# System Arkitektur Dokument (SAD)

# Spørris

# Prosjekt i DAT104, 2016

# Teknisk Arkitektur

Prosjektet skal være et dynamisk web prosjekt. Prosjektet skal lages i Eclipse og deles mellom medlemmene med Git med repository <https://github.com/2food/DAT104_Prosjekt> hvor vi også deler resten av filene vi bruker i prosjektet. Prosjektet skal også oppdateres på SVN etter hver iterasjon, repository <http://data1.hib.no:8000/svn/h16_dat104_p>?? Som databasespråk bruker vi PostgreSQL og som RDBMS bruker vi pgAdmin og databasen skal liggepå data1. Til kommunikasjon bruker vi Slack <https://sporris.slack.com/messages/general/>

# Applikasjonsarkitektur

## Pakkestruktur

Alle pakkene i prosjektet skal begynne med no.hib.dat104.xxxx

Vi skal ha følgende pakker:

* no.hib.dat104.controller
* no.hib.dat104.model
* no.hib.dat104.javabeans
* no.hib.dat104.validators
* no.hib.dat104.test

## Dokumentasjon/javadoc

Vi er mange som skal jobbe på dette prosjektet og da er det ekstra viktig og dokumentere koden slik at det blir enklere å lese og forstå koden for alle. Det skal skrives javadoc til alle klasser og metoder.

Alle metodene skal ha en kort beskrivelse av hva koden gjør, kort forklaring av parametere og returneringsverdi dersom det er aktuelt. For returneringsverdi og parameterer skal det brukes annotasjonene @return og @param. Klassene skal ha en kort beskrivelse av klassens innehold og oppgaver, @author skal også brukes for å si hvem som har opprettet klassen.

## Testing

Alle metoder skal ha automatiske enhetstester. Vi skal bruke JUnit rammeverket til enhetstesting. Testene bør skrives før metodene implementeres. Dette gjør at vi for testdrevet utvikling med stor fokus på fungerende kode fri for lus.

## URL’er

Alle URL’er skal defineres i URLmappings

## Navn

Alle navn på klasser, metoder og variabler skal være på engelsk og servlets skal ha navn på følgende måte navnServlet, f.eks. loginServlet for login

## MVC2

Applikasjonen skal bruke MVC2 arkitektur. MVC innebærer at vi har en 3-delt arkitektur, en kontroller, et view og en modell.

## Kodestandard

Vi følger vanlig javakodestandard. Eclipse formaterer koden etter denne standarden med ctrl + shift + f

## Klasser

En klasse skal ikke inneholde for mange metoder. Vi ønsker og dele opp mest mulig for å gjøre det lettere å gjenbruke og vedlikeholde.

## UML’er

skal lages i UMLet og lagres som .jpg og .uxf på github under mappen Diagrammer