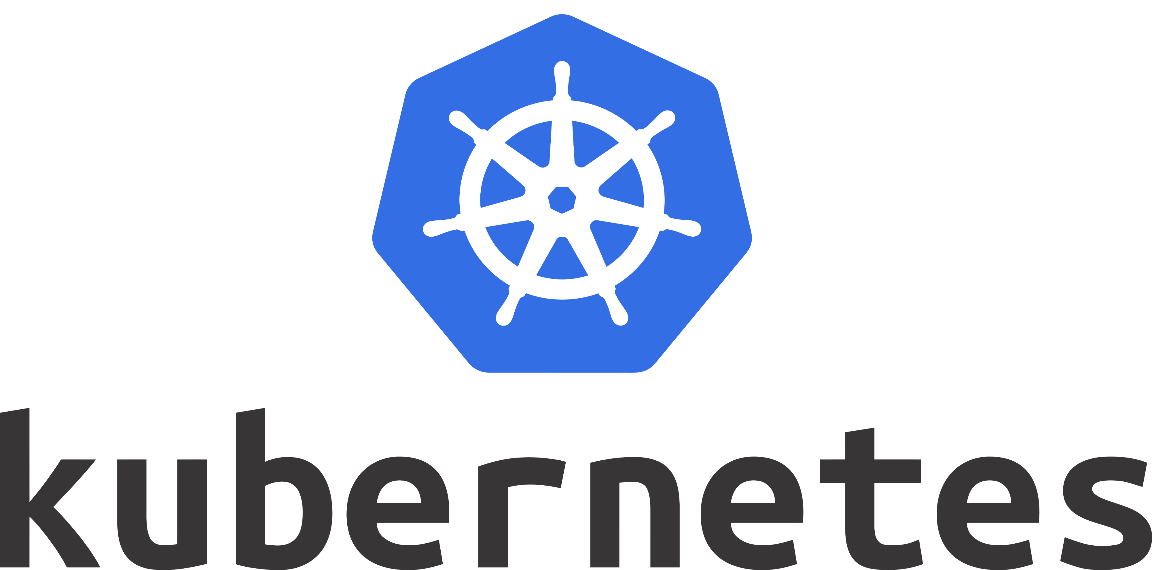
Prometheus collects metrics from instrumented applications, stores them in a time-series database, and provides a flexible query language to analyse the data. It supports various data sources, including Kubernetes, Docker, and other systems.

1. **GRAPHANA:**

Grafana is an open-source platform for visualizing and analysing metrics and logs. It supports various data sources, including Prometheus, Elasticsearch, Influx DB, Graphite, and many others. Grafana allows users to create interactive and customizable dashboards, alerts, and panels that visualize data from different sources.

**KUBERNETES:**

Kubernetes is an open-source platform for container orchestration and management. It was developed by Google. Kubernetes is built around the concept of clusters, which consist of one or more nodes that run containerized applications. Kubernetes provides various features for managing and orchestrating these nodes and the applications running on them

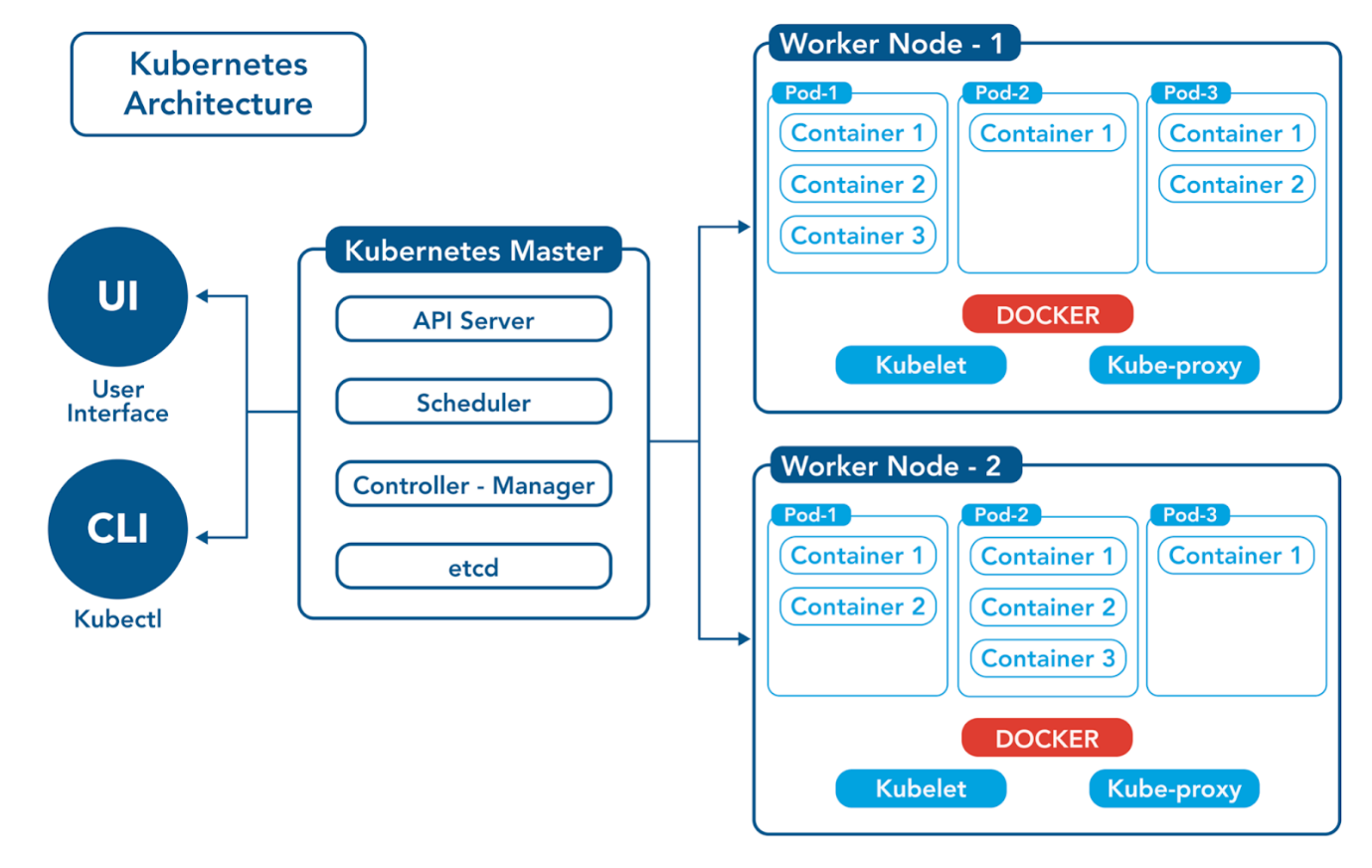
****

Kubernetes is designed to manage and deploy containerized applications across a distributed network of nodes, providing features such as load balancing, auto-scaling, and self-healing. It enables developers to abstract away the underlying infrastructure and focus on writing applications that can run on any platform, from local development environments to public clouds.

Kubernetes uses a declarative approach to managing infrastructure, where users define the desired state of the system and Kubernetes ensures that the actual state matches the desired state. This approach provides a high level of automation and reduces the risk of human error.

Kubernetes also provides a rich set of APIs and extensions that allow users to customize and extend the platform. It supports a wide range of container runtimes, including Docker, and can be used with a variety of tools and services, such as Istio for service mesh, Prometheus for monitoring, and Helm for package management.

Overall, Kubernetes is a powerful platform for deploying and managing containerized applications, enabling teams to build and deploy applications more efficiently and at scale.



**IMPLEMENTATION:**

When it comes to implementation system requirements plays a role:

The CPU and memory requirements depend on the workload and the number of nodes in the cluster.

However, a node should have at least 2 CPU cores and 2 GB of memory.

For larger workloads, the recommended minimum is 4 CPU cores and 8 GB of memory per node.

When it comes to application orchestration, Kubernetes can deploy and manage applications as a set of microservices. It can scale them up or down based on traffic or other parameters, such as CPU usage or memory consumption. Kubernetes also provides load balancing and service discovery, allowing the application to be accessed from the network.

**Create a Kubernetes cluster**: The first step is to create a Kubernetes cluster, which is a group of nodes that can run containerized applications. You can use a cloud provider like Google Cloud, Amazon Web Services, or Microsoft Azure to create a Kubernetes cluster. Alternatively, you can install Kubernetes on your own infrastructure using a tool like kubeadm.

**Deploy applications**: Once you have a Kubernetes cluster up and running, you can start deploying your applications to it. To deploy an application, you need to create a Kubernetes manifest file that describes the deployment. The manifest file typically includes information about the container image to use, the number of replicas to create, and any environment variables or configuration files that the application needs.

**Deploy databases**: In addition to deploying applications, you can also use Kubernetes to deploy databases like MySQL, PostgreSQL, or MongoDB. Kubernetes provides built-in support for stateful applications, which means that you can deploy a database as a Stateful Set instead of a Deployment. A Stateful Set provides guarantees about the ordering and uniqueness of pod names, which is important for stateful applications like databases.

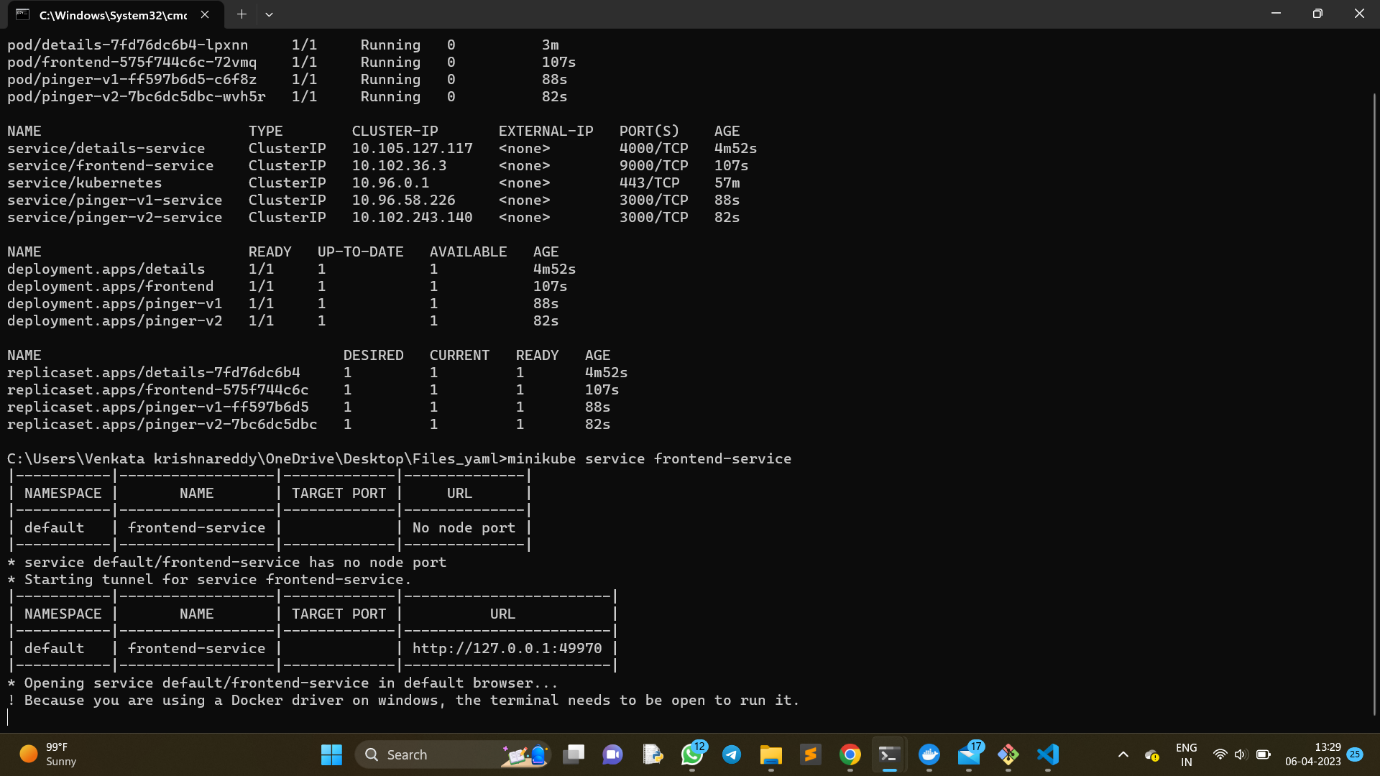
**Manage resources**: Kubernetes provides tools for managing resources like CPU, memory, and storage. You can set resource requests and limits for each application or database deployment, which helps Kubernetes schedule pods to the appropriate nodes in the cluster.

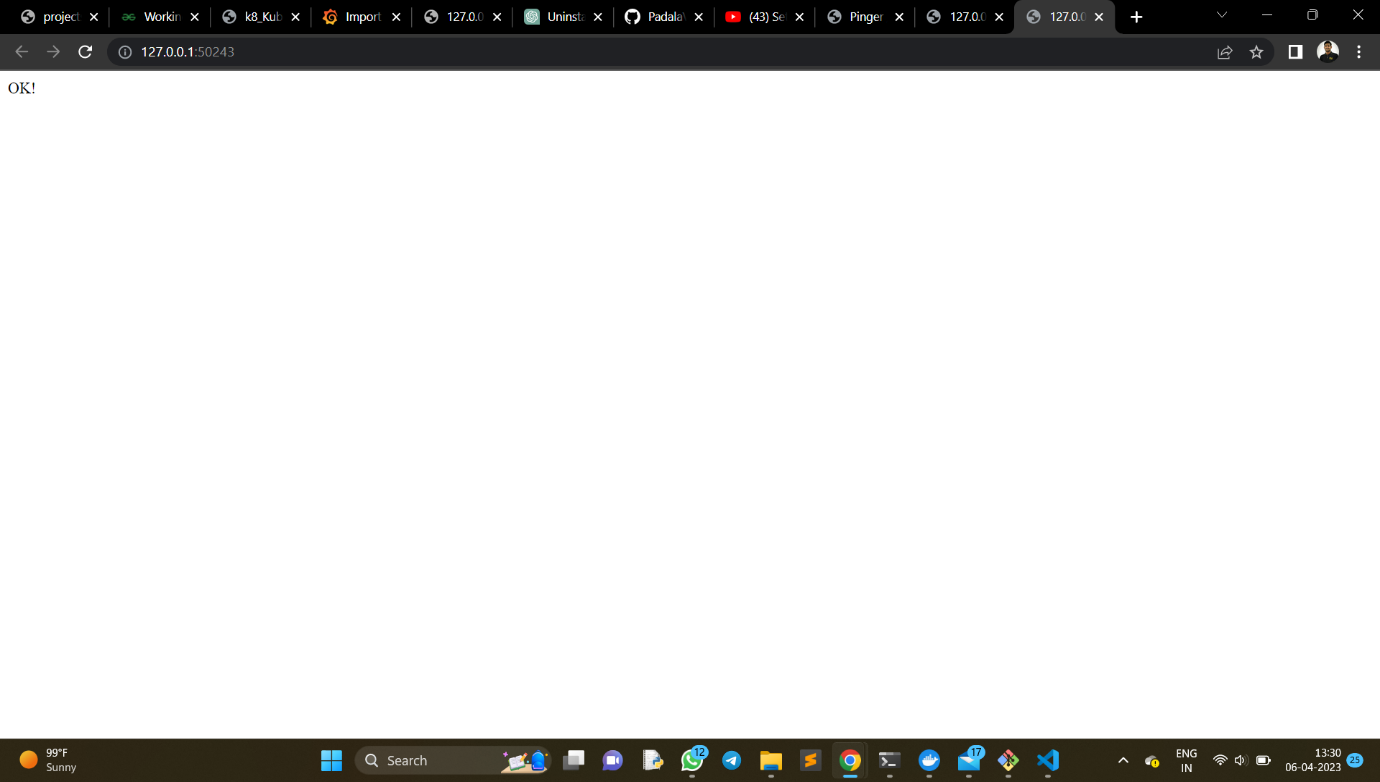
**Manage networking**: Kubernetes provides a built-in networking solution called Kubernetes Service that can be used to expose applications and databases to the outside world. You can create a Service to map a specific port on a pod to a port on the host machine or load balancer. This makes it easy to access applications and databases from outside the Kubernetes cluster.

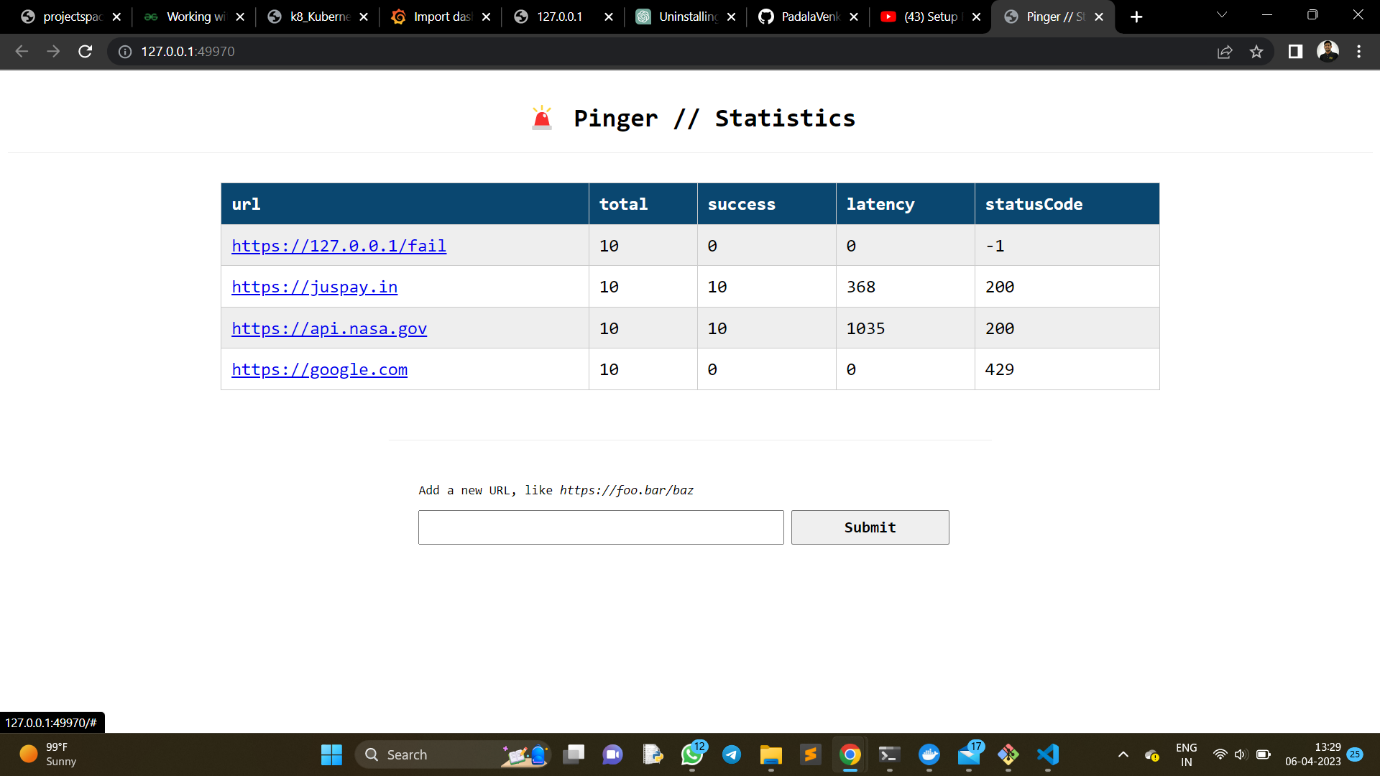
**Monitor and scale**: Finally, Kubernetes provides tools for monitoring and scaling applications and databases. You can use Kubernetes metrics to monitor the health of your applications and databases, and you can use Kubernetes autoscaling to automatically scale the number of replicas based on CPU or memory usage.

**PROOF OF CONCEPT:**

We have run various services like frontend, details and pinger and the containers are deployed and the pods are created with the help of minikube cluster.





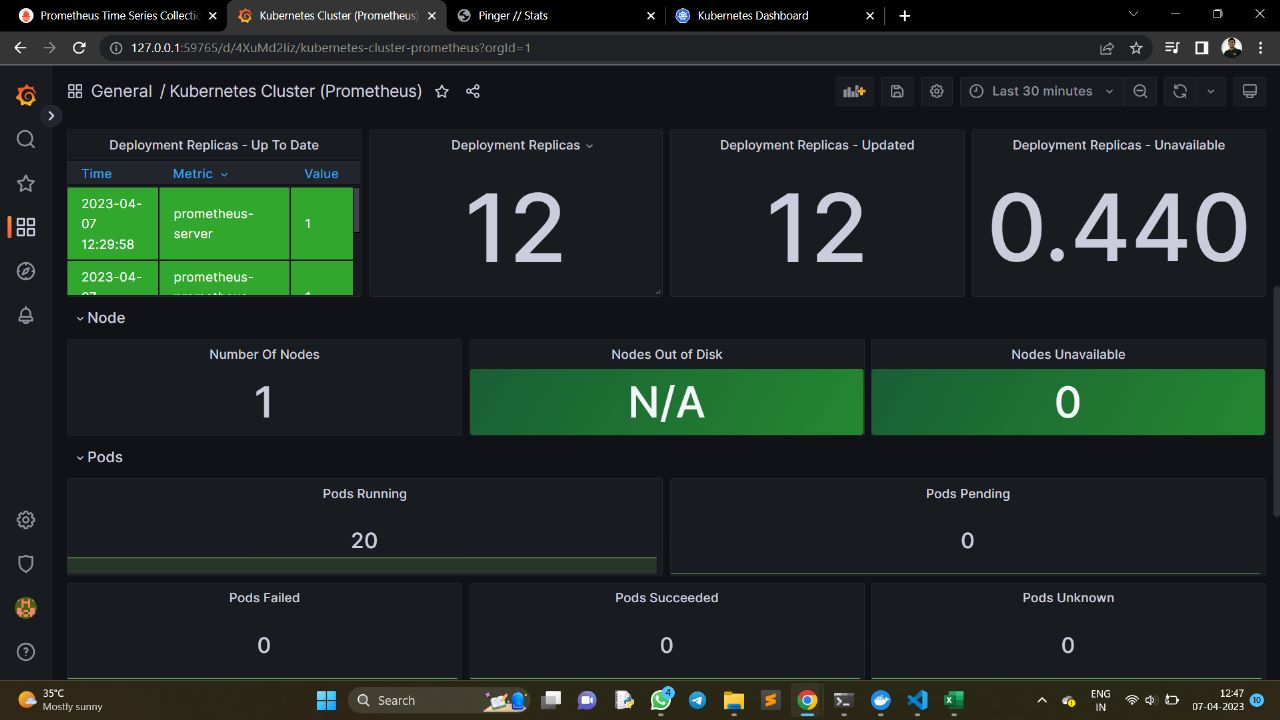


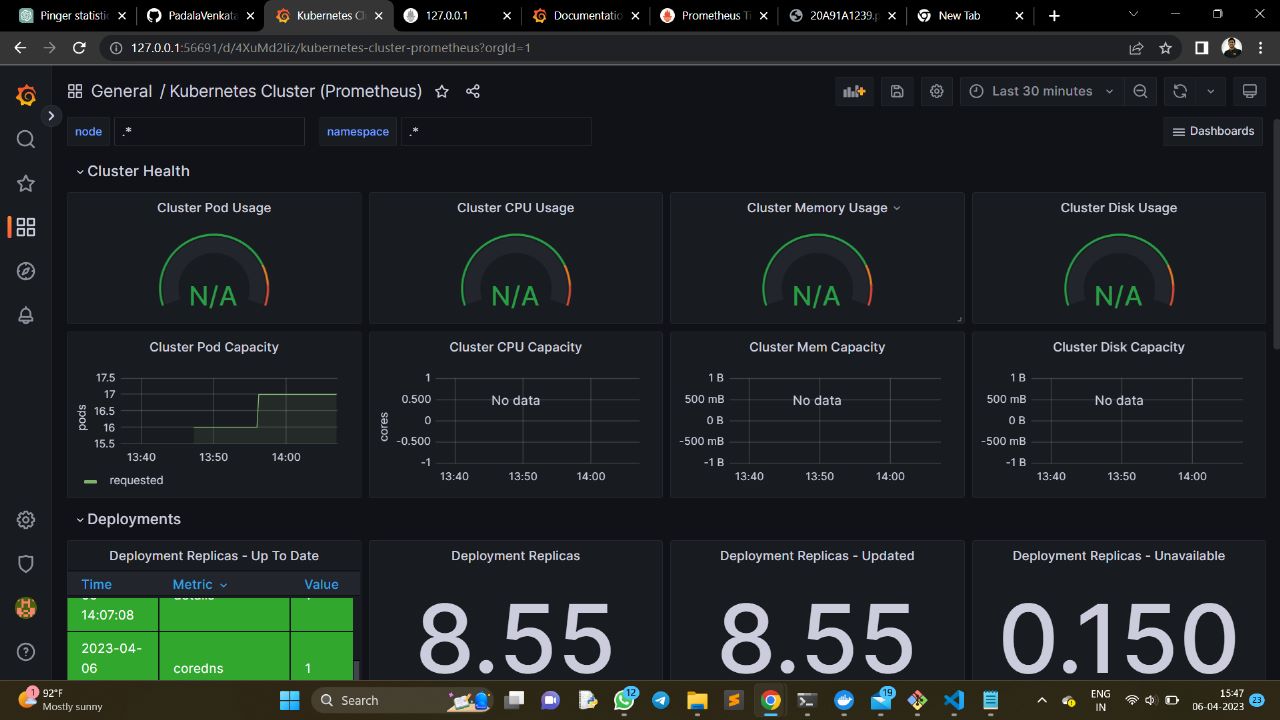
Later on, we have installed a monitoring tool called Prometheus and it has been integrated to our cluster.

So, all the metrics related to our cluster has been stored in this tool and this can be visually represented using an visualization tool called graphana.

**-**[**GITHUB-REPOSITORY**](https://github.com/PadalaVenkataKrishnareddy/BroGrammers-repo-app)

**OUTCOME:**





**CHALLENGES FACED**:

Prometheus tool integrates with the minikube cluster, it pull the images from the cluster sometimes we’ll be getting image pullback error due to network issue.

**CONCLUSION**:

In conclusion, implementing application and database orchestration using Kubernetes can provide many benefits, including improved scalability, reliability, and efficiency. By containerizing applications and deploying them on Kubernetes clusters, organizations can ensure that their applications are always available, and can easily scale up or down as needed.

Using tools like Prometheus and Grafana can help organizations monitor and analyse the performance of their applications and clusters, enabling them to identify issues and optimize their infrastructure for better performance and efficiency.

However, implementing application and database orchestration using Kubernetes requires a significant investment in time and resources, as well as a thorough understanding of Kubernetes and related tools. Organizations should carefully consider their needs and goals before deciding to adopt Kubernetes, and should plan for a gradual rollout and testing process to ensure a successful deployment.