

# Kontinuasjoneksamensoppgave

## PGR301 - tilpasset hjemmeksamen 72 timer.

---

### Fra Læreplan

---

Egenutviklet applikasjon (eller prototype) med tilhørende dokumentasjon: teller 100% av karakteren i emnet. Applikasjonen skal være utviklet, og være vedlikeholdbar, i et DevOps-miljø i skyen. Kildekode, og annen dokumentasjon, skal gjøres tilgjengelig for allmenheten."

Siden et kontinuerlig kjørende DevOps-miljø kan være kostnadsbærende for studentene vil vi lette på kravet om miljøet skal kjøre kontinuerlig i skyen. Applikasjonen må istedet være mulig å etablere i skyen ved hjelp av infrastruktur som kode (Terraform) - og enkel automatisering.

På den måten kan de som måtte ønske å ta løsningen i bruk ( for eksempel eksaminator ) bruke egen infrastruktur (og da ta kostnaden for drift av miljøet)

### Leveransekrav

---

- Applikasjonen og tilhørende DevOps infrastruktur skal gjøres tilgjengelig i offentlige GitHub repositories.
- Leveransen må være anonym
- Du skal levere et dokument (tekstfil) i wiseflow som lenker til relevante repositories i GitHub
- Infra-repository skal inneholde en `credentials_example.yml` som eksemplifiserer nødvendige hemmeligheter for pipeline.
- Det skal lages ett infrastruktur repository og ett kode-repository for applikasjonen.

Følgende to repositories *kan* brukes som startpunkt. Dette er ikke et krav.

- <https://github.com/PGR301-2018/exam-infra>
- <https://github.com/PGR301-2018/exam-app>

### Krav til applikasjonen

---

Applikasjonen må være innholdsrik nok demonstrere DevOps prinsipper og bevise ferdigheter

hos studenten. Det anbefales ikke å kun gjøre enkle modifikasjoner på eksempel-appen, men finne et eksempel fra hobby/interesse/ eller et annet prosjekt fra annet fag.

- Applikasjonen skal bestå av både av et API og en database. Minimalt et REST API med CRUD kapabilitet.
- Databasen kan være "in Memory" (h2 osv)
- Applikasjonen skal bygge med Maven
- Applikasjonen skal ha enhetstester

## Krav til build pipeline

---

Det skal lages en CI/CDpipeline for applikasjonen.

- Pipeline skal implementeres med Concourse
- Det skal være en concourse jobb som heter "infra" som oppretter nødvendig infrastruktur i skytjenesten Heroku ved hjelp av terraform-kode.
- Pipeline skal kontinuerlig deploye hver commit på master branch i applikasjons-repository til CI-miljøet
- Deployment fra CI-miljø videre til Stage og produksjon skal i utgangspunktet skje manuelt ved at man promoterer applikasjonen i Heroku web-grensesnittet

## Evaluering

---

Eksaminator gjør følgende når han får oppgaven ...

- Lager forks av de inleverte repositories. Lager deploy keys
- Endrer pipeline.yml, og setter inn sine repositories under "source"
- Døper om filen credentials\_example.yml til credentials.yml og legger inn sine egne hemmeligheter
- Endrer variables.tf eller andre filer basert på instruksjer i README filen.
- Sletter eventuelle .terraform katalog og terraform.tfstate fil
- Kjører `fly -t (min target) set-pipeline -c <infra repo>/concourse/pipeline.yml -l <infra repo>/credentials.yml -p student_name` i sitt eget Concourse-miljø.
- Kjører "infra" jobben i Concourse
- Comitter kode på master branch, og venter på at bygget skal starte av seg selv.
- Inspisere heroku, applikasjonens endepunkter og funksjonalitet.
- Leser README.

Karakter settes basert på oppnådde poeng mot mulig oppnådde poeng.

# Oppgaver

---

## Kom i gang

---

For å komme i gang kan dere følge omtrent samme prosess som vi gjorde på øvingene

- Pass på at concourse kjører- Gå til localhost:8080
- Lag en fork av repositories
- Endre /terraform/variables.tf - velg deg unike verdier for prefix, og pipeline navn
- Endre credentials.yaml for å matche applikasjonsnavn i variables.tf
- Lag deploy keys for repositoryene, installer disse, og ta vare på privat key.
- Modifiser Credentials.yml og legg inn dine hemmeligheter (blant annet deploy keys, som dere må lage)
- Eksekver `fly ....` for å laste opp pipeline.yml til concourse
- Du er nå klar for å begynne på eksamen.

Følgende oppgaver skal løses og ligger til grunn for evaluering.

## Del 1 Applikasjon og basis pipeline (20 poeng)

---

- Innleveringen skal tilfredstille krav nevnt over under "Leveransekrav".

## Del 2 Metrics (20 poeng)

---

Applikasjonen skal logge egendefinerte metrics. Med det menes at man på valgte steder i koden skal logge datapunkter. Evalueringen blir gjort basert på om Metrics er korrekt implementert på

en fornuftig måte. I evalueringen vektlegges derfor at Applikasjonen er funksjonsrik nok til å demonstrere alle nevnte typer metrics;

- Meter
- Gauge
- Counter
- Histogram
- Timer

Til dette skal applikasjonen bruke [Metrics biblioteket fra Dropwizard](#) å integrere dette i Spring boot applikasjonen ;

Metrics skal ikke leveres til en skytjeneste eller liknende. Metrics skal i stedet sendes til stdout/console ved å inkludere følgende Spring config bean i prosjektet se (<https://metrics.dropwizard.io/3.1.0/getting-started/>)

```
@Configuration
public class ReporterConfig {

    @Bean
    public MetricRegistry getRegistry() {
        return new MetricRegistry();
    }

    @Bean
    public Reporter getReporter(MetricRegistry registry) {
        ConsoleReporter reporter = ConsoleReporter.forRegistry(getRegistry())
            .convertRatesTo(TimeUnit.SECONDS)
            .convertDurationsTo(TimeUnit.MILLISECONDS)
            .build();
        reporter.start(15, TimeUnit.SECONDS);
        return reporter;
    }
}
```

## Del 3 - Docker (10 poeng)

---

- Det skal skrives en Dockerfil bygger et Container Image av Spring Boot applikasjonen.
- Eksaminator vil teste applikasjonen ved å gjøre en `docker build ...` og `docker run...`

Lykke til !