# Fra Læreplan

*"Egenutviklet applikasjon 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 utløse kostnader 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

# Krav til leveransen

- Besvarelsen skal bestå av en tekstfil med lenke til to repositories. Ett repo for applikasjon, og et for infrastruktur.
- Annonymitet er ikke påkrevet

# Krav til applikasjonen

Selve applikasjonen er ikke fokus for eksamen, *men* den må samtidig være innholdsrik nok til å demonstrere gode DevOps-prinisipper. Dere kan gjerne bruke en eksisterende applikasjon laget i et annet fag, eller et hobbyprosjekt.

- Applikasjonen skal eksponere et REST API og ha en database, gjerne "in memory" for eksempel H2
- Applikasjonen skal bygge med Maven eller Gradle
- Applikasjonen kan skrives i Java eller Kotlin
- Applikasjonen skal ha tester for REST APIet. For eksempel ved hjelp av *RestAssured* Dersom noen av testene feiler, skal bygget også feile

Applikasjonen skal være skrevet på en slik måte at drift og vedlikehold er enkelt og i henhold til prinsipper i the "twelve factor app"

- To prinsipper gjelder for eksamen *

- III Config. Ingen hemmeligheter eller konfigurasjon i applikasjonen (ingen config filer med passord/brukere/URLer osv). Vi har lært teknikker i faget for å eksternalisere konfigurasjon. Pass godt på å ikke sjekke inn API nøkler osv.

- XI Logs. Applikasjonen skal bruke et rammeverk for logging, og logge til standard-out, ikke til filer. I praksis vil dette si bruk av Logback eller Log4j via sl4j i Spring Boot, med en "Console appender". Ikke bruk System.out.println();

Brudd på overnevnte prinsipper vil gi trekk i poeng i den oppgaven der bruddet skjer.

# Oppgave 1 - Docker

Repository skal inneholde en .travis.yml fil som gjør travis i stand til å lage et docker image for hver commit som gjøres til Master branch. Dette container imaget skal deretter lastes opp til Google Container Registry. Dere skal demonstrere bruk av Docker multi-stage bygg, slik at også kompilering og pakking av applikasjonen skjer i en container.

Det skal finnes en readme som beskriver hvordan sensor setter nødvendige hemmeligheter i Travis.

# Oppgave 2 - Metrikker

Applikasjonen skal registrere egendefinerte metrics. Med det menes at man, ved hjelp av rammeverket Micrometer, skal skrive kode som lager egendefinerte metrikker.
I evalueringen legges det vekt på at applikasjonen er innholdsrik nok til å demonstrere forståelse av - og bruk av minst følgende type metrics gauge, counter, DistributionSummary, Timer, LongTaskTimer.

Applikasjonen skal være konfigurert for levering av Metrics mot InfluxDB. Dere kan anta at influxDB kjører lokalt i en egen container på eksaminator sin maskin når oppgaven rettes.

# Oppgave 3 - Logger

Denne oppgaven består av å bruke en SAAS tjeneste, Logz.io for innsamling, visualisering og analyse av logger. Dere skal utvide applikasjonen på en slik måte at logger sendes til denne tjenesten. Applikasjonen skal være konfigurert på en slik måte at den både logger lokalt til *stdout*, og leverer logger til Logz.io.

# Oppgave 4 - Infrastruktur

Det skal lages et eget repository for Terraformkode. Det skal lages en Travis pipeline som kan kjøre Terraform, og deploye infrastruktur til for eksempel (men ikke bare) Google Cloud Platform. Følgende infrastruktur skal opprettes;

- En Google Cloud Run applikasjon som skal gjøres tilgjengelig på Internet. Cloud run applikasjkonen skal kjøre container image som allerede finnes i Container Registry fra oppgave #1

## Oppgave 5 og Overvåkning og varsling

I denne oppgaven skal du implementere overvåkning, og varsling i applikasjonen ved hjelp av SAAS tjenesten StatusCake. Infrastrukturen skal opprettes ved hjelp av Terraform. https://www.terraform.io/docs/providers/statuscake/index.html

## Oppgave 6 Valgfri IAC (infrastructure as code)

Velg en provider fra Terraform https://www.terraform.io/docs/providers/index.html - Bruk fantasien og Implementer valgfri infrastruktur som dere tar i bruk i applikasjonen.