DevOps

CLOUD MODELS

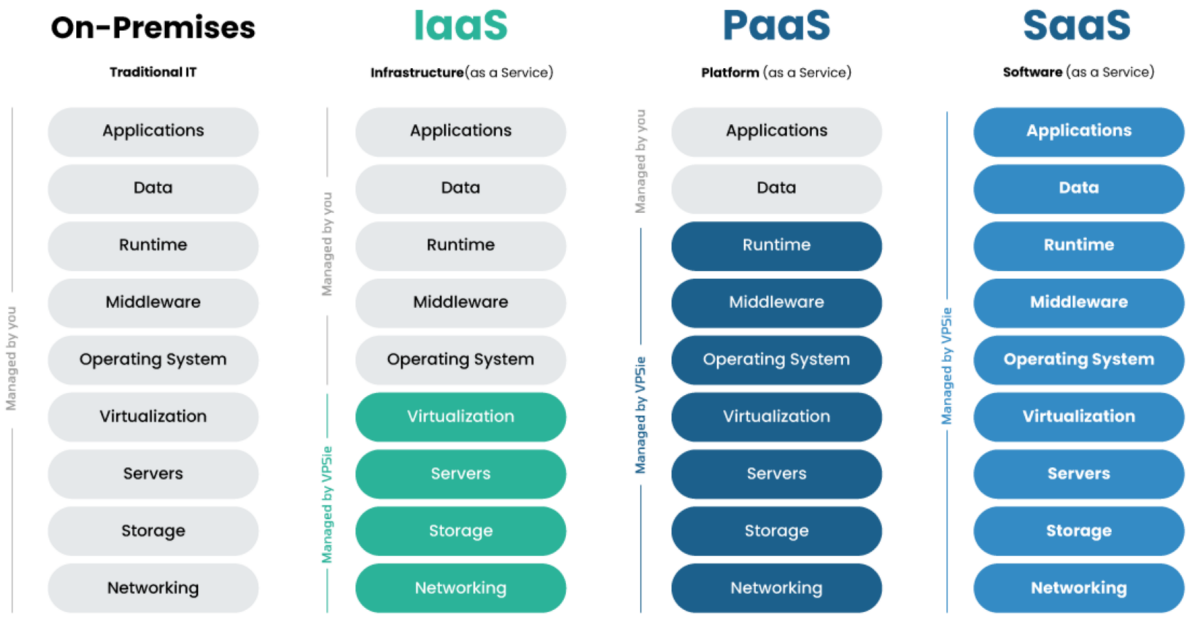
Cloud services have provided us with several benefits, such as Scalability allows you to easily scale up or down resources based on the demands of the application without having to invest in new hardware. This makes dealing with sudden increases in traffic and demand easier.

- Cost-effective :Cloud platforms employ a pay-as-you-go pricing model, so organisations only pay for the resources they use.
- Flexibility : enable the deployment of applications across many locations.
- Reliability: provide high availability and redundancy features, which ensure that the application remains operational even if hardware or network failures occur.
- It providers give strong security features to protect apps and data from illegal access and other security risks.

THERE ARE THREE MAIN CLOUD SERVICE MODELS

- Infrastructure as a Service (IaaS)
- Platform as a Service (PaaS)
- Software as a Service (SaaS)

But when when take control of every Service then it comes under our Premises we can say On Premises model



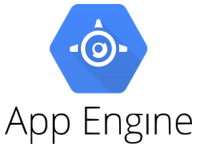
INFRASTRUCTURE AS A SERVICE (IAAS)

With an IaaS model, the cloud provider is in responsibility of the physical infrastructure, which includes data centres, servers, and networking equipment. The consumer, on the other hand, is in charge of managing the operating systems, apps, and data that operate on the virtual machines hosted in the cloud.

aws

PLATFORM AS A SERVICE (PAAS)

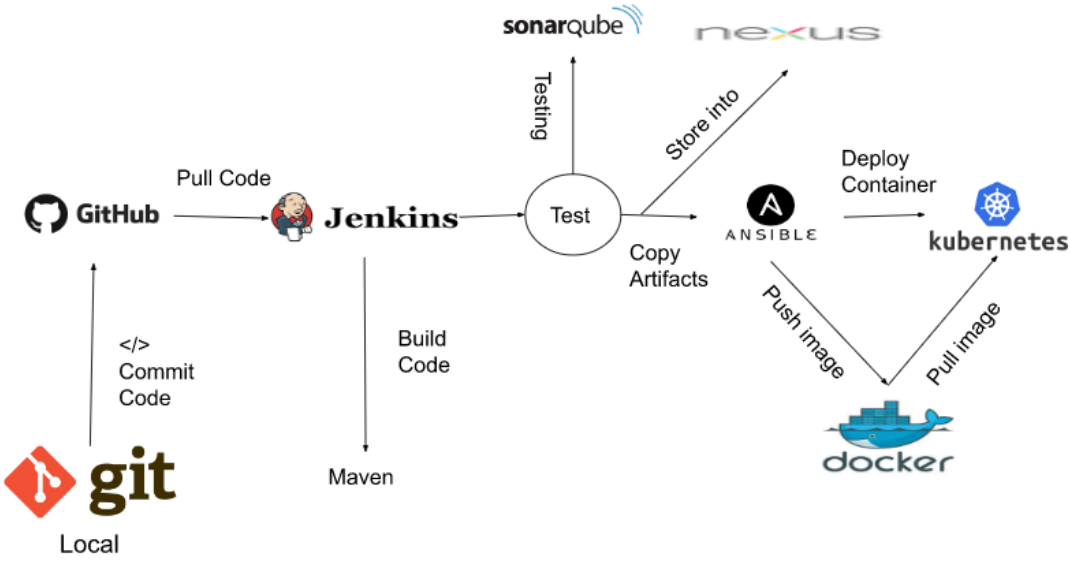
The cloud provider controls the underlying infrastructure, such as servers, storage, and networking, under a PaaS model, while the client is responsible for building and managing their applications. This enables developers to focus on developing code and creating apps rather than worrying about the underlying infrastructure.



SOFTWARE AS A SERVICE (SAAS).

With the SaaS model, the provider hosts the programme and its underlying infrastructure, while clients access the software through a web browser or mobile app.

FLOW DIAGRAM



1. Developers push code changes to a source code repository.
2. Jenkins automatically triggers a build and test of the code changes.
3. SonarQube analyzes the code for quality and security issues.
4. Then, the code is copied into artifacts using tool Nexus.
5. Ansible configures the infrastructure, including servers, networking, and storage, to support the application
6. The image and other artefacts created during the build process were pushed into Docker.
7. Kubernetes delivers and maintains application containers, scaling them up and down as required.
8. If the image is not present in the container, Kubernetes checks into Docker and then pulls it and places it in the container.