
Technical Screening



Infrastructure

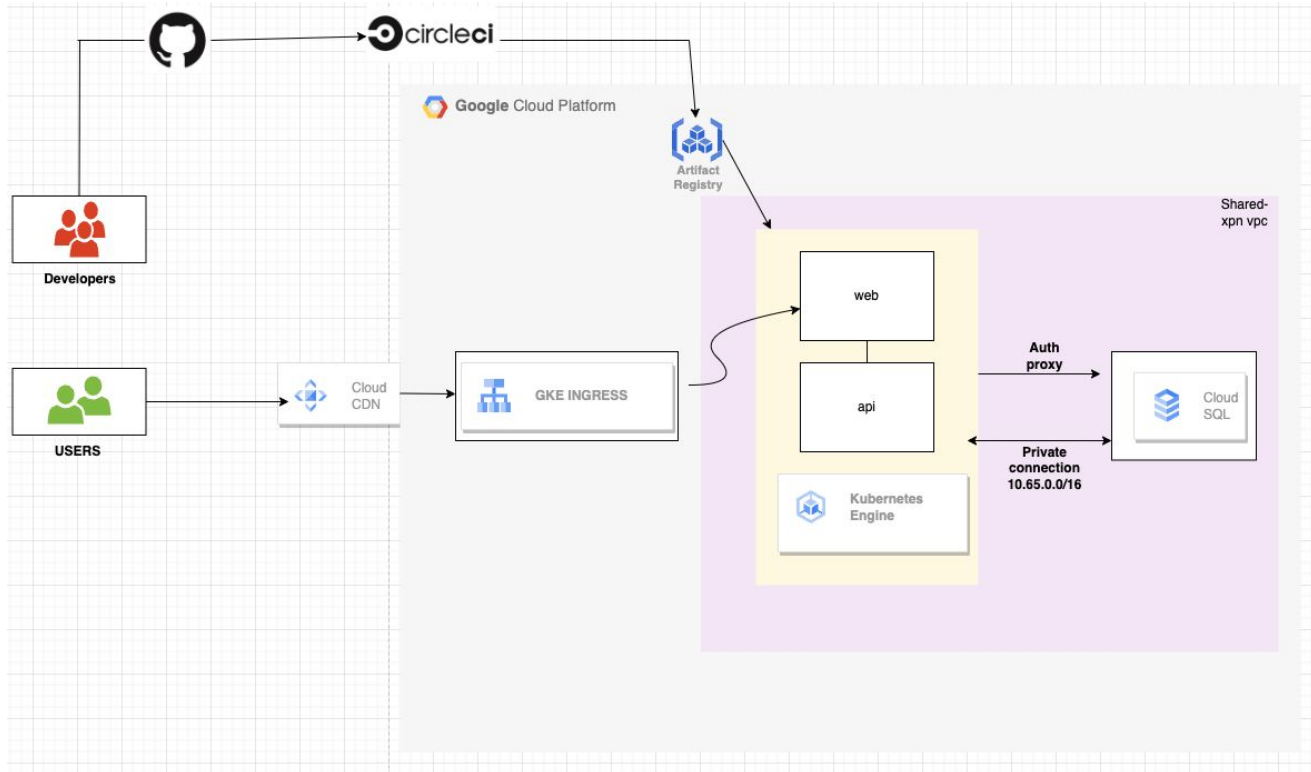
➤ How

- All infrastructure is deployed using Terraform
- Cloud Platform used is GCP
- All authentication to GCP is done via Service Account using following command:
 - `gcloud auth activate-service-account --key-file=~/.path/to/key/file`

➤ Why

- Eliminates the need for manual intervention
- Handy in case of Disaster and Recovery
- Allows teams to collaborate seamlessly on infrastructure management

High Level Diagram



GKE Autopilot

- A private GKE Autopilot Cluster is used where api and web tier of application is deployed.
- Only master authorized network ranges can access Cluster's control plane
- Outbound connections from GKE cluster are handled through Cloud NAT
- **Why?**
 - You don't need to manage underlying infra, master nodes or etcd cluster.
 - Automated scaling and Resource Management, Improved Security and Compliance
 - Benefits of GKE Autopilot Clusters in terms of handling server failures:
 - Automated Node Repair and Replacement
 - Node Pools with Multiple Zones
 - Improved Node Maintenance

Cloud SQL

- Type of Google Cloud SQL Postgres instance that is designed to provide a more secure and isolated environment for your database.
- Private Services Access is enabled for Cloud SQL
- Instance name:
 - Private-instance-7124d943
- Database name:
 - Nodejs-db

SUBNETS

STATIC INTERNAL IP ADDRESSES

FIREWALLS

ROUTES

VPC NETWORK PEERING

PRIVATE SERVICE CONNECTION

ALLOCATED IP RANGES FOR SERVICES

PRIVATE CONNECTIONS TO SERVICES

Internal IP address ranges that are allocated for services private connection [Learn more](#)

ALLOCATE IP RANGE

RELEASE

<input type="checkbox"/>	Name ↑	Internal IP range	Service producer	Connection name
<input type="checkbox"/>	private-ip-address	10.2.0.0/16	Google Cloud Platform	servicenetworking-googleapis-com

Connection between Cloud SQL and GKE Autopilot


- We use Cloud SQL Auth proxy (with private IP) to access Cloud SQL instance from application running in GKE
- GKE is connecting to Cloud SQL using GKE's Workload Identity feature.
 - This basically binds K.S.A to Google Service account
- We run cloud-sql-proxy in a sidecar pattern

```
🍏 ~/toptal/Anurag-Sharma git main
> kubectl get serviceaccount
```

NAME	SECRETS	AGE
default	0	9d
terraform-sa-ksa	0	4d7h

Artifact Registry

Filter Enter property name or value

<input type="checkbox"/>	Name ↑	Format	Type	Location	Description	Labels	Version policy ?	Encryption ?	Encryption key	Created	Updated	Size	
<input type="checkbox"/>	gcp-node-test	 Docker	Standard	us-central1 (Iowa)	Repo for storing nodejs images			Google-managed key	—	5 days ago	1 day ago	438.8 MB	^

 us-central1-docker.pkg.dev >  peak-apparatus-379619 >  gcp-node-test 

Repository Details

Format	Docker
Type	Standard

Filter Enter property name or value

<input type="checkbox"/>	Name ↑	Created	Updated
<input type="checkbox"/>	 api-nodejs-image	2 days ago	1 day ago
<input type="checkbox"/>	 web-nodejs-image	2 days ago	1 day ago

CircleCI

- CircleCI pipeline has 3 jobs basically:
 - Helm-deployment
 - Build-image
 - Deploy-to-kubernetes
- CircleCI is authenticating to Google Cloud Platform using Service Account which has been added as Environment variable
- Here is what each of the job does:
 - **Helm-deployment:** Deploys a Helm chart to a Kubernetes cluster on Google Kubernetes Engine (GKE). It initializes the Google Cloud CLI, installs Helm, and updates the version of a Helm chart based on the VERSION_PREFIX and the CircleCI build number. Then it commits the changes to the chart and pushes it to the GitHub repository.

CircleCI (contd.)

- Build-image:
 - Build and push the Docker container image to the Google Artifact Registry.
- Deploy to kubernetes:
 - Deploys Helm chart to a GKE Cluster.
 - Runs a series of steps to authenticate with GCP, install/upgrade helm deployment.

The screenshot displays a CircleCI interface for a job named 'my-custom-workflow'. The job is in a 'Success' state, indicated by a green checkmark icon. The workflow is associated with the 'circleci-project-setup' repository, specifically the '2cf8384 TDD' commit. The job was executed '14m ago' and has a duration of '2m 7s' with a '92%' completion rate. The 'Jobs' section lists three steps: 'helm-deployment' (258s, 8s), 'build-image' (259s, 37s), and 'deploy-to-kubernetes' (261s, 16s). All steps are marked as successful with green checkmarks.

Jobs	Status	Step ID	Duration
helm-deployment	Success	258	8s
build-image	Success	259	37s
deploy-to-kubernetes	Success	261	16s

CircleCI (Free version caveat)

- As we discussed earlier, access to Private GKE cluster is secure and is controlled via Master Authorized Network.
- There is a feature in CircleCI where we can whitelist CircleCI IPs on a private GKE cluster by using `circleci_ip_range: true`
 - However, it is not allowed in Free Tier of CircleCI

```
deploy-to-kubernetes:  
  circleci_ip_ranges: true  
  environment:  
    DOCKER_REPO: gcp-node-test  
    IMAGE_NAME: web-nodejs-image  
    HELM_RELEASE_NAME: nodejs  
    CHART_NAME: circle-nodejs-chart  
    VERSION_PREFIX: 0.1  
    GCP_PROJECT: peak-apparatus-379619  
    GKE_CLUSTER: gke-test-us-central1  
    GKE_REGION: us-central1  
  executor: my-custom-executor
```

Dashboard Project Branch Workflow Job

All Pipelines > test-nodejs > circleci-project-setup > my-custom-workflow > **deploy-to-kubernetes (118)**

deploy-to-kubernetes Failed CONCURRENCY LIMIT

Duration / Finished	Queued	Executor / Resource Class	Branch	Commit	Author & Message
🕒 0s / 4m ago	🕒 4m 1s	🐳 Docker / 🔗	🔗 circleci-project-setup	🔗 115c4b1	👤 silly mistake again

⚠️ This job has been blocked because the IP ranges feature is not available on your plan. Please upgrade to continue building.

STEPS TESTS TIMING ARTIFACTS RESOURCES NEW

Thank You!!!