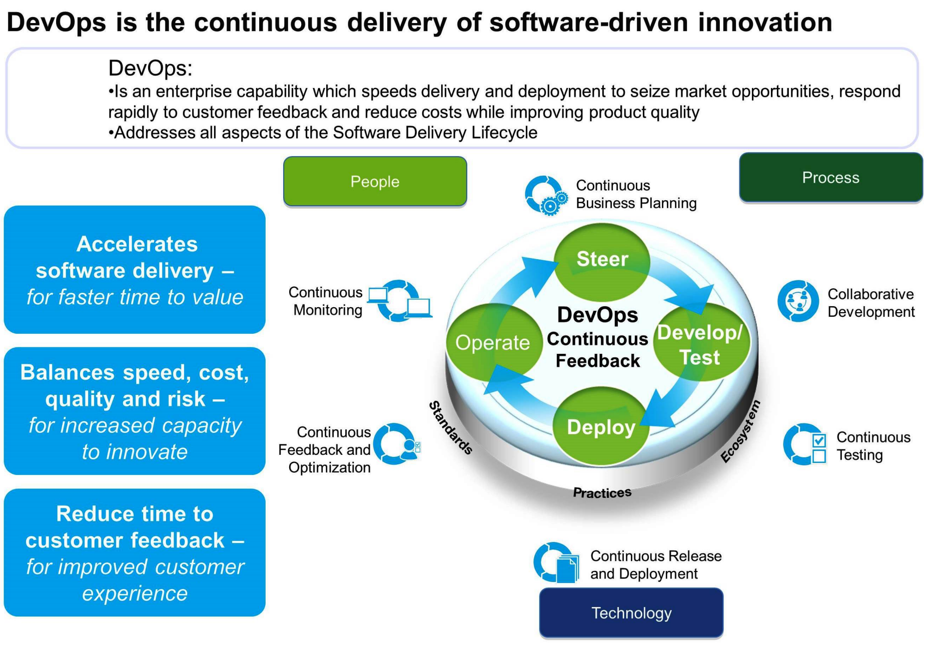
Memo - DevOps & CICD

DevOps



DevOps is the continuous delivery of software-driven innovation

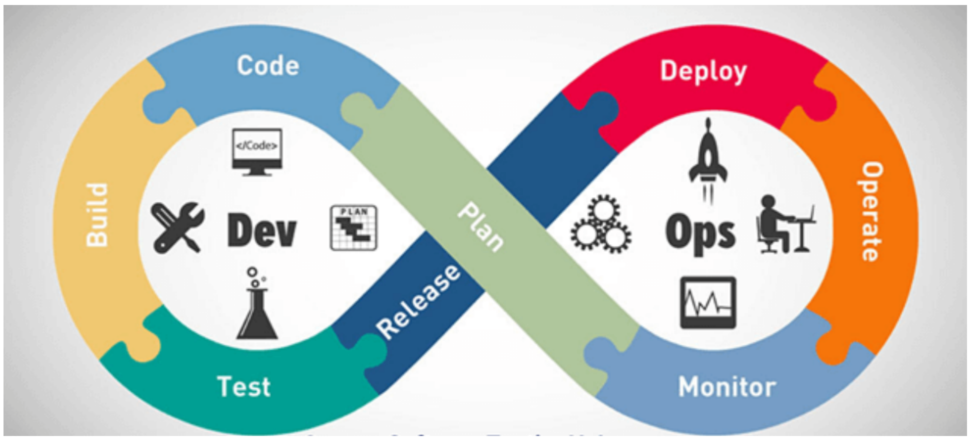
DevOps:
- •Is an enterprise capability which speeds delivery and deployment to seize market opportunities, respond rapidly to customer feedback and reduce costs while improving product quality
- •Addresses all aspects of the Software Delivery Lifecycle

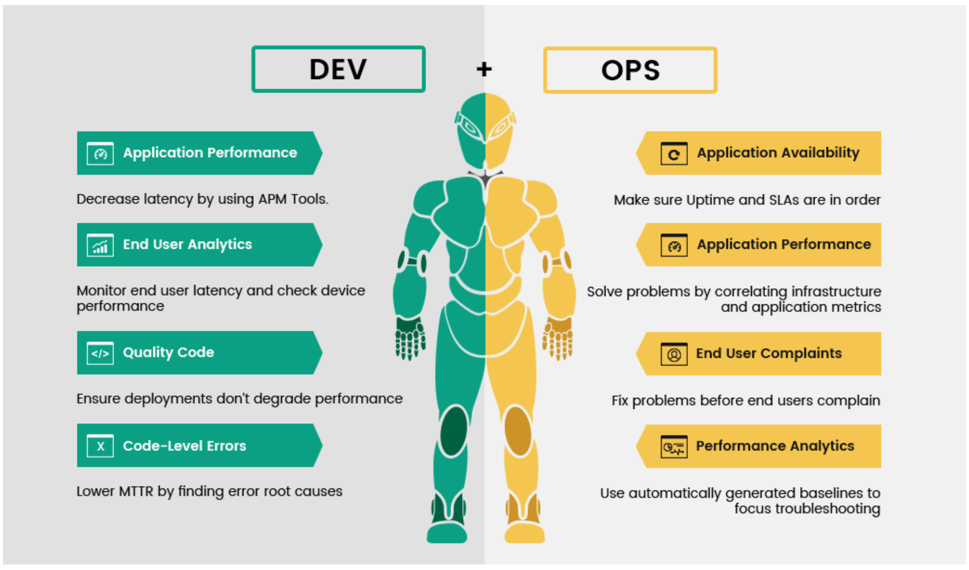
Accelerates software delivery – *for faster time to value*

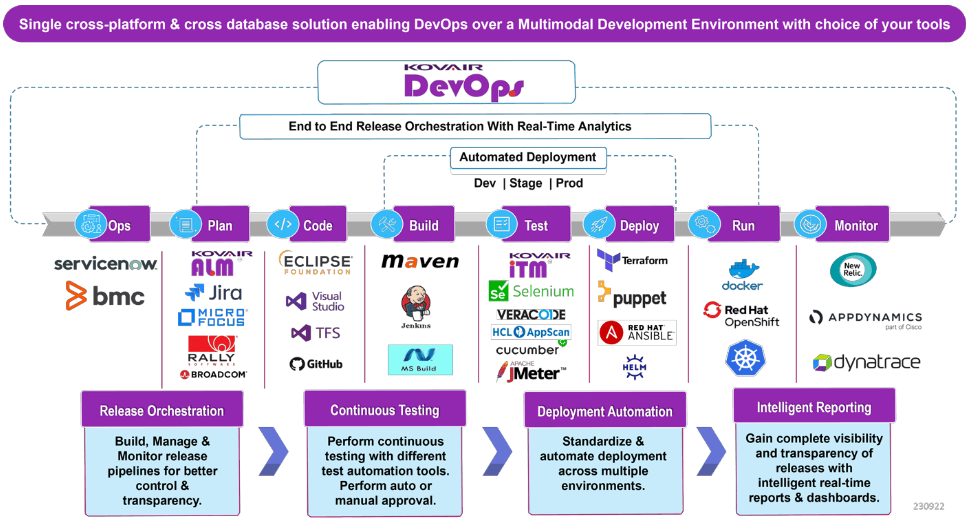
Balances speed, cost, quality and risk – *for increased capacity to innovate*

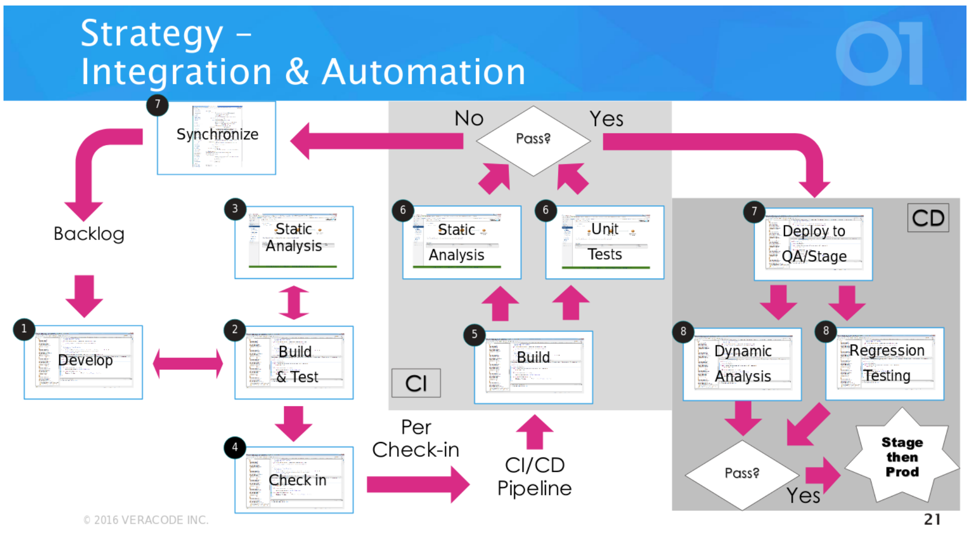
Reduce time to customer feedback – *for improved customer experience*

People
Process
Technology
Continuous Business Planning
Collaborative Development
Continuous Testing
Continuous Release and Deployment
Continuous Feedback and Optimization
Continuous Monitoring
Steer
Develop/ Test
Deploy
Operate
DevOps Continuous Feedback
Standards
Practices
Ecosystem

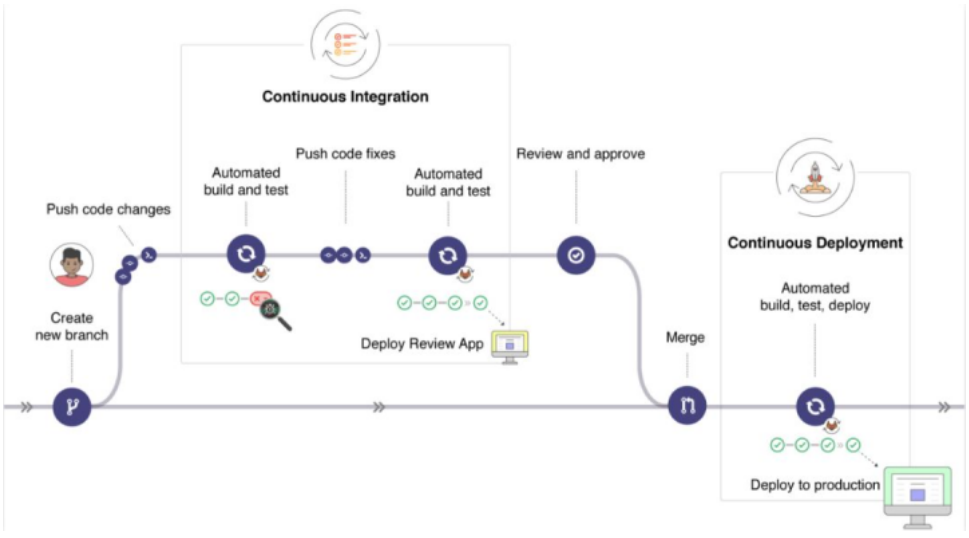


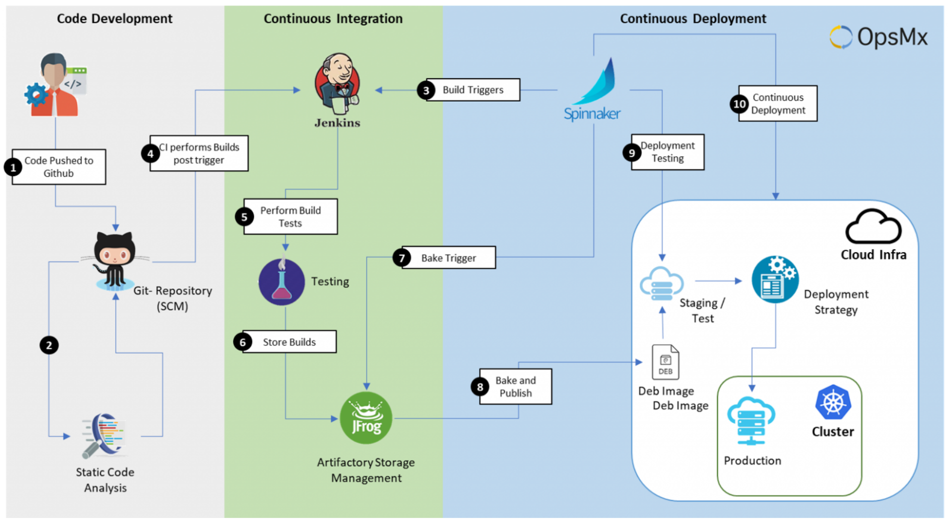

DEV
+
OPS
Application Performance
Decrease latency by using APM Tools.
End User Analytics
Monitor end user latency and check device performance
Quality Code
Ensure deployments don't degrade performance
Code-Level Errors
Lower MTTR by finding error root causes
Application Availability
Make sure Uptime and SLAs are in order
Application Performance
Solve problems by correlating infrastructure and application metrics
End User Complaints
Fix problems before end users complain
Performance Analytics
Use automatically generated baselines to focus troubleshooting

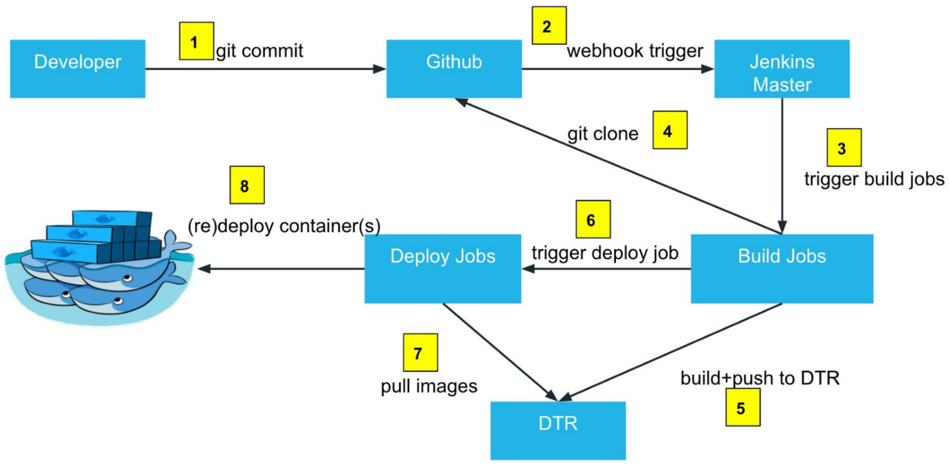

Single cross-platform & cross database solution enabling DevOps over a Multimodal Development Environment with choice of your tools
KOVAIR DevOps
End to End Release Orchestration With Real-Time Analytics
Automated Deployment
Dev | Stage | Prod
Ops
Plan
Code
Build
Test
Deploy
Run
Monitor
servicenow
bmc
KOVAIR ALM
Jira
MICRO FOCUS
RALLY
BROADCOM
ECLIPSE FOUNDATION
Visual Studio
TFS
GitHub
maven
Jenkins
MS Build
KOVAIR iTM
Selenium
VERACODE
HCL AppScan
cucumber
JMeter
Terraform
puppet
RED HAT ANSIBLE
HELM
docker
Red Hat OpenShift
New Relic.
APPDYNAMICS
dynatrace
Release Orchestration
Build, Manage & Monitor release pipelines for better control & transparency.
Continuous Testing
Perform continuous testing with different test automation tools. Perform auto or manual approval.
Deployment Automation
Standardize & automate deployment across multiple environments.
Intelligent Reporting
Gain complete visibility and transparency of releases with intelligent real-time reports & dashboards.
230922

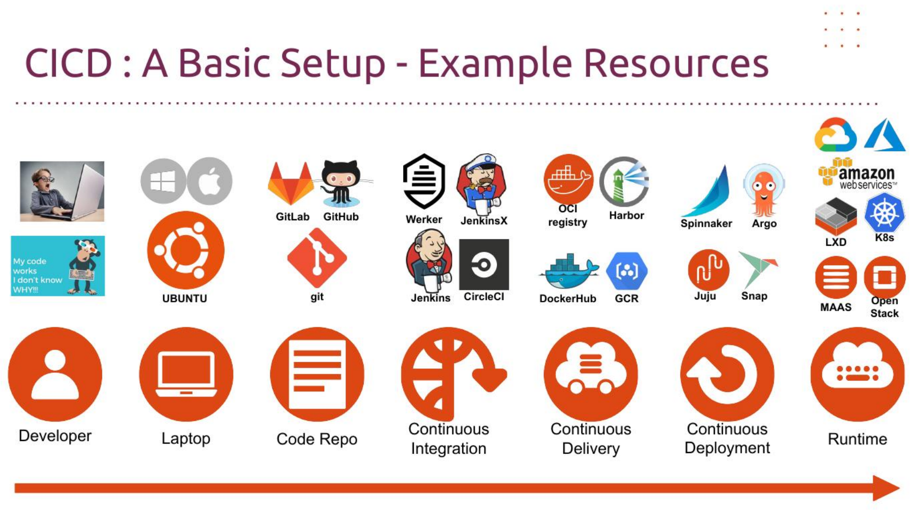

Strategy –
Integration & Automation
01
7 Synchronize
Backlog
1 Develop
2 Build & Test
3 Static Analysis
4 Check in
No
Pass?
Yes
6 Static Analysis
6 Unit Tests
5 Build
CI
Per Check-in
CI/CD Pipeline
CD
7 Deploy to QA/Stage
8 Dynamic Analysis
8 Regression Testing
Pass?
Yes
Stage then Prod
© 2016 VERACODE INC.
21

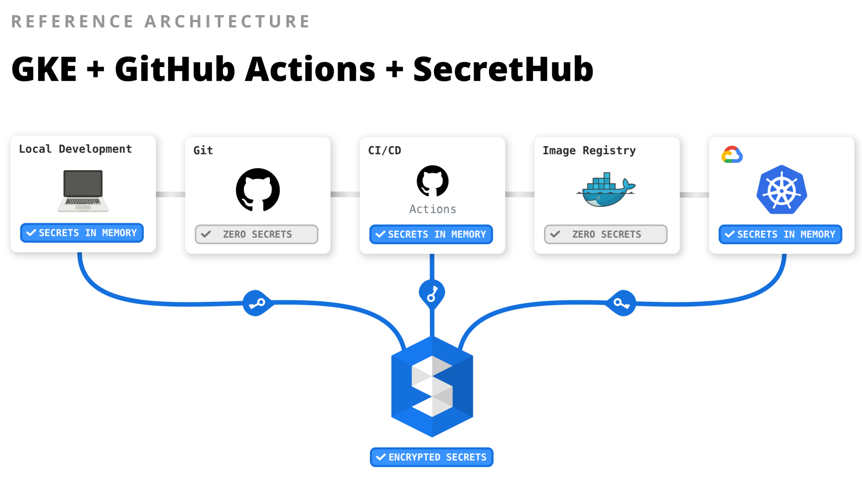
DevOps - CICD


Continuous Integration
Push code fixes
Automated build and test
Automated build and test
Review and approve
Push code changes
Create new branch
Deploy Review App
Merge
Continuous Deployment
Automated build, test, deploy
Deploy to production


Code Development
Continuous Integration
Continuous Deployment
OpsMx
1 Code Pushed to Github
2
Git- Repository (SCM)
Static Code Analysis
4 CI performs Builds post trigger
Jenkins
3 Build Triggers
5 Perform Build Tests
Testing
6 Store Builds
7 Bake Trigger
JFrog
Artifactory Storage Management
8 Bake and Publish
Spinnaker
9 Deployment Testing
10 Continuous Deployment
Cloud Infra
Staging / Test
Deployment Strategy
Deb Image
Deb Image
Production
Cluster

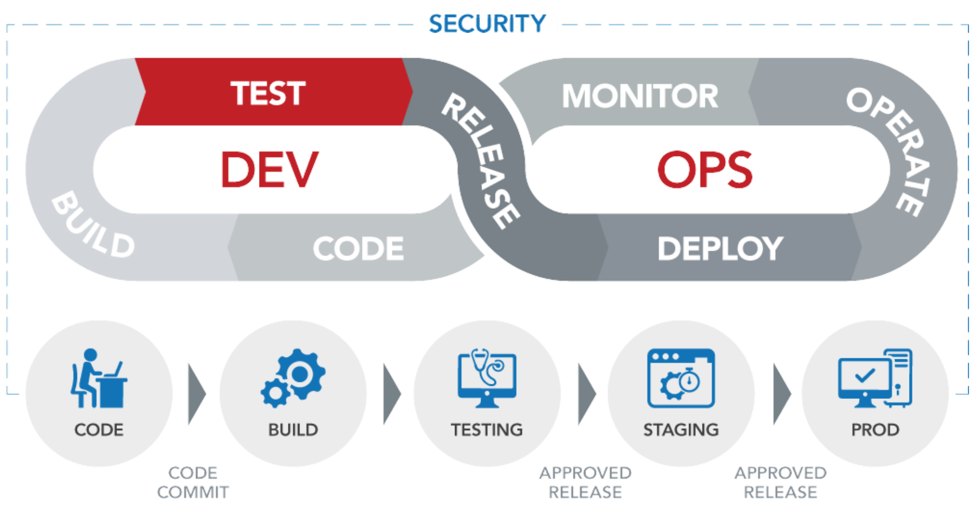

Developer
1 git commit
Github
2 webhook trigger
Jenkins Master
git clone 4
3 trigger build jobs
8 (re)deploy container(s)
Deploy Jobs
6 trigger deploy job
Build Jobs
7 pull images
build+push to DTR 5
DTR

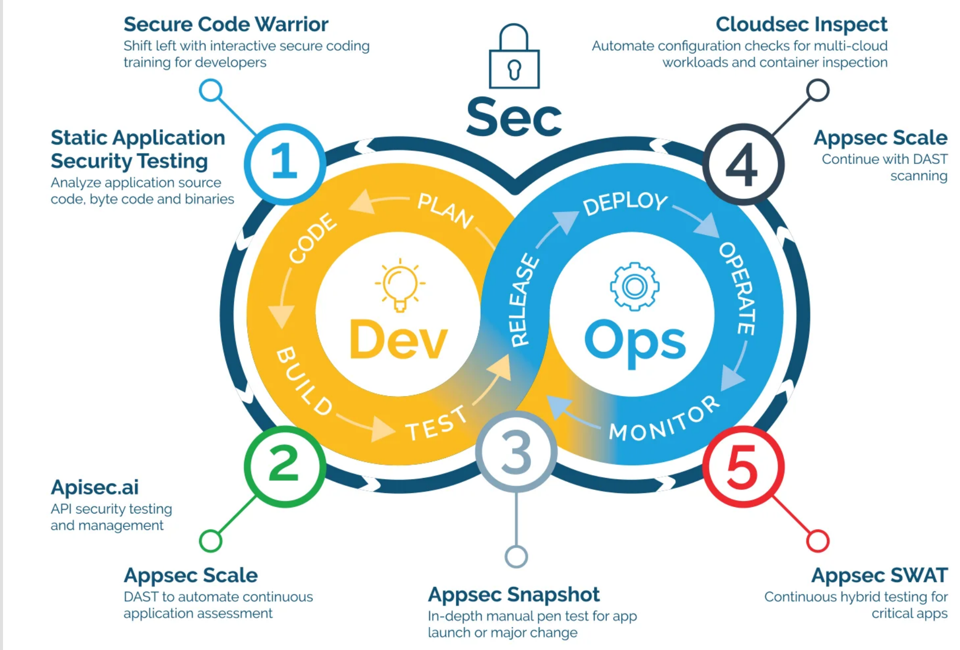

CICD : A Basic Setup - Example Resources

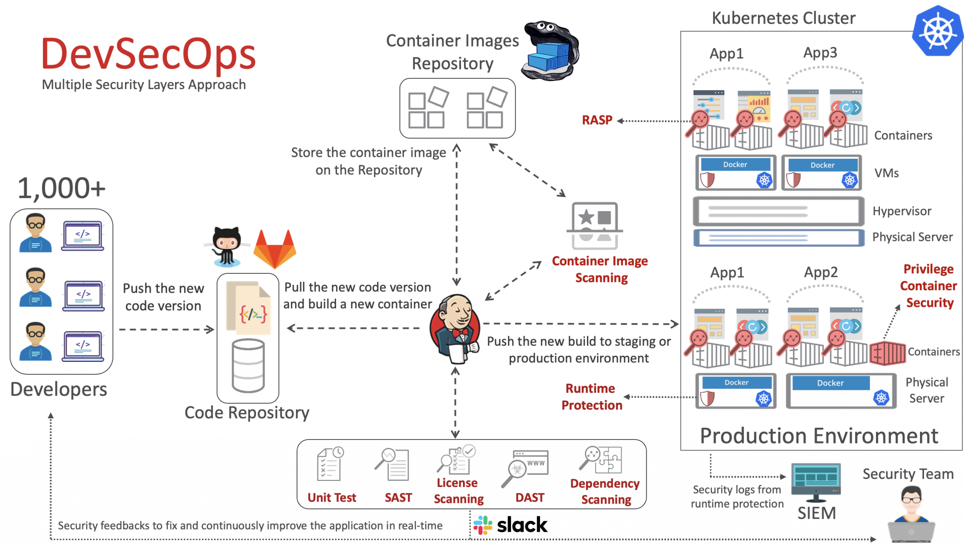

REFERENCE ARCHITECTURE

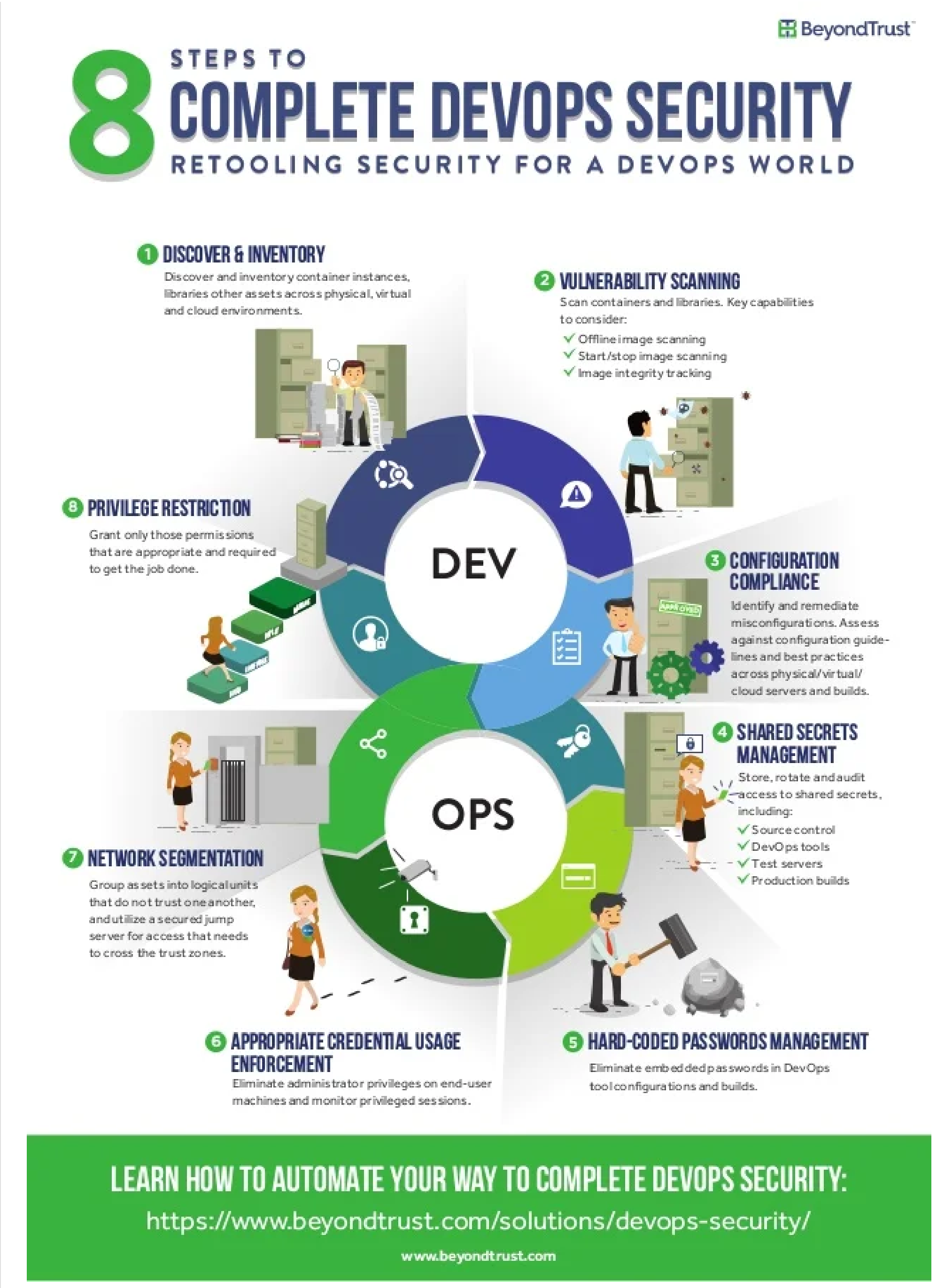
GKE + GitHub Actions + SecretHub

DevOps - Security




Secure Code Warrior
Shift left with interactive secure coding training for developers
Cloudsec Inspect
Automate configuration checks for multi-cloud workloads and container inspection
Sec
Static Application Security Testing
Analyze application source code, byte code and binaries
Appsec Scale
Continue with DAST scanning
1
4
PLAN
CODE
BUILD
TEST
Dev
RELEASE
DEPLOY
OPERATE
MONITOR
Ops
2
3
5
Apisec.ai
API security testing and management
Appsec Scale
DAST to automate continuous application assessment
Appsec Snapshot
In-depth manual pen test for app launch or major change
Appsec SWAT
Continuous hybrid testing for critical apps


DevSecOps
Multiple Security Layers Approach
Container Images Repository
Kubernetes Cluster
App1
App3
RASP
Containers
VMs
Hypervisor
Physical Server
Store the container image on the Repository
Container Image Scanning
1,000+
Developers
Push the new code version
Code Repository
Pull the new code version and build a new container
Push the new build to staging or production environment
App1
App2
Privilege Container Security
Containers
Physical Server
Runtime Protection
Production Environment
Unit Test
SAST
License Scanning
DAST
Dependency Scanning
Security logs from runtime protection
SIEM
Security Team
Security feedbacks to fix and continuously improve the application in real-time
slack



BeyondTrust

8 STEPS TO COMPLETE DEVOPS SECURITY

RETOOLING SECURITY FOR A DEVOPS WORLD

1 DISCOVER & INVENTORY
Discover and inventory container instances, libraries other assets across physical, virtual and cloud environments.

2 VULNERABILITY SCANNING
Scan containers and libraries. Key capabilities to consider:
- ✓ Offline image scanning
- ✓ Start/stop image scanning
- ✓ Image integrity tracking

3 CONFIGURATION COMPLIANCE
Identify and remediate misconfigurations. Assess against configuration guidelines and best practices across physical/virtual/cloud servers and builds.

4 SHARED SECRETS MANAGEMENT
Store, rotate and audit access to shared secrets, including:
- ✓ Source control
- ✓ DevOps tools
- ✓ Test servers
- ✓ Production builds

5 HARD-CODED PASSWORDS MANAGEMENT
Eliminate embedded passwords in DevOps tool configurations and builds.

6 APPROPRIATE CREDENTIAL USAGE ENFORCEMENT
Eliminate administrator privileges on end-user machines and monitor privileged sessions.

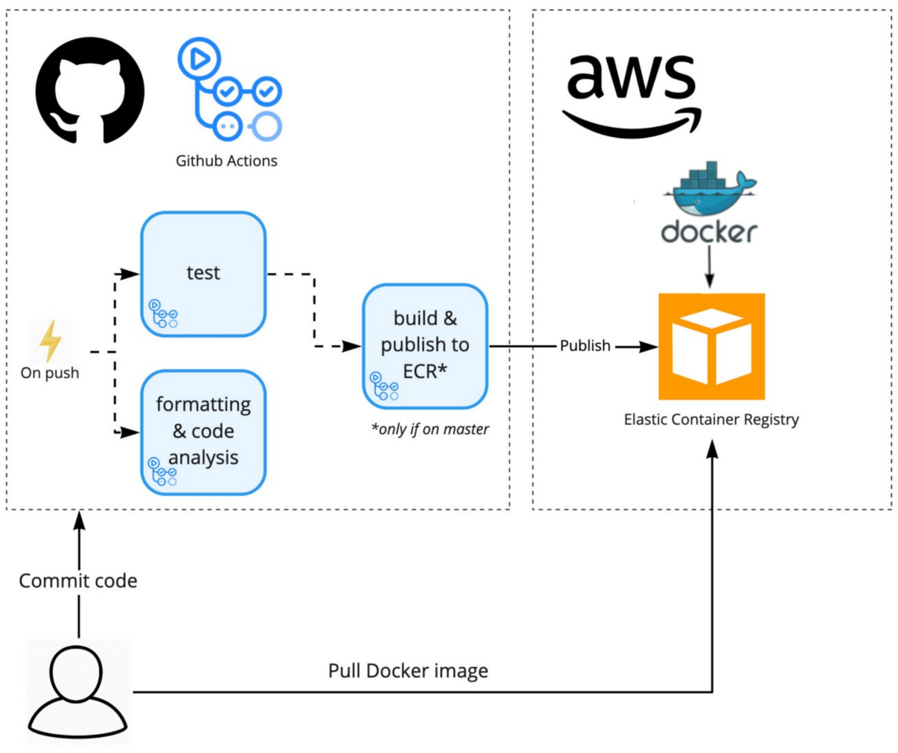
7 NETWORK SEGMENTATION
Group assets into logical units that do not trust one another, and utilize a secured jump server for access that needs to cross the trust zones.

8 PRIVILEGE RESTRICTION
Grant only those permissions that are appropriate and required to get the job done.

LEARN HOW TO AUTOMATE YOUR WAY TO COMPLETE DEVOPS SECURITY:
https://www.beyondtrust.com/solutions/devops-security/

www.beyondtrust.com

DevOps - Deployment

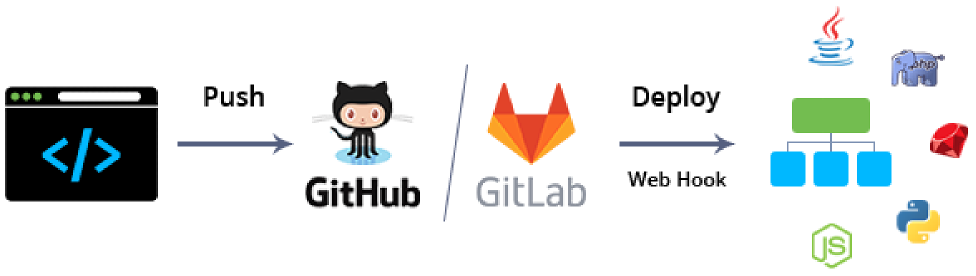


Web Deployment Preparation

- **Create A New GitHub Repository**
- **Create `.gitignore`**
- **Create `requirements.txt`**
- **Create `.env`**
- **Connect github**

Create A New GitHub Repository

GitHub Repository



Create `.gitignore`

-> venv/

-> .env

-> ...

Create `requirements.txt`

- `pip freeze`

-> CMD: pip freeze > requirements.txt

Create `.env`

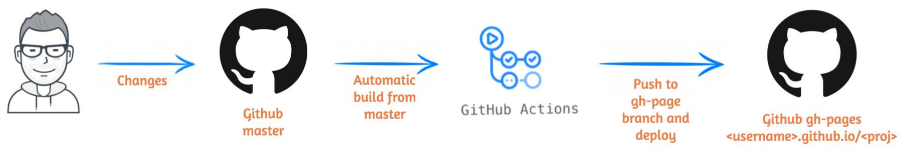
-> SECRET_KEY=**
-> DUBEG=************

Connect github

- git init
- git add README.md
- git commit -m "first commit"
- git branch -M main
- git remote add origin `git@github.com:ericarthuang/Learning_Flask_Jovin.git`
- (git remote remove origin)
- git push -u origin main

Web Service - GitHub Pages (for static website)

GitHub Pages


Changes
Github master
Automatic build from master
GitHub Actions
Push to gh-page branch and deploy
Github gh-pages <username>.github.io/<proj>

GitHub Pages WorkFlow

Reference：Getting Started with GitHub Pages

`Settings -> Pages -> Build and deployment -> Branch -> main`

Create Branch for Site

git checkout -b site

`Settings -> Pages -> Build and deployment -> Branch -> **site**`

> My Copy Learning Website
> `https://ericarthuang.github.io/{NameOfReop}/`

Web Service - Railway

> Easiest Way To Connect Django To A Postgres Database

Web Service - render

> Copy_Learning_Website_Flask

> Render Serveice Website

> - Dashboard -> create new "web server" -> connect GitHub

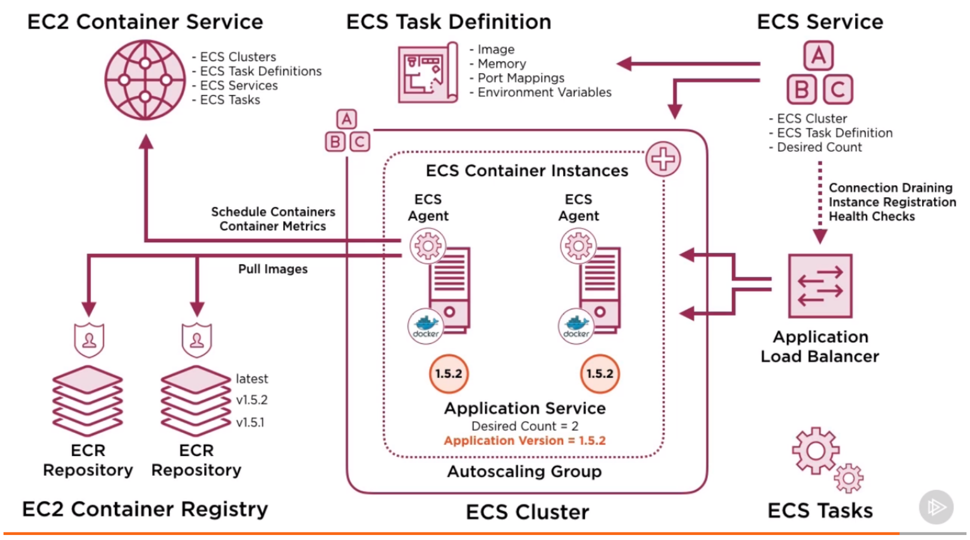
> Connect Setting
>
> - Name(Website)
> - Environment
> - Branch
> - Build Command
>
> `-> pip install gunicorn`
> `-> create requirements.txt(pip freeze > requirements.txt)`
> `-> pip install -r requirements.txt`
>
> - Start Command
>
> `-> gunicorn app:app`

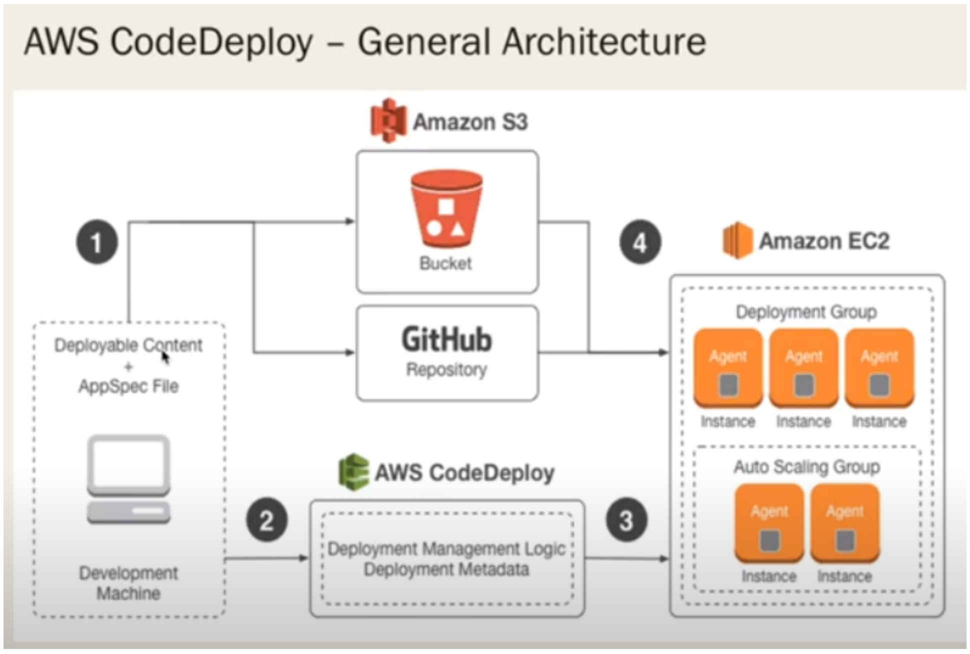
Web Service - Netlify

> Netlify Serveice Website
>
> - Netlfily Builds
> - New Site from Git -> GitHub -> select repositories -> site settings -> change sitename
>
> My Copy Learning Website
> `https://{NameOfReop}.netlify.app/`

Web Service - AWS EC2

> AWS EC2





- Launch instance

-> Name and tags
-> Application and OS Images (Amazon Machine Image)
-> Instance type
-> Key pair (login): Crerate new key pair(Keep the filename.pem)
-> Network settings
-> Configure storage
-> Launch Instance
-> Connect: EC2 serial console, RDP client:get rhe password for remote connect

- New Tenrimal

-> firewall setup
-> Windiows Denfencer Firewall
-> Advanced Setting

> -> Windiows Denfencer Firewall Properties
> -> Domain Profile/Private Profile/Public Profile: Inbound Connections: Allow

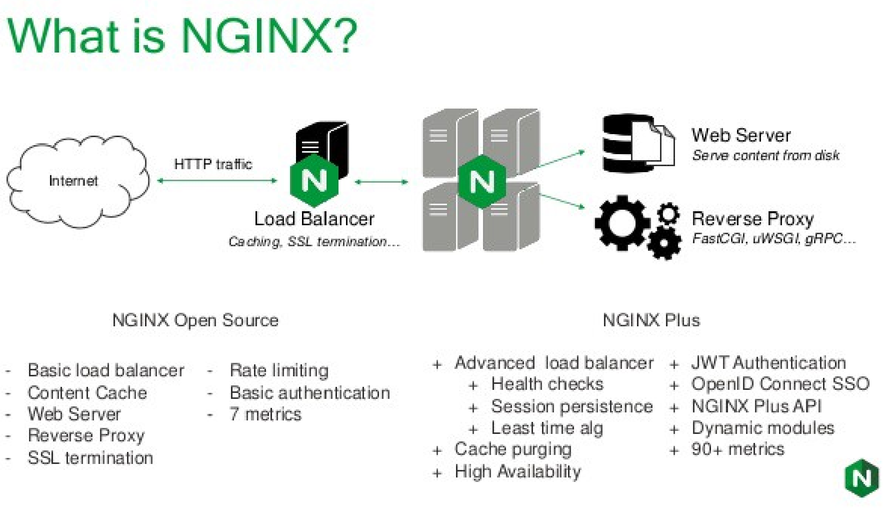
> - Connect:
>
> -> Connect Method: EC2 Instance Connect (browser-based SSH connection)
> -> Will find the `amazon-linux console`
>
> ```
> https://aws.amazon.com/amazon-linux-2/
> 13 package(s) needed for security, out of 16 available
> Run "sudo yum update" to apply all updates.
> [ec2-user@ip-172-31-44-21 ~]$
> ```
>
> - amazon-linux console
>
> -> CONSOLE: sudo yum install git -y
> -> CONSOLE: sudo amazon-linux-extras install docker -y
> -> CONSOLE: sudo systemctl enable docker.service
> -> CONSOLE: sudo systemctl start docker.service
> -> CONSOLE: sudo usermod -aG docker ec2-user
> -> CONSOLE: sudo curl -SL `https://github.com/docker/compose/releases/download/v2.12.2/docker-compose-linux-x86_64 -o /usr/local/bin/docker-compose`
> -> CONSOLE: sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose
>
> - Create key for Github
>
> -> CONSOLE: ssh-keygen -t ed25519 -b 4096
> -> CONSOLE: cat ~/.ssh/id_ed25519.pub
>
> - Create Deploy Key in GitHub Reop `settings`
>
> -> settings -> Deploy Keys -> add deploy key
>
> - clone `ssh url`
>
> -> CONSOLE: git clone git@github.com:ericarthuang/Django_Docker_Deployment.git
> -> CONSOLE: ls
> -> show: Django_Docker_Deployment
> -> CONSOLE: cd Django_Docker_Deployment
>
> - Add the configuration
>
> -> CONSOLE: cp .env.sample .env
> -> CONSOLE: vi .env
> -> modidy .env (use 'Public DNS (IPv4)' for hostname)
> -> CONSOLE: cat .env
>
> - Launch Application

> -> CONSOLE: docker-compose -f docker-compose-deploy.yml up -d

---

Nginx / Gunicron / Docker

> ## Nginx - Webserver: Act as Proxy / Handle SSL Ternimation



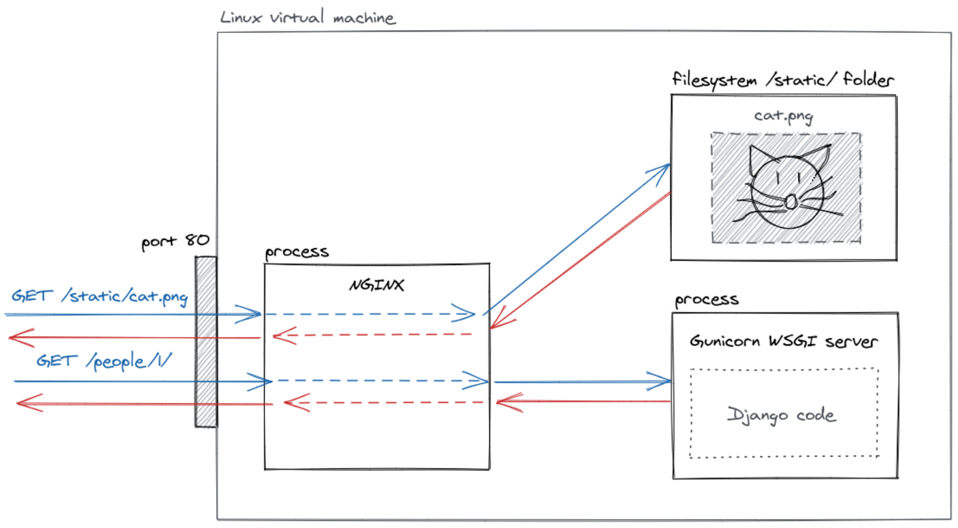
What is NGINX?

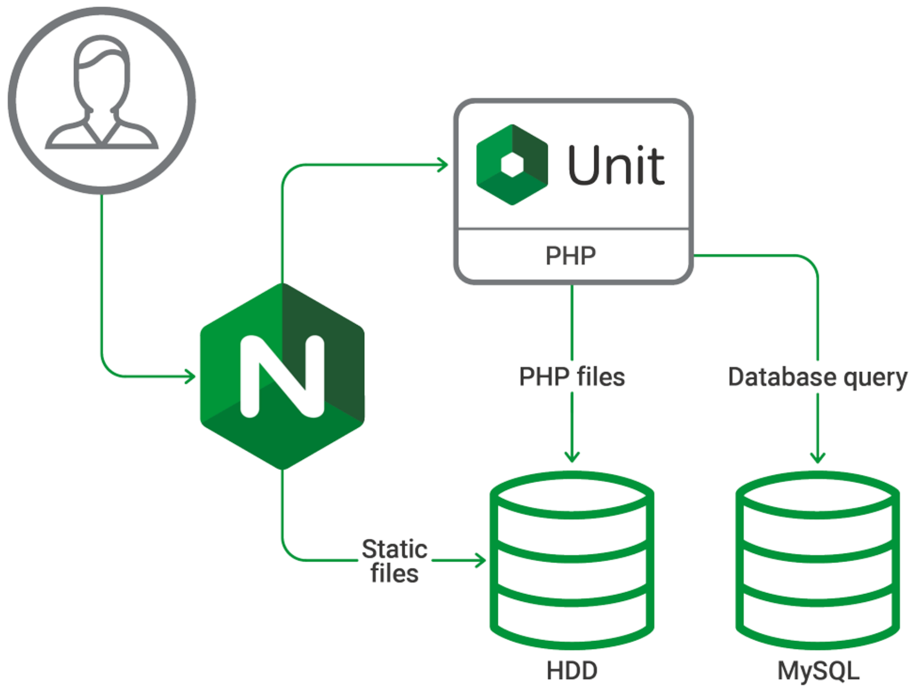
NGINX Open Source

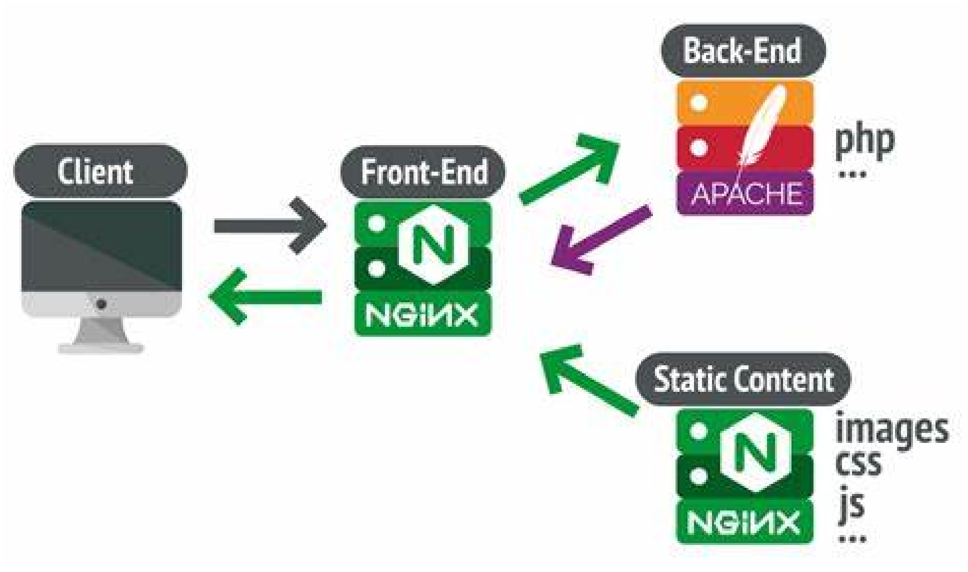
- Basic load balancer
- Content Cache
- Web Server
- Reverse Proxy
- SSL termination
- Rate limiting
- Basic authentication
- 7 metrics

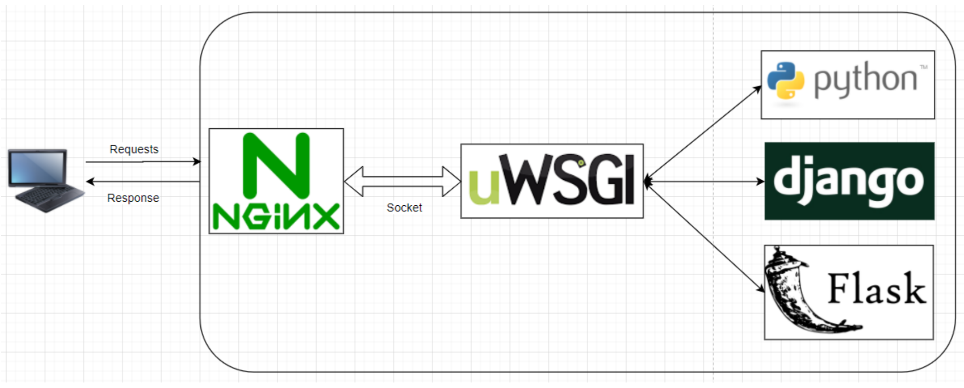
NGINX Plus

+ Advanced load balancer
  + Health checks
  + Session persistence
  + Least time alg
+ Cache purging
+ High Availability
+ JWT Authentication
+ OpenID Connect SSO
+ NGINX Plus API
+ Dynamic modules
+ 90+ metrics



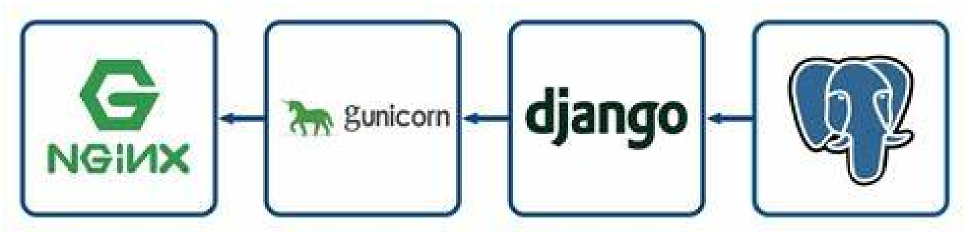

Unit
PHP
PHP files
Database query
Static files
HDD
MySQL

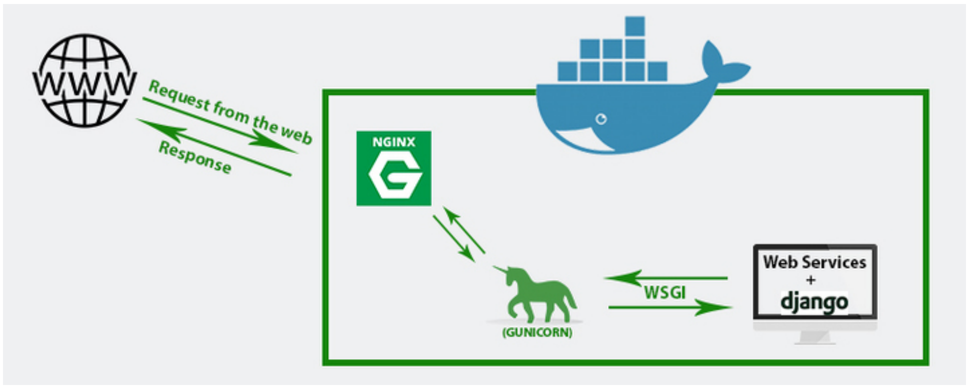

Client
Front-End
NGINX
Back-End
APACHE
php
...
Static Content
NGINX
images
css
js
...


Requests
Response
NGINX
Socket
uWSGI
python
django
Flask

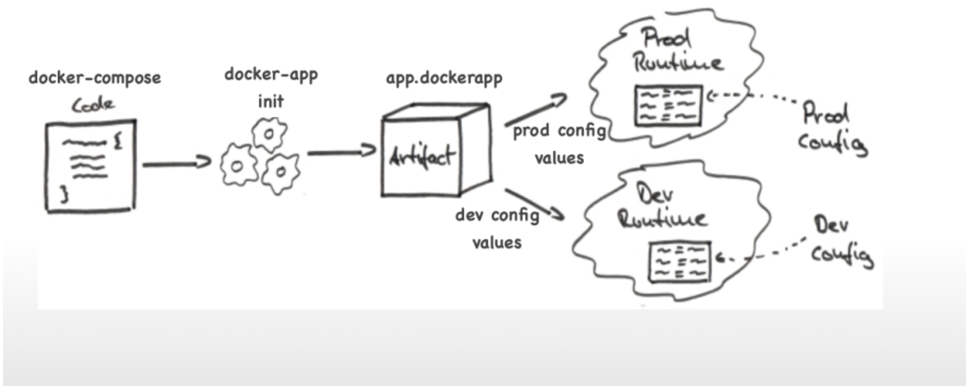
Gunicorn

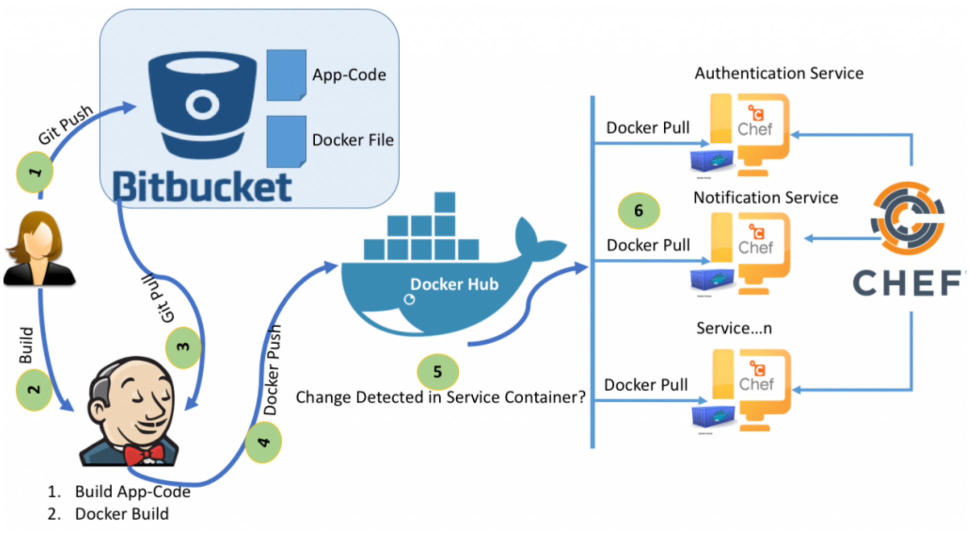
- pip install gunicron
- pip install httptools
- python.exe -m pip install --upgrade pip
- pip install uvloop
- gunicron -w 4 -k uvicorn.workers.UvicornWorker app.main:app --bind 0.0.0.0

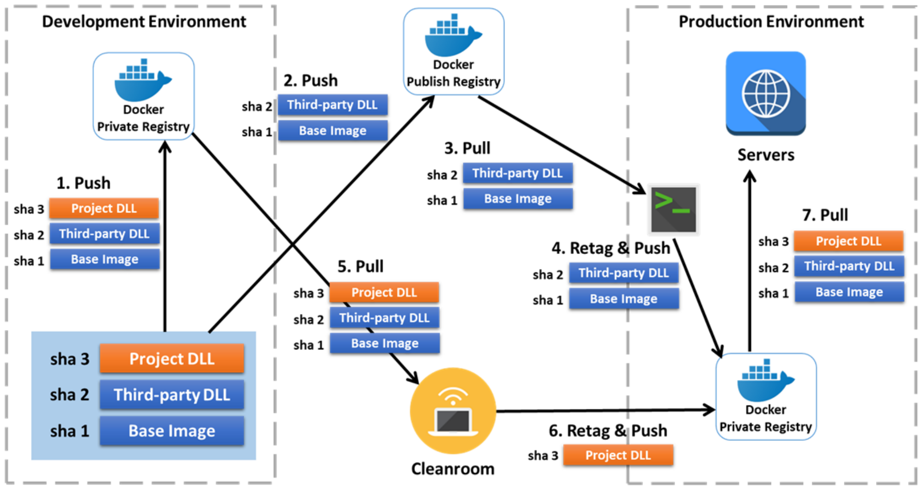


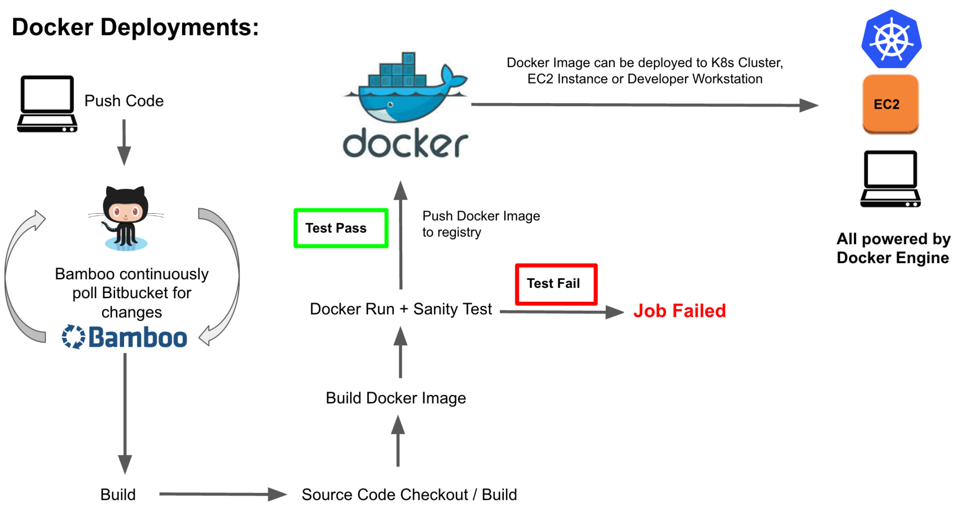


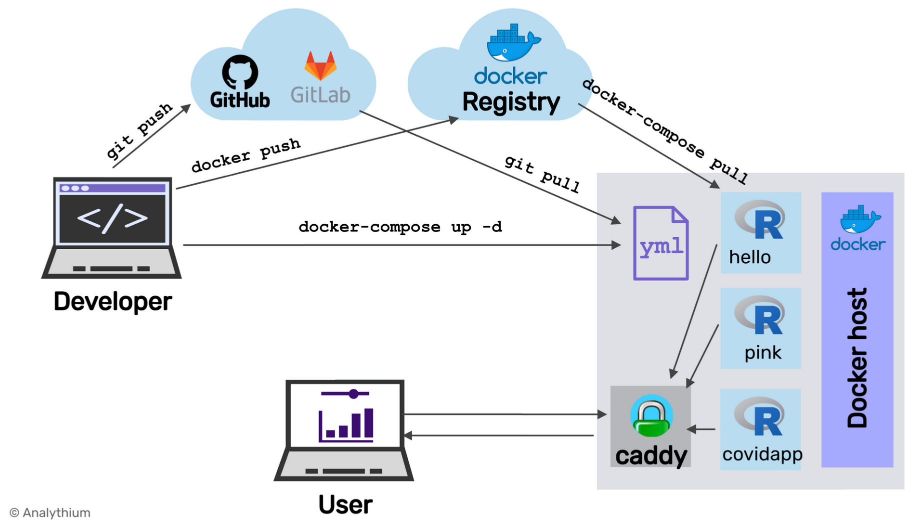
Docker

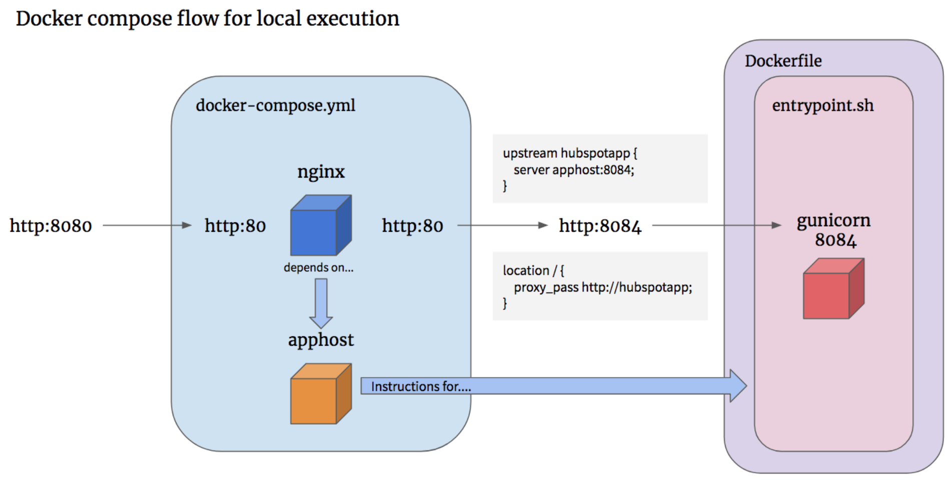



Git Push
1
App-Code
Docker File
Bitbucket
Build
2
Git Pull
3
Docker Push
4
1. Build App-Code
2. Docker Build
Docker Hub
5
Change Detected in Service Container?
Authentication Service
Docker Pull
Chef
6
Notification Service
Docker Pull
Chef
Service...n
Docker Pull
Chef
CHEF


Development Environment
Docker Private Registry
1. Push
sha 3 Project DLL
sha 2 Third-party DLL
sha 1 Base Image
2. Push
sha 2 Third-party DLL
sha 1 Base Image
Docker Publish Registry
3. Pull
sha 2 Third-party DLL
sha 1 Base Image
4. Retag & Push
sha 2 Third-party DLL
sha 1 Base Image
5. Pull
sha 3 Project DLL
sha 2 Third-party DLL
sha 1 Base Image
Cleanroom
6. Retag & Push
sha 3 Project DLL
Production Environment
Servers
7. Pull
sha 3 Project DLL
sha 2 Third-party DLL
sha 1 Base Image
Docker Private Registry


Docker Deployments:
Push Code
Bamboo continuously poll Bitbucket for changes
Bamboo
Build
Source Code Checkout / Build
Build Docker Image
Docker Run + Sanity Test
Test Fail
Job Failed
Test Pass
Push Docker Image to registry
docker
Docker Image can be deployed to K8s Cluster, EC2 Instance or Developer Workstation
EC2
All powered by Docker Engine



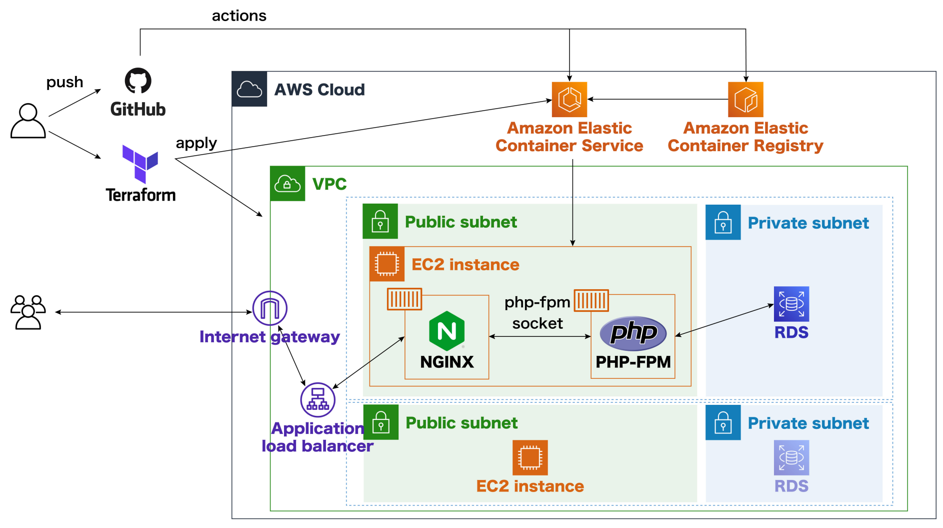


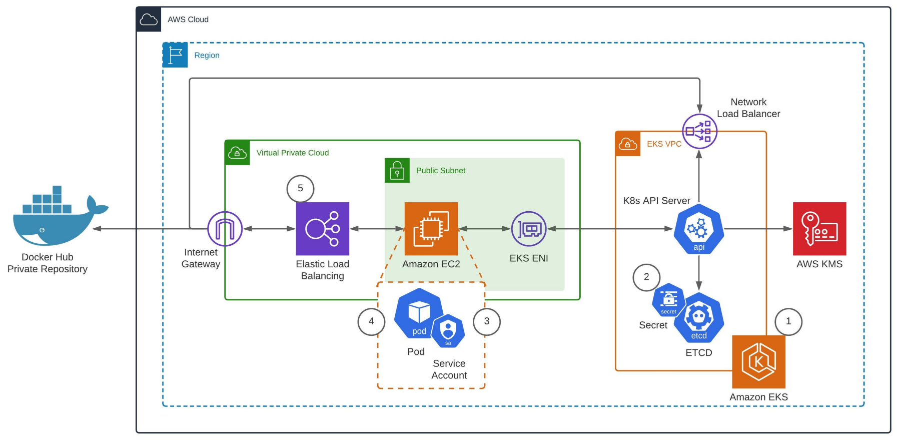
```
upstream hubspotapp {
  server apphost:8084;
}
```

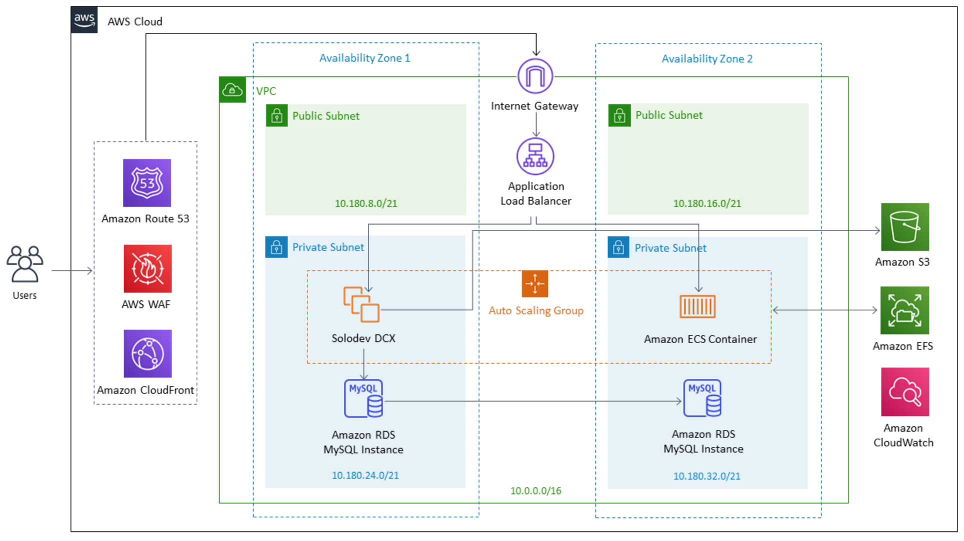
```
location / {
  proxy_pass http://hubspotapp;
}
```

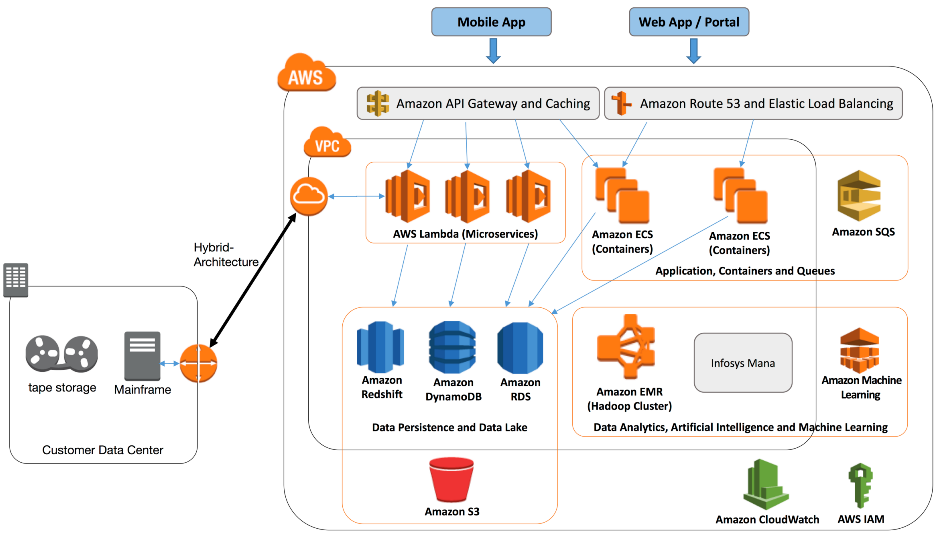
AWS Cloud

Cloud Web Services
Cloud Replication Services
Cloud Backup Services
Cloud Database Services
Managed aws
Cloud Infrastructure Services
SAP HANA
Cloud SAP HANA Services
Consultancy
Implementation
Administration
Monitoring
Support
Advanced Resource Management


actions
push
GitHub
apply
Terraform
AWS Cloud
Amazon Elastic Container Service
Amazon Elastic Container Registry
VPC
Public subnet
Private subnet
EC2 instance
php-fpm socket
NGINX
PHP-FPM
RDS
Internet gateway
Application load balancer
Public subnet
Private subnet
EC2 instance
RDS


AWS Cloud
Region
Network Load Balancer
EKS VPC
Virtual Private Cloud
Public Subnet
5
K8s API Server
Docker Hub Private Repository
Internet Gateway
Elastic Load Balancing
Amazon EC2
EKS ENI
api
AWS KMS
2
secret
Secret
etcd
ETCD
1
4
pod
sa
3
Pod
Service Account
Amazon EKS


aws
AWS Cloud
Availability Zone 1
Availability Zone 2
VPC
Internet Gateway
Public Subnet
Public Subnet
Application Load Balancer
10.180.8.0/21
10.180.16.0/21
Amazon Route 53
AWS WAF
Amazon CloudFront
Users
Private Subnet
Private Subnet
Auto Scaling Group
Solodev DCX
Amazon ECS Container
MySQL
Amazon RDS MySQL Instance
Amazon RDS MySQL Instance
10.180.24.0/21
10.0.0.0/16
10.180.32.0/21
Amazon S3
Amazon EFS
Amazon CloudWatch


Mobile App
Web App / Portal
AWS
Amazon API Gateway and Caching
Amazon Route 53 and Elastic Load Balancing
VPC
AWS Lambda (Microservices)
Amazon ECS (Containers)
Amazon ECS (Containers)
Amazon SQS
Application, Containers and Queues
Hybrid-Architecture
tape storage
Mainframe
Customer Data Center
Amazon Redshift
Amazon DynamoDB
Amazon RDS
Data Persistence and Data Lake
Amazon EMR (Hadoop Cluster)
Infosys Mana
Amazon Machine Learning
Data Analytics, Artificial Intelligence and Machine Learning
Amazon S3
Amazon CloudWatch
AWS IAM

-- Memo End --