Diploma work

Sergei Chulanov

Task description

Variant 3

Using API https://covidtracker.bsg.ox.ac.uk/about-api get all data about “By country over time" for current year.

Store it into your DB: date_value, country_code, confirmed, deaths, stringency_actual, stringency.

Output the data by country_code (the country_code is set) in form of a table and sort them by deaths (or date_value which is the same) in ascending order.

Criteria	Related Module
SCM (Source code management)	git
Language	golang (front and api)
Tests	golang tests
Quality gate	golangci-lint, tflint, sonarqube
Vulnerability	trivy
IaC	terraform
Orchestration	eks
Logging	cloudwatch
Monitoring	cloudwatch
Runtime/Deployment	gitlab-ci (test and prod)
Scalability/redundancy	eks (hpa)
Cloud and Cost efficiency	infracost

Application description

Name: goCovid

Language: golang (go version go1.17.6 linux/amd64)

Database: MariaDB (Amazon RDS)

3-tire application (front-end, back-end, database)

back-end:

- retrieve a portion of data from API (see in your Variant) and store it in a database
- update data on demand
- update DB schema if needed on app's update

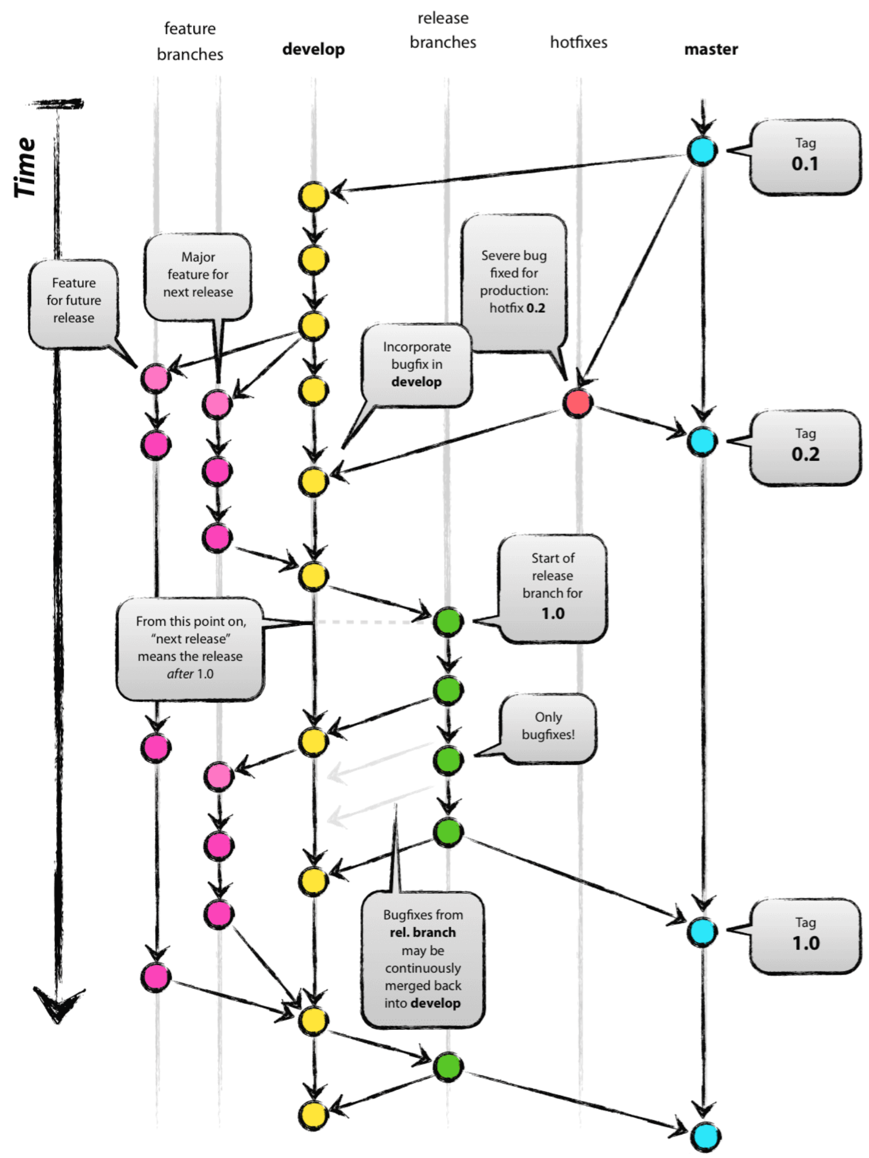
front-end:

- display any portion of the data stored in the DB
- provide a method to trigger data update process

Tools

- Git
- Make
- Docker
- Terraform
- Ansible
- Helm
- EKS
- Gitlab-ci
- Infracost
- Cloudflare
- AWS
- dbdiagram.io
- app.diagrams.net

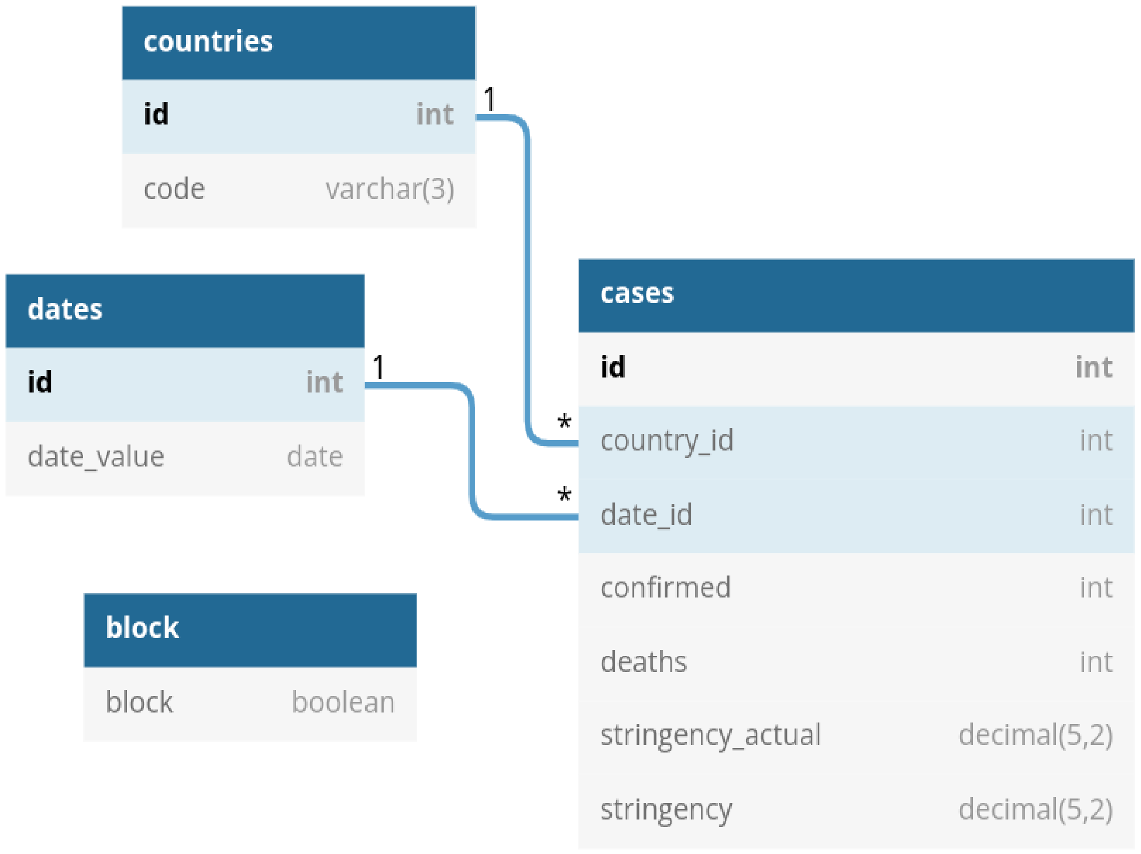
Source code management. Git. Branching strategy.



feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

- https://nvie.com/posts/a-successful-git-branching-model/
- **Master**
- **Develop**
- Feature branches
  - From develop
  - Merge to develop
- Release branches
  - From develop
  - Merge into master and develop
  - Name: release-*
- Hotfix branches
  - From master
  - Merge into master and develop
  - Name: hotfix-*

Database schema



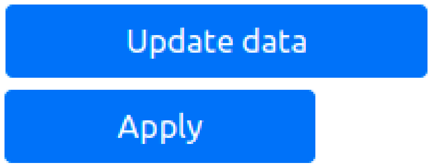
- https://dbdiagram.io/d/61a34baa8c901501c0d5ba2c

```
Table countries as C {
  id int [pk, increment] // auto increment
  code varchar(3)
}
Table cases {
  id int [pk, increment] // auto increment
  country_id int
  date_id int
  confirmed int
  deaths int
  stringency_actual decimal(5,2)
  stringency decimal(5,2)
}
Table dates as DS {
  id int [pk, increment] // auto increment
  date_value date
}
Table block {
  block boolean
}
Ref: cases.country_id > C.id
Ref: cases.date_id > DS.id
```

Backend (API)

```
/app # ls /app/configs/config.yml
/app/configs/config.yml
/app # ./api -help
Usage of ./api:
  -addr string
        API HTTP address (default "0.0.0.0:7000")
  -fill
        If it is true it fills db

/app # curl "http://localhost:7000/v1/help"
Hostname: 4abd5ea319b6

Examples:
curl -D - -s -X GET "http://localhost:7000/v1/health-check"
curl -D - -s -X GET "http://localhost:7000/v1/help"
curl -D - -s -X GET "http://localhost:7000/v1/state" (Block)
curl -D - -s -X GET "http://localhost:7000/v1/update" (Readiness)
curl -D - -s -X GET "http://localhost:7000/v1/refresh_data"
curl -D - -s -X GET "http://localhost:7000/v1/data?
countryCode=RUS&&dateFrom=2022-01-01&&dateTo=2022-01-
09&&sortBy=deaths"
```

```
SELECT
    countries.code
    ,dates.date_value
    ,cases.deaths
    ,cases.confirmed
  FROM
    cases INNER JOIN countries
      ON countries.id = cases.country_id INNER JOIN dates
      ON dates.id = cases.date_id
  WHERE
    countries.code = 'USA'
    AND date_value BETWEEN '2022-01-01' AND '2022-02-02'
  ORDER BY
    cases.deaths ASC
```



Frontend

- Generate html by golang
- Go routine to run task on backend
- JavaScript to run code on the client side
- SessionStorage
- Twitter Bootstrap

```
/app # ./web -help
Usage of ./web:
  -addr string
        HTTP address (default
"localhost:4000")
  -apiVers string
        API version (default "v1")
  -hostname string
        API hostname (default "localhost")
  -port string
        API port (default "8080")
```

Docker build with Makefile

```
$ cat Makefile.Docker
.DEFAULT_GOAL := all

# the same that .DEFAULT_GOAL for older version of make (<= 3.80)
# .PHONY: default
# default: clean;

.PHONY: all
all: build run

.PHONY: build
build:
$(info -> build:)
go mod tidy
mkdir -p ./bin
GOARCH=${GOARCH} GOOS=${GOOS} go build -o ./bin/${BINARY_NAME}-linux ${PATH_TO_SOURCE}

.PHONY: run
run:
$(info -> run:)
./bin/${BINARY_NAME}-linux

.PHONY: clean
clean:
$(info -> clean:)
go clean
rm ./bin/${BINARY_NAME}-linux

.PHONY: messages
messages:
$(info test info message)
$(warning test warning message)
$(error test error message)
```

Docker multi-stage build with Makefile

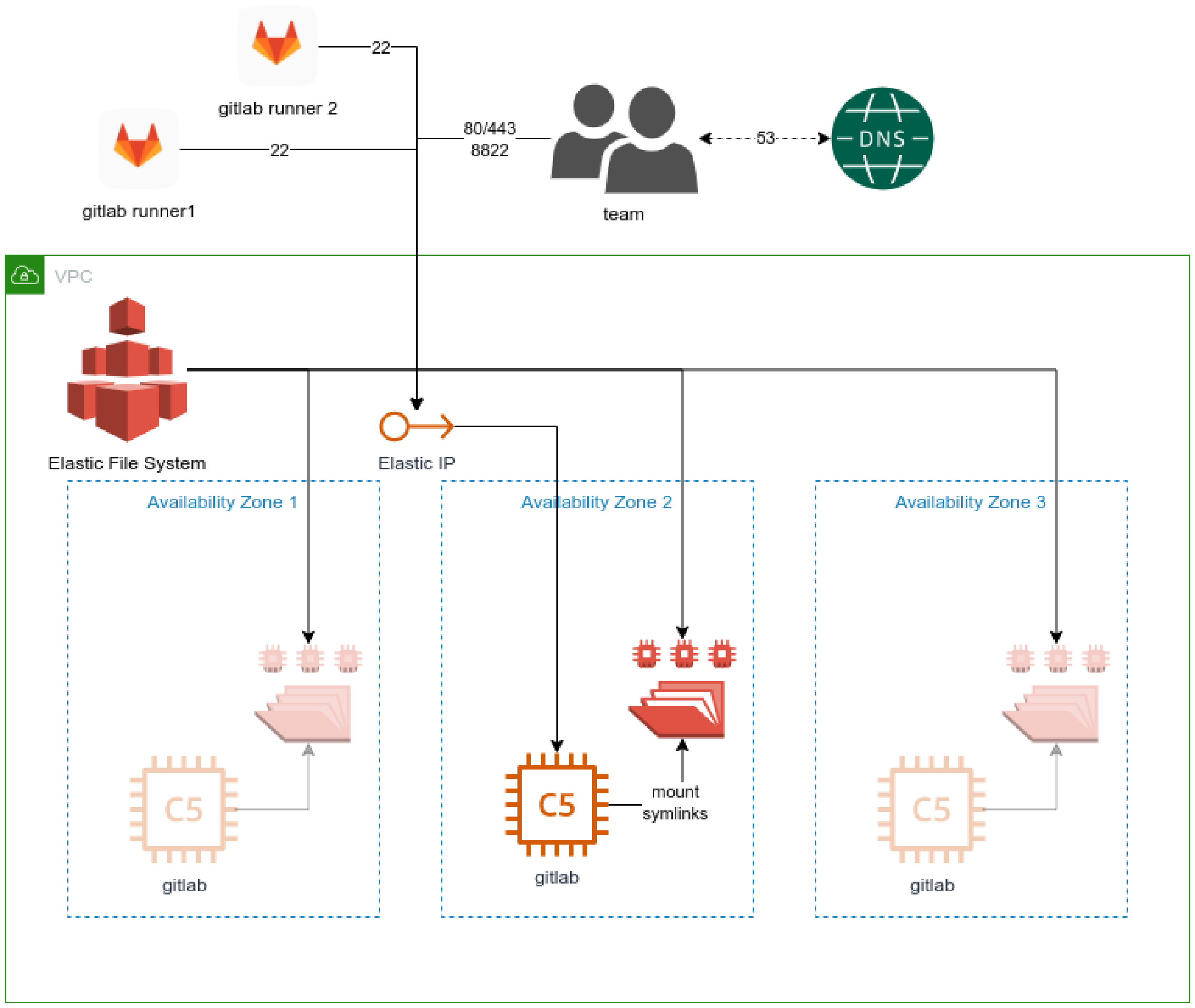
```
FROM golang:1.17.6-alpine3.15 AS build

WORKDIR /app

ARG CGO_ENABLED=0 \
  GOARCH=amd64 \
  GOOS=linux \
  BINARY_NAME=api \
  PATH_TO_SOURCE=./cmd/api


RUN apk --no-cache add make

COPY go.mod ./
COPY go.sum ./
RUN go mod download

COPY . /app/
RUN make -f Makefile.Docker build
```

```
FROM alpine:3.15

ENV DB_HOSTNAME=localhost
ENV MIGRATIONS_DIR=/app/migrations

RUN apk --no-cache add curl

COPY --from=build /app/bin/api-linux /app/api
COPY --from=build /app/migrations /app/migrations
COPY --from=build /app/configs /app/configs

WORKDIR /app

EXPOSE 5000

CMD ["/app/api"]
```

Docker compose

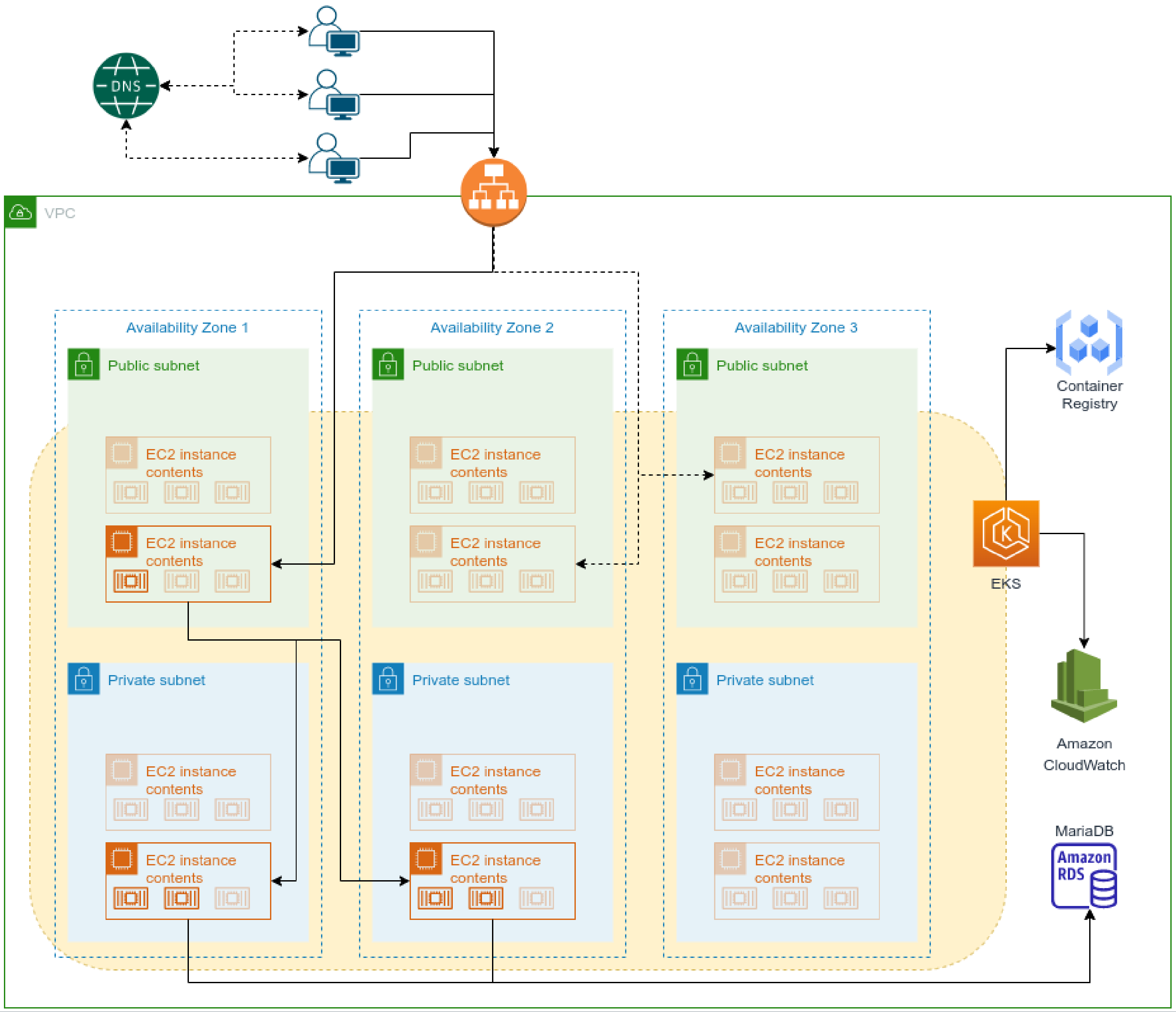
```
version: '3.1'
services:

 db:
  image: mariadb:10.7.1
  environment:
   - MARIADB_DATABASE=$MARIADB_DATABASE
   - MARIADB_USER=$MARIADB_USER
   - MARIADB_PASSWORD=$MARIADB_PASSWORD
   - MARIADB_RANDOM_ROOT_PASSWORD=yes
  ports:
   - 3306:3306
  healthcheck:
   test: ["CMD", "mysql" ,"$MARIADB_DATABASE", "-h", "localhost", "-P", "3306", "-u",
"$MARIADB_USER", "-p$MARIADB_PASSWORD", "-e", "SELECT 1"]
   interval: 3s
   timeout: 3s
   retries: 3
   start_period: 1s


 goose:
  depends_on:
   db:
    condition: service_healthy
  build: app/backend/migrations
  links:
   - db:mysql
  environment:
   - GOOSE_DRIVER=mysql
   -
GOOSE_DBSTRING=$MARIADB_USER:$MARIADB_PASSWORD@tcp(mysql:3306)/$MARIAD
B_DATABASE
  command: ["goose", "up"]

 api_data_init:
  depends_on:
   - goose
  build: app/backend
  links:
   - db:mysql
  command: ["/app/api", "-fill=true"]

 api:
  depends_on:
   db:
    condition: service_healthy
  build: app/backend
  ports:
   - 8080:7000
  links:
   - db:mysql
  environment:
   - APP_ENDPOINT=0.0.0.0:7000
  command: ["/app/api"]

 web:
  build: app/frontend
  ports:
   - 4000:4000
  links:
   - api:api
  command: ["/app/web", "-addr", "0.0.0.0:4000", "-port", "7000", "-
hostname", "api"]
```

Omnibus GitLab



- Terraform
  - cloudflare_record
  - template_file
  - EFS
  - Security groups
  - VPC
  - aws_instance
- Ansible
  - Nginx
  - Letsencript
  - Docker
  - EFS mount
  - -e "remove_gitlab_data=true"
- gitlab-ci
- Terraform state on gitlab
- Env variables (test and prod)
- Helm packages

Elastic Kubernetes Service



Terraform

- K8s
  - Cloudflare (with alb)
  - EKS Cluster
  - Policies
  - Amazon RDS (MariaDB)
  - Elastic container registry
  - Modules (openid_connect_provider)
  - etc
- Extra k8s
  - Autoscaller
  - Fluent-bit
  - Metrics-server
  - Container-insights
  - Alb controller
  - etc

Helm and manifests

```
$ ls helm/gocovid/
Chart.yaml  conf  templates  values.yaml
$ ls helm/gocovid/conf
config.yml ( 1 )
$ ls helm/gocovid/templates
api_deploy.yml  api_hpa.yml  api_secret_data_config.yml  api_service.yml
front_deploy.yml  front_hpa.yml  front_service.yml  goose_secrets.yml
_helpers.tpl  ingress.yml  job_fill.yml  namespace.yml
$ ls helm/gocovid/values.yaml
helm/gocovid/values.yaml
$ cat gocovid/templates/api_secret_data_config.yml
apiVersion: v1
kind: Secret
metadata:
  creationTimestamp: null
  name: {{ .Values.api.secretName }}
  namespace: {{ include "namespace" . }}
type: Opaque
data:
  config.yml: {{ include "apiConfig" . }} ( 4 )
```

```
$ cat helm/gocovid/conf/config.yml  ( 2 )
migration_dir: {{ .Values.api.migrationDir }}
app_endpoint: {{ .Values.api.endpoint }}
db_host: {{ .Values.dbSettings.endpoint }}
db_name: {{ .Values.dbSettings.name }}
db_port: {{ .Values.dbSettings.port }}
db_user: {{ .Values.dbSettings.user }}
db_pass: {{ .Values.dbSettings.password }}
data_reset: false

cat gocovid/templates/_helpers.tpl ( 3 )
...
{{- define "apiConfig" -}}
{{ tpl (.Files.Get "conf/config.yml") . | b64enc }}
{{- end }}
…
```

GitLab-ci

Infra

- Pipeline stages
  - init (cache)
  - validate (terraform validate, tflint)
  - build (artefacts: reports, environments)
  - Infracost (only test)
  - apply (test, prod, manual)
  - extra-init (cache)
  - extra-validate (terraform validate, tflint)
  - extra-build (artefacts: reports, environments)
  - extra-apply (test, prod, manual)
  - extra-destroy (PHASE == "destroy")
  - destroy (PHASE == "destroy")
  - Env vars
- Terraform state
- Dependencies
- Needs

Application

- Pipeline stages
  - lint (base)
  - test
  - Sonarqube (only master)
  - build (base, trivy check)
  - helm build (base, environments)
  - deploy (base) (test, prod)

  base build (extends):
  - only branches
  - only protected tags
- Env vars
- Push docker images
- Push helm chart
- Needs to increase

Logs and Monitoring by CloudWatch

Log queries:

```
fields @timestamp, @message, kubernetes.namespace_name,
kubernetes.pod_name
| sort @timestamp desc
| filter kubernetes.pod_name not like /^fluent.*.cz268$/
| limit 9999

fields @timestamp, @message, kubernetes.namespace_name,
kubernetes.pod_name
| sort @timestamp desc
| filter kubernetes.namespace_name="gocovid"
| filter kubernetes.pod_name like /^deploy-gocovid.*/
| limit 9999

fields @timestamp, @message, kubernetes.namespace_name,
kubernetes.pod_name
| sort @timestamp desc
| filter kubernetes.pod_name not like /^front.*/
| limit 9999
```

Monitoring:

- Alarms and SNS
- Build dashboards
- All LoadBalancer Metrics
- All EKS metrics
- All RDS Metrics
- Tags, env tags
- Prometheus
- Grafana

Possible project improvements

- two-factor authentication to gitlab or vpn
- change EFS with S3 bucket on gitlab
- add redis for cache
- generate full sql-query to decrease connection to database
- return errors in main.go (process them in main.go)
- Add .dockerignore file to the context directory
- Describe some tests in Makefile
- Increase tests coverage

Thank you for your attention