Devops Skills Roadmap

Mohammed Oshari, Omar Elhaj

Agendas

- Introduction - Omar
- OSI Model Understanding - Omar
- System Admin - Oshari
- Web Servers - Oshari
- Scripting - Omar
- Database Management - Omar
- Source Code Management - Oshari
- Containerization - Oshari
- Containers Orchestration - Oshari
- Cloud Computing - Omar
- Continuous Integration/ Continuous Deployment/Delivery (CI/CD) - Omar
- Infrastructure as Code (IaC) - Omar
- Secrets Management - Oshari
- Observability - Oshari

Introduction

Problem?

Gap between Developers and Operations Teams

Why Devops?

- Enforce Reliability into code by Automation
- Promote Collaboration
- Remove Blind Spots between teams
- Secure Applications

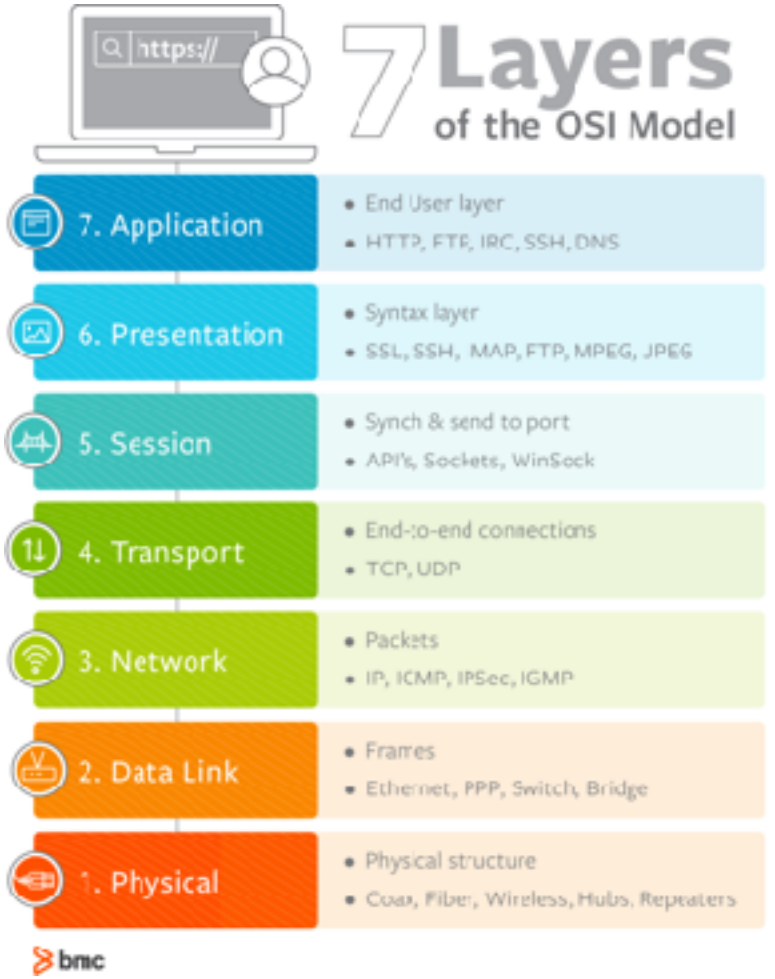
OSI Model

What?

Conceptual View that standardize NW communications into **seven** layers

Why you need it as an Engineer?

- Troubleshooting

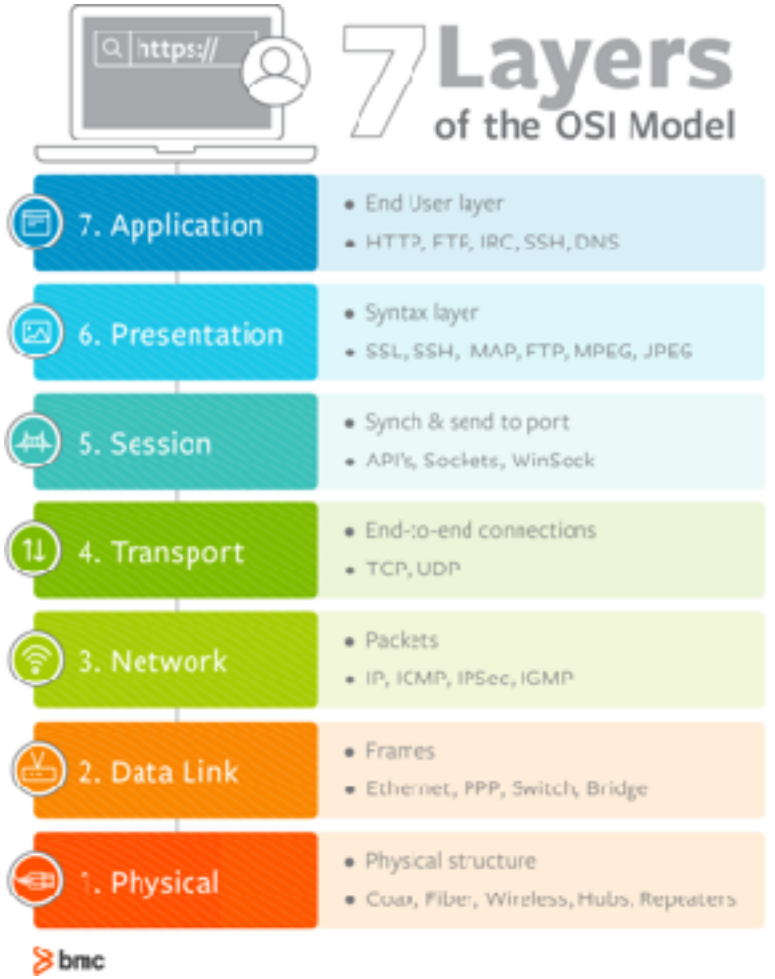


OSI Model

Troubleshooting Tools

- telnet (Layer 4)
- curl (layer 7)
- nmap (Layer3)
- traceroute (Layer3)
- Ping (Layer3)

Skills Weight - Important



Systems Administration

What?

The skill of managing, configuring, securing, and troubleshooting Linux/Windows based systems

Business Value?

- Minimal and high performing Operating Systems for Applications Runtime
- Open Source, and Enterprise options
- Wide Coverage for Resources Skills to maintain

Systems Administration

Famous Operating Systems:

Unix (FreeBSD), Linux (Redhat, Debian, SUSE), Windows (Windows Server)

Weight - Very Critical

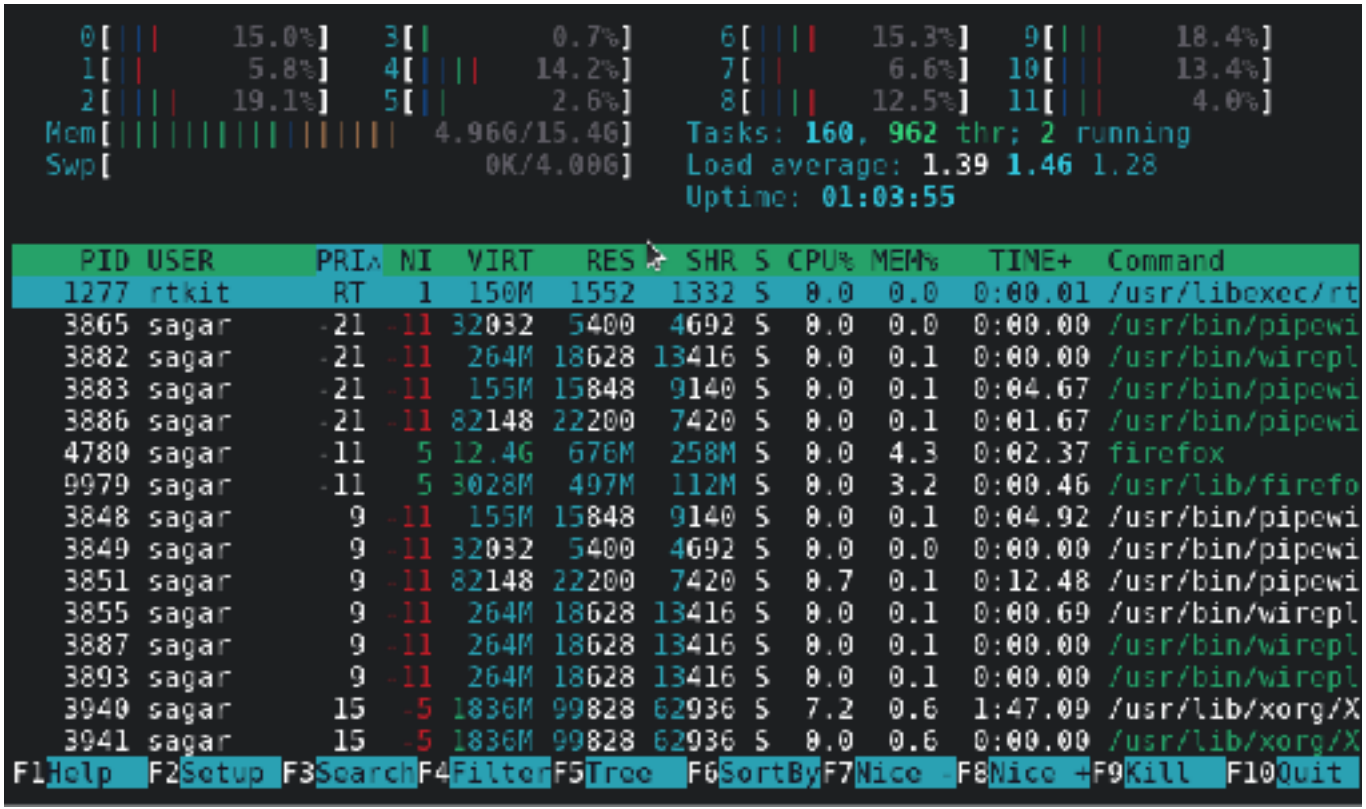
Red Hat

SUSE

FreeBSD

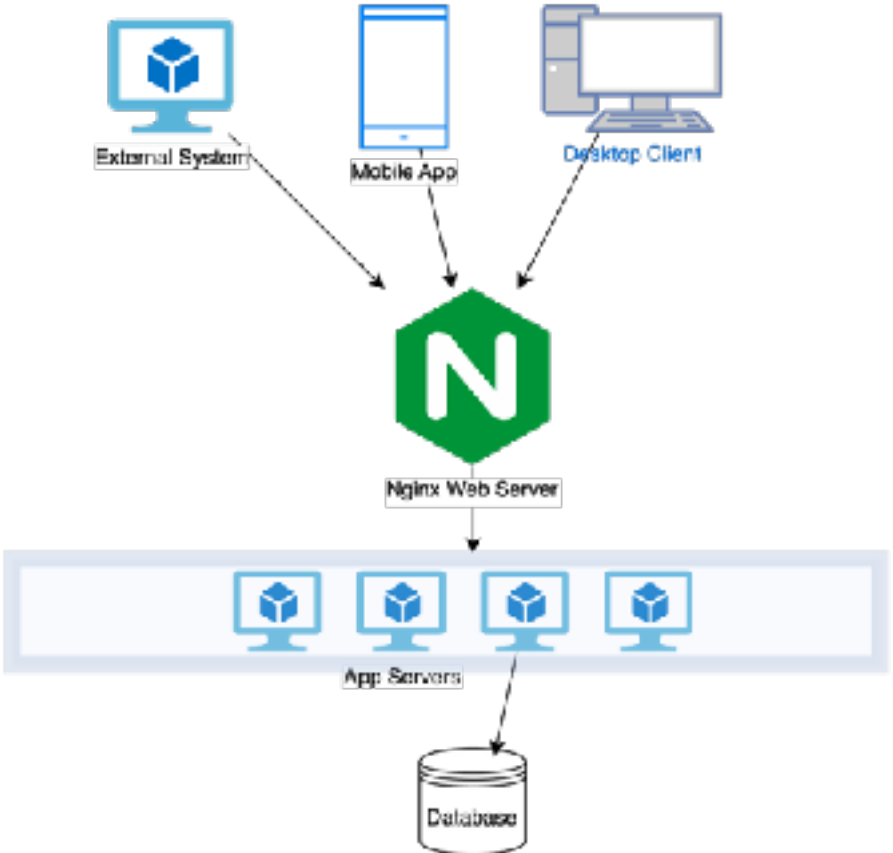
debian

Systems Administration



Systems Administration

Web Servers and Reverse Proxies



Web Servers and Load Balancers

What?

Software that is responsible for Serving Web Content (Web Sites), Hosting APIs, HTTPs, Reverse Proxying, Load Balancing

Business Value?

- Communication Encryption
- Applying Authentication/ Authorization
- Load Balancing
- High Availability Setup
- Customized Routing Layer 7 Requests

Web Servers and Load Balancers

Famous Reverse Proxies:

Nginx, Apache, Microsoft IIS, Cloud Resources

Weight - Important

Scripting

```
#!/bin/bash
# This Works with Gitlab CI/CD as shell jobs

curl -s -X POST https://api.telegram.org/bot${myBotToken}/sendMessage -d chat_id=$myGroupChatId -d parse_mode="HTML" -d text="""
New code pushed to <b>$CI_COMMIT_BRANCH</b> branch! ✅
<b>Project</b> 🧩: $CI_PROJECT_NAME
<b>Committer</b> 👤: $GITLAB_USER_NAME
<b>Revision ID</b> 📝: $CI_COMMIT_SHORT_SHA
<b>Commit Message</b> 💬:
$CI_COMMIT_MESSAGE

Check code changes at the following commit URL 🔗:
$CI_PROJECT_URL/-/commit/$CI_COMMIT_SHA
"""
```

Scripting

What?

Writing short programs (scripts) to automate tasks, often using languages like Bash, Python, or PowerShell.

Typical Use Cases

- Database Backups and Archiving
- Logs Rotation
- CI/CD Customized Jobs
- Reducing Human Error

Scripting

Top Scripting Languages:

Bash, Python, Powershell, CMD Batch

Weight - Important

Note:

Once you work a bit on OS skills,
learning this skill will make more sense

Database Management

What?

The process of storing, organizing, and accessing data using structured/ unstructured systems like relational or NoSQL databases.

Business Value/Real Life:

- Data
  - User Information in Facebook
  - Patients Information in Hospitals Systems
  - etc

Database Management

Famous Technologies

SQL/Relational Databases:

MySQL, PostgreSQL, Oracle, MSSQL, Cloud Resources

NoSQL Databases:

MongoDB, Redis, Cloud Resources

Expected from you:

- Provisioning and Maintenance
- Basic Level Queries
- Backups and Archiving
- Replication

Weight: Normal

Source Code Management

What is SCM?

Source Code Management is a practice where developers can collaboratively write code and maintain a stable code base, and it is typically the starting point of CI/CD pipelines

Why we need it?

- Where’s your change
- When did you change this
- We don’t want this feature now, let’s code it later
- I want to review your code changes
- I want to see the history of the changes in the code

Source Code Management

Famous Technologies:

Git, Gitlab, Github, Bitbucket

Best Practice:

Gitflow Branching Strategy, Push Policy, Protect main branches

Weight: Critical

Terms

Repo

Pull / Clone

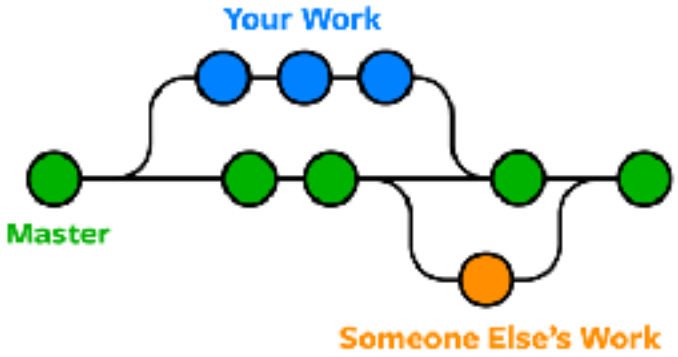
Commit

Push

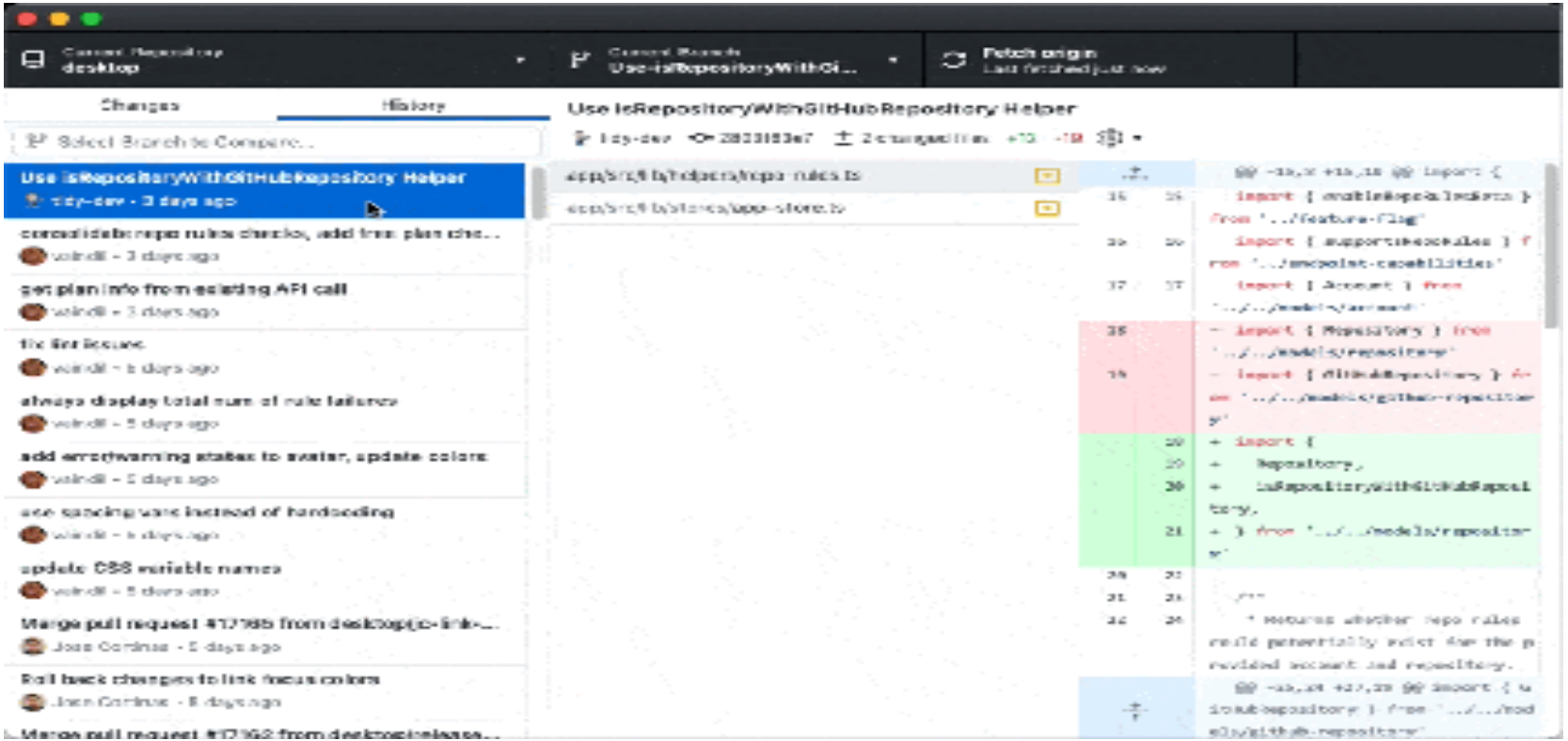
Branch

Merge

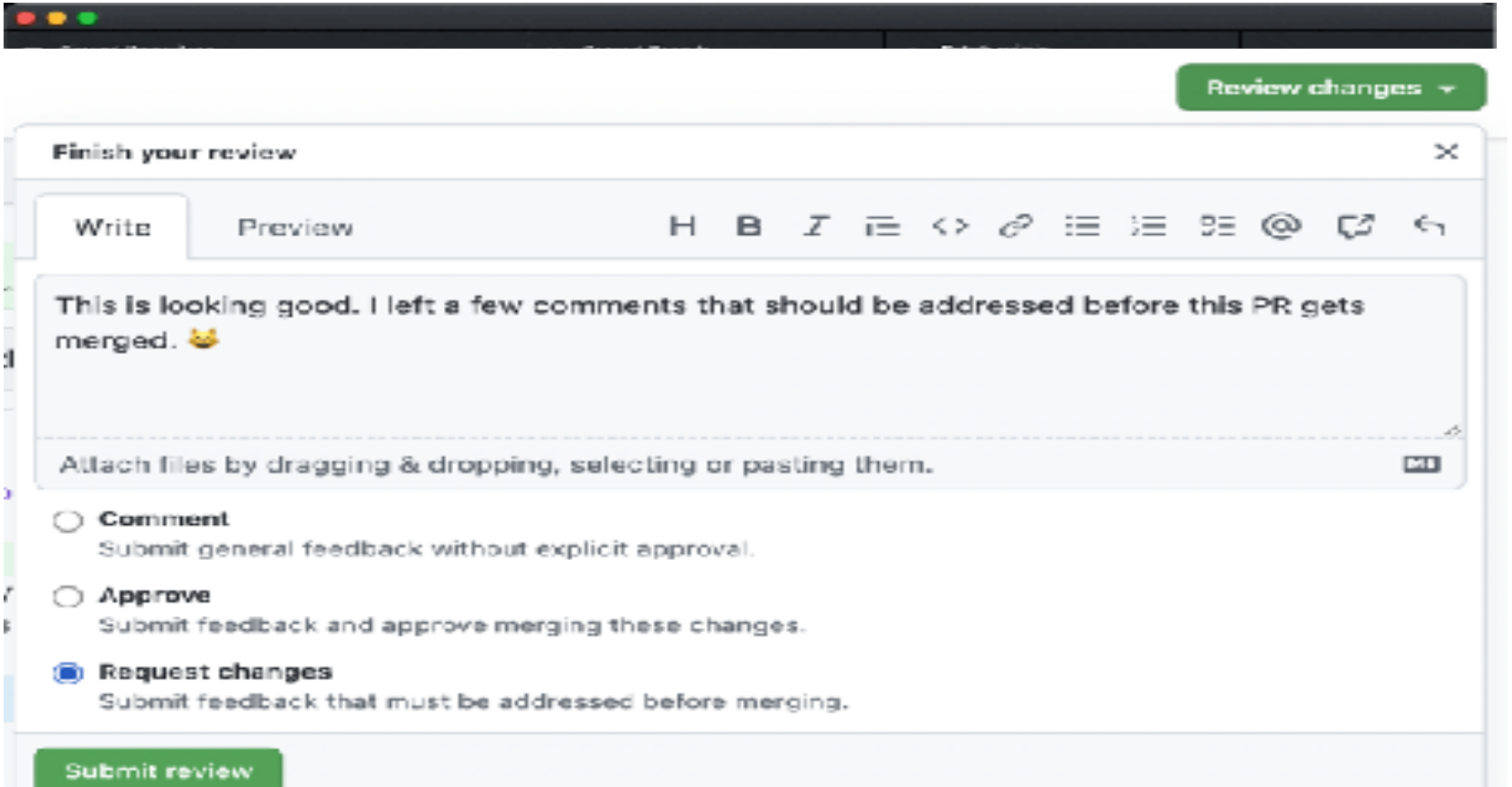
Pull Request/Merge Request



Who, When, What?



Let me Review the Change before we take it live



Cloud Computing

What?

Data Centers operated by Other companies accessed over the internet or private-network and Providing Infrastructure, Platforms, and Software as a Service

Business Value?

We Trust Microsoft Datacenters more than ours,

We want to eliminate Datacenters management overhead

We don’t want to spend over Data centers, Cooling, etc

We Want to Manage our cost in a monthly basis

We Want a Guaranteed Availability and a Scalability for our Systems

Cloud Computing

Famous Cloud Providers?

Azure, AWS, GCP, Oracle, and Digital Ocean

Weight - Important

Containerization

What is a container

Packaging applications and their dependencies into isolated units (containers) that run consistently across environments.

Why to use?

- It works on my machine
- All In One Object. Ready to run and it **will run for sure**
- portability and consistency
- Microservice Architecture

Containerization

Basic Concepts:

Image -> Container,
Registry, Build, Push

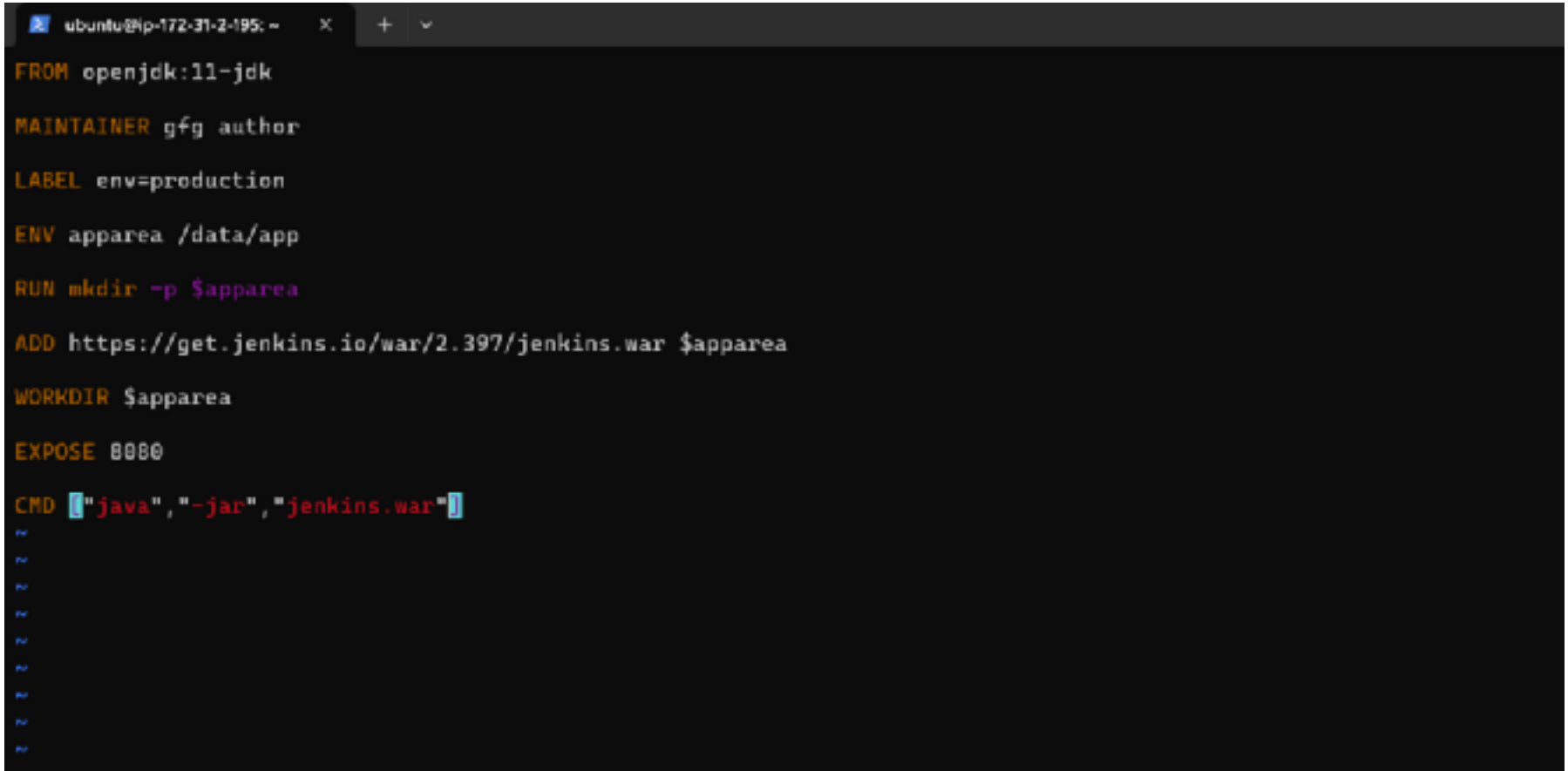
Famous Tools:

Docker, Podman

Weight: Critical

Containerization

Containerization



Containerization

```
hemantasundaray@: ~/Desktop/docker-demo $ docker build . -t sample-flask-app:v1
[+] Building 7.2s (9/9) FINISHED
 => [internal] load build definition from Dockerfile                          0.1s
 => => transferring dockerfile: 187B                                          0.0s
 => [internal] load .dockerignore                                             0.1s
 => => transferring context: 51B                                              0.0s
 => [internal] load metadata for docker.io/library/python:3.11-slim           6.8s
 => [1/4] FROM docker.io/library/python:3.11-slim@sha256:53a67c012da3b807905559fa59f  0.0s
 => [internal] load build context                                             0.0s
 => => transferring context: 190B                                             0.0s
 => CACHED [2/4] WORKDIR /app                                                 0.0s
 => CACHED [3/4] RUN pip install flask==2.3                                   0.0s
 => CACHED [4/4] COPY . /app                                                  0.0s
 => exporting to image                                                        0.0s
 => => exporting layers                                                       0.0s
 => => writing image sha256:b409d5824cd3f9f8663b473dae9f9d570a99fb6d0b4632fa3096aabf  0.0s
 => => naming to docker.io/library/sample-flask-app:v1                        0.0s
```

Containers orchestration

What?

Automating deployment, scaling, and management of containerized applications across clusters.

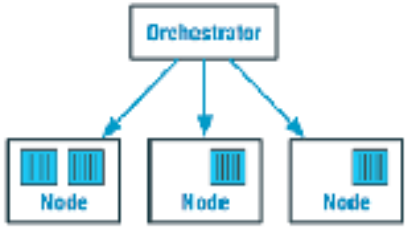
Why:

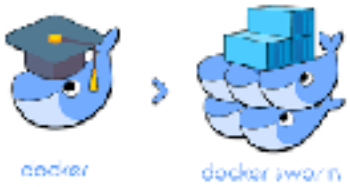
- Enables high availability and fault tolerance.
- Simplifies complex microservices management.
- Supports dynamic scaling and self-healing.

Containers orchestration

Famous Tools:

- Kubernetes (k8s)
- Docker Swarm
- Red Hat Openshift





Weight: Critical (If being used)

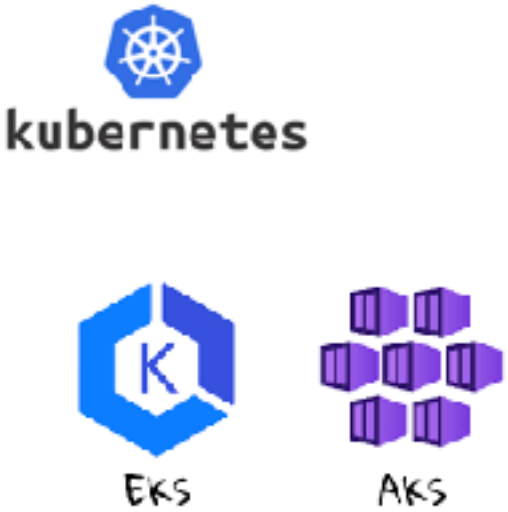
Kubernetes

De facto in Containers orchestration

Many flavours

Managed, Self-hosted

Examples



Helm

What is?

Is a packaging tool for Kubernetes-based applications

It packs both App+Infrastructure resource

Why we use it?

Declarative syntax

Desired State Management

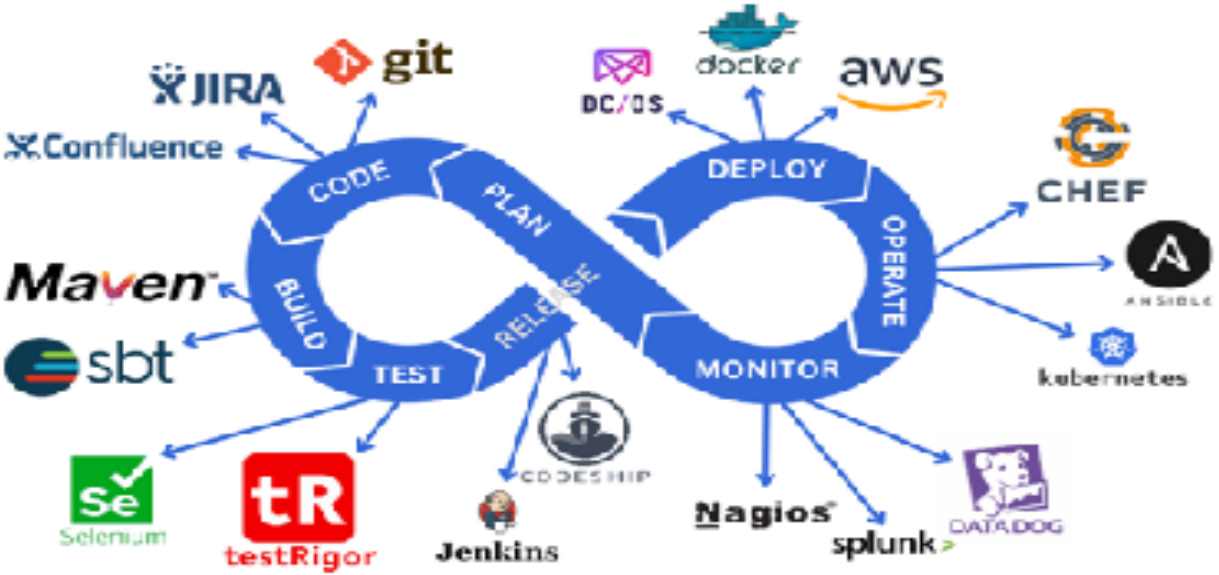
Gitops, Code Delivery approach with reviews

Weight - Important (if Kubernetes is used)

Continuous Integration Continuous Deployment/Delivery CI/CD

What?

Building an Automation Lifecycle for Applications Delivery from Static Code to Production



Continuous Integration Continuous Deployment/Delivery CI/CD

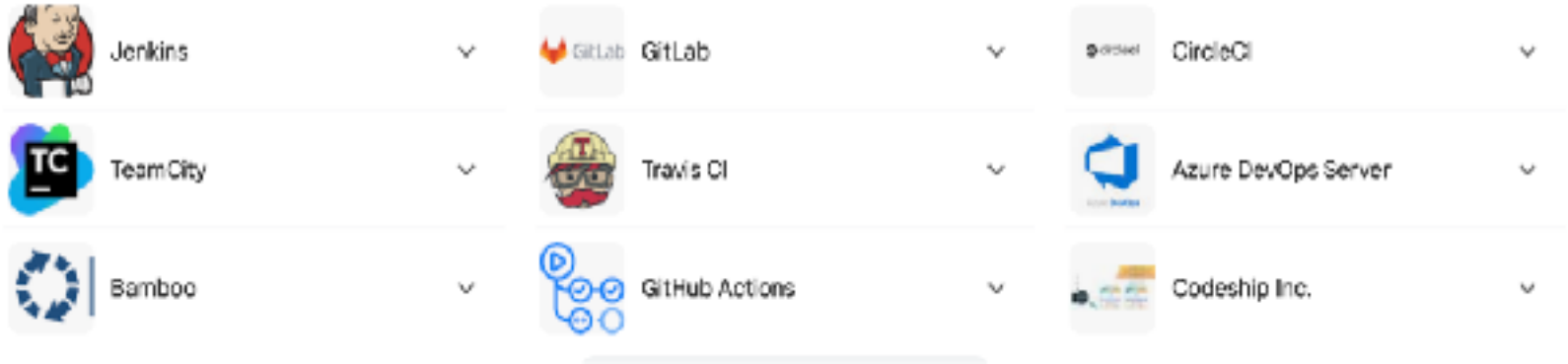
Business Value?

- Speed up release cycles.
- Improve software quality and reliability.
- Reduce manual intervention and deployment risks.
- Enforce Application Security

Continuous Integration Continuous Deployment/Delivery CI/CD

Famous Technologies?

Jenkins, Azure Devops, Github Actions, Gitlab CI, TravisCI, and CircleCI



Weight - Most Critical

Infrastructure as Code (IaC)

What?

Building an Automation Lifecycle for Infrastructure Changes from Static Code to Production

Streamlining Infrastructure Changes through CICD pipelines, using declarative syntaxes for Desired State Management

Business Value?

Famous Technologies?

Weight

Infrastructure as Code (IaC)

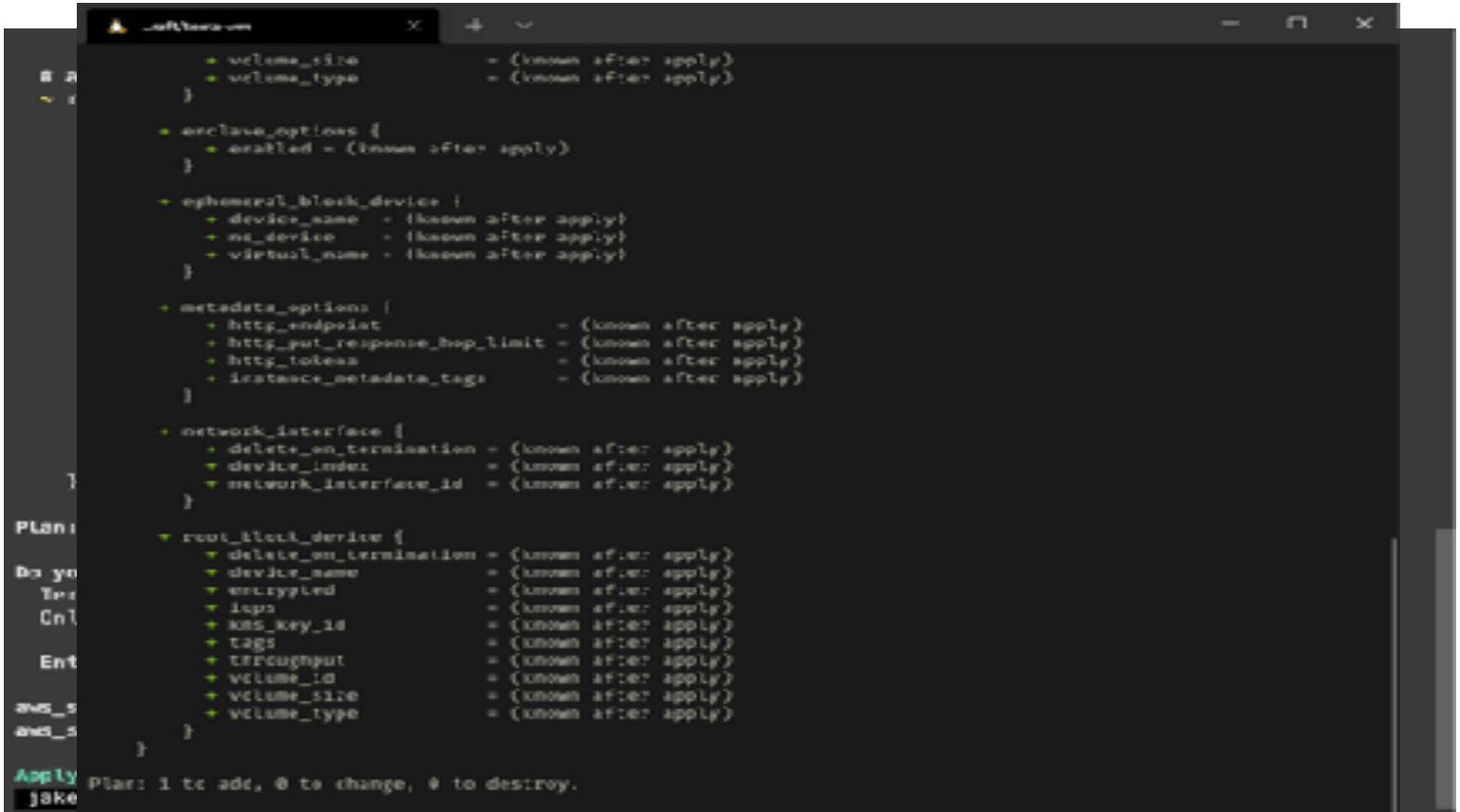
What?

Building an Automation Lifecycle for Infrastructure Changes from Static Code to Production

Business Value?

- Enables repeatable and consistent environments.
- Supports version control and automation.
- Reduces configuration drift and manual errors.

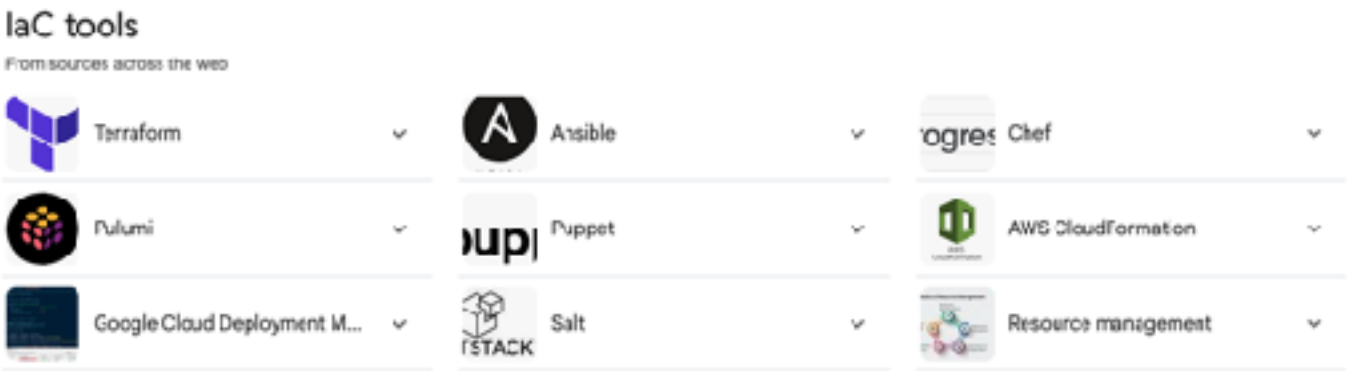
Infrastructure as Code (IaC)



Infrastructure as Code (IaC)

Famous Technologies?

Terraform, Azure RM, AWS Cloudformation



Weight - Normal

Configuration Management

What is it
Manage your infra from central point

Why we need it
Govern your infrastructure management

Examples

Ansible

Weight - Critical if Used

Secret Management

What is Secret Management?

Storing, distributing, rotating, and revoking secrets securely

Why we need it?

- Prevent credential leaks
- Preventing unauthorized access to secrets
- Removing Hardcoded passwords in Source

Code

Secret Management

Features:
Centralized Secret Store, Automation, Compliance

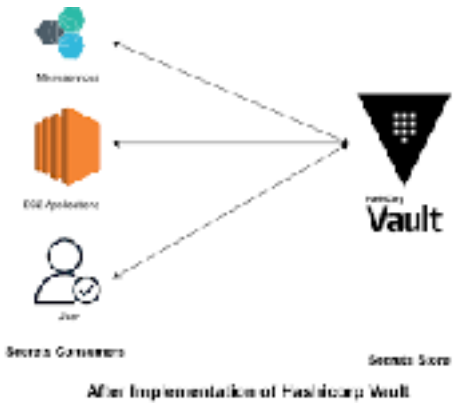
Examples:

CI/CD Secretes, Azure Key Vault, Hashicorp Vault

What is expected from Devops Engineers:

Provisioning, Managing AuthN/AuthZ, Configuration

Weight - Critical

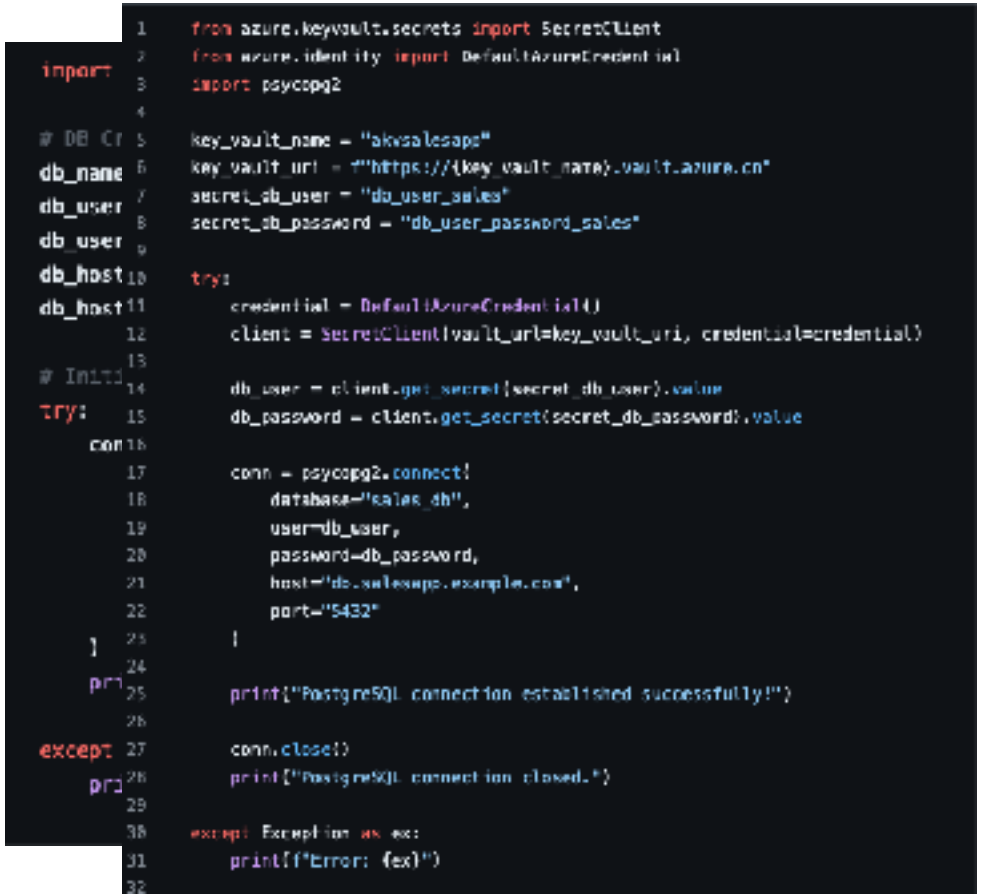


After Implementation of Hashicorp Vault

Secret Management

Before: ❌

After: ✅



```python
from azure.keyvault.secrets import SecretClient
from azure.identity import DefaultAzureCredential
import psycopg2

key_vault_name = "akvsalesaps"
key_vault_uri = f"https://{key_vault_name}.vault.azure.cn"
secret_db_user = "db_user_sales"
secret_db_password = "db_user_password_sales"

try:
    credential = DefaultAzureCredential()
    client = SecretClient(vault_url=key_vault_uri, credential=credential)

    db_user = client.get_secret(secret_db_user).value
    db_password = client.get_secret(secret_db_password).value

    conn = psycopg2.connect(
        database="sales_db",
        user=db_user,
        password=db_password,
        host="db.salesaps.example.com",
        port="5432"
    )

    print("PostgreSQL connection established successfully!")

    conn.close()
    print("PostgreSQL connection closed.")

except Exception as ex:
    print(f"Error: {ex}")
```

Observability

What is it?

- The ability to understand system behavior and performance by analyzing telemetry data (logs, metrics, traces)

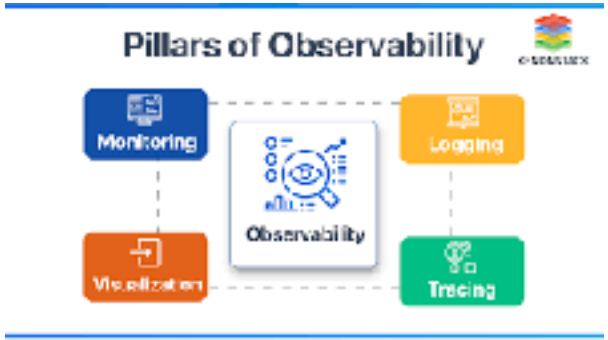
Why we need it?

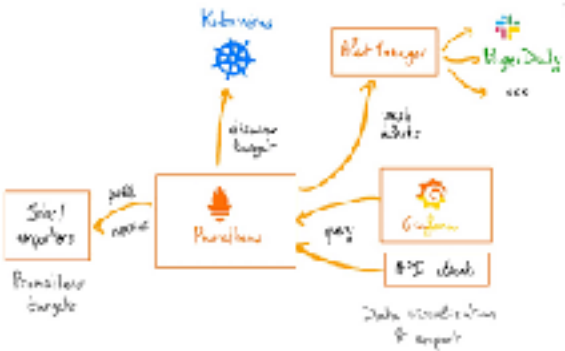
- Accelerates root cause analysis and incident response.
- Improves reliability and user experience.
- Enables proactive performance tuning and capacity planning.

Observability

Famous Technologies

- Prometheus, Grafana, Loki
- Datadog
- Open Telemetry





Terms

Query Language

Metrics

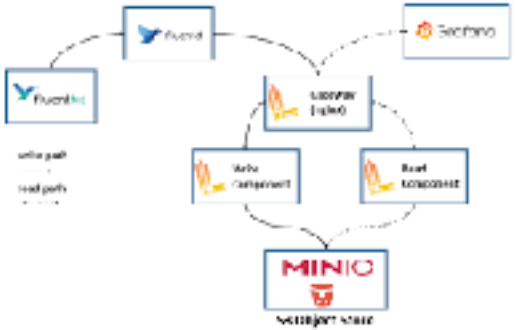
Values

Logs

Logs

Traces

Track request



Dashboards

What is it

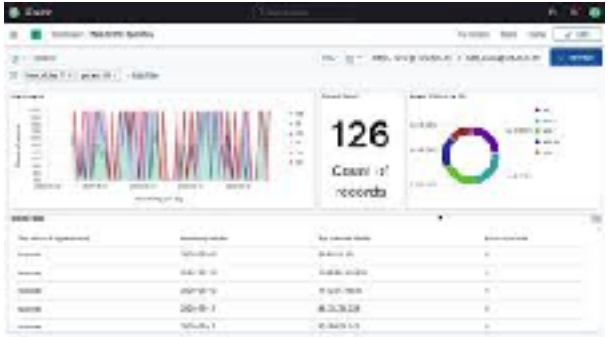
Prepared visualizations of Operational Logs

Aggregations

Ex: Failure Rate, Response Time

Why we need it

Golden Signals, Anomalies Detection, Alerting



Weight - Normal for Devops Engineers

Weight - Critical for SREs

Extras - Gitops

What?

Manage both App and Infrastructure changes through Code Review Process

Business Value?

Reduce Manual Intervention Errors

Improve Developers Productivity

Adopt Code Review into Cloud Native applications changes

“**I** could make an Error, but **We** are less likely to do so”

Extras - Gitops

Famous Technologies?

ArgoCD, FluxCD, Helm

Weight - Normal

Extras - DevSecOps

What?

Enforcing Application Security Scans into CI/CD pipelines

Business Value?

Secure Applications from early development stages

Utilize Quality Gates for Secure Releases

Famous Technologies?

SonarQube, OWASP ZAP, Synk

Weight - Normal

How To start and get into a job

- There are no Jr. Devops Roles
- Make yourself an Experience from hands-on projects
- Work with your hands-on
- Oracle Virtual Box
- Github, and Gitlab
- Kodekloud for Skills
- PluralSight for Certificates
- AWS/ Azure Free Plans

Recommendations

- Change doesn’t happen between a day and night?
- Always, think, speak, and be driven by business value/terms
- Focus on Concepts Only. Commands or Configurations are available everywhere
- Once you land a job:
  - Make your Standard and Enforce it “Work Smart not Hard”
  - Don’t embrace workarounds, they will make you “work hard”
  - Follow Devops Institute, subscribe to their mailing list
  - Read about Site Reliability Engineering

Sponsors

- https://nogalliance.org/