Ghost Blog on AWS

Iftikhar Hossain
https://www.linkedin.com/in/engiftikhar

Terraform

Terraform is an open-source infrastructure as code software tool created by HashiCorp. Users define and provide data center infrastructure using a declarative configuration language known as HashiCorp Configuration Language, or optionally JSON.

- Write declarative config files
- Installable modules
- Plan and predicts changes
- State Management
- Provision Infrastructure in familiar languages
- Manage infrastructure across clouds

Terraspace

Terraspace is a Terraform Framework that optimizes for Infrastructure-as-code (IaC) happiness.

- An organized project structure.
- Keeping your code DRY by building Terraform projects from your app and config/terraform files and easily manageable multiple environments.
- The automated creation of backend buckets.
- CLI hooks: Call custom commands before and after terraform commands.
- You can deploy all or some infrastructure in a single command: *terraspace all up*.
- Built-in Test Framework.
- Terraform Cloud and Terraform Enterprise are also supported.

AWS Services

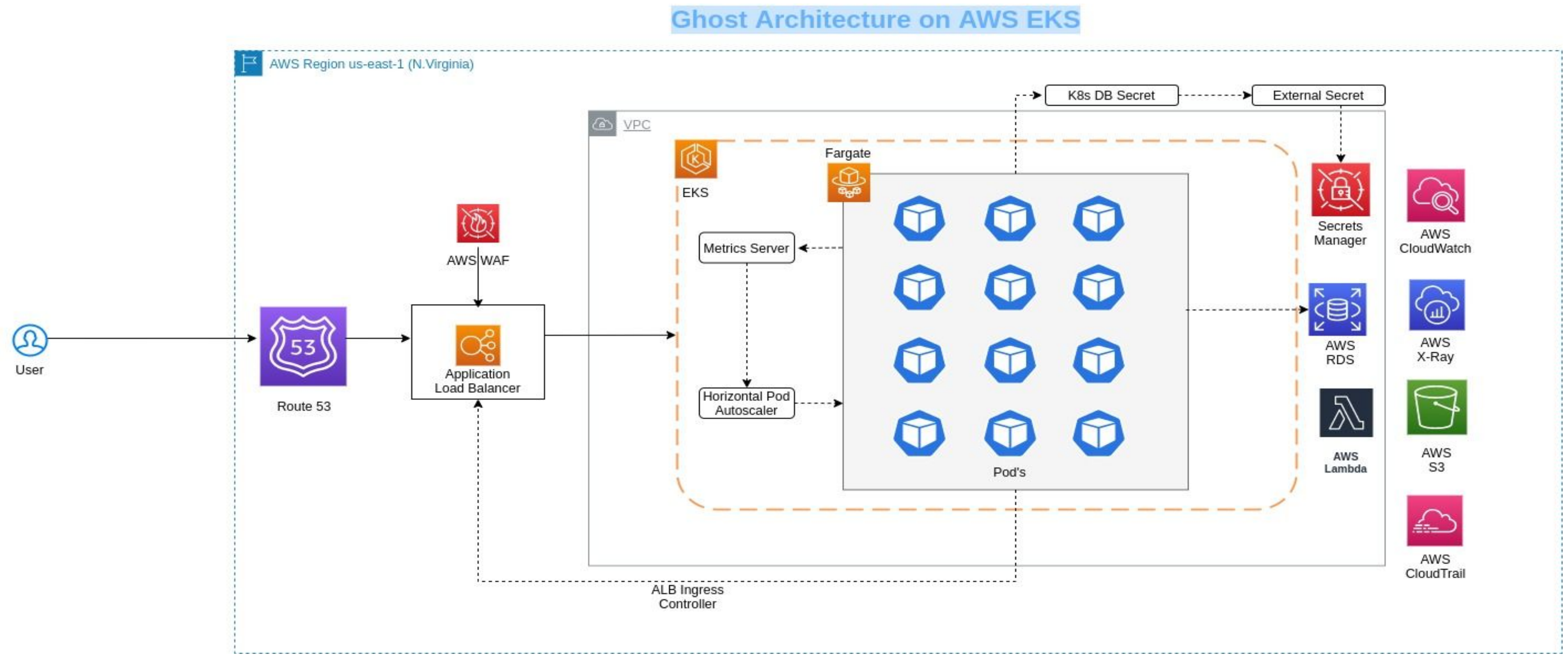
- Create VPC [Public & Private Subnet, Nat Gateway, Internet Gateway]
- Use Multi-AZ MySQL RDS / AWS Aurora [Private Subnet]
- Create EKS Cluster with Fargate [Private Subnet]
- AWS Secret Manager to Manage Sensitive data [like Terraform DB credentials]
- ALB can be Configured to save access logs to S3
- Static images can be serve via AWS S3 via AWS Cloudfront
- AWS Lambda will be responsible for processing access logs
- AWS WAF is responsible for traffic control [Skip AWS Shield Advanced for pricing]

Helm and ArgoCD

- Create Kubernetes manifest files to deploy ghost applications
- Install Ingress Controller to serve the frontend applications using Helm Charts
- Install Kubernetes external secret using Helm Charts to manage sensitive data like credentials.
- Install ArgoCD tool using Helm Charts.
- Install metrics server on EKS and use horizontal pod autoscaler to scale the traffic.

Ghost Blog Architecture on AWS EKS

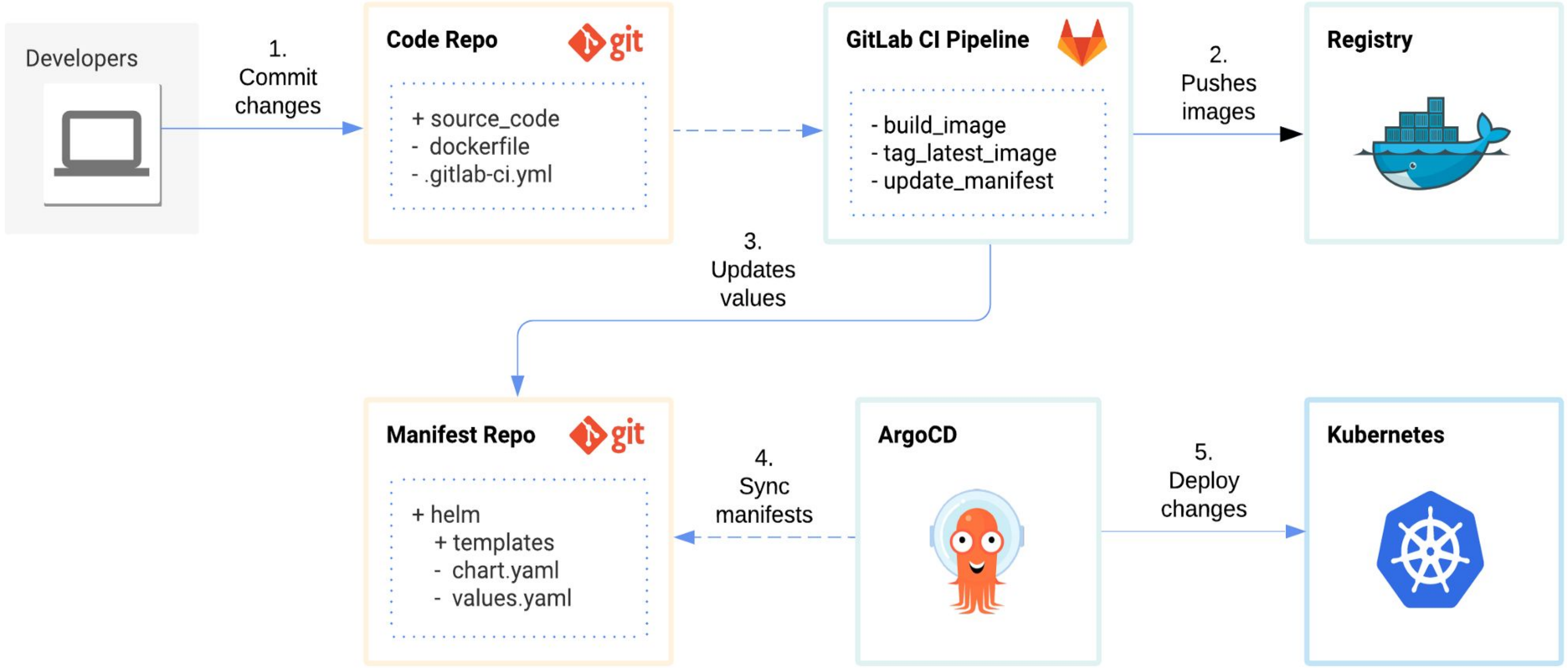
Ghost is an open source, professional publishing platform built on a modern Node.js technology stack designed for teams who need power, flexibility and performance.



CI/CD implementation for Ghost blog

- Using Gitlab CI, integrated with Git repo
- Build and test the code
- Push the docker images with tag in Dockerhub/ AWS ECR
- Create another job with update manifest files in deploy stage.
- This clones and commits news changes to Helm manifest repository.
- Argo CD automatically detects this change and triggers the sync to change the state of resources as per the Git repo
- Sync action is triggered through a webhook action from Jenkins CI after commit or after a regular polling interval.pository.
- Argo CD runs as a controller on the Kubernetes cluster, which continuously monitors running applications and compares the current state with the desired target state (as specified in the GitOps repository).
- The controller detects an out-of-sync application state, which is done by diff-in Helm manifest template outputs and optionally takes corrective action.

CI/CD implementation for Ghost blog



Security and Monitoring

- ECR Scan allowed to check the docker images
- Implement WAF to protect the frontend applications
- Allowed Office IP/VPN IP with specific ports only Security Groups
- Implement AWS Lambda functions to delete the posts
- Implement Monitoring solution like AWS Prometheus, AWS Grafana, AWS X-Ray, Cloudwatch
- Implement logging solution with Cloudwatch logs, AWS Elasticsearch
- For Security, use Cloudtrail and AWS Guard Duty and AWS Inspector.

Disaster Recovery

- Create Nat Gateway per AZ to protect Availability zone failure
- Deploy the same architecture on another region.
- Implement AWS Global Accelerator provides static IP addresses that act as a fixed entry point to both ALB's and change GA point to AWS Route53.
- EKS cluster are running on different region and create VPC peering between them.
- Use AWS RDS (MYSQL/Aurora) Multi-AZ for replication, automatic failover and backups.

Questions

- Stateful set/ headless Service (https://kubernetes.io/docs/tutorials/stateful-application/basic-stateful-set)
- Storage class (https://kubernetes.io/docs/concepts/storage/storage-classes/)
- Manage Terraform state files
- Terraform testing (https://terratest.gruntwork.io/)
- Kube practise
- Security Context on Pod (https://kubernetes.io/docs/tasks/configure-pod-container/security-context/)