

# Systemdokumentasjon

Elektronisk valgadministrasjon - EVA Admin



2025

## Innhold

- Introduksjon
  - Overordnet systembeskrivelse - EVA
    - EVA - Bruksområder
    - Kontekstdiagram
    - Verdikjedekontekst
- Overordnet systembeskrivelse - EVA Admin
  - EVA Admin
  - Bruksområder
  - Kontekstdiagram
  - Verdikjedekontekst
  - Sikkerhetsstrategi
    - Kontekst for sikkerhetsstrategi
    - Sikkerhet gjennom lagdeling
  - Policies og prosedyrer
    - Policy
    - Prosedyre
    - Valgdirektoratets policies
  - Kommunikasjon over internett
    - Kryptering av kommunikasjon over internett
    - Sertifikater / PKI
    - Klientsertifikater
- Funksjonelle moduler
  - Konfigurasjonsmodul
    - Formål
    - Modulbeskrivelse
  - Stemmegivningsmodul
    - Formål
    - Modulbeskrivelse
  - Opptellingsmodul
    - Formål
    - Modulbeskrivelse
  - Valgoppgjørsmodul
    - Formål
    - Modulbeskrivelse
    - Kandidatkåring og mandatfordeling
    - Fordeling av utjevningsmandater
  - Rapportmodul
    - Formål
    - Modulbeskrivelse
- Valgdomenet
  - Domenemodell - geografi og valghierarki
  - Domenemodell - listeforslag
  - Domenemodell - stemmegivning
  - Domenemodell - opptelling
  - Domenemodell - valgoppgjør - utjevningsmandater
  - Domenemodell - valgoppgjør - kandidatmåring
- Arkitektur
  - Innledning
  - EVA Admins utvikling
  - Modularisering
  - Konfigurasjonsstyrt forretningsmodell
    - Sentral konfigurasjon
    - Lokal konfigurasjon
  - Applikasjonsarkitektur
  - Skalering
  - Sikring av infrastruktur
    - Tilgangsbegrensning
    - Redundans
  - Lagdeling - EVA Admin
    - Fysisk tilgang
    - Logisk tilgang
  - Moduler i EVA Admin
  - Frontend - presentasjonsmodul
    - Javaklasser
    - JSF Ajax (Java Server Faces)
    - Primefaces
    - Applikasjonsserver

- Sikkerhets- og sesjonshåndtering
- Autentisering - Id-porten
- Backend - forretningsprosessmodul
- Backendprosesser
  - Javaklasser
  - Context og Dependency injection - CDI
  - Java Persistence API - Hibernate
  - Applikasjonsserver
  - Autorisering - Rollebasert tilgangskontroll
  - Logging
- Kommunikasjon frontend - backend
  - Javaklasser
- Database
  - Databaseskjema
- Rapportering
  - Formål
  - Modulbeskrivelse
- Kodeorganisering - prinsipper og retningslinjer
  - Domenedrevet design (DDD)
  - Kodeeksempel
- Koden som understøtter det nye skjermbildet for å administrere brukere
  - Kildekodefil: operatorAdmin.xhtml
  - Kildekodefil: OperatorAdminController.java
  - Eksempel på applikasjonstjeneste
  - Eksempel på entitet med domenelogikk
  - Eksempel på domenetjeneste
  - Eksempel på repository klasse
- Integrasjoner
  - Mottak av telleresultater fra EVA Skanning
    - Mottakstjeneste - telleresultater
    - Sikkerhet ved overføring
  - Id-porten - brukerautentisering
  - Rapportering av opptellingsresultat til EVA Resultat
  - Rapporter - Jasper Server
  - Folkeregisteret - manntall
    - Innledning
    - Beskrivelse
  - Digitaliseringsdirektoratets tjenester for elektronisk distribusjon
    - Innledning
    - Valgkortmodul
    - Beskrivelse
  - Kartverket - kartdata
- Vedlegg
  - Systemkrav

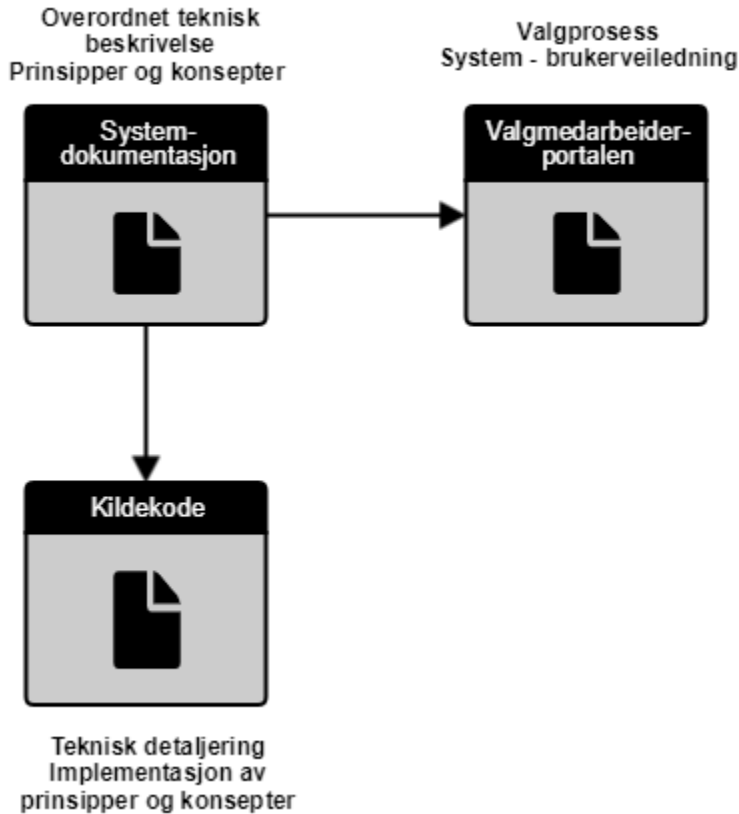
## Introduksjon

Systemdokumentasjonen gir en overordnet beskrivelse av EVA -systemet fra et funksjonelt og teknisk perspektiv.

Dokumentet gir beskrivelser på et konseptuelt nivå, og er ikke en uttømmende teknisk beskrivelse av EVA. Konsepter og prinsipper gir i sin tur bakgrunn til kildekode og konfigurasjon som er hoveddokumentasjonen for systemet på et detaljert teknisk nivå.

Inngående beskrivelser av selve valgprosessen er gitt på valgmedarbeiderportalen, med brukerveiledning, prosess- og rutinebeskrivelser, samt skjematiske framstillinger av prosess.

*Skisse som illustrerer hvordan dokumentasjonen er organisert:*



## Overordnet systembeskrivelse - EVA

EVA - (Elektronisk valgadministrasjon) er Valgdirektoratets IT-støttesystemportefølje som kommuner og fylkeskommuner benytter seg av for å forenkle valg gjennomføringen. EVA er ikke et saksbehandlingssystem og kan ikke erstatte valgstyrene sitt ansvar for å påse at valghendelsene blir gjennomført etter gjeldende regelverk, men fungerer som et støtteverktøy for kommuner og fylkeskommuner i de ulike fasene av valg gjennomføring. EVA har funksjonalitet som støtter opp om gjennomføringen av

- Stortingsvalg
- Fylkestingsvalg
- Kommunestyrevalg
- Direkte valg til kommunedelsutvalg
- Sametingsvalg
- Lokalvalg til Longyearbyen

Systemporteføljen brukes også av Valgdirektoratet til oppfølging av valg gjennomføring, samt til rapportering til 3. parter.

## EVA - Bruksområder

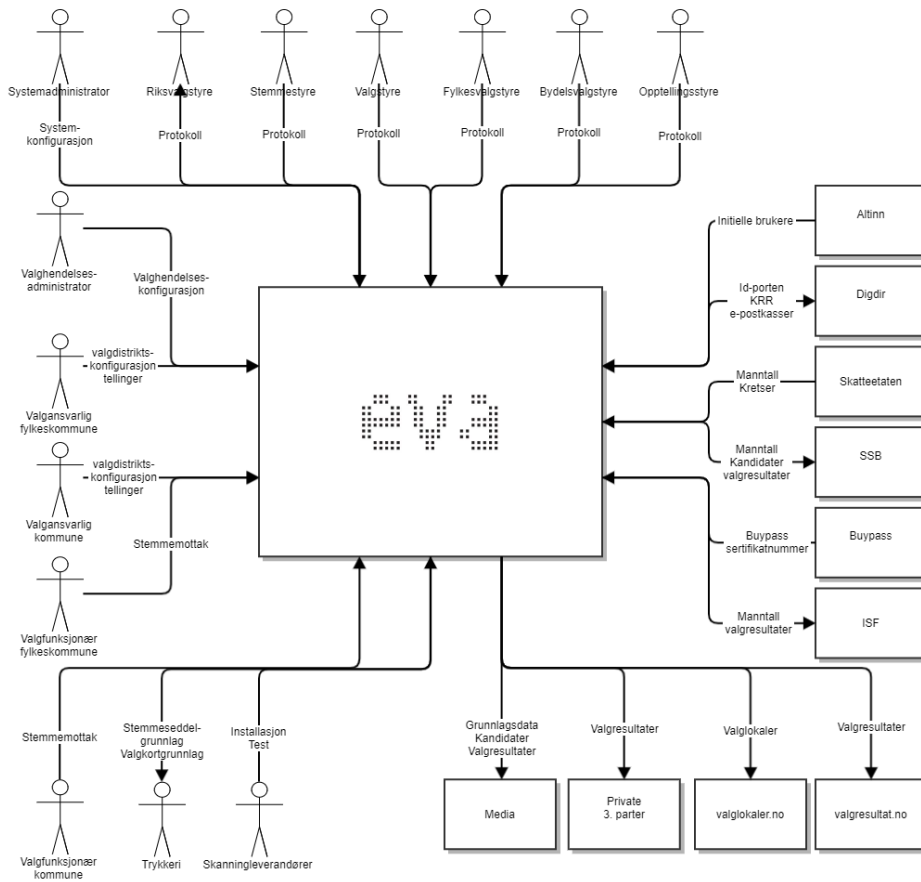
Valg utviklingen foregår over 4 faser og EVA porteføljen støtter alle disse fasene:

- Forberedelsesfasen
  - EVA Admin
- Stemmegivningsfasen
  - EVA Admin
- Opptellingsfasen
  - EVA Admin
  - EVA Skanning
  - EVA Resultat
- Valgoppgjør fasen
  - EVA Admin
  - EVA Resultat

Som det framgår av oversikten over brukes EVA Admin i alle faser, mens andre deler av porteføljen kun brukes for utvalgte faser.

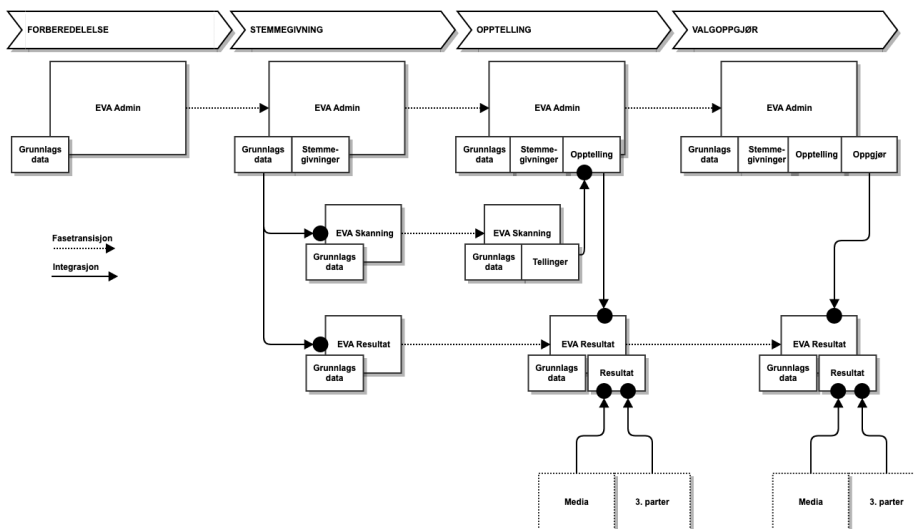
## Kontekstdiagram

Diagrammet under beskriver de aktører og systemer / etater som har interaksjon og/eller interesser i/for EVA porteføljen.



## Verdikjedekontekst

Diagrammet beskriver EVA systeminteraksjon i valg gjennomføringsverdikjeden, kun hovedmoduler og aktører er inkludert i diagrammet.



Dette dokumentet omhandler kun EVA Admin, de andre modulene beskrives i separate dokumenter spesifikke for den enkelte modul.

# Overordnet systembeskrivelse - EVA Admin

## EVA Admin

EVA Admin er hovedsystemet i Valgdirektoratets EVA-portefølje og gir IT-støtte til valg gjennomføringen fra valget forberedes til det er gjennomført. Valgdirektoratet oppretter valg i EVA Admin basert på lovpålagte krav, før kommuner og fylkeskommuner legger inn informasjon om hvordan valgstyret ønsker å gjennomføre valget innenfor sitt ansvarsområde.

EVA Admin inneholder informasjon om manntallsførte velgere og benyttes til å registrere stemmegivninger på velger. Stemmegivningene vurderes manuelt, men forkastes eller godkjennes i EVA, eventuelt mot et papirmanntall for de kommunene som registrerer stemmegivninger i et papirmanntall på valgdagen(e).

Når kommunene er i fasen for å telle og partifordelte stemmesedlene, legges tall og rettelser gjort på stemmeseddelen inn i EVA Admin før de blir godkjent og rapportert. Når alle stemmesedlene er talt opp i sitt valgdistrikt, kan det gjennomføres et valgoppgjør med mandatfordeling og kandidatføring.

Underveis og avslutningsvis er det mulig å hente ut ulike rapporter fra EVA Admin.

## Bruksområder

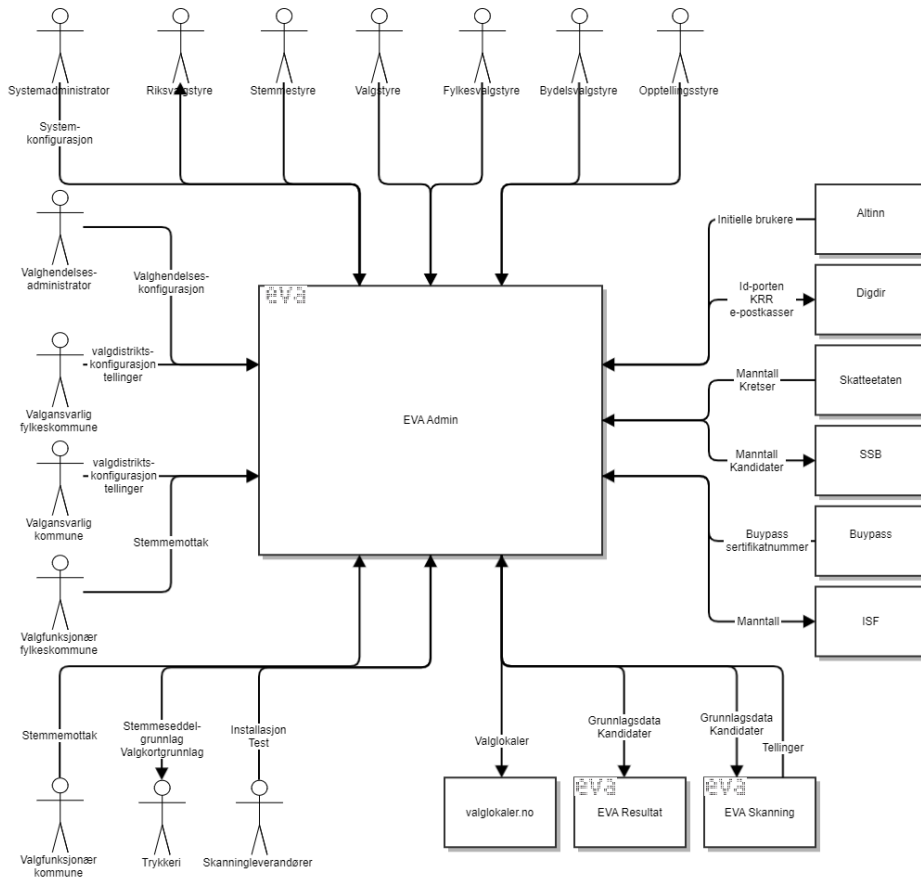
Valggjennomføringen foregår over 4 faser og EVA Admin tilbyr IT-systemstøtte for alle disse fasene for gitte funksjonsområder:

- Forberedelsesfasen
  - Grunnlagsdata - målform, opptellingsadministrative forhold, manntallsform, forhånds- og valgtingsstemmesteder, samt valgkortinformasjon relatert til disse
  - Listeforslag - administrasjon av listeforslag, underskrifter og stemmeseddelgrunnlag
  - Manntall - søk og behandling av søknader om innføring
- Stemmegivningsfasen
  - Stemmegivninger i tidlig- og forhåndsstemmeperioden
  - Stemmegivninger på valgting
- Opptellingsfasen
  - Manuell opptelling av
    - Forhåndsstemmer
    - Valgtingsstemmer
  - Mottak av maskinelle telleresultater for
    - Forhåndsstemmer
    - Valgtingsstemmer
  - Protokollføring av valggjennomføring for aktuell valggjennomføringsinstans
    - Stemmestyre
    - Valgstyre
    - Fylkesvalgstyre
    - Bydelsutvalg
    - Opptellingsvalgstyret for Sametingsvalget
    - Valg til lokalstyret i Longyearbyen
- Valgoppgjørfasen
  - Endelig valgoppgjør
  - Beregning av mandatfordeling
  - Beregning av kandidatføringer

For alle faser tilbyr EVA Admin relevant rapportering til sluttbrukere og valgmyndigheter.

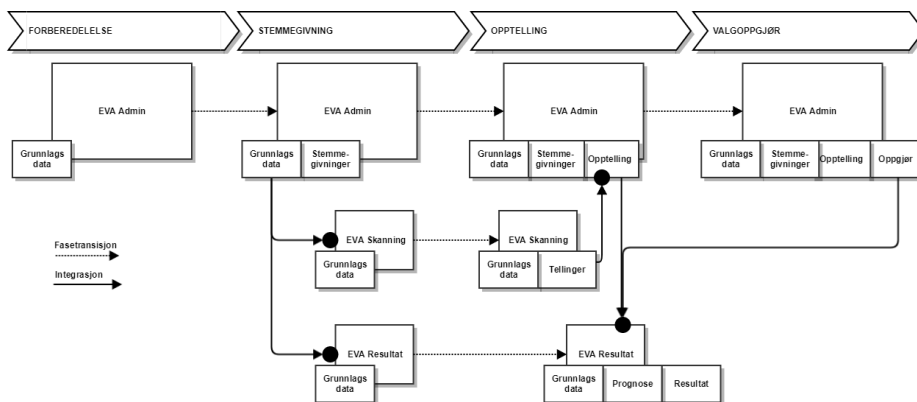
## Kontekstdiagram

*Kontekstdiagrammet beskriver de aktører og systemer som bruker EVA Admin.*



## Verdikjedekontekst

Diagrammet beskriver EVA systeminteraksjon i valg gjennomførings verdikjeden med utgangspunkt i EVA Admin



## Sikkerhetsstrategi

### Kontekst for sikkerhetsstrategi

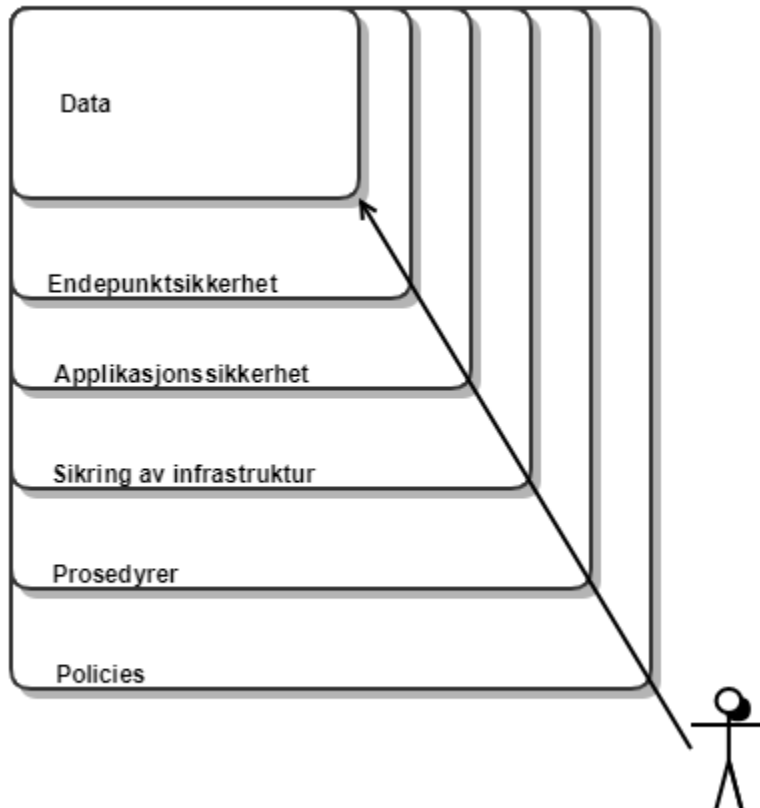
Sikkerhetsstrategien for EVA porteføljen inngår som en del av Valgdirektoratets policies for informasjonssikkerhet, sikkerhet og styring og kontroll.

### Sikkerhet gjennom lagdeling

Applikasjonene sikres gjennom lagdeling som konseptuelt kan deles inn som følger:

- Policies
- Prosedyrer
- Sikring av infrastruktur
- Applikasjonsikkerhet
- Endepunktsikkerhet

Skisse som illustrerer lagdelingen som er beskrevet over



## Policies og prosedyrer

Valgdirektoratet har utarbeidet policies og prosedyrer for sikkerhet.

### Policy

En policy beskriver overordnede føringer og prinsipper for etablering, oppfølging og forbedring av et funksjonsområde.

### Prosedyre

En prosedyre beskriver konkret framgangsmåter for å etterkomme en eller flere policies.

### Valgdirektoratets policies

Valgdirektoratet har utarbeidet følgende policies med tilhørende prosedyrer:

- "Policy for informasjonssikkerhet"
- "Policy for sikkerhet"
- "Policy for styring og kontroll"

i tillegg til et sett med prosedyrer og retningslinjer, samt implementasjoner som oppfyller prinsipper og retningslinjer gitt i policies.

## Kommunikasjon over internett

### Kryptering av kommunikasjon over internett



Samtlige av Valgdirektoratets EVA produkter kommuniserer over internett, enten som avsender av informasjon eller som mottaker.

EVA produktene kommuniserer utelukkende på HTTPS-protokollen (Hypertekst Transfer Protocol Secure) der kommunikasjonen skjer over åpne nettverk (internett).

For en nærmere beskrivelse av HTTPS, se [Store norske leksikon](#) sin beskrivelse av protokollen, det gis ingen utdypende forklaring av protokollen i systemdokumentasjonen - annet enn at protokollen brukes.

## Sertifikater / PKI

Valgdirektoratet har anskaffet nødvendige sertifikater fra *anerkjente sertifikatutstedere* (CAs) for å understøtte sikker kommunikasjon over internett og ivareta behovet for:

- Autentisering
- Kryptering (asymmetrisk)

For en nærmere beskrivelse av sertifikater/PKI se [Store norske leksikon](#) sin beskrivelse av PKI.

## Klientsertifikater

Valgdirektoratet utsteder klientsertifikater som alle som skal kommunisere med EVA produktene må installere, dette er en ekstra sikkerhetsmekanisme for å begrense tilgangen til EVA produktene samt tilleggs-autentisere alle brukere av systemene. Dette påvirker ikke mekanismene beskrevet over.

## Funksjonelle moduler

Funksjonelle moduler beskriver modulene som brukerne interagerer med og som manifesterer seg som skjermbilder eller rapporter. EVA Admin er delt inn i 5 moduler:

- Konfigurasjon - grunnlagsdata
- Stemmegivning
- Opptelling
- Valgoppgjør
- Rapportering

Utover beskrivelsene som er gitt under henvises det til <http://valgmedarbeiderportalen.valg.no>, brukerveiledning EVA for nærmere beskrivelser av funksjonalitet i EVA Admin.

## Konfigurasjonsmodul

### Formål

Konfigurasjonsmodulen understøtter den første delen av valgprosessen som består i å konfigurere valget. De første stegene utføres sentralt (dvs. av direktoratet) og innebærer å opprette valghendelsen og andre sentrale grunnlagsdata. Deretter får kommunene tilgang til systemet for å legge inn underskrifter, lister og listekandidater. I tillegg legger kommunene inn konfigurasjon rundt hvordan opptellingen skal foregå, sentralt eller kretsvis, og informasjon om målform og adressen til valgstyret. Konfigurasjonen skal være ferdigstilt rundt 1. juni i valgåret.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for konfigurasjon bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i konfigurasjonsmodulen.

#### Geografi og valghierarki

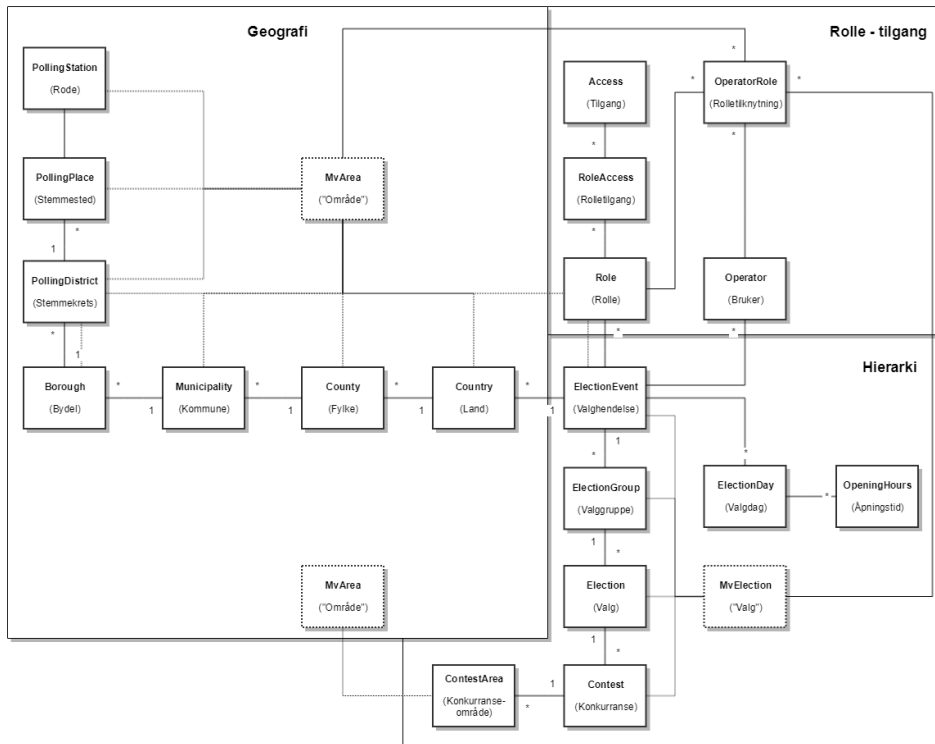
To helt sentrale konsepter som funksjonaliteten i EVA Admin bygger på, er *områdehierarkiet* eller "*Geografi*" og *valghierarkiet*.

All valgkonfigurasjon og alle valgprosesser i EVA Admin utføres for:

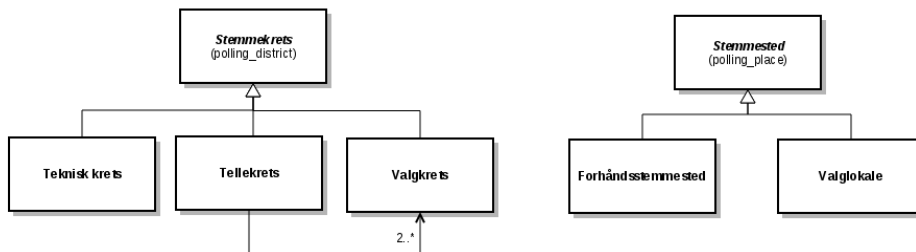
- Et gitt nivå i valghierarkiet
- Et gitt nivå i områdehierarkiet, dvs. for et gitt geografisk område.

Områdehierarkiet vises i venstre del av diagrammet under. Her settes Norges geografi opp, fra alle fylker ned til stemmesteder. På kodenivå er det introdusert en områdeabstraksjon, kalt `mv_area`, som mye av områdeinteraksjonen skjer gjennom. Abstraksjonen er implementert på databasenivå ved hjelp av et "materialized view" som samler alle områdenivåer i en tabell.

Valghierarkiet vises i høyre del av diagrammet under. Her konfigureres valghendelse, valggruppe, valg og valgdistrikt. På kodenivå er det introdusert en valgabstraksjon, kalt `mv_election`, som mye av valginteraksjon skjer gjennom som er implementert på tilsvarende måte som `mv_area`.



Stemmekrets og stemmested i områdehierarkiet er abstraherte konsepter, og kan ytterligere konkretiseres:



## Valghendelse

Rotentiteten i valghierarkiet:

- Setter opp valgdager og datoer for valget
- Er knyttet til geografikonfigurasjon
- Er knyttet til bruker-tabell, slik at settet med brukere er angitt per valghendelse. (Her blir som oftest personopplysningene hentet fra velger-tabellen.)
- Stil (storting, sameting, test)

## Valgruppe

Her settes optellingmåter opp for valget.

- Er knyttet til stemmegivninger (Voting)
- Er knyttet til sentralt definerte optellingskategorier som angir hvilke optellingskategorier kommunene kan velge
- Flagg for papirstemmegivning og elektronisk stemmegivning
- Start og sluttdato for ulike stemmegivningsperioder (forhånd utenriks, forhånd innenriks, etc)
- Skal knyttes til manntalls- og velger-tabell

Merknad: Den (mest) praktiske funksjonen for valggruppe er at man kan bruke ett manntall for flere valg, for eksempel kommunestyre- og fylkestingsvalg.

## Valg

Her settes regler opp for valget, som om det er tillatt med personstemmer, slengere og strykninger. Hvis det er sametingsvalg, angis dette her (ved bruk av flagg "single area" og "penultimate recount").

- Områdenivå for valget (kommune eller fylke)
- "Single area"-flagg (vanlige valg er "single area", sametingsvalg er det ikke)
- Renummerering, strykning, tilføyelse (writein)
- Aldersgrense
- Flagg for om kandidater må ha stemmerett
- Diverse regler for valgoppgjør (blant annet "first divisor in Sainte-Lague")
- Flagg som angir hvem som utfører andre telling (brukes i Sametingsvalget)

### Valgdistrikt/Delvalg (Contest)

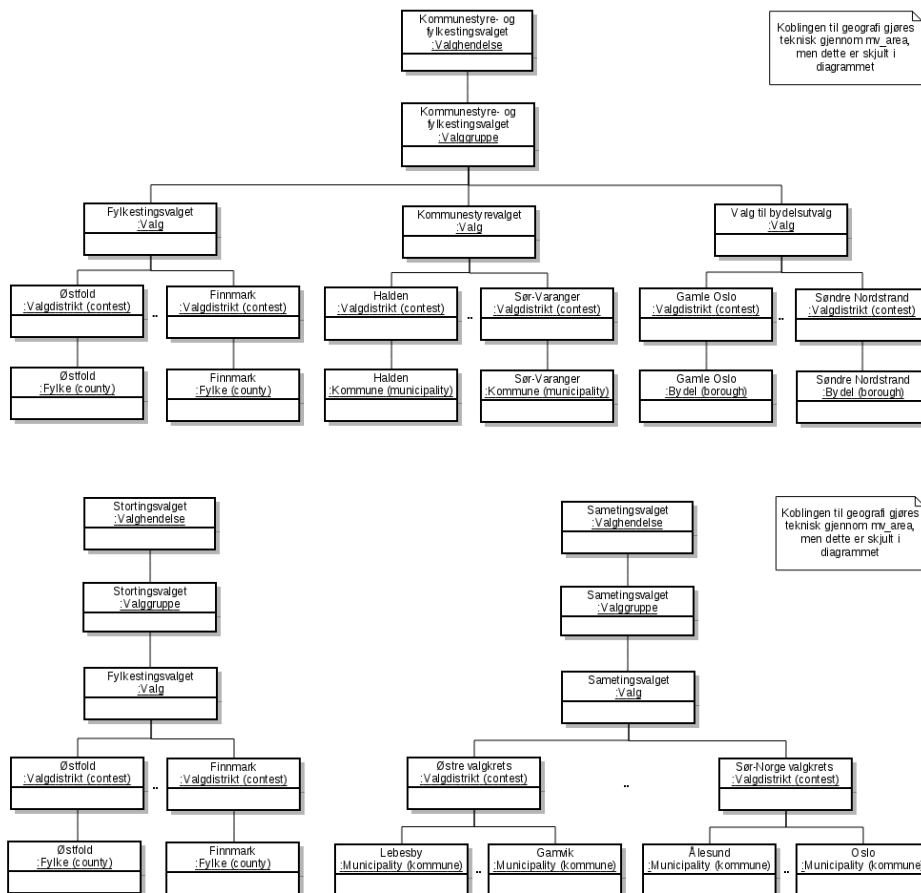
Her konfigureres geografien. I tillegg kan det gjøres spesialtilpasninger på noe av det som er konfigurert høyere opp i valghierarkiet.

- Knyttet til tellinger (via "contest report")
- Knyttet til valgoppgjør (settlement)
- Kan overskrive en del instillinger på valg, f.eks aldersgrense
- Maks antall kandidater, stemmegivninger, etc
- Flagg som angir hvem som utfører andre telling (kan overstyre verdien satt på valgnivå)
- Knyttet til stemmeseddel/partiliste

### Manntall

- Velger kan være i flere manntall, "gjenbruk" av persondata
- En valggruppe har ett manntall
- Innføres for å støtte at vi kan ha Fylkestingsvalg, Kommunestyrevalg og Bydelsvalg i samme Valghendelse

Objektdiagrammene under viser hvordan valghierarkiet konfigureres for de forskjellige typene valg som systemet støtter:



### Partikoderegister

Partikoderegisteret inneholder informasjon om partiene som kan stille til valg. Vi skiller mellom *partier* og *lister*. Et *parti* er den "globale" oppføringen i partikoderegisteret. *Lista* er partiets lokallag (f.eks. Venstre i Halden).

Informasjon som lagres i partikoderegisteret inkluderer:

- Partinavn
- Partinummer (4-sifre unik identifikator av partiet)
- Partikode (unik partikode for partiet)
- Type parti (stortingsparti, landsdekkende parti, lokalt parti)
- Godkjenning (må være på plass for at et parti skal kunne stille liste i et valg)
- Forenklet behandling (har betydning for hvor mange underskrifter som trengs for å godkjenne en liste)

### Lokale partier

I motsetning til stortingspartier og landsdekkende partier, som er tilgjengelig i hele landet, er lokale partier koblet opp mot et valgdistrikt ved hjelp av tabellen `party_contest_area`. Denne knytter partiet til et kommunenummer (municipality id) eller fylkesnummer (county id) som igjen brukes for å finne valgdistrikter (contest) partiet skal kunne stille lister i. Når det skal opprettes et listeforslag for et eksisterende parti, vil lokale partier kunne velges dersom de er knyttet til valgdistriktet og de ikke har opprettet liste allerede i dette valgdistriktet.

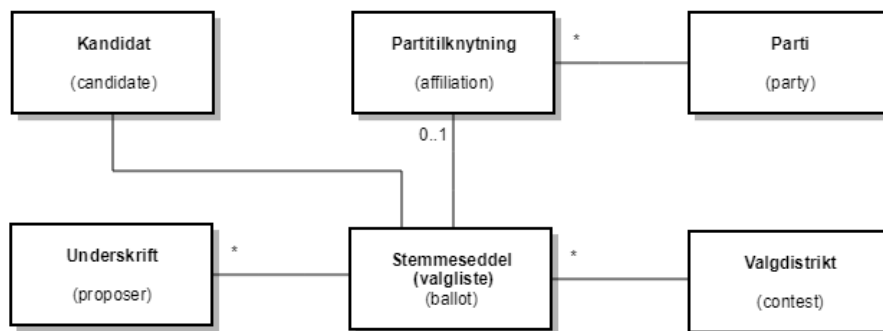
### Automatisk tildeling av partinummer (short\_code)

Ved opprettelse av et nytt parti, blir partinummeret tilordnet automatisk. Dette gjøres av trigger-funksjonen `set_party_number_aka_short_code`. Denne forutsetter at tabellen `party_number` er konfigurert for gjeldende valghendelse. `party_number`-tabellen blir opprettet når en ny valghendelse opprettes ved kopiering av en annen og det markeres at Valghierarkiet skal tas med i kopieringen. Dersom ikke denne tabellen er konfigurert, så blir ikke partinummer for det nye partiet satt.

### Listeforslag

Listeforslagsmodulen har funksjonalitet for å opprette og redigere listeforslag. Ved opprettelse av et listeforslag, knyttes det til et parti (`Affiliation`) og et valgdistrikt (`Contest`).

Redigering av listeforslag innebærer å legge til eller ta bort kandidater eller underskrifter. Når listeforslaget er ferdig kan det godkjennes. På et tidspunkt vil partitilknytningen i listeforslaget gitt av `Affiliation` manifestere seg til partiet og kandidatene som vises på stemmeseddelen (`Ballot`).



## Stemmeavgivningsmodul

### Formål

Modulen har som formål å tilby funksjonalitet til valgmedarbeiderne i kommunene for å registrere og prøve stemmeavgivninger i forhåndsstemmeperioden og på valgdagen(e). Modulen hjelper stemmemottakerne med å ha kontroll på om en stemmegiver allerede har avgitt stemme eller ikke.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for opptelling bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i stemmeavgivningsmodulen.

#### Måter å avgi stemme på

I valg gjennomføringen finnes det to typer stemmer som er knyttet til stemmeavgivningsprosessen. Forenklet er dette:

- Stemmeavgivninger avgitt i konvolutt
- Stemmeavgivninger avgitt i urne

Proessen for stemmegivninger avgitt i konvolutt er i stor grad en papirprosess med enkel systemstøtte. For begge prosessene er endepunktet mot systemet at stemmegivningen skal registreres som mottatt.

*MERK! Det registreres ingen knytning (implisitt eller eksplisitt) mellom stemmegivning (at velger har stemt) og stemmeseddel (hva velger har stemt).*

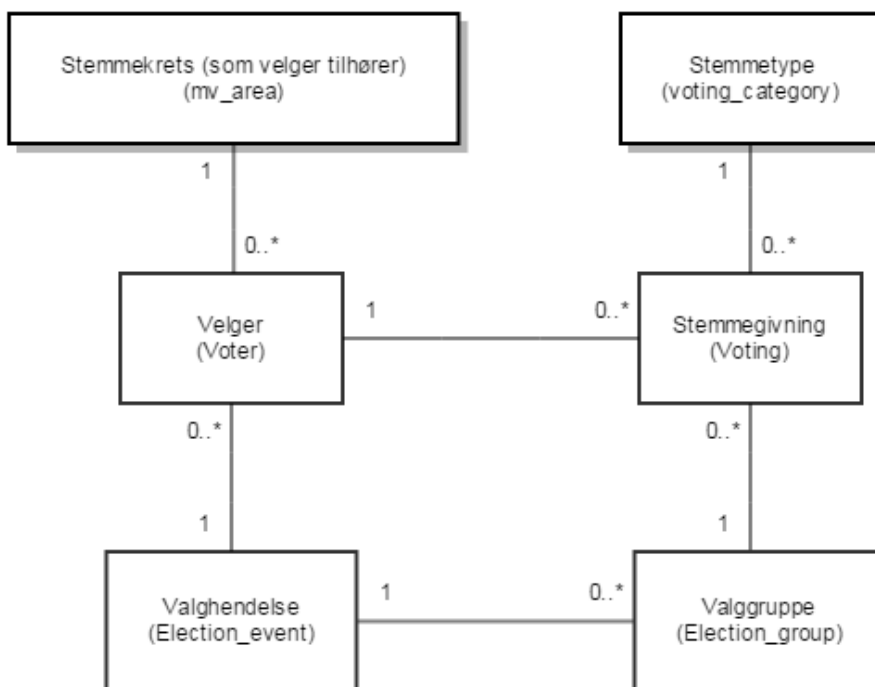
## Registrering av stemmegivning

Manntallet for en gitt valggruppe ligger (i voter-tabellen) i EVA Admin. Det er kun de stemmeberettigede som stemmemottakerne kan søke opp ved registrering av stemmegivninger.

Alle forhåndsstemmegivninger registreres elektronisk (i voting-tabellen) i EVA Admin. Under valgdagene er det kun stemmegivninger i kommuner med elektronisk avkryssning i manntall som registreres i EVA Admin. Kommuner som ikke har elektronisk avkryssning i manntall på valgdagen må skrive ut avkryssningsmanntallet etter endt forhåndsstemmeperiode for å ha oversikt over stemmegivere som allerede har avgitt godkjent stemme.

En stemmegivning (voting) knyttes opp mot valggruppe (election\_group), som f.eks. Kommunestyre- og fylkestingsvalget, og ikke opp mot hvert enkelt valg (election) kommunevalget, fylkestingsvalget eller bydelsvalget. Dette gjøres for å støtte flere valg mot samme manntall, som er tilfelle ved kommunestyre- og fylkestingsvalg.

Det kan kun registreres én godkjent stemmegivning for en person, påfølgende stemmegivninger vil avvises i systemet og følges opp i en manuell prosess regulert av lov/forskrift.



## Stemmetyper

Stemmegivninger er delt inn i forskjellige typer. Dette gjøres hovedsaklig av to grunner:

- For å kunne gruppere stemmetyper - primært for å kunne levere statistikk på en hensiktsmessig oppløsning
- For å kunne styre funksjonalitet i brukergrensesnitt - forskjellige stemmetyper krever forskjellig brukerinteraksjon

Tabellen angir hvilke stemmetyper stemmegivninger kan tilhøre

Stemmetype - ID	Stemmetype - beskrivelse	Forhåndsstemme	Valgting
FI	Forhåndsstemmer innenriks	JA	
FU	Forhåndsstemmer utenriks	JA	
FB	Brevstemmer utenriks	JA	
FE	Stemmer i beredskapskonvolutt	JA	
FA	Forhåndsstemmer til andre kommuner	JA	
VO	Valgtingsstemmer		JA

VF	Fremmedstemmer		JA
VB	Stemmer i beredskapskonvolutt		JA
VS	Stemmer i særskilt omslag		JA

## Oversikt - registrering av stemmegivninger i EVA Admin

Tabellen angir knytningen mellom stemmetype og valg gjennomføringsprosess, samt styring av funksjonalitet i brukergrensesnittet

Link for registrering i EVA Admin	Stemmetype	Registrering	Prøving*
Registrering (bare "rett i urne")	FI	Stemmegivning registreres ute på allment tilgjengelig forhåndsstemmed	automatisk
Mottatte konvolutter egen kommune	FI	Stemmegivning registreres på konvolutt-mottaket i kommunen som velger tilhører	manuelt
Mottatte konvolutter egen kommune	FU	Stemmegivning registreres på konvolutt-mottaket i kommunen som velger tilhører	manuelt
Mottatte konvolutter egen kommune	FB	Stemmegivning registreres på konvolutt-mottaket i kommunen som velger tilhører	manuelt
Mottatte konvolutter egen kommune	FE	Stemmegivning registreres på konvolutt-mottaket i kommunen som velger tilhører	manuelt
Mottatte konvolutter egen kommune	FA	Stemmegivning registreres ute på forhåndsstemmed i en annen kommune enn den velger tilhører	manuelt
Registrering	VO	Stemmegivning registreres ute på et valgtingsstemmed	automatisk
Registrering	VF	Stemmegivning registreres ute på et valgtingsstemmed som er i en annen krets enn den velger tilhører	automatisk
Sentral registrering av stemmegivninger	VB	Stemmegivning registreres ute på et valgtingsstemmed	manuelt
Sentral registrering av stemmegivninger	VS	Stemmegivning registreres ute på et valgtingsstemmed	manuelt/automatisk

\*Automatisk = Valgprosessen bestemmer at registrering av stemmegivningen og godkjenning gjøres i samme operasjon.

\*Manuelt = man må aktivt inn i EVA og fortelle systemet hvorvidt en stemmegivning faktisk skal godkjennes eller forkastes.

## Prøving

Alle stemmegivninger som ikke legges rett i urne skal prøves, dvs. at stemmegivningens gyldighet skal vurderes. Alle stemmegivninger som ikke legges rett i urne vil være lagt i en konvolutt.

Konvoluttstemmegivninger registreres i EVA Admin og får tildelt et løpenummer, som lar seg søke opp når stemmegivninger skal prøves. Prøving kan ha to utfall:

- Stemmegivningen godkjennes
- Stemmegivningen forkastes

Dersom en stemmegivning forkastes gjøres dette med utgangspunkt i lov og forskrift, forkastelsesgrunn angis ved forkasting av en stemmegivning.

Da stortingsvalg baserer seg på ny valglov av 2024 mens sametingsvalg baserer seg på gammel valglov av 2002, brukes forskjellige forkastelsesgrunner i de to valgene, som vist i denne tabellen:

Valgtype	Forkastelsesgrunn - ID	Forkastelsesgrunn - beskrivelse	Forhånd /Valgting
Stortingsvalg	G0	Feilregistrert stemmegivning	Begge
Stortingsvalg	G1	Velgeren er ikke manntallsført i kommunen § 10-2 (1) a	Begge
Stortingsvalg	G2	Velgeren er ikke innført i manntallet § 10-2 (1) a	Begge
Stortingsvalg	G3	Det kan ikke fastslås hvem velgeren er § 10-2 (1) b	Begge
Stortingsvalg	G4	Stemmegivningen er ikke levert et sted der velgeren kan stemme § 10-2 (1) c	Begge
Stortingsvalg	G5	Det er sannsynlighetsovervekt for at omslagskonvolutten er åpnet § 10-2 (1) d	Begge
Stortingsvalg	G6	Velgeren har tidligere fått godkjent en stemmegivning § 10-2 (1) e	Begge
Stortingsvalg	G7	Stemmegivningen har ikke kommet inn til valgstyret etter at forhåndsstemmingen har startet eller innen klokken 17 dagen etter valgdagen § 10-2 (1) f	Begge
Sametingsvalg	F0	Feilregistrert stemmegivning	Forhånd

Sametingsvalg	FA	Velgeren er ikke innført i manntallet i kommunen § 10-1 (1) a	Forhånd
Sametingsvalg	FB	Stemmegivningen inneholder ikke tilstrekkelige opplysninger til å fastslå hvem velgeren er § 10-1 (1) b	Forhånd
Sametingsvalg	FC	Stemmegivningen er ikke avgitt til rett tid § 10-1 (1) c	Forhånd
Sametingsvalg	FD	Stemmegivningen er ikke levert til rett stemmemottaker § 10-1 (1) d	Forhånd
Sametingsvalg	FE	Omslagskonvolutten er åpnet eller forsøkt åpnet § 10-1 (1) e	Forhånd
Sametingsvalg	FF	Velgeren har allerede avgitt godkjent stemmegivning § 10-1 (1) f	Forhånd
Sametingsvalg	FG	Stemmegivningen er ikke kommet inn til valgstyret innen kl. 17 dagen etter valgdagen §10-1 (1) g	Forhånd
Sametingsvalg	FH	Velgeren er ikke innført i manntallet i kommunen § 10-1a (1) a	Forhånd
Sametingsvalg	FI	Velgeren har allerede avgitt godkjent stemmegivning § 10-1a (1) c	Forhånd
Sametingsvalg	V0	Feilregistrert stemmegivning	Valgting
Sametingsvalg	VA	Velgeren er ikke innført i manntallet i kommunen § 10-2 (1) a	Valgting
Sametingsvalg	VC	Velgeren har avgitt godkjent stemmegivning § 10-2 (1) c	Valgting

## Opptellingsmodul

### Formål

Opptellingsmodulen har ansvar for å understøtte valgrutinene for opptelling av stemmesedler. Opptellingen for en kommune gjennomføres for flere kategorier og områder i kommunen avhengig av kommunens konfigurasjon. For hvert område og kategori telles det minst to ganger for kommunestyre- og sametingsvalg og minst 3 ganger for fylkestings- og stortingsvalg. En opptelling kan enten være manuell eller en import fra EVA Skanning.

Merk at dokumentet beskriver opptellinger som utført på stemmekrets, selv om valgloven tilsier at de skal telles på valglokale. Dette opprettholdes ved at forholdet mellom valglokaler og stemmekretser er 1:1, altså vil tellinger fordelt på krets alltid være tilknyttet ett, og bare ett, valglokale.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for opptelling bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

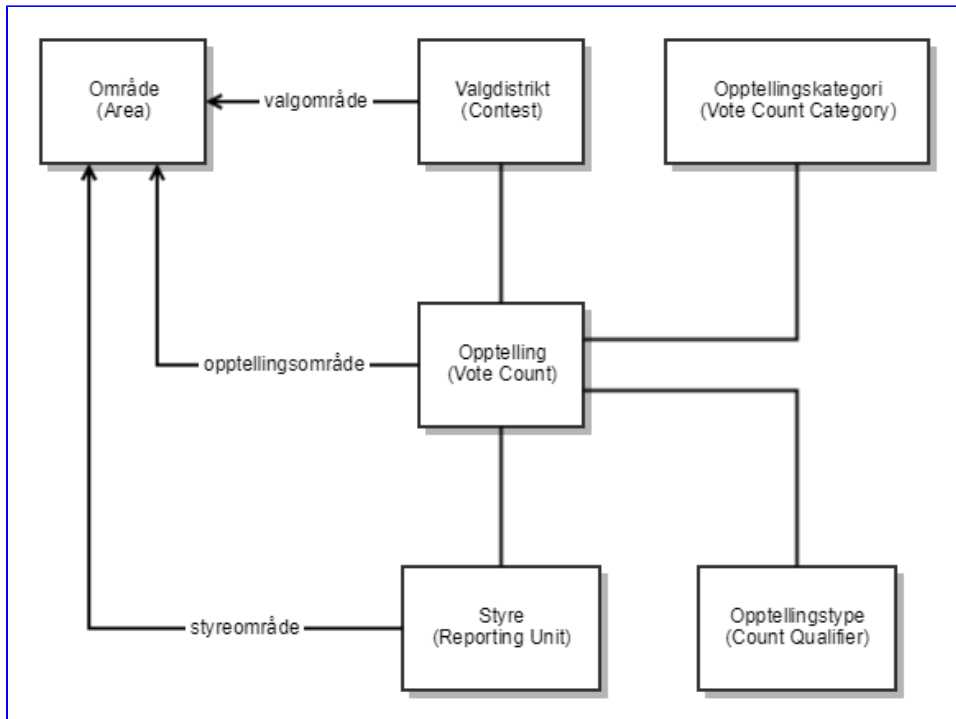
### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i opptellingsmodulen.

#### Tilknytninger

En opptelling har flere tilknytninger:

1. **Område** - kan være:
  - a. *opptellingsområde*. Kan være en stemmekrets, teknisk krets, bydel eller kommune.
  - b. *styreområde*. Kan være en stemmekrets (Ansvarlig valglokale - merk at dette ikke er et fullverdig styre), en kommune (Valgstyret) eller en fylkeskommune (Distriktsvalgstyret).
  - c. *valgområde*. Kan være kommune (Kommunestyrevalg/Sametingsvalg), fylke (Fylkestingsvalg/Stortingsvalg) eller bydel (Valg til bydelsutvalg/kommunedelsutvalg) og områdene det telles på ligger under dette området.
2. **Styre** - styret det telles for. Kan være et valgstyre for en kommune, et distriktsvalgstyre for et fylke eller et opptellingsvalgstyre for valgkretser (=valgdistrikter) i Sametingsvalget. Ansvarlig valglokale bruker også denne strukturen, selv om det ikke er et fullverdig styre.
3. **Valgdistrikt** - valget det telles for, f.eks. Kommunestyrevalget i én kommune.
4. **Opptellingskategori** - f.eks. valgtingsstemmer - fra valglokalene.
5. **Opptellingstype** - kan være en urnetelling, en urne- og første telling, en andre telling (for kommunen) eller en fylkeskommunens kontrolltelling (for fylkeskommunen i valg på fylkesnivå, dvs. et fylkestingsvalg eller et stortingsvalg)



### Områdetyper

Områdetype	Beskrivelse
Fylke	Område for fylkestingsvalg og stortingsvalg.
Kommune	Område for kommunestyrevalg, sametingsvalg og opptellinger som telles sentralt samlet.
Bydel	Område for "valg til bydelsutvalg/kommunedelsutvalg" og opptelling som telles sentralt samlet for dette valget.
Stemmekrets	Område for opptellinger fordelt på krets. Brukes til forhåndsstemmer og valgtingsstemmer fra valglokalene (Sametinget omtaler sine valgdistrikter som valgkretser. Dette er ikke det samme som områdetypen stemmekrets).
Teknisk krets	Område for opptelling fordelt på tekniske kretser. Brukes i enkelte kommuner med stort antall forhåndsstemmer - øvrige (Bergen, Trondheim og Oslo).

### Opptellingskategorier

En opptelling telles for en gitt kategori. Det er 3 kategorier for forhånd og 2 kategorier for valgting i stortings-, fylkestings- og kommunestyrevalg, mens sametingsvalget har 2 kategorier for forhånd og 2 kategorier for valgting. Opptellingskategoriene er beskrevet under.

Opptellingskategori	Kode	Valg	Fase	Beskrivelse
Forhåndsstemmer - fra valglokalene	FO	ST F K B	Forhånd	Stemmesedler kommet inn i tidligstemmeperioden og forhåndsstemmeperioden i allment tilgjengelig valglokale.
Forhåndsstemmer - øvrige	FT	ST SA F K B	Forhånd	Stemmesedler mottatt i konvolutt eller i ikke allment tilgjengelig valglokale.
Forhåndsstemmer - telt etter kl. 17 dagen etter valgdagen	FS	ST SA F K B	Forhånd	Stemmesedler registrert mottatt av kommunen i posten på valgdagen og stemmesedler fra FT lagt til side.
Valgtingsstemmer - fra valglokalene	VO	ST SA F K B	Valgting	Stemmesedler mottatt i urne på valgdagene.
Valgtingsstemmer - øvrige	VS	ST SA F K B	Valgting	Stemmesedler mottatt i konvolutt på valgdagen som ikke kunne legges rett i urne.
Bydelsfremmedstemmer	BF	B	Valgting	Stemmesedler til andre bydeler enn den bydelen stemmekretsen ligger i. Brukes kun for "Valg til bydelsutvalg".

ST=stortingsvalg, SA=sametingsvalg, F=fylkestingsvalg, K=kommunestyrevalg, B=valg til bydelsutvalg

### Opptellingstyper



Avhengig av valg og kategori telles det med forskjellige opptellingstyper. Disse er beskrevet i tabellen under.

Opptellingstype	Styret som teller	Valg	Opptellingsområde	Beskrivelse
Urnetelling	Ansvarlig valglokale	<b>ST SA F K B</b>	Stemmekrets	Initiell urnetelling i valgkrets. Gjelder kun for opptellingskategorien VO.
Urne- og første telling	Ansvarlig valglokale	<b>ST SA F K B</b>	Stemmekrets	Kombinert urne- og første telling utført i valgkretsen. Gjelder kun for opptellingskategorien VO.
Første telling	valgstyre	<b>ST SA F K B</b>	Stemmekrets Teknisk krets Kommune Bydel	Første partifordeling av stemmesedler. Utført for stemmekretser, tekniske kretser, for hele kommunen eller bydelen avhengig av kommunens konfigurasjon av valget og oppsett av geografi. Gjelder for alle opptellingskategorier.
Andre telling (Kommunens andre telling)	valgstyre	<b>ST SA F K B</b>	Stemmekrets Teknisk krets Kommune Bydel	Andre partifordeling av stemmesedler. Utført for stemmekretser, tekniske kretser, for hele kommunen eller bydelen avhengig av kommunens konfigurasjon av valget og oppsett av geografi. Gjelder for alle opptellingskategorier. Merk at selv om den kalles "andre" telling kan det finnes flere enn én andre telling. Kun én andre telling kan godkjennes.
Fylkeskommunens kontrolltelling	distriktsvalgstyre	<b>ST F</b>	Stemmekrets Teknisk krets Kommune	Fylkeskommunens kontroll av partifordeling av stemmesedler. Utført for stemmekretser, tekniske kretser, og for hele kommunen avhengig av kommunens konfigurasjon av valget og oppsett av geografi. Gjelder for alle opptellingskategorier.
Opptellingsvalgstyrets andre telling	opptellingsvalgstyre	<b>SA</b>	Stemmekrets Kommune	Opptellingsvalgstyrets andre telling med partifordeling av stemmesedler. Utført for stemmekretser og for hele kommunen avhengig av kommunens konfigurasjon av valget og oppsett av geografi. Gjelder for alle opptellingskategorier.

**ST**=stortingsvalg, **SA**=sametingsvalg, **F**=fylkestingsvalg, **K**=kommunestyrevalg, **B**=valg til bydelsutvalg

## Opptellingsmåter

Kommunen konfigurerer hver opptellingskategori på én av fire måter. Det er begrensning i systemet som gjør at kun noen kategorier kan konfigureres på en gitt måte. Opptellingsmåtene er beskrevet i tabellen under.

Opptellingsmåte	Beskrivelse	Brukes av opptellingskategori	Første telling sentralt	Første telling på stemmekrets	Første telling på teknisk krets
Lokalt fordelt på krets	Opptellingene fordeles på stemmekretser. Ansvarlig for valglokale teller urne- og første tellinger. Valgstyret, distriktsvalgstyret (for fylkestingsvalg og stortingsvalg) og opptellingsvalgstyret (for sametingsvalg) teller andre tellinger.	<b>VO</b>	NEI	JA	NEI
Sentralt fordelt på krets	Opptellingene fordeles på stemmekretser. Valgstyret teller første og andre tellinger. Distriktsvalgstyret (for fylkestingsvalg og stortingsvalg) og opptellingsvalgstyret (for sametingsvalg) teller andre tellinger.	<b>FO</b>	JA	JA	NEI
Sentralt fordelt på teknisk krets	Opptellingene fordeles på tekniske kretser. Valgstyret teller første og andre tellinger. Distriktsvalgstyret (for fylkestingsvalg og stortingsvalg) og opptellingsvalgstyret (for sametingsvalg) teller andre tellinger.	<b>FO</b>	JA	NEI	JA
Sentralt samlet	Opptellingene fordeles på kommune eller bydel. Valgstyret teller første og andre tellinger samlet for kommune eller bydel (gjelder kun for valg til bydelsvalg). Distriktsvalgstyret (for fylkestingsvalg og stortingsvalg) og opptellingsvalgstyret (for sametingsvalg) teller andre tellinger.	<b>FS, FT, VT</b>	JA	NEI	NEI

**MERK!** For sametingsvalg er det kun opptellingsvalgstyrene som teller andre tellinger.

## Opptellingsstatus

En opptelling har en av fem statuser beskrevet i tabellen under.

Opptellingsstatus	Brukes av opptellingstyper	Beskrivelse
<b>Ny</b>	<b>U F A K</b>	Angir at opptellingen ikke er lagret ennå. Forekommer ikke i persistent lager.
<b>Lagret</b>	<b>U F A K</b>	Angir at opptellingen er lagret.
<b>Godkjent</b>	<b>U F A K</b>	Angir at opptelling er godkjent.

<b>Til valgoppgjør</b>	<b>A K</b>	Angir at tellingen er klar til valgoppgjør. Gjelder kun for valgets siste, godkjente andre eller kontrolltelling. Dvs. valgstyrets andre telling for kommunestyrevalg og valg til bydelsutvalg, fylkeskommunens kontrolltelling for fylkestingsvalg og stortingsvalg og opptellingsvalgstyrets andre telling i sametingsvalg.
<b>Opphevet</b>	<b>U F A K</b>	Angir at en godkjent opptelling har fått godkjenningen opphevet. En opptelling kan oppheves om den er godkjent eller klar til valgoppgjør.

**U:** Urnetelling, **F:** Første telling, **A:** Andre telling (enten kommunens andre telling eller opptellingsvalgstyrets andre telling), **K:** Fylkeskommunens kontrolltelling

## Valgoppgjørsmodul

### Formål

Valgoppgjørsmodulen har ansvar for å gjennomføre valgoppgjør som i praksis vil si å gjøre nødvendige beregninger for mandatfordeling og kandidatføring.

For å kunne foreta valgoppgjør må alle opptellinger være ferdige, og foreslåtte forkastede stemmesedler må være ferdig behandlet.

Valgoppgjør foretas av øverste kontrollinstans for et gitt valg, f.eks vil distriktsvalgstyret gjennomføre valgoppgjør for et gitt valgdistrikt på fylkesnivå.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for opptelling bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i valgoppgjørsmodulen.

Modulen har tre hovedfunksjoner:

- Kontrollere status på opptellinger som er relevante for det gitte valgoppgjøret
- Gjennomføre kandidatføring
- Beregne mandatfordeling samt utjevningsmandater dersom det er relevant (konfigurert i valget)

### Avhengigheter

Modulen er avhengig av modulene for opptelling og konfigurasjon og benytter data fra disse domeneene for å foreta valgoppgjøret. Modulen kontrollerer selv at grunnlag for valgoppgjør er på plass:

- At alle relevante andre- og kontrolltellinger er gjennomført
- At behandling av ev. forkastede stemmesedler er gjennomført

## Kandidatføring og mandatfordeling

### Opptelling av listestemmer

I valg hvor "slengere" (kandidater fra andre lister) ikke er tillatt beregnes listestemmer som antall stemmesedler et parti har fått. I valg hvor "slengere" er tillatt beregnes listestemmer som antall stemmesedler multiplisert med antall mandater i valgdistriktet. I tillegg blir et antall listestemmer trukket fra den listen de er skrevet opp på, og lagt til den listen de kommer fra.

### Opptelling av kandidatstemmer

For hver kandidat i hvert parti beregnes kandidatstemmer etter kategori (grunnstemme, personstemme, slenger, renummerering og strykning). I noen valg, f.eks. kommunestyrevalg, er det vanlig å forhåndskumulere én eller flere kandidater øverst på listen. Disse utheves på stemmeseddelen og gis et sett grunnstemmer basert på antall stemmesedler partiet har fått i dette valgdistriktet.

Antall grunnstemmer finnes ved å multiplisere antall partistemmer med en faktor konfigurert for valget, f.eks. 0,25. For renummereringer er antallet gruppert på hvilken posisjon det er renummerert til. For slengere er antallet gruppert på partiet til listen slengeren er oppført på (ikke partiet kandidaten tilhører).

### Kandidatføring

Kandidater rangeres per liste på følgende måte:

- Synkende etter antall kandidatstemmer
- Ved likt antall kandidatstemmer rangeres kandidatene etter kandidatnummer på listen

Noen valg, f.eks. fylkestingsvalg, har en sperregrense for om en kandidats stemmer skal telle med i rangeringen av kandidaten. Angis som en andel av listens stemmer i dette valgdistriktet og antall kandidatstemmer tas med i beregning av rangeringen kun dersom kandidaten får mer enn andelen, f.eks. 8 %, av listens stemmer. En sperregrense på 0 % angir at alle kandidatstemmer skal telle med i rangeringen.

#### *Kandidatkåring for valg med renummereringer og strykninger*

Rangeringen av kandidater gjennomføres hovedsakelig i to steg for hver liste:

For hver kandidat telles det opp et antall stemmer for hver kandidatplass basert på eksplisitte (renummererte) og implisitte kandidatposisjoner på stemmeseddelen. Plassene fordeles på følgende måte

- Plass 1 fordeles til kandidaten med høyest antall stemmer for 1. plass
- Plass 2 fordeles til kandidaten med høyest antall stemmer for 1. og 2. plass (kandidaten på 1. plass utelukkes)
- Plass n fordeles til kandidaten med høyest antall stemmer for plass 1-n (kandidatene på plass 1-(n-1) utelukkes)
- Hvis to eller flere kandidater har samme antall stemmer for en gitt plass gis plassen til kandidaten med lavest kandidatnummer.

For å beregne korrekte kandidatplasser må hver rettede stemmeseddel behandles individuelt. Kandidatplasser beregnes på følgende måte:

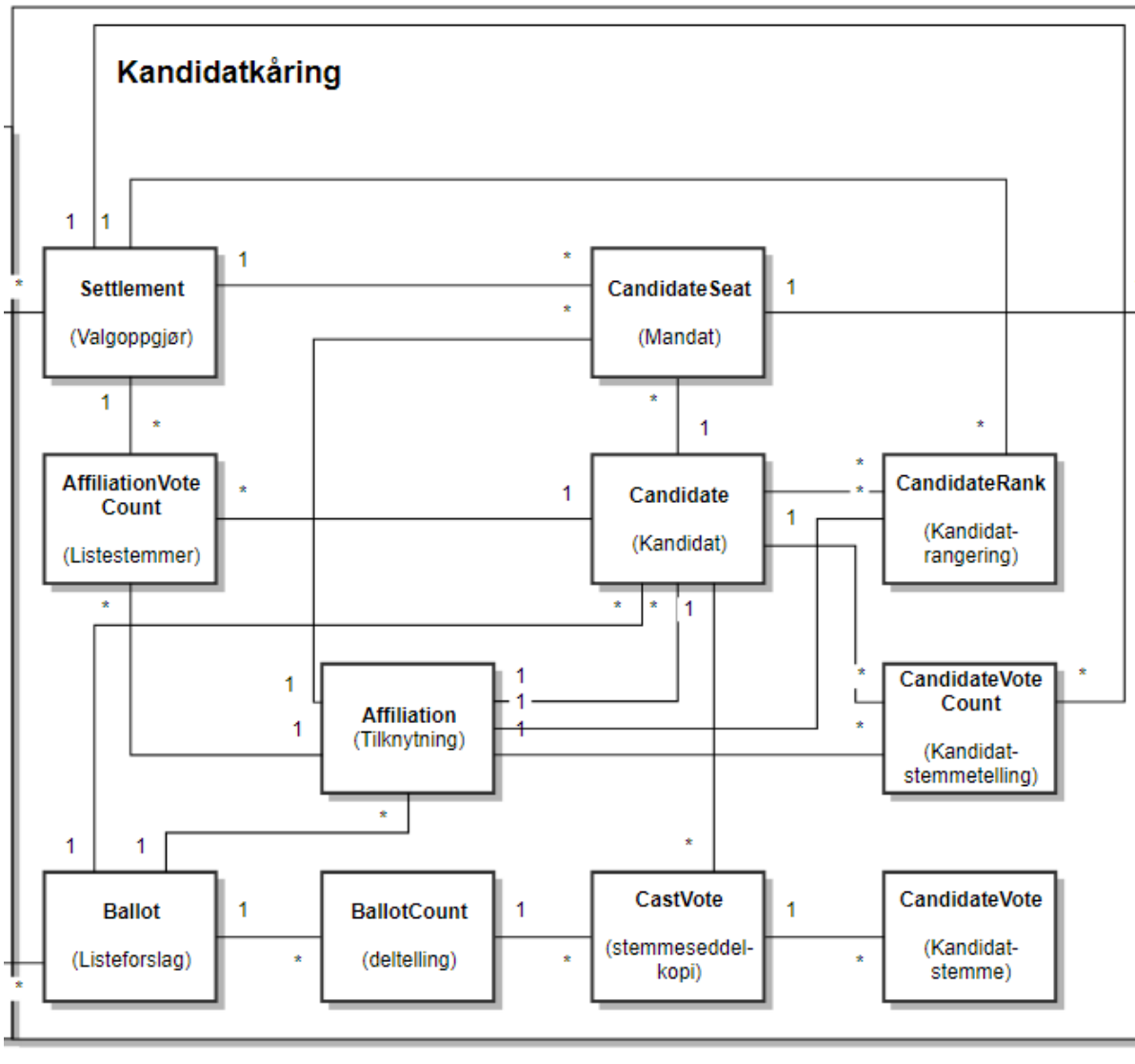
1. Strøkne kandidater fjernes fra listen (dvs. de får ingen stemmer på noen plasser fra stemmeseddelen)
2. Renummererte kandidater fjernes fra listen (dvs. uendrede kandidatene gis en initiell ny ordning)
3. Renummererte kandidater ordnes etter ny plassering og settes inn igjen i listen på ny plass ordnet fra lavest til høyest. Dette fører til at alle kandidater på eller under denne gitte plassen blir skjøvet en plass ned. En renummerering til en plass lenger en listens lengde + 1 behandles som en renummerering til listen lengde + 1.

### **Mandatfordeling**

Mandater fordeles etter Sainte-Laguës modifiserte metode. Se eksempel her:

([Sainte-Laguës metode](https://no.wikipedia.org/wiki/Sainte-Lagu%C3%ABs_metode)([https://no.wikipedia.org/wiki/Sainte-Lagu%C3%ABs\\_metode](https://no.wikipedia.org/wiki/Sainte-Lagu%C3%ABs_metode)))

*Utsnitt av domenemodellen som understøtter prosessene beskrevet over:*



## Fordeling av utjevningsmandater

Overordnet følger fordeling av utjevningsmandater følgende steg:

1. Partistemmer for hele stortingsvalget som ett valgdistrikt telles opp
2. Lokale partier og partier som faller under sperregrensen (4% av stemmene på landsbasis) utelukkes fra fordelingen av utjevningsmandater
3. Distriktsmandater for hele landet beregnes/telles opp
4. Mandater fordeles etter Sainte-Lagües modifiserte metode for hele landet som ett valgdistrikt
5. Partier som har fått flere distriktsmandater (punkt 3) enn antall mandater når hele landet ses på som ett valgdistrikt (punkt 4) utelukkes fra konkurransen om utjevningsmandater. Punkt 4 og 5 gjentas inntil det ikke er flere partier som utelukkes.
6. Antall utjevningsmandater beregnes for hvert parti (antall mandater fra punkt 4 minus antall distriktsmandater fra punkt 3)
7. Beregning av kvotienter for utjevningsmandater (lokale partier og partier under sperregrensen inkluderes i beregning av kvotienter) etter (ikke modifisert) Sainte-Lagües metode
8. Utjevningsmandater fordeles til partier og valgdistrikter (begrenses til ett per valgdistrikt og et maks antall per parti som beregnet i punkt 6)

## Optelling av partistemmer og distriktsmandater for hele stortingsvalget

Listestemmer og distriktsmandater fra valgoppgjør i alle valgdistrikt i stortingsvalget (for Stortingsvalget 2021 var det 19 valgdistrikter for 18 fylker + Oslo) telles opp og grupperes per parti. Alle partier inkluderes, også lokale partier (dvs. ikke landsdekkende partier eller stortingspartier) og partier som ikke kommer over sperregrensen.

## Mandatfordeling for hele stortingsvalget og beregning av antall utjevningsmandater per parti

Partier som har fått færre distriktsmandater enn de ville fått om hele landet ses på som ett valgdistrikt blir inkludert i fordelingen av utjevningsmandater. Lokale partier og partier som ikke kommer over sperregrensen utelukkes fra fordelingen av utjevningsmandater

## Beregning av kvotienter for utjevningsmandater

For hvert parti og valgdistrikt beregnes det kvotienter som brukes i fordelingen av utjevningsmandater. Kvotienten regnes ut på følgende måte:

- $\text{dividend} = \text{partistemmer} / (\text{distriktsmandater} * 2 + 1)$
- $\text{divisor} = \text{antall godkjente stemmer i valgdistrikt} / \text{antall mandater i valgdistriktet}$
- $\text{kvotient} = \text{dividend} / \text{divisor}$

### Fordeling av utjevningsmandater

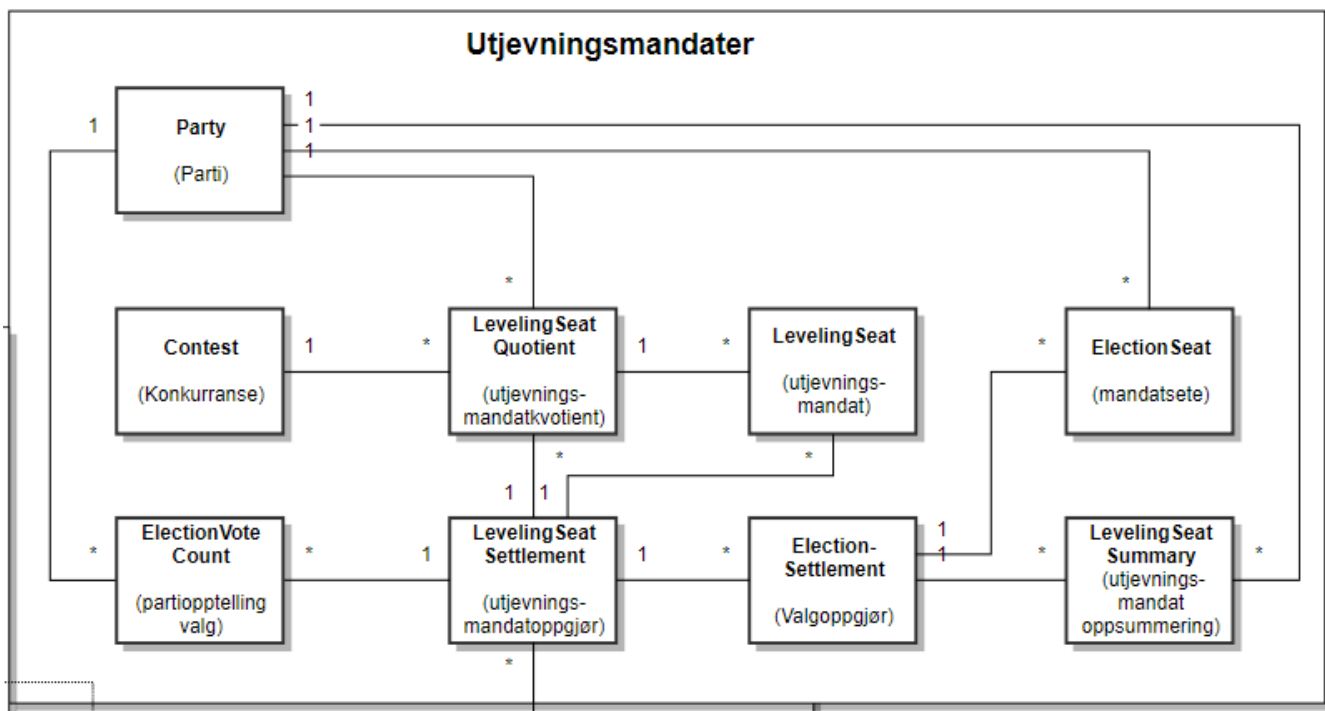
Kvotientene rangeres på følgende måte:

- Kvotienter fra høyest til lavest
- Antall partistemmer, høyest til lavest, hvis kvotientene er like
- Loddrekning dersom både kvotienter og antall partistemmer er like

### Sperregrense

For at parti skal kunne være med i fordelingen av utjevningsmandater, må det få et antall stemmer som er mer enn eller lik med 4 % av totalt antall godkjente stemmesedler (blanke stemmesedler unntatt). Sperregrensen for utjevningsmandater konfigureres når valget settes opp i EVA Admin.

Utsnitt av domenemodellen som understøtter prosessen beskrevet over:



## Rapportmodul

### Formål

Rapportmodulen understøtter uttak av rapporter og statistikk gjennom hele valgprosessen. I hovedsak kan rapporter deles inn i 2 grupper:

- Statistikk - statistikkrapporter tas ut av brukerne av systemet for å understøtte beslutningsprosesser eller for å innhente informasjon om valgavviklingen.
- Protokoller - protokollrapporter kalles som oftest "møtebøker", dette er de offisielle protokollene for valgavviklingen på et gitt nivå i valgdistriktet eller for valget. Disse brukes kun av sametingsvalget, da andre valg har en egen protokolleringsløsning.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for oppstilling bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i rapporteringsmodulen.

Rapportmodulen gir tilgang til statistikk og valginformasjon gjennom hele valg gjennomføringen. Avhengig av valgprogresjonen hentes informasjon fra en eller flere av fasene:

- Forberedelse
- Stemmegivning
- Opptelling
- Valgoppgjør

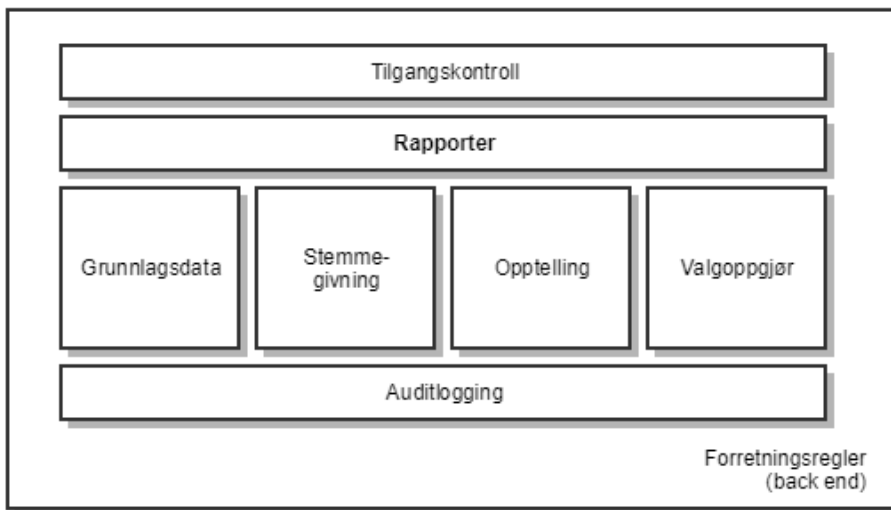
Rapportene tas ut som PDFer og/eller excel-filer, avhengig av bruksområde.

### Datagrunnlag

Rapporter og statistikk er typisk sammenstilling av data fra flere domeneområder, for å oppnå dette opererer rapporteringsmodulen på hele domenet i EVA Admin.

Typisk vil en "møtebok" (valgprotokoll) genereres med data fra hele valg gjennomføringen fra grunnlagsdata til valgoppgjør.

Skissen illustrerer at rapportering skjer basert på data fra en eller flere faser avhengig av valgprogresjon

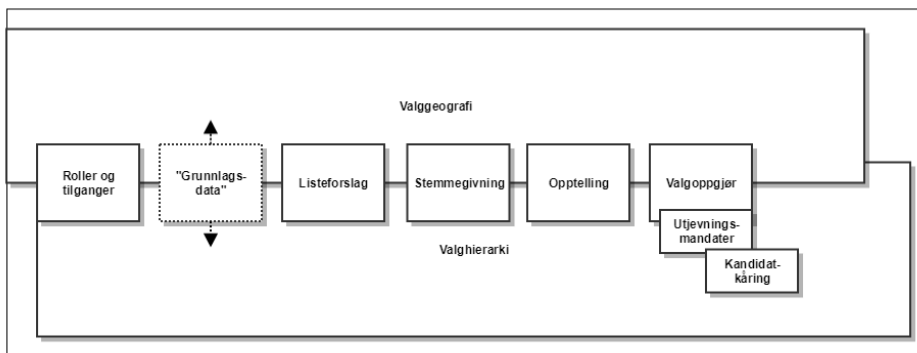


## Valgdomenet

Hoveddomenet for EVA Admin er valg, for å ikke forholde seg til hele valgdomenet på en gang er valgdomenet brutt ned i sub-domener som skissert under.

Sub-domener er tegnet over sub-domeneene Valggeografi og Valghierarki fordi alle sub-domene operer i konteksten av Valggeografi og/eller Valghierarki.

"Grunnlagsdata" er ikke et tydelig sub-domene, men er et begrep som brukes av og mot kommuner og fylkeskommuner. "Grunnlagsdata" er den informasjonen kommuner og fylkeskommuner selv registrerer i valggeografien og valghierarkiet.



## Domenemodell - geografi og valghierarki

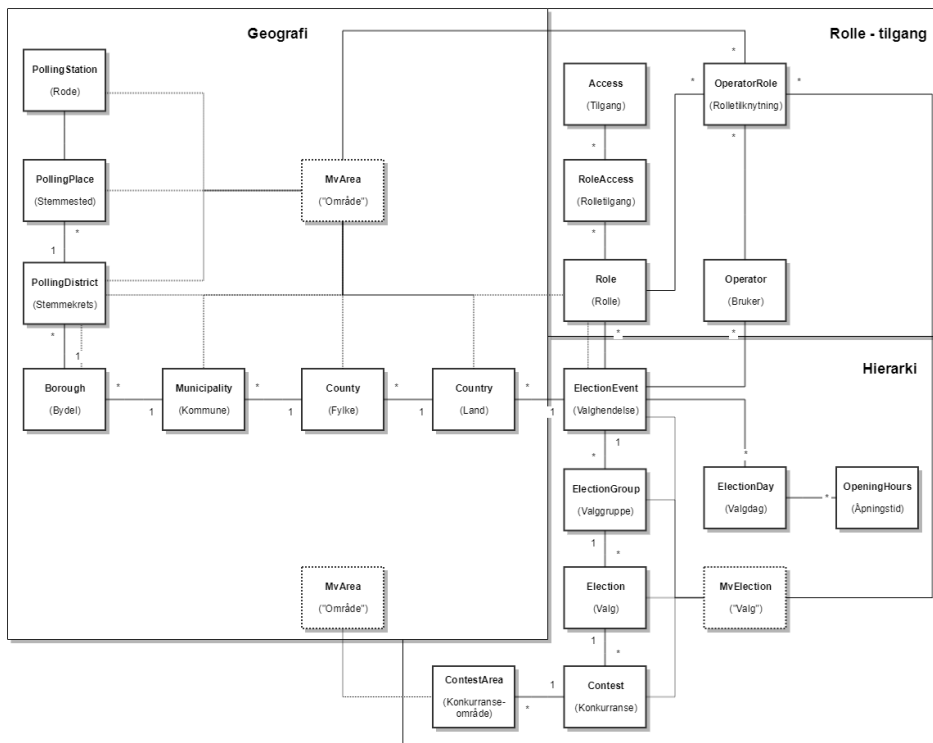
To helt sentrale konsepter som funksjonaliteten i EVA Admin bygger på, er *områdehierarkiet* eller "Geografi" og *valg*hierarkiet.

All valgkonfigurasjon og alle valgprosesser i EVA Admin utføres for:

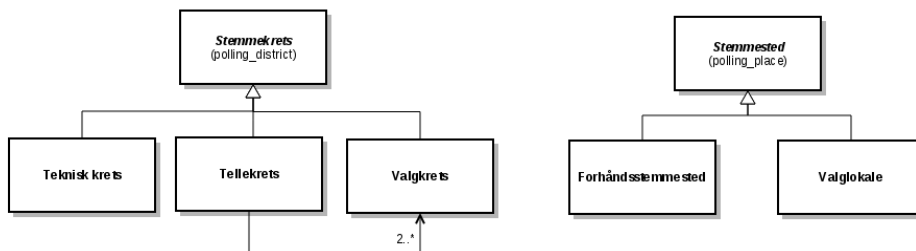
- Et gitt nivå i valg
- Et gitt nivå i områdehierarkiet, dvs. for et gitt geografisk område.

Områdehierarkiet vises i venstre del av diagrammet under. Her settes Norges geografi opp, fra alle fylker ned til stemmesteder. På kodenivå er det introdusert en områdeabstraksjon, kalt `mv_area`, som mye av områdeinteraksjonen skjer gjennom. Abstraksjonen er implementert på databasenivå ved hjelp av et "materialized view" som samler alle områdenivåer i en tabell.

Valghierarkiet vises i høyre del av diagrammet under. Her konfigureres valghendelse, valggruppe, valg og valgdistrikt. På kodenivå er det introdusert en valgabstraksjon, kalt `mv_election`, som mye av valginteraksjon skjer gjennom som er implementert på tilsvarende måte som `mv_area`.



Stemmekrets og stemmested i områdehierarkiet er abstraherte konsepter, og kan ytterligere konkretiseres:



## Valghendelse

Rotentiteten i valghierarkiet:

- Setter opp valgdager og datoer for valget
- Er knyttet til geografikonfigurasjon
- Er knyttet til bruker-tabell, slik at settet med brukere er angitt per valghendelse. (Her blir som oftest personopplysningene hentet fra velger-tabellen.)
- Stil (storting, sameting, test)

## Valggruppe

Her settes optellingmåter opp for valget.

- Er knyttet til stemmegivninger (Voting)
- Er knyttet til sentralt definerte opptellingskategorier som angir hvilke opptellingskategorier kommunene kan velge
- Flagg for papirstemmegivning og elektronisk stemmegivning
- Start og sluttdato for ulike stemmegivningsperioder (forhånd utenriks, forhånd innenriks, etc)
- Skal knyttes til manntalls- og velger-tabell

Merknad: Den (mest) praktiske funksjonen for valggruppe er at man kan bruke ett manntall for flere valg, for eksempel kommunestyre- og fylkestingsvalg.

## Valg

Her settes regler opp for valget, som om det er tillatt med personstemmer, slengere og strykninger. Hvis det er sametingsvalg, angis dette her (ved bruk av flagg "single area" og "penultimate recount").

- Områdenivå for valget (kommune eller fylke)
- "Single area"-flagg (vanlige valg er "single area", sametingsvalg er det ikke)
- Renummerering, strykning, tilføyelse (writein)
- Aldersgrense
- Flagg for om kandidater må ha stemmerett
- Diverse regler for valgoppgjør (blant annet "first divisor in Sainte-Lague")
- Flagg som angir hvem som utfører andre telling (brukes i Sametingsvalget)

## Valgdistrikt/Delvalg (Contest)

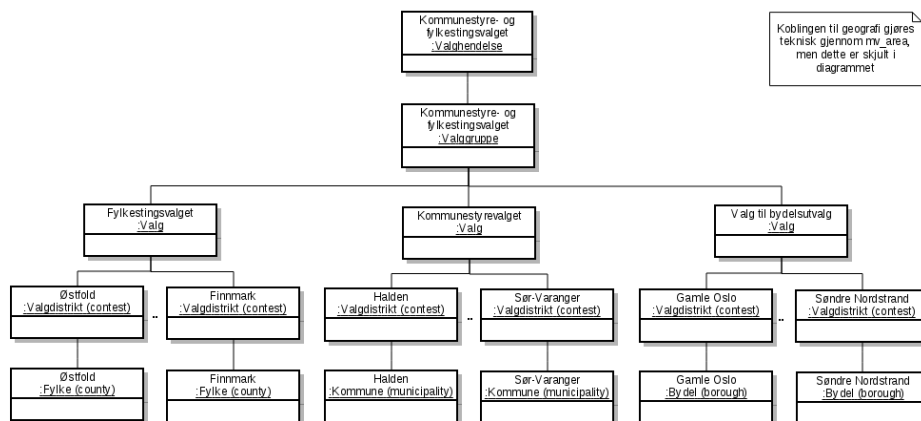
Her konfigureres geografien. I tillegg kan det gjøres spesialtilpasninger på noe av det som er konfigurert høyere opp i valghierarkiet.

- Knyttet til tellinger (via "contest report")
- Knyttet til valgoppgjør (settlement)
- Kan overskrive en del instillinger på valg, f.eks aldersgrense
- Maks antall kandidater, stemmegivninger, etc
- Flagg som angir hvem som utfører andre telling (kan overstyre verdien satt på valgnivå)
- Knyttet til stemmeseddel/partiliste

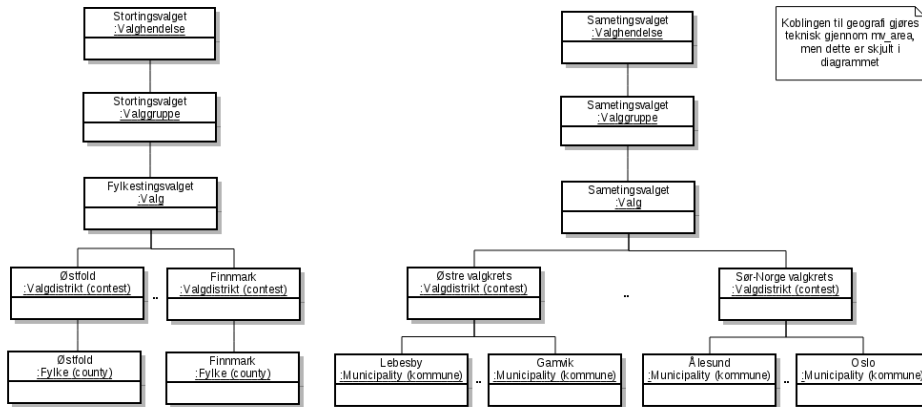
## Manntall

- Velger kan være i flere manntall, "gjenbruk" av persondata
- En valggruppe har ett manntall
- Innføres for å støtte at vi kan ha Fylkestingsvalg, Kommunestyrevalg og Bydelsvalg i samme Valghendelse

Objektdiagrammene under viser hvordan valghierarkiet konfigureres for de forskjellige typene valg som systemet støtter:

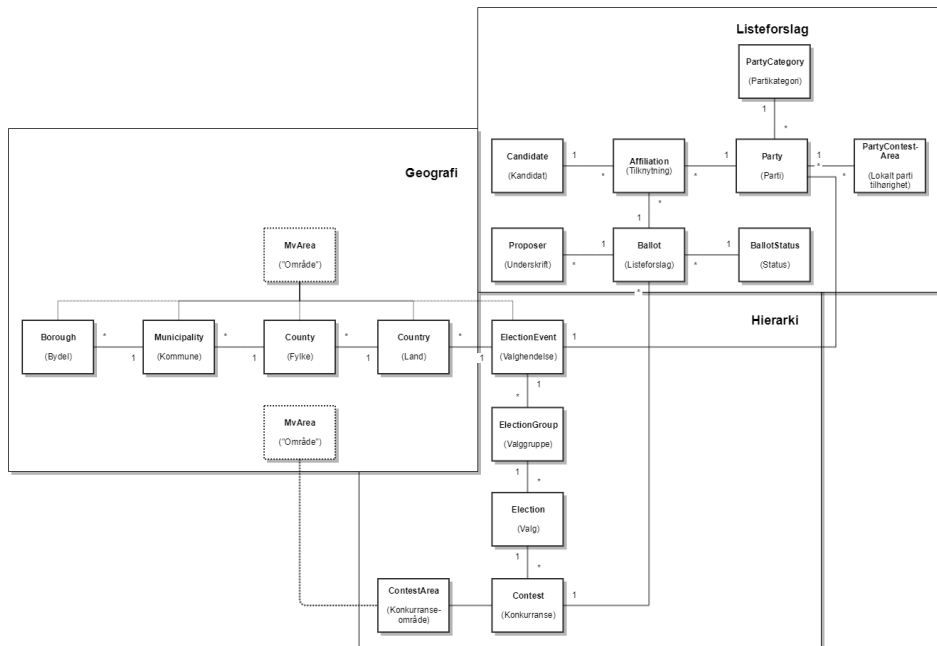






## Domenemodell - listeforslag

Skisse som illustrerer de sentrale konseptene i listeforslagsdomenet:



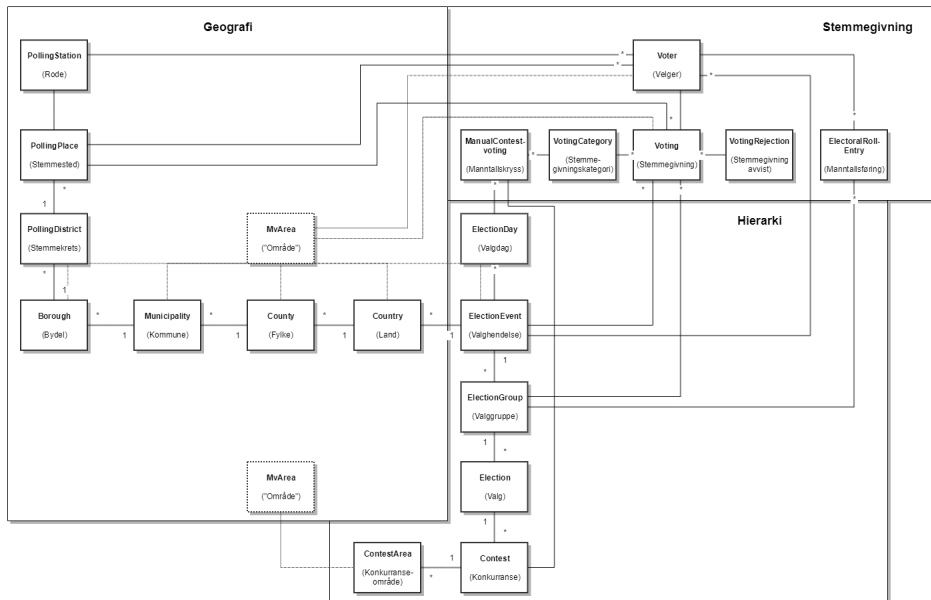
Et listeforslag inneholder følgende elementer:

- Listeforslaget som vil være gjenstand for behandling og eventuell godkjenning
- En forslagsstiller, dette er den eller de som utarbeider listeforslaget - vanligvis et politisk parti
- Listeforslaget kobler sammen parti og kandidater som potensielt velges inn i det politiske organet som det skal gjennomføres valg for
- I tillegg har et listeforslag en kobling til et gitt valgdistrikt, f.eks en fylkeskommune dersom det er stortingsvalg.

Merk at saksbehandlingen av listeforslagene ikke gjøres i EVA Admin, dette er en prosess som foregår i andre systemer.

## Domenemodell - stemmegivning

Skisse som illustrerer de sentrale konseptene i stemmegivningsdomenet



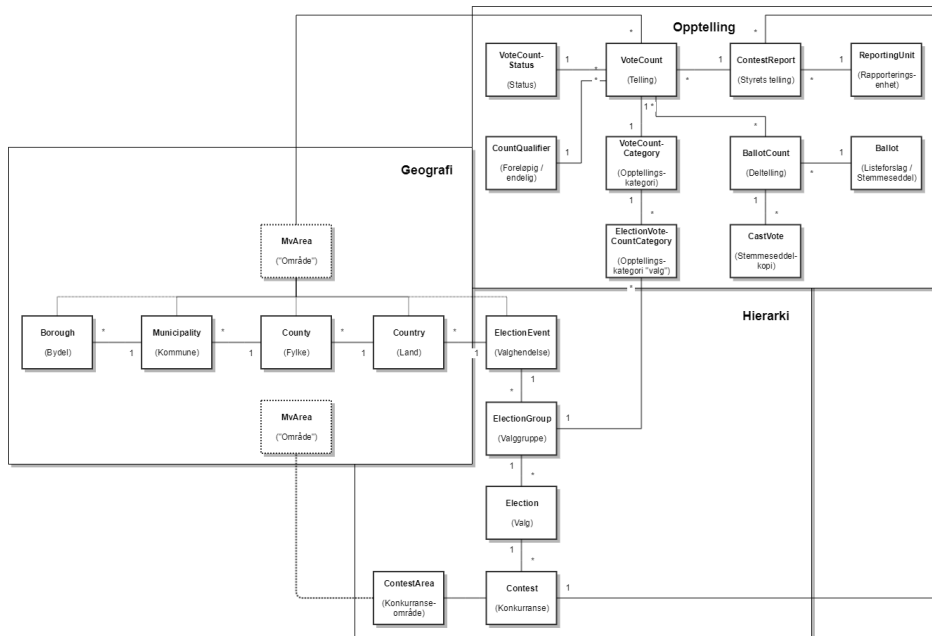
En stemmegivning representerer *stemmegivningshandlingen*, det vil si handlingen som utføres når stemme avgis, ikke stemmeseddelen.

En stemmegivning involverer følgende:

- En stemmegiver, det vil si den personen som skal stemme
- Et sted for avgivelse av stemmegivningen, dette vil typisk være et valglokale (stemmested), med eller uten roder (A-P, P-W osv)
- Når stemme avgis oppstår en stemmegivning knyttet til stemmegiver, stemmegivningen kategoriseres, f.eks som en valgtingsstemme
- Potensielt kan en stemmegivning avvises dersom valgloven tilsier at dette skal gjøres
- Dersom det ikke benyttes elektronisk avkryssing i manntall brukes ikke mesteparten av dette domenet i EVA Admin, alt skjer da på papir. I forbindelse med opptelling skrives i så fall antall manntallskryss for de forskjellige kategoriene inn i EVA Admin (ManualContestVoting) som aggregerte tall.

## Domenemodell - opptelling

Domenemodellen for subdomenet opptelling er vist i figuren under.



## "Styrets telling"/ contest\_report

- Knyttet til valgdistrikt (contest). Et valgdistrikt har mange "Styrets telling". Rapporteringenheter på ulike nivå (Valgstyre, Distriktsvalgsstyre) lager en "Styrets telling" som er resultatet for et valgdistrikt
- Har mange tellinger fordelt på kategori (forhånd, valgting), kvalifikator (urne, første, andre) og status (godkjent, forkastet, til godkjenning..)
- Inneholder telleinformasjon som senere skal i møteboka

## Telling

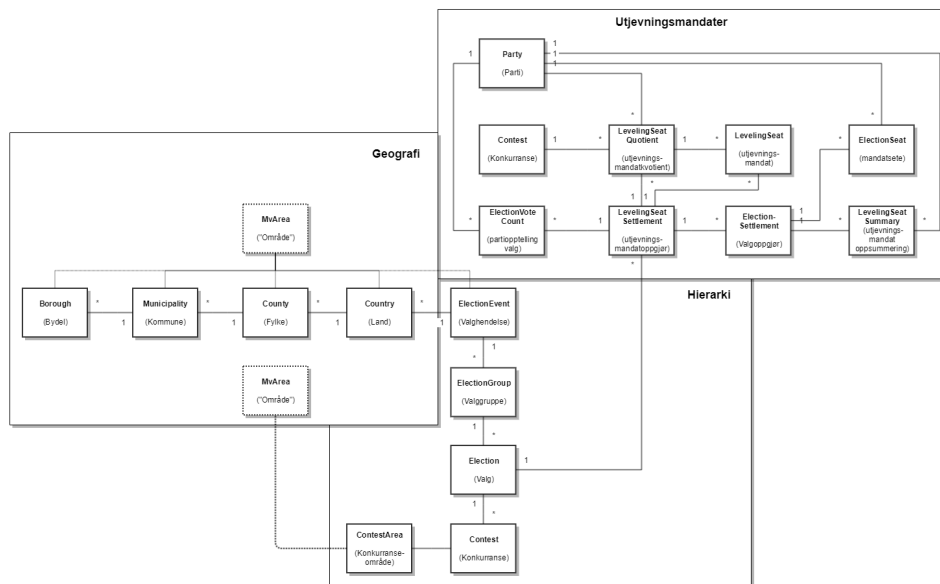
- Representerer en telling for en gitt kategori på et visst nivå (kvalifikator: første, andre)
- Er knyttet til en stemmekrets. I valgloven av 2024 skal tellinger gjøres per valglokale, men dette løses ved at alle kretser hvor det skal gjøres opptellinger har et 1:1-forhold med valglokaler.
- Telling har mange deltellinger

## Deltelling

- Dette er en telling av stemmene for et gitt parti eller for et gitt alternativ i en folkeavstemning.
- Har mange "cast\_vote" som kan ses på som en kopi av stemmeseddelen

## Domenemodell - valgoppgjør - utjevningsmandater

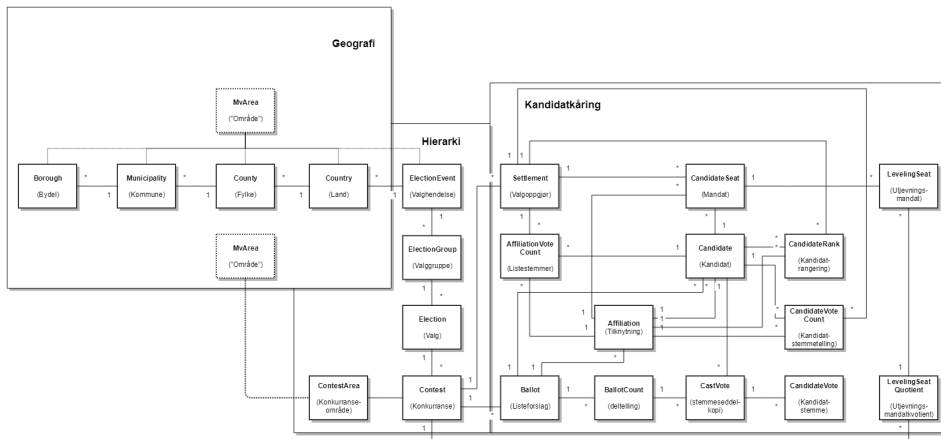
Skisse som illustrerer domenet for valgoppgjør for utjevningsmandater



Se beskrivelse av funksjonell modul for utjevningsmandater for forklaring til domenenmodellen.

## Domenemodell - valgoppgjør - kandidatføring

Skissen illustrerer fomenet for valgoppgjør - kandidatføring



Se funksjonell modul for mandatberegning og kandidat kåring for forklaring til domenemodellen.

## Arkitektur

### Innledning

EVA Admin ble opprinnelig utviklet som administrasjonsmodul for e-valgsforsøkene i 2011 av Ergo Group (nåværende TietoEVRY). I 2013 overtok det offentlige forvaltning og videreutvikling av produktet (ved KMD).

Da Valgdirektoratet ble opprettet 01.01.2016 overtok direktoratet forvaltningen av produktet.

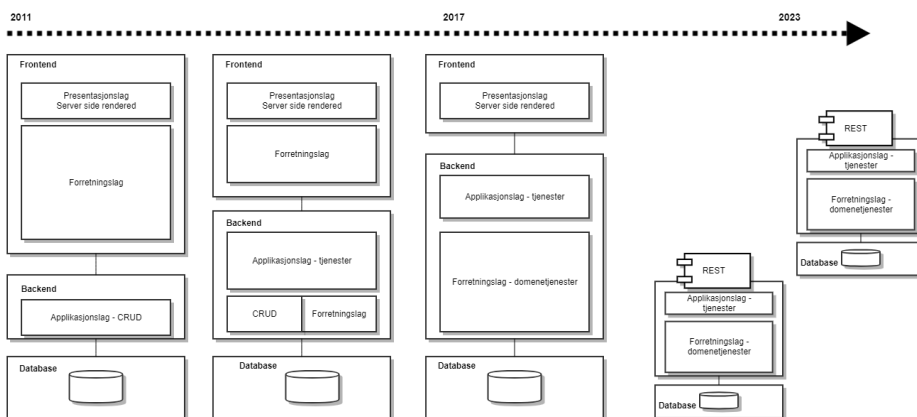
Historikken dikterer EVA Admins arkitektur og det arbeides kontinuerlig med å tilpasse systemet til et nyere målbilde avstemt med Difis overordnede arkitekturprinsipper.

### EVA Admins utvikling

EVA Admin ble i utgangspunktet levert som en lagdelt applikasjon med frontend og backend. Backend gjorde basale CRUD operasjoner mot databasen, mens frontend inneholdt presentasjonslag, forretningslag og applikasjonslag.

Det er en pågående prosess å styrke backend med applikasjons- og forretningslogikk, mens frontend spisses mot presentasjon.

Skissen er en forenklet framstilling av utviklingen av EVA Admin mot et mer hensiktsmessig målbilde.



### Modularisering

Ny funksjonalitet utvikles som moduler snarere enn å implementeres i eksisterende applikasjon og aksesseres via REST-ressurser. Inntil videre er nye moduler laget med samme teknologiportefølje som EVA Admin.

## Konfigurasjonsstyrt forretningsmodell

EVA Admin er i stor grad et konfigurasjonsdrevet system. En forretningsprosess består av en gitt gjennomføring av operasjoner der detaljene og valideringene i forretningsprosessen i stor grad er definert gjennom konfigurasjon.

Et konfigurasjonsstyrt system har noen fordeler:

- Systemet er ekstremt fleksibelt med henblikk på endringer i variable som er en del av valggjennomføringen
- Forretningsprosesser inneholder ingen informasjon om hvilken type valg som gjennomføres - dette gjør også at tilfeller av "hardkoding" og spesialregler er på et minimum

Noen ulemper er at:

- Det er en viss terskel for å sette seg inn i hvordan systemet fungerer ifbm. forvaltning og videreutvikling
- Det finnes få referanser i kildekoden til hva som utgjør forretningsprosessene for en gitt type valg,

## Sentral konfigurasjon

Valgdirektoratet gjennomfører den sentrale konfigurasjonen av EVA Admin i forbindelse med produksjonssetting av systemet før valgåret. Dette er typisk:

- Instansiering av valghendelse der global konfigurasjon for alle underliggende valggrupper defineres, herunder:
  - Geografi som skal brukes i underliggende valggrupper/valg
- Instansiering av valggruppe, der global konfigurasjon for alle underliggende valg defineres som f.eks:
  - Om elektronisk manntall skal kunne benyttes
  - Hvorvidt forhåndsstemmer skal avgis rett i urne eller ikke
- Instansiering av valg der global konfigurasjon per valg defineres som f.eks:
  - Hvilke geografiske områder som skal utgjøre valgdistrikt
  - Hvorvidt listekandidater må være manntallsførte i valgdistriktet eller ikke
  - Hvilke personvalgregler som gjelder for valget
  - Hvilke regler for listeforslag som gjelder
  - Variabler for valgoppgjør (herunder antall utjevningsmandater om relevant)

I sum utgjør denne konfigurasjonen hovedreglene for valggjennomføringen og informasjon om hvorvidt valget som skal gjennomføres er et stortingsvalg, kommune- og fylkestingsvalg eller et sametingsvalg. I praksis vil denne konfigurasjonen gjelde valghierarkiet, samt knytningen mellom dette og valggeografien via contest (valgdistrikt) i domenemodellen for grunnlagsdata.

## Lokal konfigurasjon

Innenfor rammene av den sentrale konfigurasjonen definerer kommuner og fylkeskommuner den lokale konfigurasjonen som er reglestett som kan anvendes innenfor rammene av den sentral konfigurasjonen som f.eks:

- Målform
- Lokale krav til listeforslag
- Hvilke optellingsmåter som skal benyttes

I praksis vil denne konfigurasjonen gjelde hovedreglene for den lokale valggjennomføringen der kommuner og fylkeskommuner definerer de deler av valggjennomføringen de har råderett over, og ansvar for. I praksis vil denne konfigurasjonen gjelde geografi i domenemodellen under.

Se forøvrig kapittel om funksjonelle moduler: [Konfigurasjonsmodul](#)

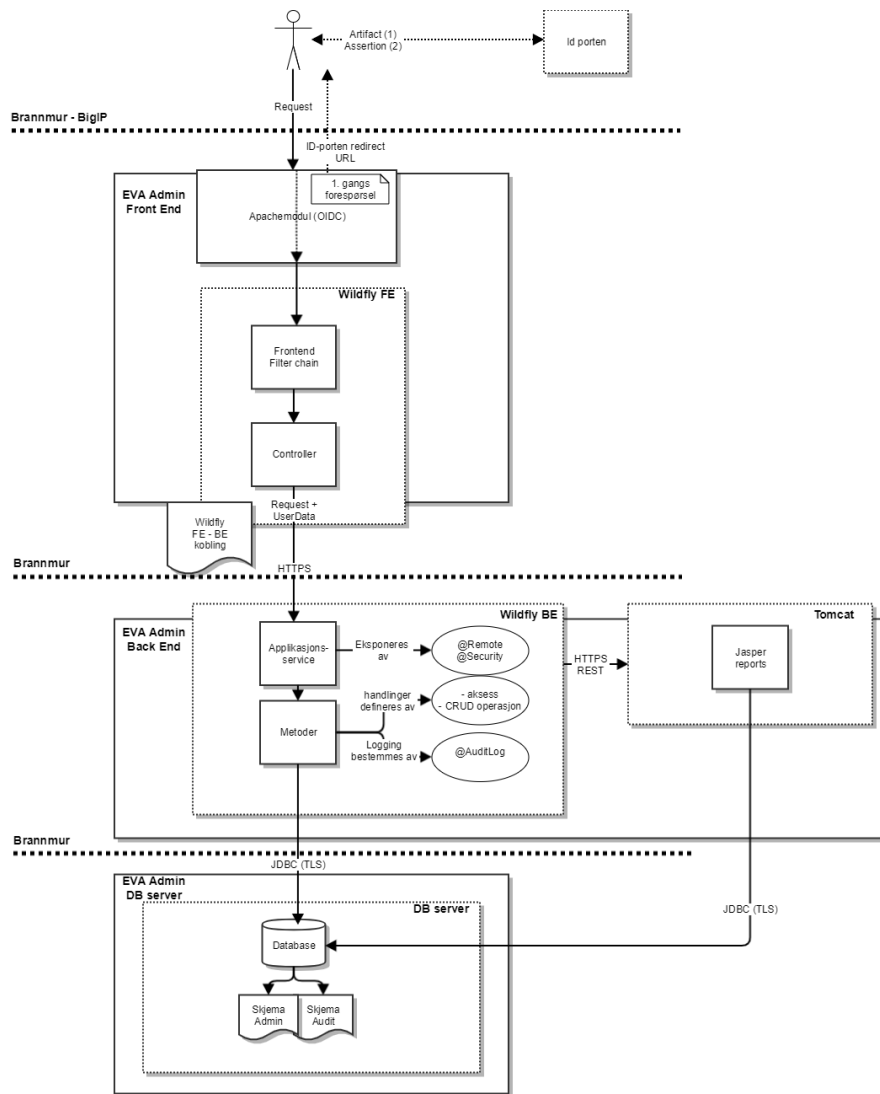
## Applikasjonsarkitektur

EVA Admin applikasjonen er en standard 3 lags Java EE webapplikasjon som driftes sentralt av Valgdirektoratet.

Frontend og backend er koblet i "par", skalering oppnås ved å opprette nodepar av frontend og backends, alle koblet mot samme databaseserver. Lastbalansering gjøres av lokal lastbalanser.

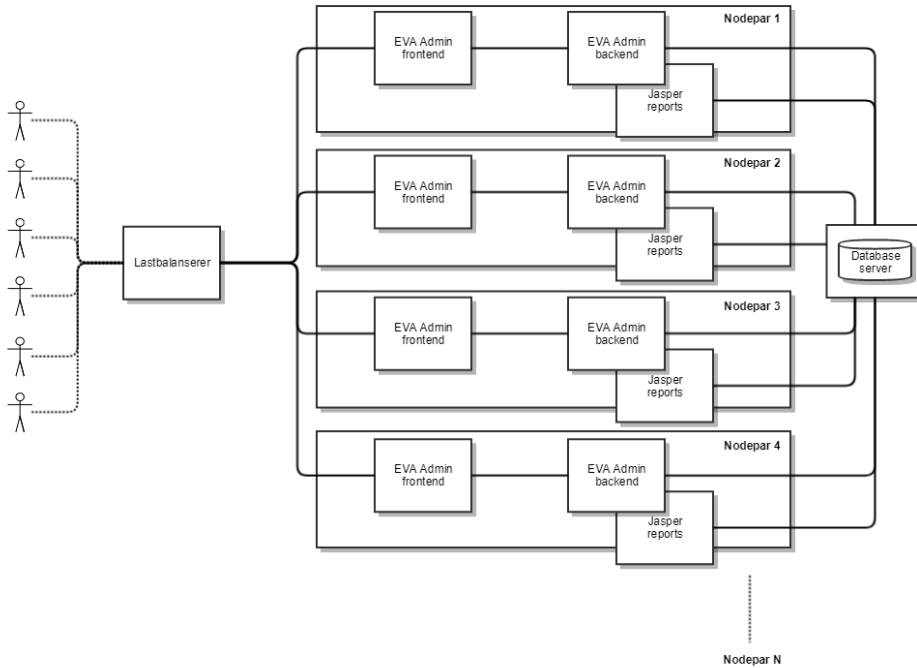
Rapporter genereres med JasperSoft rapportserver. Uavhengig av antall nodepar av frontend og backend finnes det kun én rapportserver.

*Diagrammet gir en forenklet framstilling av lagdelingen av EVA Admin - Frontend -Backend -Database samt sentrale sikkerhetsmekanismer*



## Skalering

Skalering oppnås som beskrevet ved å skalere nodepar av frontend- og backendnoder, lasten balanseres av 3. parts lastbalanserer

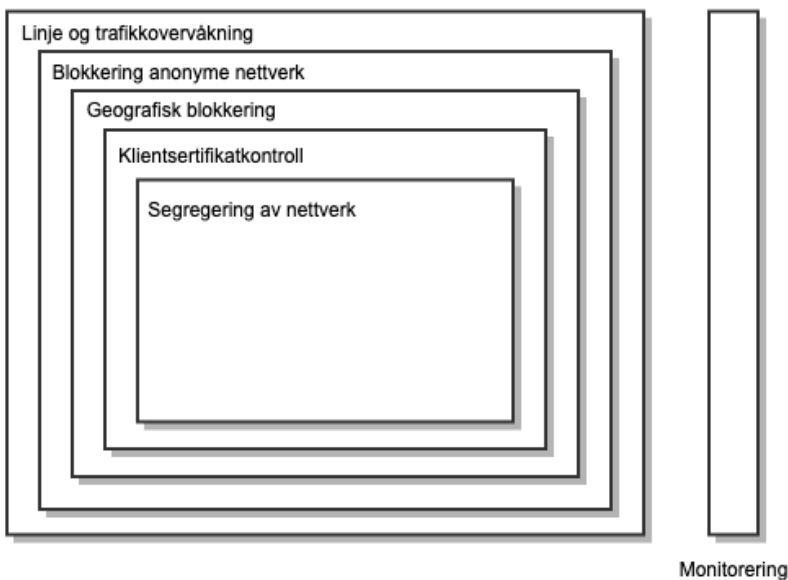


## Sikring av infrastruktur

### Tilgangsbegrensning

Tilgang til Valgdirektoratets sentral driftede IT-applikasjoner er regulert gjennom forskjellige mekanismer for å snevre inn hvem som gis tilgang.

Skisse som forenklet illustrerer tilgangsbegrensning gjennom infrastruktur:



### Tilgangskontroll

- Geografisk:
  - Tilgang fra anonyme nettverk forsøkes blokkert
  - Tilgang fra andre nasjoner er blokkert
- Personlig
  - Tilgang der man ikke har klientsertifikat utstedt fra valgdirektoratet er blokkert
- Logisk:
  - Tilgang til frontend servere er beskyttet av brannmur / i VLAN
  - Tilgang fra frontend til backend server er beskyttet av brannmur / i VLAN
  - Tilgang fra backend til database er beskyttet av brannmur i/VLAN
  - Frontend/backend er koblet i nodepar og isolert i VLAN

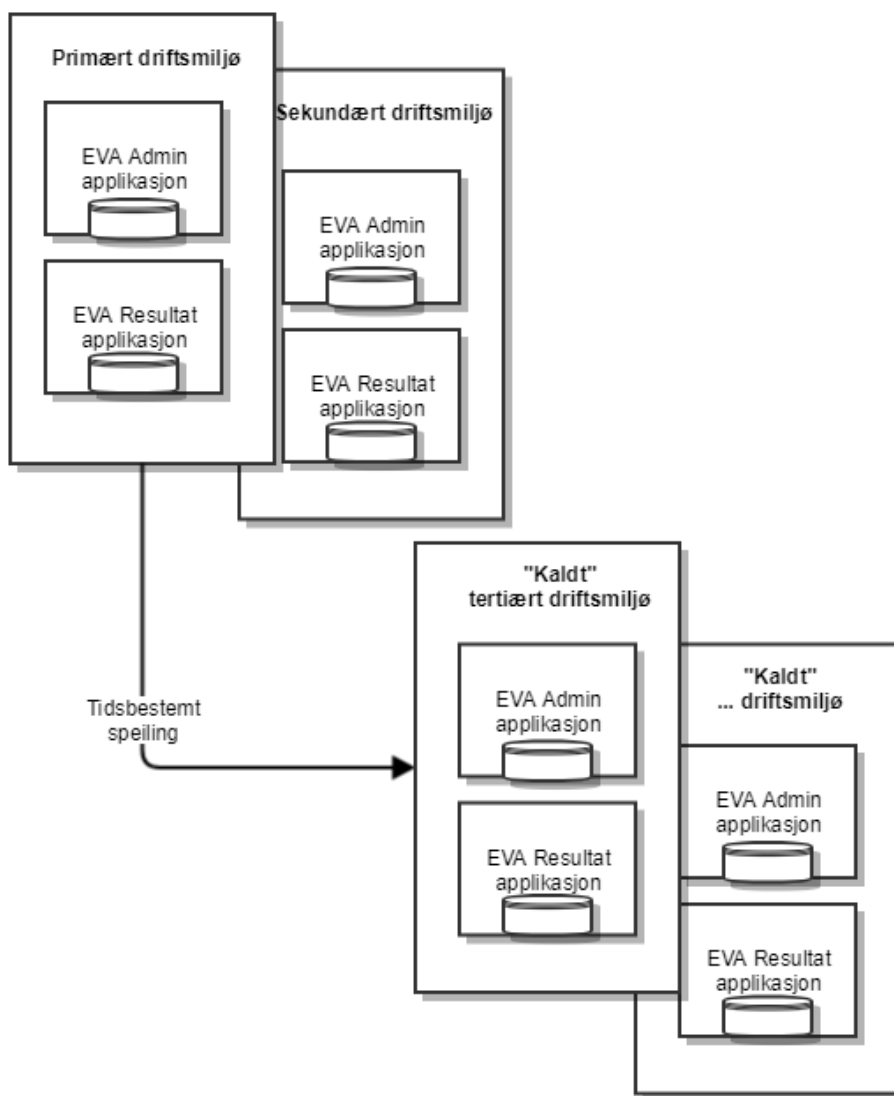
## Monitorering

Alle ledd monitoreres med varslinger ved avvik

## Redundans

Valgdirektoratets sentralt driftede IT-applikasjoner er sikret mot brudd i tjenestene med redundans for driftsmiljøene for disse applikasjonen.

Skissen illustrerer prinsippet for redundans av driftsmiljø



I en krise eller beredskapssituasjon hvor det primære driftsmiljøet ikke lenger fungerer vil man kunne ta i bruk det sekundære driftsmiljøet i løpet av svært kort tid. Miljøet speiler det primære driftsmiljøet i sanntid og således umiddelbart er klart til bruk uten datatap.

Om det sekundære driftsmiljøet ikke lenger fungerer vil man kunne ta i bruk det tertjære driftsmiljøet etter noe lenger tid. At miljøet er "kaldt" betyr at det ikke speiler produksjonsmiljøet i sanntid og ikke umiddelbart er klart til bruk.

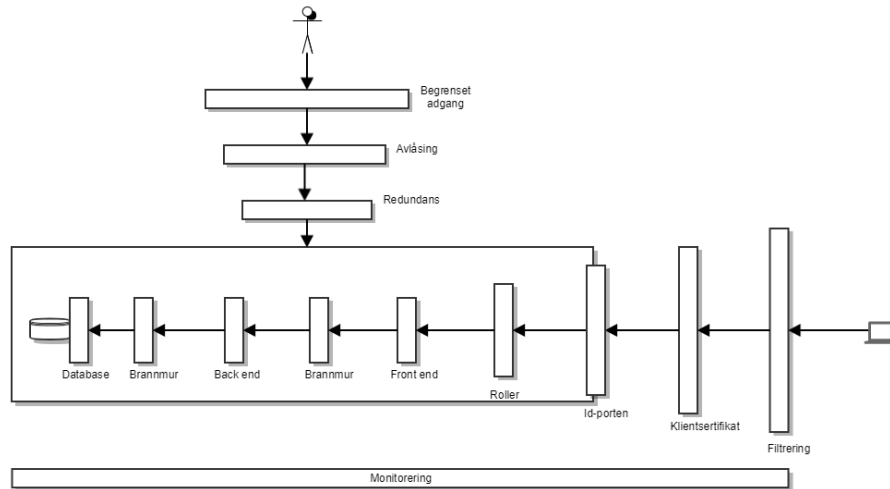


## Lagdeling - EVA Admin

EVA Admin er lagdelt på flere nivåer for på den måten å skape sikkerhetsbarrierer. Prinsipielt skal ikke brudd på en enkelt barriere føre til at systemet ligger åpent.

EVA Admin er beskyttet gjennom infrastruktur og gjennom hvordan applikasjonen er bygget opp. De forskjellige mekanismene er omtalt mer detaljert i etterfølgende kapiteler.

Skissen viser en forenklet framstilling av barrierene som beskytter EVA Admin



## Fysisk tilgang

En bruker som forsøker å aksessere EVA Admins driftssenter møter et sett av barrierer:

- Driftshallers beliggenhet er ikke offentlig kjent
- Lokaler er avlåst med brann og innbruddsalarm
- Kun personer med særskilt autorisasjon har tilgang til driftshaller hvor systemet kjører
- Driftsmiljø er satt opp med flere nivåer av redundans

## Logisk tilgang

En bruker som forsøker å aksessere EVA Admin møter et sett av barrierer:

- Filtrering - trafikk fra uønskede domener filtreres vekk i brannmur
- Bruker må ha klientsertifikat utstedt av valgdirktoratet for å få tilgang til EVA Admin innloggingsside
- Bruker må autentisere seg gjennom ID-portens autentiseringstjeneste
- Bruker må ha en rolle i applikasjonen for å få tilgang til funksjoner i EVA Admin
- Applikasjonen er lagdelt:
  - Frontend snakker med back end via intern brannmur
  - Backend snakker med databasen via intern brannmur
  - Kommunikasjon mellom lagene i applikasjonen er kryptert (SSL)
- Nodene er isolert, en frontendnode kan kun snakke med sin makker backend node
- Alle ledd i infrastruktur og applikasjon monitoreres

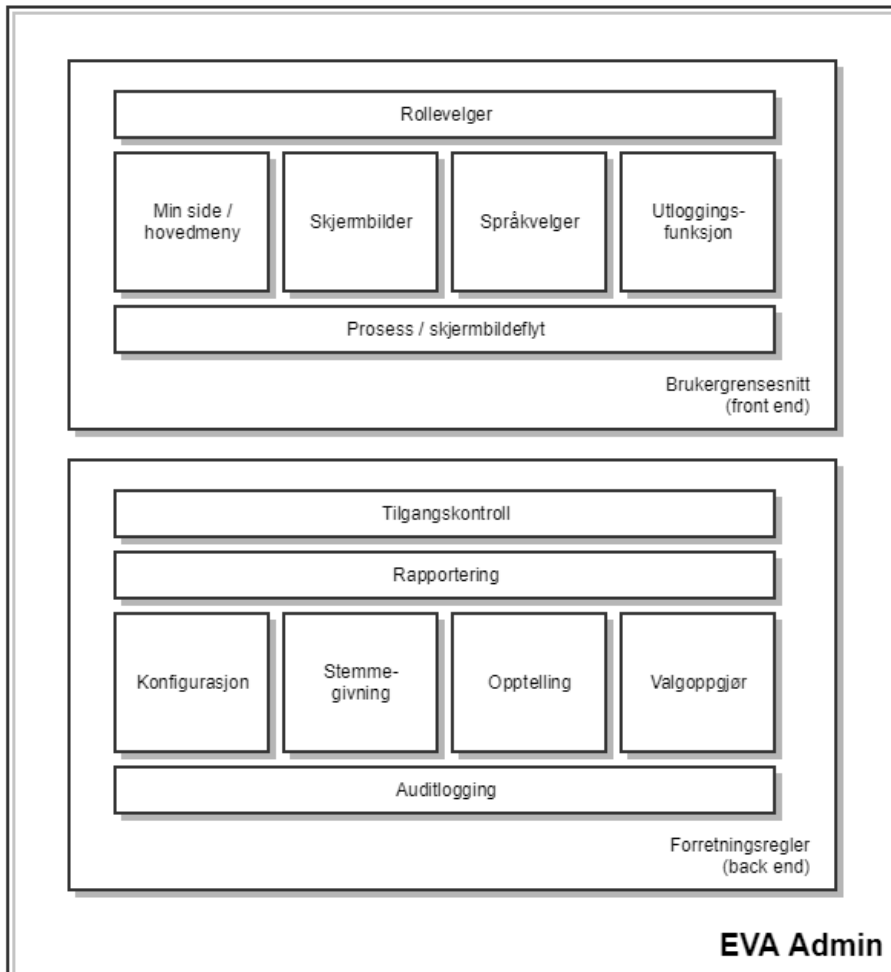
## Moduler i EVA Admin

Som skissert er EVA Admin delt inn i moduler fordelt på frontend og backend, forretningsprosessene ivaretas av:

- Rollevelger, skjermbilder og prosess/skjermbildeflyt i frontend, i samspill med
- Forretningsstjenestene for Konfigurasjon/grunnlagsdata, stemmegivning, opptelling og/eller valgoppgjør, samt
- Rapportering for rapporter knyttet til forretningsområdene.

Brukertilgang styres av rollevelger og tilgangskontroll, logging håndteres av Auditloggmodulen.

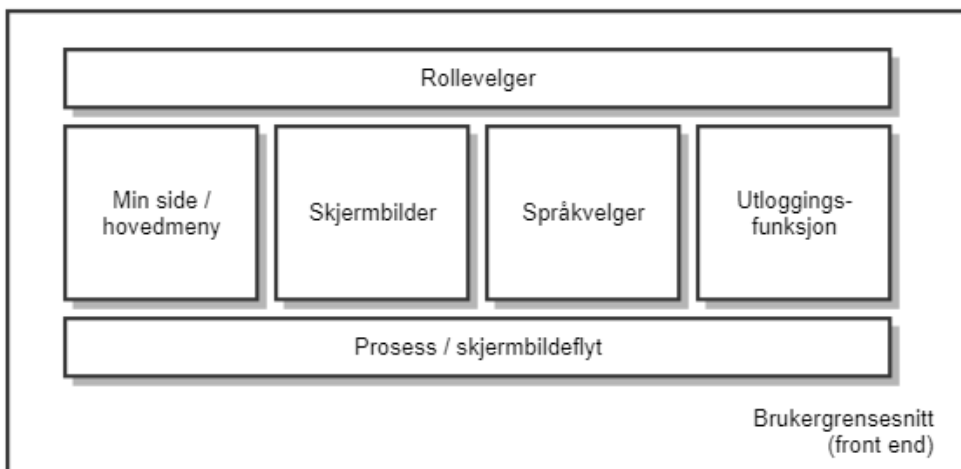
## Skisse over moduler i EVA Admin



## Frontend - presentasjonsmodul

EVA Admin frontend er modulen der brukerinteraksjon skjer gjennom web-grensesnittet frontend tilbyr.

Skisse over frontend modulen



Frontend modulen utøver følgende funksjoner:

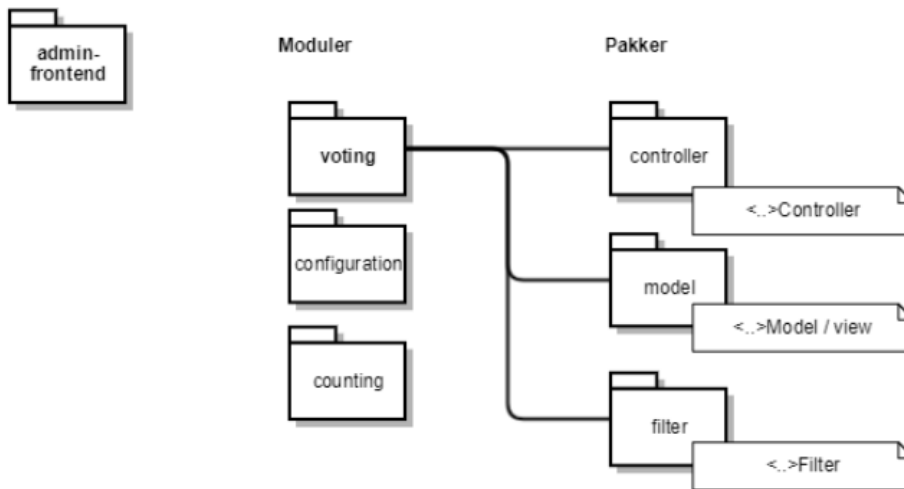
- Mottar og administrerer id-porten autentiseringsresultat
- Administrerer sesjoner for innlogget bruker
- Administrerer rollevalg for innlogget bruker
- Administrerer språkvalg
- Presenterer brukergrensesnittet for forretningsprosessene i EVA Admin
- Håndterer skjermbytte i forretningsprosesser
- Kommuniserer rollevalg og språkvalg til backend for innloggede brukere
- Kommuniserer med backend for data og forretningsprosessregler

EVA Admin frontend inneholder funksjonalitet for brukerinteraksjon gjennom EVA Admins brukergrensesnitt, dette gjøres gjennom et sett av komponenter for å bygge og presentere brukergrensesnittet.

## Javaklasser

EVA Admin frontend er bygget opp basert på standard Java EE komponenter og JSF og koden er organisert etter prinsipper for domenedrevet design, samt Model View Controller (MVC) konseptet.

Skissen illustrerer oppbygning av kode som følger nytt målbilde:



Primærinndelingen skjer ved inndeling i domene, ikke funksjon, for hvert domene organiseres koden ihht. ansvarsområder (funksjon):

- Controller - betjener view og modell
- Model/view - data og brukergrensesnitt
- Filter - filtrerer og håndhever sikkerhet og tilgang

## JSF Ajax (Java Server Faces)

Hensikten med denne beskrivelsen er ikke å gi en utførlig beskrivelse av teknologien, men belyse at/hvordan den brukes i applikasjonen.

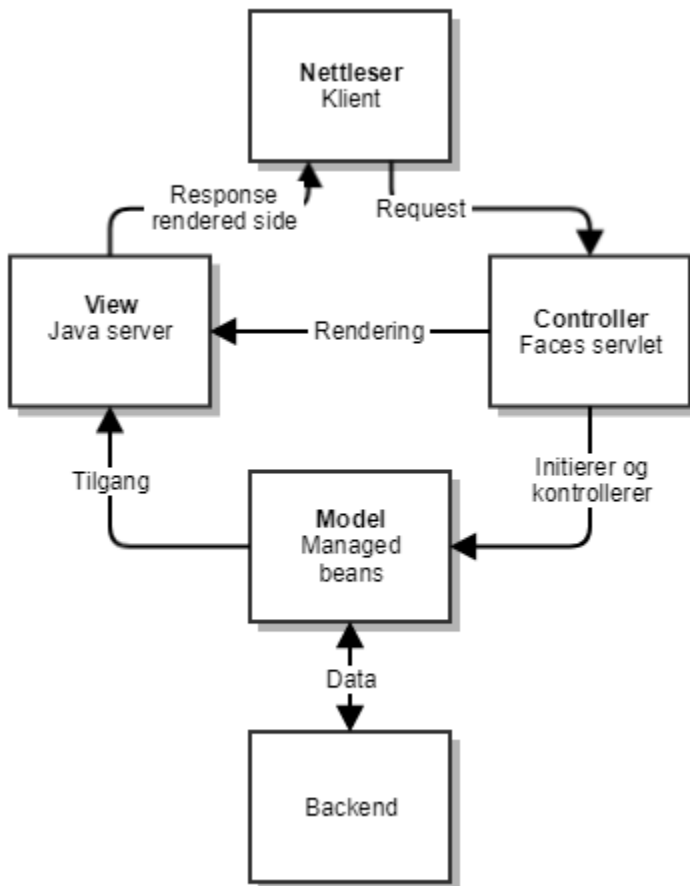
JSF er i motsetning til flere andre mer moderne frontendrammeverk serverbasert. Frontend serverer en ferdig oppsatt side til nettleseren og bruker i liten grad nettleseren for å bygge opp presentasjonen av nettsiden, såkalt "server side rendering".

Ajax (Asynchronous JavaScript and XML) gjør det mulig å oppdatere siden med innhold uten at siden lastes på nytt.

JSF bruker et designpattern kalt Model-view-controller (MVC), i korthet gå dette ut på at funksjonaliteten er delt inn i 3 deler:

- Model - data
- View - brukergrensesnitt
- Controller - bindeledd mellom view og model som initierer prosesser basert på input/output

Skissen viser en forenkling av hvordan brukerinteraksjon håndteres i JSF i et MVC regime



## Primefaces

*Hensikten med denne beskrivelsen er ikke å gi en utførlig beskrivelse av teknologien, men belyse at/hvordan den brukes i applikasjonen.*

Primefaces brukes i tillegg til standard JSF komponenter for å berike brukergrensesnittet med UI-komponenter der det er behov for det. Det i tillegg brukt Javascript for å berike funksjonalitet i skjermbilder der funksjonalitet må tilpasses utover der som tilbys innenfor JSF/Primefaces.

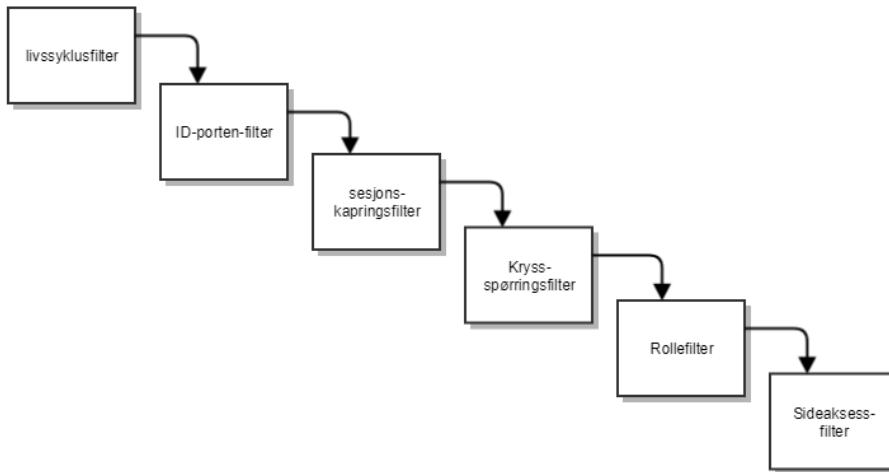
## Applikasjonsserver

EVA Admin kjører i Wildfly som implementerer Java EE spesifikasjonen.

## Sikkerhets- og sesjonshåndtering

EVA Admin frontendmodulen er konfigurert med en såkalt "filter chain" for å håndtere sikkerhet og sesjoner.

*Figuren er en forenklet illustrasjon av filterkjeden som er konfigurert i EVA Admin*



Overordnet har filtrene følgende oppgaver:

Filter	Oppgave
<b>Livssyklusfilter</b>	Filterets oppgave er å sjekke at systemet er klart til bruk. Typiske sjekker er at alle sertifikater er på plass og at system passord er satt.
<b>ID-porten-filter</b>	Filterets oppgave er å håndtere OIDC innlogging (via ID porten). OIDC request håndteres og session opprettes. I tillegg håndterer filteret bytte av rolle for allerede innlogget bruker.
<b>Sesjonskapringsfilter</b>	Filterets oppgave er å sjekke at brukerens identitet stemmer med applikasjonsserverens sesjon og OIDC-sesjonen
<b>Krysspørringsfilter</b>	Filterets oppgave er å sikre at alle HTTP POST requester har med <a href="#">CSRF</a> token. Requester som mangler dette vil få returnert HTTP 400 (BAD REQUEST).
<b>Rollefilter</b>	Filterets oppgave er å håndtere valg eller bytte av rolle for nyinnlogget eller allerede innlogget bruker.
<b>Sideaksessfilter</b>	Filterets oppgave er å hindre tilgang til sider brukeren, gitt sin rolle, ikke har tilgang til.

## Autentisering - Id-porten

Brukere i EVA Admin må autentiseres, dvs. at brukerens identitet må kontrolleres.

Dette gjøres i ID-porten. På denne måten trenger ikke EVA Admin å forvalte brukeridentiteter, men lar i stedet brukerne bruke sin egen, elektroniske ID som f.eks. MinID, BankID eller Buypass.

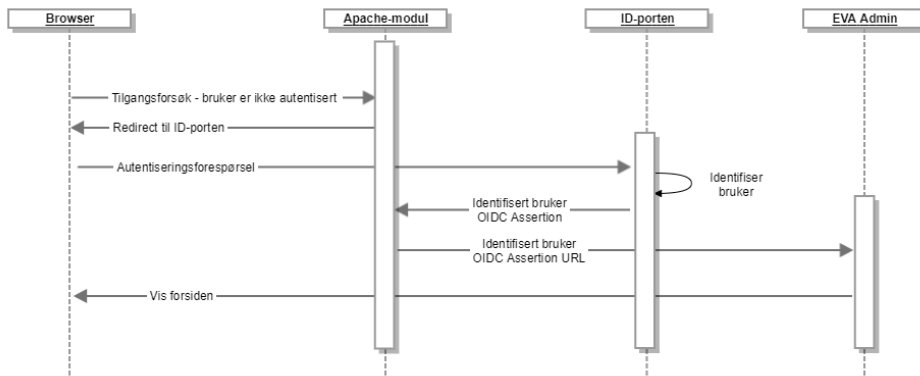
### Protokoll

Autentiseringstjenesten bruker OpenID Connect (OIDC)-protokollen mot ID-porten.

### ID-porten dialog

Apache-modulen mod\_auth\_ldap brukes som OIDC-megler mot ID-porten.

*Forenklet skisse over dialog med ID-porten*



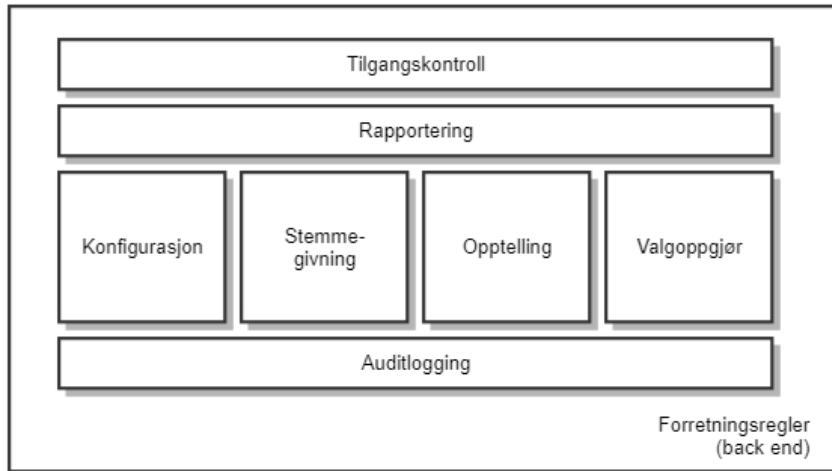
- Bruker forespør EVA Admin innloggingside frontend. Apachemodul videresender forespørslene til ID-porten
- ID-porten identifiserer og autentiserer bruker
- Frontend Apachemodul videresender svar på autentiseringsforespørsel til EVA Admins rollevalgside hvor alle brukerens roller i EVA Admin kan velges mellom før man videre redirigeres til "min side".

## Backend - forretningsprosessmodul

### Backendprosesser

EVA Admin backend er modulen der forretningsregler håndheves og data formidles mellom frontend og database.

Skisse over backendmodulen



Backend modulen utøver følgende funksjoner:

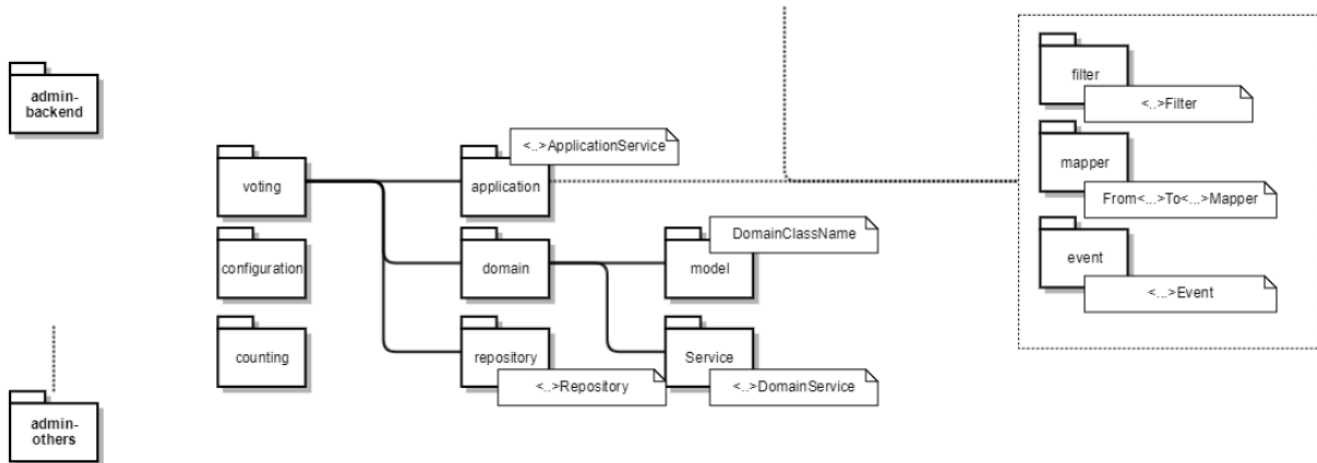
- Håndhever tilgangskontroll til data i henhold til rollebasert tilgangskontroll
- Utfører forretningsprosesser på forespørsler fra frontend
- Håndhever forretningsregler for uthenting og oppdatering av data fra frontend mot databasen
- Formidler forespørsler om data fra frontend til databasen
- Formidler forespørsler om oppdatering av data fra frontend til databasen
- Utfører auditlogging av forretningsprosesser
- Formidler forespørsler til og resultater fra rapporteringsmodul til frontend

EVA Admin backend inneholder funksjonalitet for å betjene frontends behov for data og forretningsprosesser.

### Javaklasser

EVA Admin er bygget opp basert på standard Java EE komponenter og koden er organisert etter prinsipper fra domenedrevet design.

Skissen illustrerer oppbygging av kode som følger nytt målbylde:



Primærinndelingen skjer ved inndeling etter domene, ikke funksjon, for hver domene organiseres koden iht. ansvarsområder (funksjon):

- ApplicationServices: Tjenester som er domeneuavhengige og orkestrerer prosessflyt og infrastruktur
- DomainClasses: Klasser som representerer domenet med ansvar for forretningsregler for seg selv
- DomainServices: Tjenester som er knyttet til domene og der kjennskap til domene går utover eller på tvers av flere DomainClasses
- Repository: Oppslags- og persistenshåndtering

## Context og Dependency injection - CDI

EVA Admin bruker applikasjonsserveren Wildfly sitt CDI rammeverk for context og dependency injection, Weld.

CDI er ikke ytterligere forklart eller beskrevet i dokumentasjonen da bruk av CDI anses som industristandard.

## Java Persistence API - Hibernate

EVA Admin bruker Hibernate sin implementasjon av spesifikasjonen for Java Persistence API (JPA) for relasjonsmapping (ORM) og persistens. Hibernate er ikke ytterligere forklart eller beskrevet i dokumentasjonen da bruk av Hibernate (eller liknende JPA/ORM-rammeverk) anses som industristandard.

## Applikasjonsserver

EVA Admin kjører i Wildfly som implementerer Java EE spesifikasjonen.

## Autorisering - Rollebasert tilgangskontroll

### Beslutningsmodell - rollebasert tilgangskontroll

Tilgang til data er fordelt på to ansvarsområder:

- En modul / funksjon *håndhever* tilgangskontrollen
- En modul / funksjon *avgjør* hvorvidt bruker har tilgang

### Tilgangshåndhevelse

Håndhevingsfunksjonen er ansvarlig for å fange forespørsler fra bruker og håndheve avgjørelsen som tas av tilgangsavgjørelsesfunksjonen.

### Tilgangsavgjørelse

Tilgangsavgjørelsesfunksjonen er ansvarlig for å evaluere hvorvidt brukeren har tilgang til forespurte funksjon og tilhørende data. Evalueringer gjøres i to steg:

### Funksjonskallavgjørelse

Tilgangsavgjørelsesfunksjonen evaluerer hvorvidt brukeren har tillatelse til å forespørre en gitt funksjon. Denne avgjørelsen baserer seg på hvilke sikrede objekter som kreves for å utføre funksjonen og hvilke sikrede objekter som er tildelt brukeren gjennom rollen.

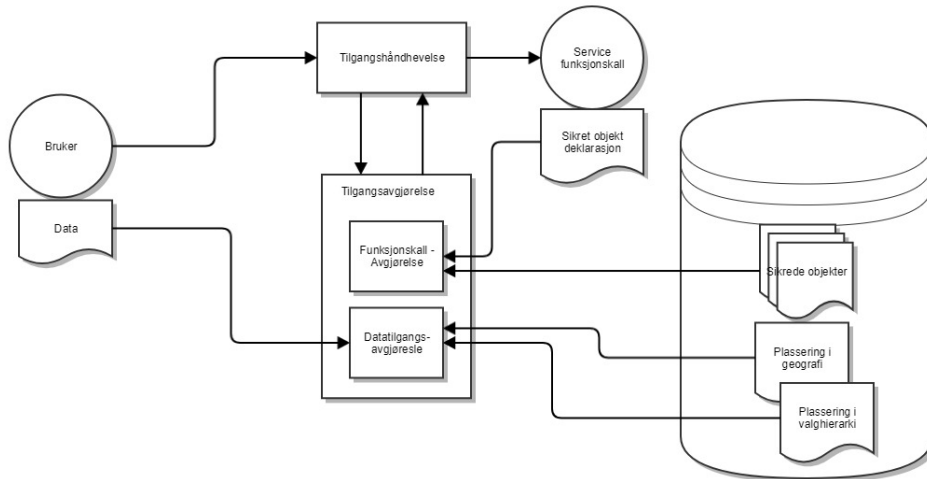
Brukeren gis tilgang til funksjonen dersom brukeren er tildelt påkrevde sikrede objekter.

## Datatilgangsavgjørelse

Funksjonen for avgjørelse om datatilgang evaluerer hvorvidt bruker har tillatelse til å utføre en gitt funksjon *på de spesifikke data som forespørres*. Avgjørelsen baserer seg på om forespurte data er knyttet til samme geografisk nivå og valgnivå som brukeren er knyttet til gjennom rollen.

Brukeren gis tilgang til dataene dersom knytninger stemmer overens.

Skissen beskriver beslutningsmodellen for tilgangskontroll til funksjoner og data



## Datamodell - rollebasert tilgangskontroll

Rollemodellen styrer hvilke funksjoner og data bruker gis tilgang til i systemet.

Konseptene er som følger:

- En bruker er knyttet til en eller flere *roller*
- En *rolle* er knyttet til en eller flere aksesser, dvs. tilganger til data og manipulering av data i systemet
- En *rolle* er i tillegg knyttet til et *geografisk område*
- En *rolle* er i tillegg knyttet til et *nivå i valghierarkiet*

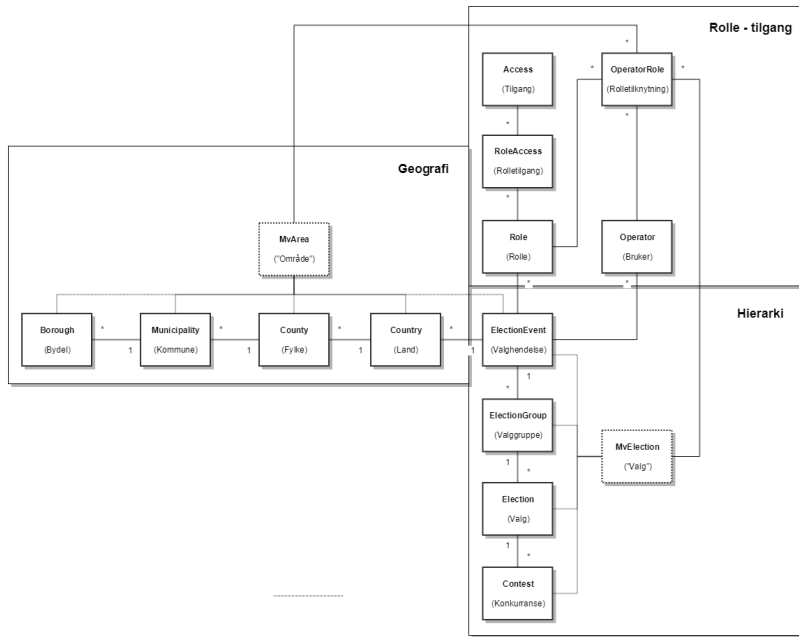
I sum gjør dette det mulig å definere roller med begrensede systemtilganger ut fra rollens definerte arbeidsoppgaver.

Dette gir to gevinster:

- Sikkerhet - en gitt bruker med en gitt rolle har kun tilgang til funksjoner som er relevante og tiltrudde rollen
- Optimalisering av arbeidsflyt - en gitt bruker med en gitt rolle forholder seg kun til de funksjoner som er relevante, og kjente for brukeren

Skisse som illustrerer de sentrale konseptene i rolle- og tilgangsmodellen





Se kapittel rollebasert tilgangskontroll for forklaring av domenemodellen.

*Eksempel:*

Kjell skal ta imot stemmegivninger i valglokalet Fram i Volden krets i Tønsberg kommune ved Stortingsvalget:

- Kjell opprettes som bruker i EVA Admin
- Rollen stemmemottaker valgting er forhåndsdefinert med følgende attributter:
  - Valgnivå - som kan være på nivå Valggruppe (flere valg), Valg (et gitt valg), Valgdistrikt
  - Geografisk nivå - Høyeste geografiske nivå rollen kan brukes mot (Nasjon, fylke, kommune, krets, valglokale)
  - Funksjonstilgang - stemmemottaker har kun tilgang til funksjonen registrer stemmegivning valgting
- Kjell knyttes til rollen stemmemottaker valgting, og kombinasjonen Kjell og rollen stemmemottaker valgting knyttes til det geografiske området Fram i Volden krets (i Tønsberg kommune)

Kjell har nå tilgang til funksjonen register stemmegivning i EVA Admin - og kun den funksjonen.

## Logging

### Auditlogging

Alle operasjoner brukeren gjør i Admin blir sporet i Admin sin *auditlogg*. Auditloggen sendes til et sentralisert verktøy for logging og overvåking.

Hovedprinsippene for omfanget av auditlogging er at følgende spores:

- Alle handlinger som medfører at data endres (feks registrering av en telling, eller endring av en konfigurasjonsparameter)
- Alle handlinger som henter ut personopplysninger eller andre sentrale data (feks søk etter en person i manntall)
- Alle handlinger som henter ut data som er viktige deler av valgprosessen (feks uthenting av avkrysningsmanntall eller møtebok)

Det logges til én logg:

Navn	Beskrivelse
Audit.log	Logger aksess til alle tjenestekall som er annotert/merket med "auditlog".

Formatet i audit-logg er key-value-basert. Objekter logges på JSON-format.

**audit.log**

```
time="2014-12-19 14:23:52,219 +0100", client=127.0.0.1, process=Authentication, eventType=OperatorLoggedIn,
outcome=Success, uid=06052937823, securityLevel=3
time="2014-12-19 14:24:13,709 +0100", client=127.0.0.1, electionEvent=750403, process=Authentication,
eventType=OperatorSelectedRole, outcome=Success, uid=06052937823, securityLevel=3, role=election_event_admin,
roleAreaPath=750403, roleElectionPath=750403
time="2014-12-19 14:35:18,989 +0100", client=127.0.0.1, electionEvent=750403, process=CentralConfiguration,
objectType=ElectionDay, eventType=Update, outcome=Success, uid=06052937823, securityLevel=3,
role=election_event_admin, roleAreaPath=750403, roleElectionPath=750403, auditObject="{\"date\":\"2014-10-22\", \"star
tTime\":\"1970-01-01 08:00\", \"endTime\":\"1970-01-01 23:59\"}"
```

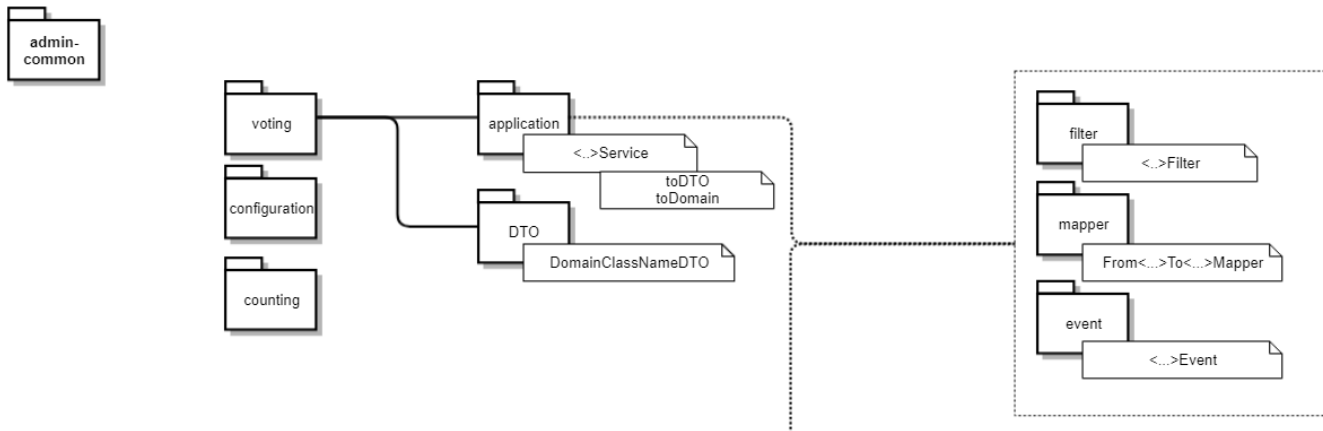
## Kommunikasjon frontend - backend

Av historiske årsaker kommuniserer frontend og backend gjennom Java RMI og JNDI. Dette skjer ved at backend eksponerer et sett med tjenester i en felles modul som også inkluderes i frontend.

### Javaklasser

EVA Admin er bygget opp basert på standard Java EE komponenter, API-koden er organisert etter prinsipper for domenedrevet design

Skisse illustrerer oppbygging av kode som følger nytt mål bilde:



Primærinnordningen skjer gjennom domeneområder, for hver domene organiseres koden iht. ansvarsområder:

- ApplicationServices (Interfaces) - backendtjenester som eksponeres til frontend
- DTO - backendtjenestenes payload
- Mapper - transformasjonsmapping mellom DTO (fra/til frontend) og domeneklasser (fra/til backend)
- Filter / event - tilleggsfunksjoner

## Database

EVA Admin applikasjonen bruker PostgreSQL database for lagring av data. Databaseen er en standard relasjonsdatabase.

### Databaseskjema

Databasen er delt i to skjema, ett for applikasjonsbruk, og ett for endringssporing

#### Applikasjonsbruk

- Tables - tabeller / materialiserte views
- views - views (denormaliseringer)
- routines- prosedyrer (forretningslogikk / auditskjema)
- sequences - sekvenser (nøkler)
- object\_types - typer

Av historiske årsaker finnes det fremdeles noe forretningslogikk i databaseprosedyrer, i tillegg håndterer prosedyrene oppdatering av:

- Matrialiserte views
  - mv\_area
  - mv\_election
- Innslag i auditskjema / audittabeller

Samtlige forekomster i tabeller identifiseres ved:

- Id - f.eks kommunenummer
- Primærnøkkel generert ut i fra tilhørende sekvenser

For nærmere beskrivelse av relasjons- og databasemodell se domenemodeller.

### Auditskjema

- Tabeller

Alle skriveoperasjoner mot databasen logges og foregående utgave av en forekomst lagres i audit-skjema. Dette gjør det mulig å opprettholde *full sporing på alle endringer som er av prinsipiell betydning for valggjennomføringen*.

## Rapportering

### Formål

Rapportmodulen understøtter uttak av rapporter og statistikk gjennom hele valgprosessen. I hovedsak kan rapporter deles inn i 2 grupper:

- Statistikk - statistikkrapporter tas ut av brukerne av systemet for å understøtte beslutningsprosesser eller for å innhente informasjon om valgavviklingen.
- Protokoller - protokollrapporter kalles som oftest "møtebøker", dette er de offisielle protokollene for valgavviklingen på et gitt nivå i valgdistriktet eller for valget. Disse brukes kun av sametingsvalget, da andre valg har en egen protokolleringsløsning.

### Valgrutiner og brukerveiledning

Valgmedarbeiderportalen beskriver valgrutinene for gjennomføring av valg og inneholder en brukerveiledning med beskrivelse av de ulike funksjonene i EVA Admin. Rutinene og brukerveiledningen for opptelling bør leses i sammenheng med systemdokumentasjonen for å få et klart bilde av prosessen.

### Modulbeskrivelse

I det følgende beskrives de viktigste konseptene i rapporteringsmodulen.

Rapportmodulen gir tilgang til statistikk og valginformasjon gjennom hele valggjennomføringen. Avhengig av valgprogresjonen hentes informasjon fra en eller flere av fasene:

- Forberedelse
- Stemmegivning
- Opptelling
- Valgoppgjør

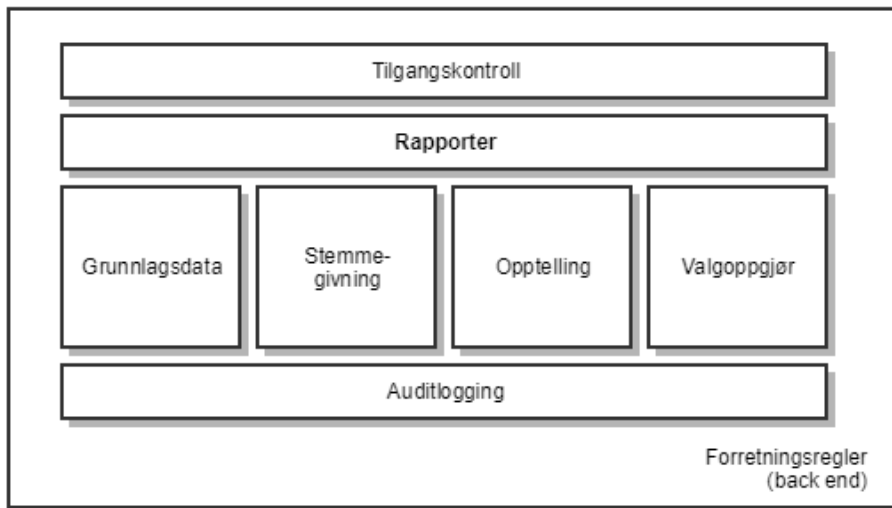
Rapportene tas ut som PDFer og/eller excel-filer, avhengig av bruksområde.

### Datagrunnlag

Rapporter og statistikk er typisk sammenstilling av data fra flere domeneområder, for å oppnå dette opererer rapporteringsmodulen på hele domenet i EVA Admin.

Typisk vil en "møtebok" (valgprotokoll) genereres med data fra hele valggjennomføringen fra grunnlagsdata til valgoppgjør.

*Skissen illustrerer at rapportering skjer basert på data fra en eller flere faser avhengig av valgprogresjon*



## Kodeorganisering - prinsipper og retningslinjer

### Domenedrevet design (DDD)

Domenedrevet design (DDD) er en tilnærming til softwareutvikling der det utvikles modeller for å beskrive og implementere ulike forretningsområder og prosesser. For et gitt forretningsområde eller prosess benyttes modellen for å designe funksjonalitet, samtidig som modellen og implementasjonen gjenspeiler hverandre. Modellen er sentral i kommunikasjonen mellom funksjonelle eksperter og utviklere, og det er viktig at funksjonelle eksperter og utviklere benytter en felles terminologi når det kommuniseres rundt modellen.

#### "Byggeklosser"

**entitet:** et objekt som har en id og som har en utvikling over tid

**verdiobjekt:** et objekt der bare verdien(e) er viktige, utvikler seg ikke over tid. Verdiobjekter endrer seg ikke etter at de er opprettet. Det er anbefalt å benytte verdiobjekter der det er mulig. Det at de ikke endrer seg gjør koden enklere og mindre utsatt for feil.

**tjeneste/service:** funksjonalitet som ikke kan modelleres som en ting

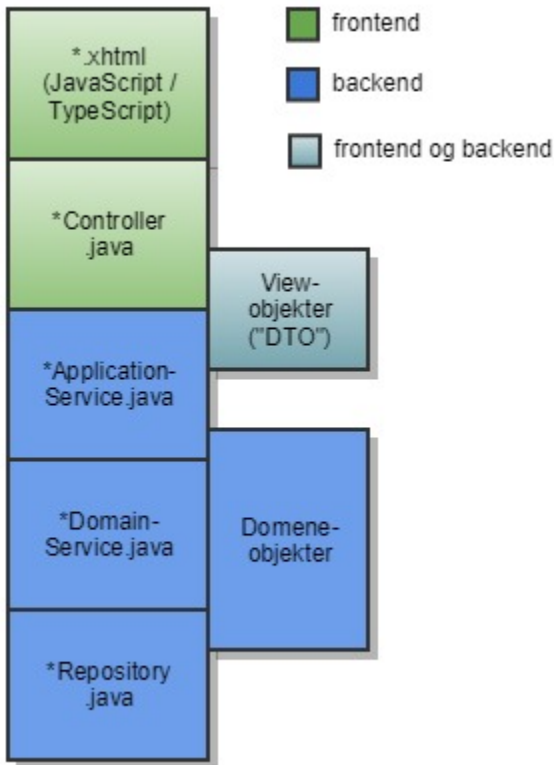
**aggregat:** en rotentitet som kan inneholde andre entiteter eller verdiobjekter, skal være transaksjonelt konsistent, ContestReport f.eks.

**repository:** en abstraksjon av f.eks en database, brukes for å hente ut aggregater

**bounded context:** en avgrensning rundt en modell, et stort system består ofte av flere modeller som er konsistente og har mening innenfor en kontekst.

### Kodeeksempel

Dette er et eksempel på kode som følger det gjeldende målbilde. Ovenfra og ned i følgende diagram vil det vises kodeeksempler fra hvert individuelle lag. Annen kode som følger nytt målbilde har samme inndeling og vil kunne navigeres på en tilsvarende måte.



Vi tar utgangspunkt i et av de nye skjermbildene som ble innført i 2015 for administrasjon av brukere.

## Koden som understøtter det nye skjermbildet for å administrere brukere

Figuren under viser skjermbildet.



Oslo, Olemic | Valgadministrator kommune, Lunner Kommune  
**Skanning test**  
[Hjelp](#) [Bytt rolle](#) [Logg ut](#) Norsk bokmål ▼

Min side ► Administrere bruker

### Administrere bruker

Fylkeskommune: Oppland | Kommune: Lunner

[+ Opprett ny bruker](#)
[Importer fra Excel](#)

	Navn	Fødselsnr	Alle roller	Alle steder	Telefonnr	E-post
<a href="#">Rediger</a>	Sand, Karen-Anne-britt	02025918498	Skanning	Lunner	12345678	
<a href="#">Rediger</a>	Oslo, Olemic	26125826627	Valgadministrator kommune	Lunner	12345678	
<a href="#">Rediger</a>	Ahmed, Embla-Turi	19034030084	Skanning	Lunner	18159901	Embla-Turi.Ahmed@test.no
<a href="#">Rediger</a>	Rosslund, Torkel-Thor	21125529302	Valgadministrator kommune	Lunner	12551728	Torkel-Thor.Rosslund@test.no

**Kildekodefil: operatorAdmin.xhtml**

```
<ui:composition template="/templates/layout.xhtml">

  <ui:param name="helpId" value="@help.rbac"/>

  <ui:define name="breadCrumb">
    <p:menuItem value="#{msgs['@rbac.handle.operator']}" url="operatorAdmin.xhtml"/>
  </ui:define>

  <ui:define name="content">

    <div id="page-operator-admin" class="page" data-view="OperatorAdmin">
      <h1 class="page-title">
        <h:outputText value="#{msgs['@rbac.handle.operator']}" />
      </h1>
      <!--No need to guard pageTitleMeta here-->
      <widget:pageTitleMetaTemplate models="#{operatorAdminController.pageTitleMeta}" />

      ...

      <ui:include src="#{operatorAdminController.view.template}" />
    </div>
  </ui:define>
</ui:composition>
```

I starten av kildekoden for siden refereres det til det generelle sideoppsettet som brukes for alle sidene: "/templates/layout.xhtml". I dette oppsettet settes det inn ulike innholdsfragmenter, blant annet `helpId`, `breadCrumb` og `content`.

Div-elementet (her med `id="page-operator-admin"`) er sentralt i frontend-arkitekturen. Det angir at JavaScript-filen `OperatorAdmin.js` understøtter datavisningen for denne siden. `OperatorAdmin.js` refererer igjen til verdien av `id`-attributten, `page-operator-admin`.

Uttrykket `{operatorAdminController.pageTitleMeta}` er et av mange knytninger eller kall mot JavaBean-klassen som understøtter skjermbildet. I EVA Admin betegnes disse klassene som kontrollere, `*Controller.java`.

## Kildekodefil: `OperatorAdminController.java`

```

@Named
@ViewScoped
public class OperatorAdminController extends BaseController {

    private static final int MAX_EXCEL_ERRORS = 10;

    @Inject
    private UserData userData;
    @Inject
    private PageTitleMetaBuilder pageTitleMetaBuilder;
    @Inject
    private OperatorImportHelper importHelper;
    @Inject
    private OperatorListController listController;


    ...

    @PostConstruct
    public void init() {
        userAgent = userAgentParser.parse(getFacesContext());
        listController.initOperatorLists();
        setView(RbacView.LIST);
    }

    ...

    public List<PageTitleMetaModel> getPageTitleMeta() {
        return pageTitleMetaBuilder.area(userData.getOperatorMvArea());
    }
}
    
```

Weld er et rammeverk for blant annet "å injisere" avhengigheter bestående av klasseinstanser som håndteres av kontaineren. Det er en implementasjon av CDI (Context and dependency injection) spesifikasjonen i JEE. `@Inject` over `UserData`-attributten er et eksempel på bruk av CDI. `UserData` er et såkalt session scoped objekt som opprettes i forbindelse med at en bruker logger på, og det har levetid så lenge sesjonen til brukeren er aktiv.

 Merk at vi her bruker såkalt "field injection". "Constructor injection" er bedre da vi samler alle avhengighetene en komponent trenger i konstruktøren. Det anbefales å endre fra "field injection" til "constructor injection". Under er et eksempel fra kodebasen på "constructor injection":

```

@Inject
public CertificateRevocationListService(CertificateService certificateService) {
    this.certificateService = certificateService;
}
    
```

Uttrykket `#{operatorAdminController.pageTitleMeta}` fra xhtml-siden evalueres til returverdien av metoden `getPageTitleMeta` i `OperatorAdminController`.

Skjermbildet har flere modi. Kontrolleren holder rede på hvilken modus som er gjeldende. Elementet

```
<ui:include src="#"#{operatorAdminController.view.template}"/>
```

legger til innhold for aktuell modus. Initielt er dette visning av en liste av operatører som kan redigeres. Visningen av listen er implementert i `operatorList.xhtml` som understøttes av `OperatorListController`. Etter at `OperatorAdminController` er opprettet i kontaineren vil metoden markert med `@PostConstruct` bli kalt (se kildekode for `OperatorAdminController` over). Denne kaller igjen:

## OperatorListController

```
@Inject
private AdminOperatorService adminOperatorService;

...

public void initOperatorLists() {
    setOperatorList(new ArrayList<>(transform(adminOperatorService.operatorsInArea(userData, userData.
getOperatorAreaPath()),
        new Function<Operator, OperatorWrapper>() {
            @Override
            public OperatorWrapper apply(final Operator operator) {
                return new OperatorWrapper(operator);
            }
        }
    )));
    operatorIds.clear();
    for (OperatorWrapper wrapper : operatorList) {
        operatorIds.put(wrapper.getValue().getPersonId().getId(), wrapper.getValue());
    }
    extractRolesAndAreasFromOperatorList();
    setFilteredOperatorList(operatorList);
}
```

som henter ut listen med operatører fra databasen ved å benytte en applikasjonstjeneste i admin backend.

## Eksempel på applikasjonstjeneste

Applikasjonstjenester implementerer grensesnittet mellom frontend og backend. Ansvarsområdene for applikasjonstjenester er:

- sikkerhet / tilgangskontroll
- transaksjonshåndtering (ved hjelp av kontainer)
- auditlogging
- aggregering og mapping av data - data fra entiteter som er knyttet til databasetabeller mappes til objekter som returneres til admin frontend. I den nye arkitekturen ønsker vi ikke at databasetabellmappings skal eksponeres i admin frontend.



### AdminOperatorApplicationService

```

@Remote(AdminOperatorService.class)
@Stateless(name = "AdminOperatorService")
public class AdminOperatorApplicationService implements AdminOperatorService {

    private OperatorRepository operatorRepository;
    private OperatorRoleRepository operatorRoleRepository;
    private MvAreaRepository mvAreaRepository;
    private RoleRepository roleRepository;
    private RoleAreaService roleAreaService;
    private OperatorDomainService operatorDomainService;
    private VoterRepository voterRepository;

    @Inject
    public AdminOperatorApplicationService(OperatorRepository operatorRepository, OperatorRoleRepository
operatorRoleRepository, RoleRepository roleRepository,
        MvAreaRepository mvAreaRepository, RoleAreaService roleAreaService, OperatorDomainService
operatorDomainService, VoterRepository voterRepository) {
        this.operatorRepository = requireNonNull(operatorRepository);
        this.operatorRoleRepository = requireNonNull(operatorRoleRepository);
        this.roleRepository = requireNonNull(roleRepository);
        this.mvAreaRepository = requireNonNull(mvAreaRepository);
        this.roleAreaService = requireNonNull(roleAreaService);
        this.operatorDomainService = requireNonNull(operatorDomainService);
        this.voterRepository = requireNonNull(voterRepository);
    }
}

```

@Remote og @Stateless indikerer at applikasjonstjenesten er en EJB 3-komponent. Dette er viktig fordi tjenesten kan kalles fra en annen virtuell maskin (der admin frontend kjører) og kontaineren sørger for at tjenestekallet foregår i en transaksjon.

Applikasjonstjenesten har en rekke avhengigheter som injiseres på samme måte som en kontrollør får injisert andre kontrollere eller applikasjonstjenester. Her bruker vi injisering via konstruktør fremfor feltvis injisering. På denne måten slipper vi å skrive kode som bare testene benytter, og testene kjører samme kode som kjøres i produksjonsmiljøet.

Applikasjonstjenesten `operatorsInArea` henter ut listen med `Operator`-instanser som frontend spør etter. `Operator` som returneres er her ikke en hibernate-mappet entitet, men et domeneobjekt som brukes for å understøtte datautveksling mellom frontend og backend samt for å understøtte visningslogikk i frontend. Objektene av denne typen kan også ha annen hensiktsmessig logikk knyttet til dataene. `Operator` er altså mer enn et enkelt dataoverføringsobjekt (DTO) siden det også inneholder logikk. Tradisjonelt har EVA Admin vært preget av å ha en anemisk domenemodell der entitetene var enkle databærere. Det nye målbildet etterstreber å ha logikk i domenemodellen for å øke testbarhet, hindre duplisering av kode og kommunisere forretningsregler via modellen. Applikasjonstjenesten gjør først dataaggregering, hvor den henter ut en `MvArea` entitet (som er hibernate-mappet og bare bør eksistere i backend-modellen) fra verdiobjektet `AreaPath`. Deretter hentes det opp en liste med `OperatorRole`-instanser ved hjelp av domenetjenesten `OperatorDomainService`. Til slutt sammenfattes resultatene til en liste med `Operator`-instanser som returneres.

`AreaPath` er et verdiobjekt. Det betyr at det ikke har noen id eller livssyklus - det er bare verdien `AreaPath` inneholder som betyr noe. `AreaPath` brukes for å understøtte datautveksling mellom frontend og backend og inneholder en del logikk knyttet til områdestier. Det er ikke muterbart, altså vil det lages en ny instans av objektet dersom verdien endres.

### Eksempel på entitet med domenelogikk

Applikasjonstjenester bruker domenetjenester, repositories, entiteter og verdiobjekter. Domenelogikk bør gjerne implementeres på entiteter og verdiobjekter. Følgende eksempel viser hvordan en applikasjonstjeneste benytter et repository og en entitet for å oppdatere kontaktinformasjonen til en bruker.

### AdminOperatorApplicationService

```

@SecObj(secObjs = { SecObjEntity.ANY_USER })
@Override
@AuditLog(eventClass = ContactInfoChangedAuditEvent.class, eventType = AuditEventTypes.ContactInfoChanged)
public void updateContactInfoForOperator(UserData userData, ContactInfo contactInfo) {
    operatorRepository.findByPk(userData.getOperator().getPk()).updateContactInfo(contactInfo);
}

```

Her hentes det frem en `Operator`-entitet ved hjelp av `OperatorRepository`.

### Operator

```
public void updateContactInfo(ContactInfo contactInfo) {
    setContactInfoConfirmed(true);
    setTelephoneNumber(contactInfo.getPhone());
    setEmail(contactInfo.getEmail());
}
```

Operator har en `updateContactInfo`-metode for å oppdatere kontaktinformasjonen. I tidligere implementasjoner var det en annen klasse som oppdaterte de tre feltene på `ContactInfo`. Sett fra et objektorienteringsperspektiv er det en mye bedre implementasjon at `Operator` selv oppdaterer sine interne felter. Forretningsregelen om at flagget `contactInfoConfirmed` skal settes blir også ivaretatt. Dette er et enkelt eksempel på kode som følger retningslinjene for domene-drevet design (DDD). Setterne som kalles i `updateContactInfo` bør ideelt sett ha privat aksess (være "private").

## Eksempel på domenetjeneste

Domenetjenestene er den delen av forretningslogikken i Admin som ikke enkelt kan knyttes til entiteter eller verdiobjekter. Domenetjenestene implementerer forretningsregler.

### AdminOperatorApplicationService

```
@Override
@SecObj(secObjs = { SecObjEntity.INSPECT, SecObjEntity.OPERATOR })
public List<Operator> operatorsInArea(UserData userData, AreaPath areaPath) {
    MvArea mvArea = findMvArea(requireNonNull(areaPath));
    List<OperatorRole> operatorsRoles = operatorDomainService.operatorRolesInArea(mvArea, userData);
    return OperatorViewToDomainMapper.toOperatorWithRoleAssociations(operatorsRoles);
}
```

I eksemplet over benytter applikasjonstjenesten domenetjenesten `operatorRolesInArea`. Denne returnerer `OperatorRole`-entiteter som mappes til `Operator`, som returneres til klienten.

### OperatorDomainService

```
public List<OperatorRole> operatorRolesInArea(MvArea mvArea, UserData userData) {
    List<OperatorRole> operatorRoles = canAccessAreasInsideOperatorsArea(userData) ? operatorRoleRepository.
    findDescOperatorsRoles(mvArea)
        : operatorRoleRepository.operatorRolesByArea(mvArea);

    boolean includeUserSupportOperator = AreaPath.from(mvArea.getAreaPath()).isRootLevel();
    if (!includeUserSupportOperator) {
        operatorRoles = removeUserSupportRole(operatorRoles);
    }

    return operatorRoles;
}
```

Domenetjenesten implementerer følgende forretningsregler:

- hent alle operatører på og under operatørens områdenivå dersom brukeren er valgadmin kommune eller valghendelsesadmin (implementert ved å sjekke på en bestemt tilgang disse brukerne er konfigurert til å ha)
- hent alle operatører på operatørens områdenivå for andre nivå, det vil si brukere på fylkesnivå. Disse skal bare kunne endre på andre brukere på fylkesnivå (og ikke på alle underliggende kommuner)
- returner også brukerstøttebrukeren om operatørens områdenivå er på toppnivået (det vil si at operatøren er valghendelsesadmin)

## Eksempel på repository klasse

Domenetjenester og applikasjonstjenester kan benytte repository-klasser for å hente ut entiteter.

### OperatorRoleRepository

```
public List<OperatorRole> operatorRolesByArea(MvArea mvArea) {
    TypedQuery<OperatorRole> query = getEm()
        .createNamedQuery("OperatorRole.findOperatorRolesByArea", OperatorRole.class)
        .setParameter("mvArea", mvArea);
    List<OperatorRole> operatorRoles = query.getResultList();
    Collections.sort(operatorRoles);
    return operatorRoles;
}
```

Her benyttes en Hibernate-spørring for å hente ut OperatorRole-entiteter knyttet til et gitt område.

Repositories kan gjerne ha et grensesnitt som minner om "collection"-grensesnittet i Java.

### CertificateRevocationListRepository

```
public void add(CertificateRevocationList certificateRevocationList) {
    ..
}

public CertificateRevocationList certificateRevocationListByIssuer(X500Principal issuer) {
    ..
}
```

Slike implementasjoner kalles collection-oriented repositories og abstraherer lesing og skriving til databasen som operasjoner mot en "collection".

## Aggregater

Et aggregat er en samling entiteter som hører sammen. Det vil si at endringer på entitetene skal skrives til databasen i samme transaksjon. De fleste aggregatene i EVA Admin er for finkornede. ContestReport er et eksempel på et aggregat som har en mer hensiktsmessig størrelse. Det finnes et repository for å aksessere ContestReport-entiteter. Når tellingene som hører til en ContestReport skal endres, gjøres dette via denne entiteten. En VoteCount eller BallotCount skal dermed ikke ha et eget repository.

## Repositories eller tjenester som gjør databaseoppslag?

I EVA Admin bruker vi en del steder en repository-abstraksjon der det i mange tilfeller ville vært mer hensiktsmessig å bruke en tjeneste som gjør et direkte databaseoppslag.

## VotingRepository

```

public long findNotRejectedVotingCountByMunicipalityAndCategoriesAndLateValidation(
    Municipality municipality,
    no.valg.eva.admin.common.voting.VotingCategory[] categories,
    boolean lateValidation) {
    return getEm()
        .createNamedQuery("Voting.findNotRejectedVotingCountByMunicipalityAndCategoriesAndLateValidation",
Long.class)
        .setParameter("municipalityPk", municipality.getPk())
        .setParameter(VOTING_CATEGORY_IDS, toStringList(categories))
        .setParameter(LATE_VALIDATION_FILTER, lateValidation)
        .getSingleResult();
}

public Collection<Voting> findApprovedVotingsByPollingDistrictAndCategories(
    PollingDistrict pollingDistrict,
    no.valg.eva.admin.common.voting.VotingCategory[] categories) {
    List<Object[]> rows = getEm()
        .createNamedQuery("Voting.findApprovedVotingsByPollingDistrictAndCategories", Object[].class)
        .setParameter("pollingDistrictPk", pollingDistrict.getPk())
        .setParameter(VOTING_CATEGORY_IDS, toStringList(categories))
        .getResultList();
    List<Voting> result = new ArrayList<>(rows.size());
    for (Object[] row : rows) {
        Voting voting = new Voting();
        voting.setCastTimestamp((DateTime) row[0]);
        voting.setApproved((Boolean) row[1]);
        voting.setVotingCategory((VotingCategory) row[2]);
        // CSOFF: MagicNumber
        voting.setMvArea((MvArea) row[3]);
        // CSON: MagicNumber
        result.add(voting);
    }
    return result;
}
    
```

Koden over er fra `VotingRepository` i admin-counting-prosjektet. Den øverste metoden bør sannsynligvis være i en tjenesteimplementasjon siden den returnerer et tall og ikke en `Voting`-entitet. Metoden under er et annet eksempel. Her returneres et antall "amputerte" `Voting`-entiteter fordi klienten ikke trenger noe annet enn `timestamp`-verdien for hver `Voting`. En tjeneste som returnerer tidspunktene for stemmegivninger ville vært en mer hensiktsmessig tilnærming her.

## Integrasjoner

### Mottak av telleresultater fra EVA Skanning

#### Mottakstjeneste - telleresultater

EVA Admin eksponerer en tjeneste til EVA Skanning der telleresultat kan mottas og konsumeres.

Telleresultatene er XML-filer som er strukturert etter samme oppbygning som EMLen EVA Skanning bruker for konfigurasjon av valgoppsett:

- Sesjonstoken, samt buypasssignatur
- Informasjon om hvem som overfører (identifikasjon og rolle)
- Informasjon om hvilket valg opptellingen gjelder
- Informasjon om tilhørighet i valget (f.eks Tønsberg i Vestfold fylke), samt rapporteringsenhet (styre)
- Opptellingsfilen med:
  - Opptellingskategori (Forhåndsstemmer, valgtingsstemmer osv.)
    - Parti
    - Stemmetall - gyldige stemmesedler
    - Stemmetall - rettede stemmesedler (også inkludert i gyldige stemmesedler)
  - Totalt antall opptalte stemmesedler
  - Avviste stemmesedler fordelt på gyldige avvisningsårsaker
- Fil med rettede (personstemmer, strykninger m.m) og foreslått forkastede stemmesedler:
  - Opptellingskategori (Forhåndsstemmer, valgtingsstemmer osv.)

- Parti
  - Kandidat
  - Personstemmer / slengere / strykninger / renummereringer
  - Bilde av stemmeseddelen med retting
- Opptellingskategori (Forhåndsstemmer, valgtingsstemmer osv.)
  - Bilde av foreslått forkastet stemmeseddel

Mottatte opptellingsresultater fra EVA Skanning lagres i EVA Admin opptellingstabeller. Opptellingsmodulen i EVA Admin støtter den videre prosessen med oppfølging av endrede og foreslått forkastede stemmesedler.

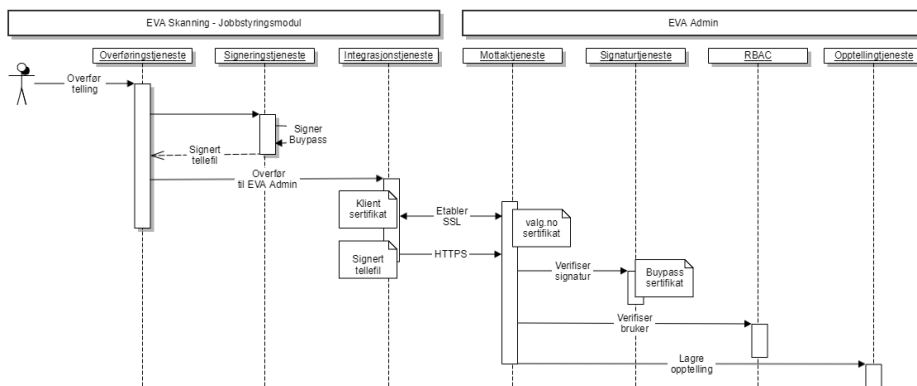
## Sikkerhet ved overføring

Ved ferdigstilling av en gitt telling skal telleresultatet oversendes til EVA Admin.

Overføringen av opptellingsfilen sikres på flere måter:

- Ved roller: Det er kun valgansvarlig som kan overføre opptellingsfiler fra EVA Skanning til EVA Admin
- Ved signering av telleresultat: Valgansvarlig må signere tellefilen ved hjelp av Bypass signeringstjeneste med Bypasskort
- Ved tilgangskontroll: EVA Skanning må inneha et klientsertifikat for å få tillatelse til å kommunisere med EVA Admin
- Ved kryptering: Tellefilen oversendes over HTTPS med asymmetrisk kryptering
- Ved verifisering i EVA Admin: Ved mottak av tellefil vil EVA Admin verifisere at avsenders signatur har opphav i forventet Bypass-sertifikat
- Ved autentisering: EVA Admin: Ved mottak av tellefil vil EVA Admin kontrollere om signatur fra avsender stemmer overens med bruker som er definert som valgansvarlig i EVA Admin

Skissen illustrer en forenklet framstilling av oversending av opptellingsresultat fra EVA Skanning (jobbstyringsmodulen) til EVA Admin med fokus på sikkerhetsmekanismer (skissen inneholder begge systemer for å sikre helheten)



## Id-porten - brukerautentisering

Brukere i EVA Admin må autentiseres, dvs. at brukerens identitet må kontrolleres.

Dette gjøres i ID-porten. På denne måten trenger ikke EVA Admin å forvalte brukeridentiteter, men lar i stedet brukerne bruke sin egen, elektroniske ID som f.eks. MinID, BankID eller Buypass.

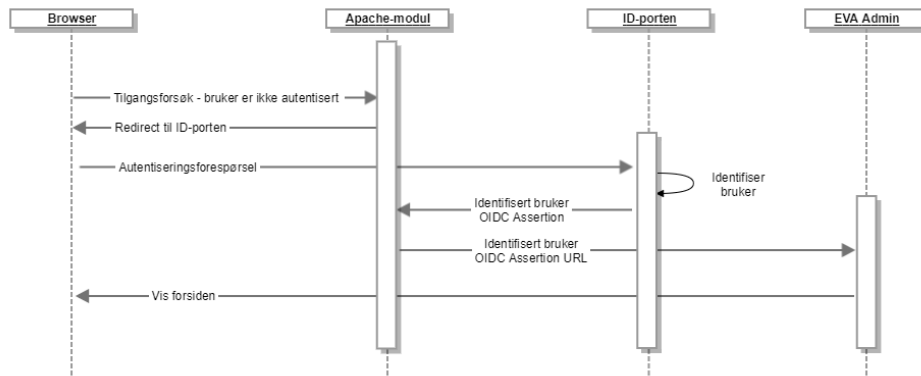
### Protokoll

Autentiseringstjenesten bruker OpenID Connect (OIDC)-protokollen mot ID-porten.

### ID-porten dialog

Apache-modulen mod\_auth\_ldap brukes som OIDC-megler mot ID-porten.

Forenklet skisse over dialog med ID-porten



- Bruker forespør EVA Admin innloggingside frontend. Apachemodul videresender forespørselene til ID-porten
- ID-porten identifiserer og autentiserer bruker
- Frontend Apachemodul videresender svar på autentiseringsforespørsel til EVA Admins rollevalgside hvor alle brukerens roller i EVA Admin kan velges mellom før man videre redirigeres til "min side".

## Rapportering av opptellingsresultat til EVA Resultat

### Formål

Formålet med admin-valgnatt, modulen som rapporterer til EVA Resultat (tidligere Valgnatt), er å produsere og oversende ulike rapporter, også kalt skjemaer, til EVA Resultat. Det finnes totalt fire typer skjema: To for grunnlagsdata, som rapporteres initielt før andre rapporter, og to for valgresultater. Reglene for når ulike skjemaer er klare for å rapporteres er også implementert, sammen med håndtering av metadata for de ulike skjemaene.

### Rapporttyper

Følgende rapporttyper inngår:

- Grunnlagsdatarapporter
  - geografi (stemmekretser), antall stemmeberettigede og kobling til valgdistrikter
  - valgdistrikter, partier og kandidater
- Resultatrapporter
  - stemmeskjema
  - metadata for rapportering (oppgjørsskjema)

### Rapportering av grunnlagsdata

For "geografi og stemmeberettigede" vil informasjon om hver stemmekrets oversendes. Dette gjelder både valgtingsstemmekretser og forhåndskretser. Tidligere var det mulig for kommuner å telle sentralt samlet for alle kategorier, i hvilket tilfelle kun informasjon om kommunekretsen 0000 ble oversendt, men dette er ikke lenger tilfelle.

### Rapportering av stemmeskjema

EVA Resultat må være initialisert med grunnlagsdata og tellinger må være utført og godkjent før stemmeskjema kan rapporteres. I praksis utføres dette automatisk når opptellingen godkjennes.

Opptellingsmåtene påvirker hvordan rapporteringen foregår. Ved sentralt samlet havner alle tellinger på kommunekretsen (krets 0000) og rapporteres fra denne kretsen. Dette vil gjelde forhåndsstemmer/valgtingsstemmer - øvrige, samt forhåndsstemmer talt etter klokken 17 dagen etter valgdagen (såkalt "sent innkomne"). Når det telles fordelt på krets, altså i kategoriene forhåndsstemmer/valgtingsstemmer - fra valglokalene, fordeles stemmene utover de ulike kretsene/valglokalene der de ble mottatt.

## Rapporter - Jasper Server

Rapporter i EVA Admin produseres via Jasperserver og kan hentes ut av bruker på PDF-, Excel-, eller CSV-format, avhengig av rapporttype.

### Kjøring av rapporter

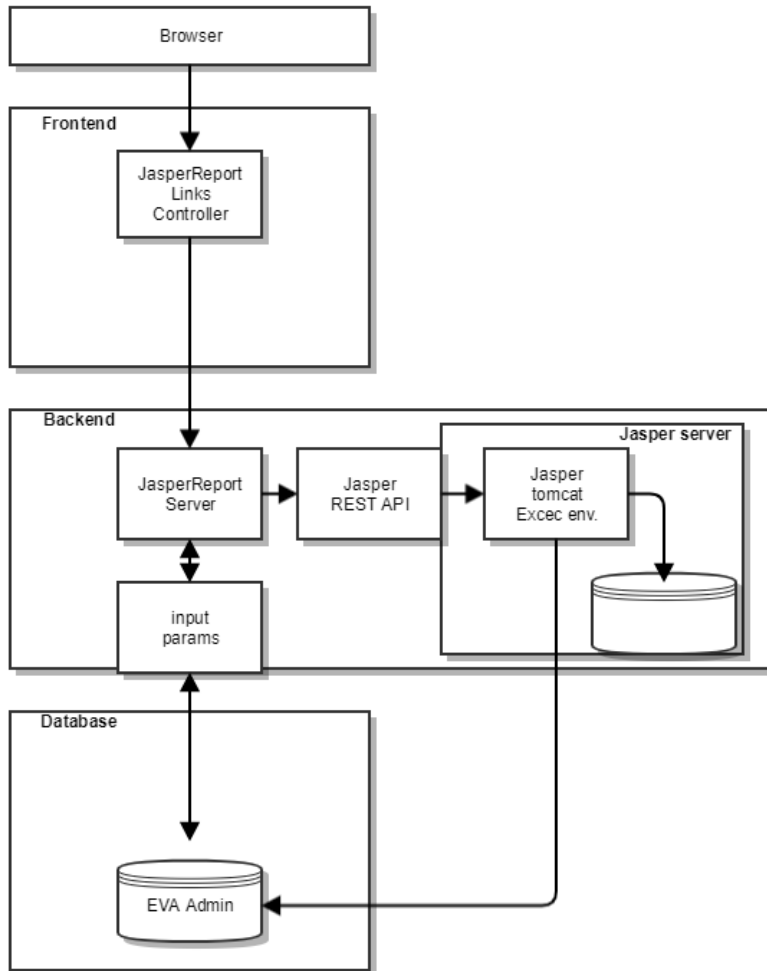
Linker for å produsere rapportene er samlet på siden "Alle rapporter" som er tilgjengelig fra hovedmenyen i EVA Admin. Tilgang til produksjon av rapporter er rollestyrt på lik linje med andre funksjoner i EVA Admin.

Input-parametre til rapportproduksjon hentes fra EVA Admin dersom dette er påkrevd.

### Integrasjon

JasperServer er en webapplikasjon som kjøres i en Apache Tomcat servlet container, med en egen PostgreSQL database som holder på konfigurasjon og rapportmaler. EVA Admin kommuniserer med JasperServer via dennes REST-api. Det kjøres én JasperServer uavhengig av hvor mange backend-noder som kjøres, og den kjøres på en egen server, separat fra resten av EVA Admin.

Skissen er en forenklet representasjon av integrasjonen med Jasperserver:



## Folkeregisteret - manntall

### Innledning

Valgdirektoratet mottar manntall fra folkeregisteret gjennom folkeregisterets automatiserte delingstjenester.

### Beskrivelse

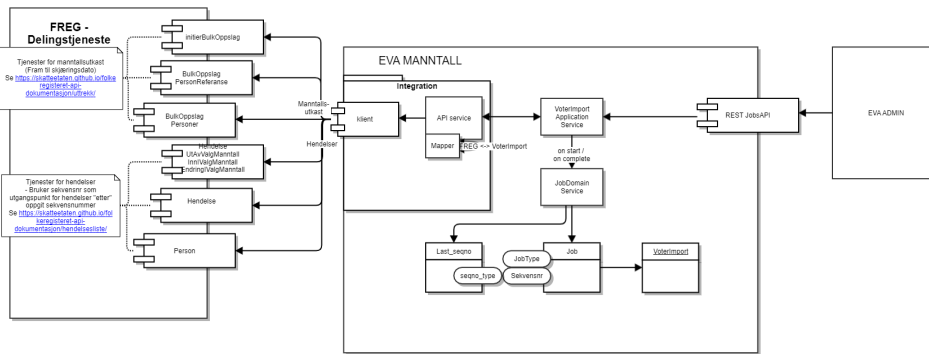
Folkeregisteret tilbyr 2 delingstjenester for valgmanntall:

- Tjeneste for bulkoppslag av manntall, som gir et fullt manntall på et gitt tidspunkt.
- Tjeneste for endringer i manntall, som gir endringer i manntall over tid.

Folkeregisterets delingstjenester er beskrevet her: <https://www.skatteetaten.no/deling/opplysninger/folkeregisteropplysninger/>

Det er utviklet en modul, EVA Manntall, for integrasjon mot folkeregisterets delingstjenester for å kunne hente bulkoppslag ved behov, og for fra et gitt tidspunkt å kunne hente endringer i manntall. EVA Admin er i sin tur integrert mot EVA Manntall. Dette skillet er gjort for å sikre optimal tilgjengelighet for funksjoner som krever manntall i EVA Admin uavhengig av folkeregisterets tilgjengelighet.

Integrasjonen mellom EVA Admin og EVA Manntall er basert på "jobber", og manntallsførte hentes og knyttes til disse.



Vedlagt en [presentasjon av EVA Manntall](#) som ble brukt i en avdelingsintern show-and-tell-sesjon onsdag 10/2 2021.

## Digitaliseringsdirektoratets tjenester for elektronisk distribusjon

### Innledning

Valgkort distribueres i økende grad elektronisk gjennom Digitaliseringsdirektoratets tjenester for post til innbyggere.

### Valgkortmodul

Det er utviklet en modul for integrasjon mot Digidirs kontakt- og reservasjonsregister (KRR) for å kunne distribuere valgkort elektronisk eller trykket på papir.

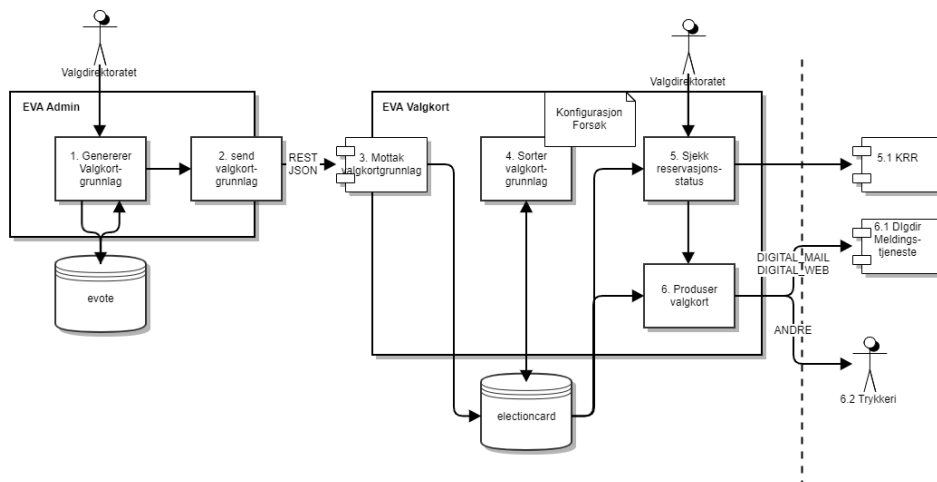
Inntil videre produserer modulen filer.

### Beskrivelse

Proessen kan kort beskrives som følger:

1. EVA Admin genererer valgkortgrunnlag, dvs. informasjonen som skal påføres den enkelte velgers valgkort
2. Valgkortgrunnlaget sendes EVA Valgkort
3. og mottas og lagres av EVA Valgkort
4. Valgkortgrunnlag der kommuner ikke er definert som kandidater for elektronisk distribusjon av valgkort merkes med "papir" - valgkort skal trykkes
5. Valgkortgrunnlag der kommuner er definert som kandidater for elektronisk distribusjon av valgkort vaskes mot kontakt- og reservasjonsregisteret for å avgjøre reservasjonsstatus og postkasseleverandør. Reserverte og utgåtte merkes med "papir" - valgkort skal trykkes, resten er kandidater for elektronisk distribusjon av valgkort
6. Fil for trykking av valgkort eksporteres og oversendes trykkeri. Fil for elektroniske valgkort eksporteres - full integrasjon med Digidir er inntil videre ikke på plass.

Skissen gir en overordnet illustrasjon av løsningen for elektronisk distribusjon av valgkort



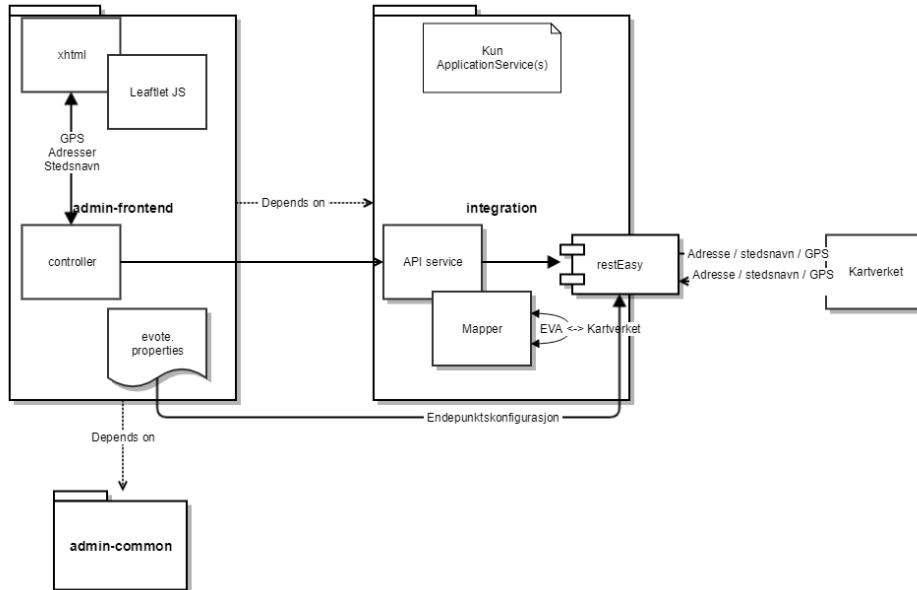


## Kartverket - kartdata

EVA Admin bruker Kartverkets åpne API for oppslag i kart i funksjonen for å definere forhånds- og valgtingstemmested. Det primære målet med å bruke denne tjenesten er å kunne hente ut GPS-koordinater som lagres på stemmestedene.

EVA Admin kjører integrasjonen i frontendmodulen, ikke lokalt i nettleser hos bruker.

Skisse over integrasjonen mot Kartverket:



## Vedlegg

### Systemkrav

Her beskrives anbefalinger for maskinvare for bruk av EVA Admin - klient for valget i 2025

Maskinvare	Beskrivelse
Bærbar PC med internetttilgang	<ul style="list-style-type: none"> <li>Minimum 4 GB internminne</li> <li>Minimum skjermopløsning på 1280x1024 pixler</li> <li>Støtte for utskrivning i lokalet                             <ul style="list-style-type: none"> <li>– for utskrift av informasjonskort, møtebøker (der det er relevant), og rapporter</li> </ul> </li> </ul>
Anbefalt tilbehør	<ul style="list-style-type: none"> <li>USB-mus</li> <li>Ekstra strømadapter pr. 5. PC</li> <li>Håndholdt USB strekkodeskanner                             <ul style="list-style-type: none"> <li>– HID kompatibel – dersom elektronisk manntall er valgt</li> </ul> </li> </ul>
Valgfritt tilbehør	<ul style="list-style-type: none"> <li>Innebygd smartkortleser dersom smartkort skal brukes for pålogging via ID-porten</li> <li>USB smartkortleser dersom smartkort skal brukes for pålogging via ID-porten</li> </ul>
Nødvendig programvare	<ul style="list-style-type: none"> <li>Nettleser: Chrome (siste versjon)</li> <li>For nettlelere: Javascript må være tillatt</li> </ul>

#### Forklaringer

**HID:** Human Interface Device – strekkodeleseren skal fungere som forlengelse av tastaturet slik at strekkoden leser inn tall i felt for manntallsnummer. HID-produkter er merket med forkortelsen HID eller Human interface Device.

**Javascript:** EVA bruker JavaScript. JavaScript er et skriptspråk som kjører i brukerens nettleser for å gi bedre funksjonalitet. EVA krever at JavaScript er aktivert. Hvis JavaScript er deaktivert vil ikke EVA fungere.

Valgdirektoratet anbefaler at du tar kontakt med egen IT-ansvarlig for spørsmål rundt anbefalingene.