2017

Kravspesifikasjon

ELEKTRONISK BESØKSREGISTER FOR NC-SPECTRUM

ERIK LI	S236777
ANDREAS STENSRUD	S236742
JOAKIM F. MØLLER	S196960
EMIL R. NEDREGÅRD	S236767

GRUPPE 19 | HØGSKOLEN I OSLO OG AKERSHUS

INNHOLD

1 PRESENTASJON

1.1 PRESENTASJON

Oppdragsgiver	NC-Spectrum	
Prosjekttittel	Elektronisk besøksregister for NC-Spectrum	
Oppgave	Prosjektet går ut på å utvikle et elektronisk besøksregister for NC-Spectrums operasjonssentral som kan erstatte deres nåværende utdaterte fysiske journalbok.	
Bedriftsnettside	https://www.nc-spectrum.no/	
Gruppenummer	19	
Gruppemedlemmer	Erik Li	s236777
	Andreas Stensrud	s236742
	Joakim F. Møller	s196960
	Emil R. Nedregård	s236767
Veileder	Terje Gjøsæter	
Gruppenettside	http://student.cs.hioa.no/~s236767/bachelornettside/	
Prosjektperiode	05.01.2017 – 24.05.2017	

1.2 GRUPPE

Gruppe 19 består av Erik Li, Andreas Stensrud, Joakim F. Møller og Emil R. Nedregård. Vi er en blanding av studenter som går på bachelorlinjene, ”Anvendt Datateknologi” og ”Informasjonsteknologi” ved Høgskolen i Oslo og Akershus, fra Høsten 2014 til Våren 2017.

Emil, Andreas og Joakim har arbeidet sammen gjennom studiet, men det er første gang Erik jobber sammen med dem. Vi har ulike kompetanser med tanke på hva vi kan og ikke kan. Erik er mer erfaren i programmering/back-end, Emil er ganske erfaren på front-end, GIT og serverhåndtering mens Joakim og Andreas er erfarne innenfor testing og dokumentasjon. Vi kommer nok til å fordele oppgavene slik som oppgitt, samtidig som alle kommer til å gjøre noe innenfor alle oppgavene og feltene.

2 FORORD

Kravspesifikasjonen er en beskrivelse av de kravene som NC-Spectrum har fremlagt til det elektroniske besøksregisteret som skal lages. Dokumentet vil inneholde beskrivelser av formål, behov, systemet, rammebetingelser og -krav, funksjonelle og ikke-funksjonelle krav og til slutt grafiske modeller av ulike kravspesifikasjoner.
Kravspesifikasjonsdokumentet kan sees på som en mer detaljert utdypning av forprosjektrapporten, som må godkjennes av begge parter som en modell for hvordan systemet skal videreutvikles og vil bli sett på som et styringsdokument. Dette vil da virke som en kontrakt mellom partene.

2.1 BAKGRUNN

NC-Spectrum er et konsulentselskap som holder til i Kviteseid, Telemark. Fokuset til bedriften er blant annet drift av elektronikk, infrastruktur i bredbåndsnett og etablering. Eksempler på kunder er små og mellomstore e-verk, vannverk og renseanlegg hvor cybersikkerhet på nettverk- og operativsystemer må ivaretas.

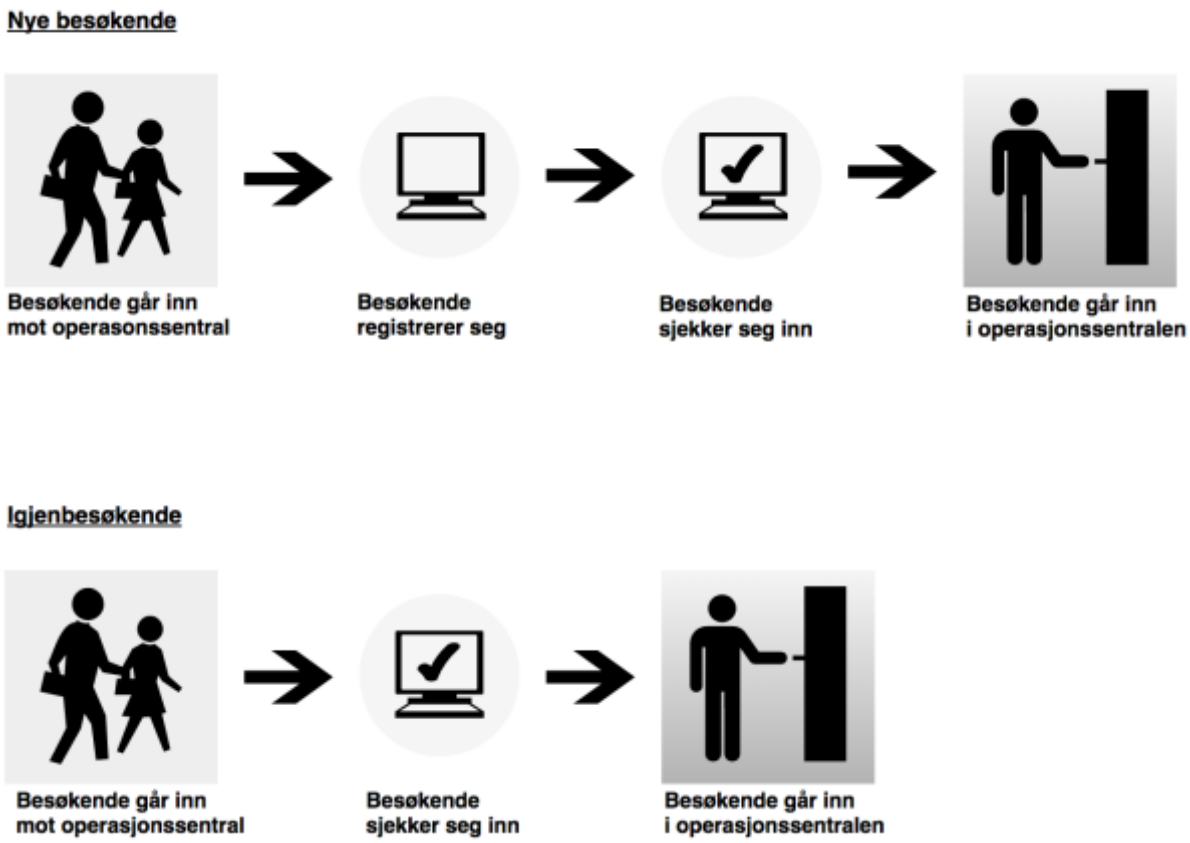
Hos NC-Spectrum har de operasjonssentralen som inneholder et møterom. Per dags dato gjøres registreringen via en fysisk bok. Vår oppgave er å digitalisere dette med hensyn til krav til sikkerhet og personvern slik at NC-Spectrum i fremtiden kan bruke denne webløsningen for registrering av besøkende. Et av de ønskede kravene er at webløsningen skal utvikles med PHP som hovedspråk, men ellers sto oppgaven relativt fritt.

For den besøkende skal webløsningen gjøre akkurat det samme som i boken, men med enklere registrering for brukere og mange automatiske bakgrunnsprosesser som logging, registrering og orden for senere bruk av administratorer. Hovedoppgaven til brukeren er å sjekke seg inn og ut samt opprette møter med ansatte. Administrator skal ha en egen sikker innlogging og sørge for at systemet holdes vedlike. Som nevnt skal admin kunne registrere brukere og ansatte, se historikk fra databasen og endre webløsningen ved behov.

2.2 SYSTEMBESKRIVELSE

Hensikten med vår oppgave er å lage et elektronisk besøksregister som håndterer besøkende hos NC-Spectrum slik at de ansatte til enhver tid vet hvem som er på deres operasjonssentral og hvor de besøkende kommer fra. Informasjon besøkende må registrere er; fullt navn, telefonnummer, e-post og hvilken bedrift de kommer fra. Besøkende kan ligge i systemet fra før av og skal da slippe å registrere seg på nytt neste gang vedkommende er hos bedriften. Man trenger dermed bare å søke på sitt eget navn. Når den vedkommende er registrert skal man kunne sjekke seg inn og ut som besøkende/ikke besøkende ved å dra eller klikke på navnet sitt. En registrert besøkende skal i tillegg ha muligheten til å opprette et møte med en ansatt. Den besøkende finner da den ansatte sitt navn via en liste og bestemmer hvor lenge møtet skal vare.

Administratorer skal ha tilgang til en egen del av webløsningen. Her kan de registrere nye gjester og ansatte, få oversikt og endre på informasjon om ansatte og gjester, se ulik statistikk over de besøkende og de skal kunne sjekke logg og historikk for alle hendelser som automatisk gjøres av systemet. Det er kun en egen superadministrator som skal ha muligheten til å opprette nye administratorer. Det skal også være mulig for administrator å se en grafisk fremstilling av databasen gjennom et «pai-diagram» slik at administrator kan se, endre og slette data direkte mot databasen. Det blir til slutt også tatt høyde for at administrator skal kunne endre webløsningen i fremtiden ved behov.



Systemmodell som viser hvordan nye og eksisterende besøkende må bruke besøksregisteret for å kunne sjekke seg inn i operasjonssentralen.

3 FUNKSJONELLE KRAV

De funksjonelle kravene beskriver hva systemet skal kunne gjøre og hvordan det skal oppføre seg. Det skal gi en god beskrivelse av hvilke funksjoner systemet skal tilby for sine brukere.

3.1 BESØKENDE

Følgende funksjonelle krav beskriver hvilke funksjoner som skal tilbys for de besøkende som skal benytte seg av løsningen:

	Krav	**Forklaring**	**Viktighetsgrad**
F1	Registrere seg	Besøkende skal kunne registrere seg som ny besøkende når vedkommende ankommer.	Absolutt
F2	Engangsregistrering	Besøkende skal kunne endre seg fra “ikke besøkende” til “besøkende” uten å registrere seg på nytt hvis vedkommende allerede er registrert i systemet.	Absolutt
F3	Sjekke inn	Besøkende skal kunne endre seg fra “ikke besøkende” til “besøkende” når de ankommer operasjonssentralen.	Absolutt
F4	Sjekke ut	Besøkende skal kunne endre seg fra “besøkende” til “ikke besøkende” når de forlater operasjonssentralen.	Absolutt
F5	Opprette møte	Besøkende skal ha muligheten til å opprette avtaler med de ansatte hos bedriften. De besøkende skal velge hvilken ansatt de skal møte samt varigheten på møtet.	Absolutt
F6	Liste over tidligere besøkende	Besøkende skal ha muligheten til å få opp en liste over tidligere besøkende slik at man kan finne seg igjen hvis man har vært der før.	Høy
F7	Livesearch	Besøkende skal ha muligheten til å søke på besøkende som ligger registrert fra før.	Høy
F8	Stå oppført som “ikke besøkende”	Besøkende skal stå som “ikke besøkende” hvis vedkommende er forhåndsregistrert eller har vært der tidligere.	Middels

3.2 ADMINISTRATORER

Følgende funksjonelle krav beskriver hvilke funksjoner som skal være mulig for administratorene å bruke:

	Krav	**Forklaring**	**Viktighetsgrad**
F9	Registrere besøkende og ansatte	Administrator skal ha muligheten til å registrere nye besøkende og ansatte	Absolutt
F10	Opprette nye administrator-brukere	En superadministrator skal ha muligheten til å opprette nye admin-brukere for f.eks ansatte. Ingen andre enn superadministratorer skal ha muligheten til å opprette nye administrator-brukere.	Absolutt
F11	Endre på og slette informasjon	Administrator skal ha muligheten til å føre endringer på registrerte besøkende og ansatte i databasen samt slette brukere. Innlogget administrator skal også ha mulighet til å endre på informasjon om seg selv.	Absolutt
F12	Superadministrator	En superadministrator må eksistere for å kunne opprette, endre og slette administrator-brukere.	Absolutt
F13	Sette gjester som ikke besøkende	Administrator skal ha muligheten til å fjerne gjester som "besøkende" hvis de har glemt å sjekke seg ut selv.	Høy
F14	Oversikt over nåværende besøkende	Administrator skal ha en oversikt over nåværende besøkende via database.	Høy
F15	Oversikt over alle registrerte personer.	Administrator skal ha en oversikt over alle registrerte, både gjester, ansatte, admins.	Høy
F16	Oversikt over alle besøk	Administrator skal ha en oversikt over alle besøk som skal skje eller har skjedd.	Høy
F17	Oversikt over historikk	Administrator skal ha et brukergrensesnitt som viser frem all historikk over all interaksjon med systemet.	Høy
F18	Sette profilbilde	Administrator skal ha muligheten til å legge til et profilbilde/avatar.	Middels
F19	Tilbakestille passord	Administrator skal ha muligheten til å tilbakestille passordet sitt.	Middels

3.3 SYSTEM

Følgende funksjonelle krav beskriver hva som forventes av selve systemet:

	Krav	**Forklaring**	**Viktighetsgrad**
F20	Ta vare på brukerinformasjon	Webløsningen skal ta vare på brukerinformasjon og det skal ligge trygt.	Absolutt
F21	Lagre registrerte personer	Webløsningen skal lagre alle registrerte personer i en database	Absolutt
F22	Lagre avtalte besøk	Webløsningen skal lagre alle nye avtalte besøk.	Absolutt
F23	Status	Webløsningen skal ha en tabell for administratorer som viser status (hvilke personer som er satt som sjekket inn og ikke).	Absolut
F24	Lagre oversikt over deltakere i hvert besøk	Webløsningen skal ha en tabell som inneholder en oversikt over hvem som deltar i hvert besøk.	Høy
F25	Logge alle hendelser	Webløsningen skal til enhver tid ha en bakgrunnsprosess som logger all aktivitet gjort av både brukere og administratorer.	Høy

4 IKKE-FUNKSJONELLE KRAV

Ikke-funksjonelle krav gir en beskrivelse av hvilke egenskaper systemet bør ha for å fungere optimalt. Dette inkluderer hvordan sikkerheten for systemet skal ivaretas, hvordan man skal sørge for at systemet er så pålitelig som mulig og hvordan systemet skal være så enkelt som mulig å bruke for alle personer:

	Krav	**Forklaring**	**Viktighetsgrad**
IF1	Programmeringsspråk	Webløsningen skal utvikles i PHP, HTML, CSS (Bootstrap), JavaScript (JQuery, JSON, AJAX) og bruke MySQL database.	Absolutt
IF2	Rammeverk	Som rammeverk for PHP skal Laravel brukes for utviklingen.	Absolutt
IF3	Server	Webløsningen med alt innhold skal ligge på oppdragsgivers egne servere for å ivareta sikkerhet	Absolutt
IF4	Universell utforming	Webløsningen må tilpasses alle typer enheter og nettlesere.	Høy
IF5	Sikkerhet	Webløsningen skal være sikker og beskyttet mot ulike angrep. Blant annet sikker mot SQL-injection og CSRF.	Høy
IF6	Skalerbarhet	Webløsningen skal ha lav kobling og høy kohesjon, slik at senere bygging på systemet kan gjøres uten problemer.	Høy
IF7	Nettleserkompatibilitet	Webløsningen skal være kompatibel med alle nettlesere	Høy
IF8	Ytelse	Webløsningen skal ha god ytelse og rask responstid.	Middels
IF9	MVC	Model-View-Controller skal brukes for å skille brukergrensesnitt og data fra hverandre. Laravel bygger på MVC-prinsippene.	Middels

5 RAMMEKRAV

5.1 KRAV TIL DESIGN

- Utvikle webløsningen slik at de fleste fonter i webløsningen er like og i passelig størrelse.
- Utvikle webløsningen slik at fargene har god kontrast i forhold til hverandre og være komfortabel for øynene.
- Utvikle webløsningen med passelig størrelse på knapper og andre interaksjons- gjenstander.
- Utvikle webløsningen slik at all interaksjon kommer tydelig frem, er selvforklarende og virker naturlig.

5.2 KRAV TIL UNIVERSELL UTFORMING

- Språk skal være engelsk
- Utvikle webløsningen med hensyn til lite IT-erfarne brukere og eventuelle funksjonshemninger. Løsningen skal i så stor grad som mulig følge WCAG 2.0 sine prinsipper.
- Tilpasse slik at det blir god skalerbarhet med alle typer enheter.

5.3 KRAV TIL DET TEKNISKE

- God struktur og dokumentering.
- Utvikle webløsningen med oversiktlig, forståelig og selvforklarende kode.
- Webløsningen skal være forståelig nok slik at den lett kan endres på av NC-Spectrum sine ansatte senere, etter prosjektslutt.

5.4 KRAV TIL SIKKERHET

- Skal kun være tilgjengelig på NC-Spectrum sitt intranett.
- Feltvalidering med Regular Expression.
- Passord skal krypteres sammen med tilfeldig generert salt i databasen.
- Unngå SQL-Injection
- Beskyttelse mot Cross-site Request Forgery.
- Besøkende skal kun få tilgang på sider som er relevant for dem.
- Bare superadministratorer og administratorer skal ha tilgang på administratorsider
- All data skal lagres trygt i en database
- Validering av data skal foregå på både klient og server.
- Kun administratorer skal kunne endre på brukere for besøkende og ansatte.
- Kun superadministrator skal kunne endre på andre administratorer.

5.5 KRAV TIL LAGRING I DATABASE

- Bygge opp databasen slik at den er tilpasset for normalisering.
- Bygge opp databasen slik at den er godt strukturert og inneholder naturlige tabeller.
- Utvikle webløsningen slik at man alltid har nok kapasitet for lagring.
- Sørge for at man alltid har backup av alt innhold i tilfelle ødeleggelse, tyveri etc.
- Databasen skal ikke lagre unødvendig og urelevant informasjon.

6 BRUK OG BRUKERVENNLIGHET

Ved at et brukergrensesnitt er ryddig, enkelt, forståelig og brukervennlig økes effektiviteten og opplevelsen til brukeren. Det gir fleksibilitet og læringsprosessen blir rask og behagelig for brukerne av systemet.

Som besøkende av NC-Spectrum ønsker man at registreringsprosessen er så kort og enkel som mulig. Derfor er det viktig med et ryddig og brukervennlig system, noe NC-Spectrum klart og tydelig spesifiserte. Minimalt med tekst, kontrastfylte farger, tydelige figurer og knapper, og et universelt design er mye av nøkkelen for å oppnå kortest mulig læringsprosess slik at brukerne raskt og enkelt forstår hva som må gjøres for å oppnå målet i systemet de interagerer med.

Administratorgrensesnittet som administratorene bruker vil ha mye mer funksjonalitet som gjør at brukergrensesnittet blir mer komplekst, mindre effektivt og læringsprosessen vil ta lengre tid. Administratorene har derimot mer tid til å bli kjent og lære seg deres brukergrensesnitt. Derfor er optimal funksjonalitet førsteprioritet, og optimal brukergrensesnitt er andreprioritet.

7 SIKKERHET

Det settes krav til sikkerhet da brukerne av systemet legger ved sensitiv informasjon som må sikres. Vi har i forprosjektet snakket om at webløsningen skal ligge på NC-Spectrums egne servere, slik at man får tilgang til portalen bare når man er tilkoblet på deres nett. Dette reduserer sannsynligheten for angrep over internett, og det er heller ikke nødvendig at den ligger på nett da man kommer til å bruke webløsningen bare når man er fysisk tilstede ved NC-Spectrum.

En av hoved sikkerhetsteknikkene som skal benyttes i webløsningen vår er passordkryptering og salting. Dette skal vi skrive litt om nedenfor. Mer grundig og detaljert forklaring for resten av sikkerhetsteknikkene/ -mekanismene som benyttes vil bli beskrevet i sluttrapporten vår.

Alle passord som lagres i systemet skal krypteres med godkjente krypteringsalgoritmer som for eksempel MD5 (Message-Digest Algorithm 5) eller SHA (Secure Hash Algorithm) og ved bruk av tilfeldig-genererte salter. Fordelen med dette er at; dersom noen får tak i passord-databasen vil man ikke kunne se hva passordet faktisk er, da en slik type kryptering kun går “en vei” og er svært vanskelig om ikke umulig å benytte seg av “reverse engineering”. Krypteringsalgoritmene må være godkjente, sikre og ikke minst bestått “the test of time”. Godkjente krypteringsalgoritmer er som regel bygget opp av avanserte matematiske formler som gjør at det vil ta svært lang tid å gjette seg til det riktige passordet (brute-force attack).

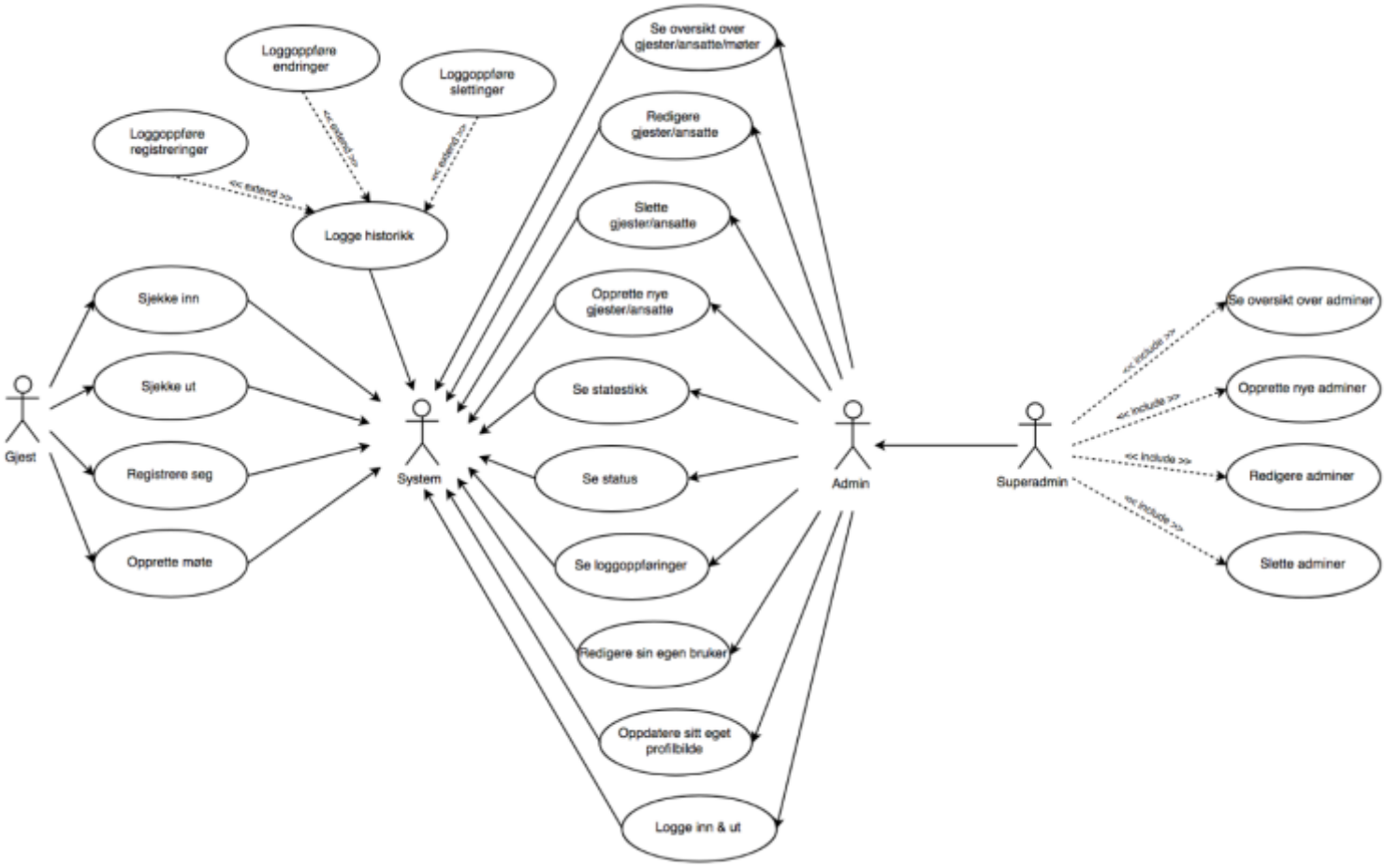
8 FREMTIDIG UTVIDELSE AV SYSTEMET

Som nevnt tidligere kommer vi til å utvikle webløsningen i PHP. Dette var ønskelig fra NC-Spectrum da PHP var noe de selv var komfortable med. Vi skal også bruke Laravel som er et PHP rammeverk. Vi kom også til enighet om at koden må utvikles på en veldig lett og forståelig måte, med god dokumentering og generelle uttrykk, variabel- og funksjonsnavn som alle utviklere vil kunne kjenne igjen. Dette er nyttig for at oppdragsgiver kan endre funksjonalitet der de mener det er hensiktsmessig etter at vi har fullført bachelorprosjektet og de skal ta over webløsningen.
Vi kommer også til å lage diagrammer som klassediagram, ER-modell, Use Case osv, for å vise hvordan koden er strukturert. Dette gjør det lettere for oppdragsgiver å få et helhetlig bilde om hvordan systemet ser ut slik at de slipper bruke mye tid på å lete i koden hvis de ønsker å gjøre endringer.

9 DIAGRAMMER

I dette avsnittet skal vi vise frem og skrive litt om tre ulike diagrammer/modeller som vi har laget som representerer tre ulike lag av systemet.

9.1 USE CASE (BRUKSTILFELLE)



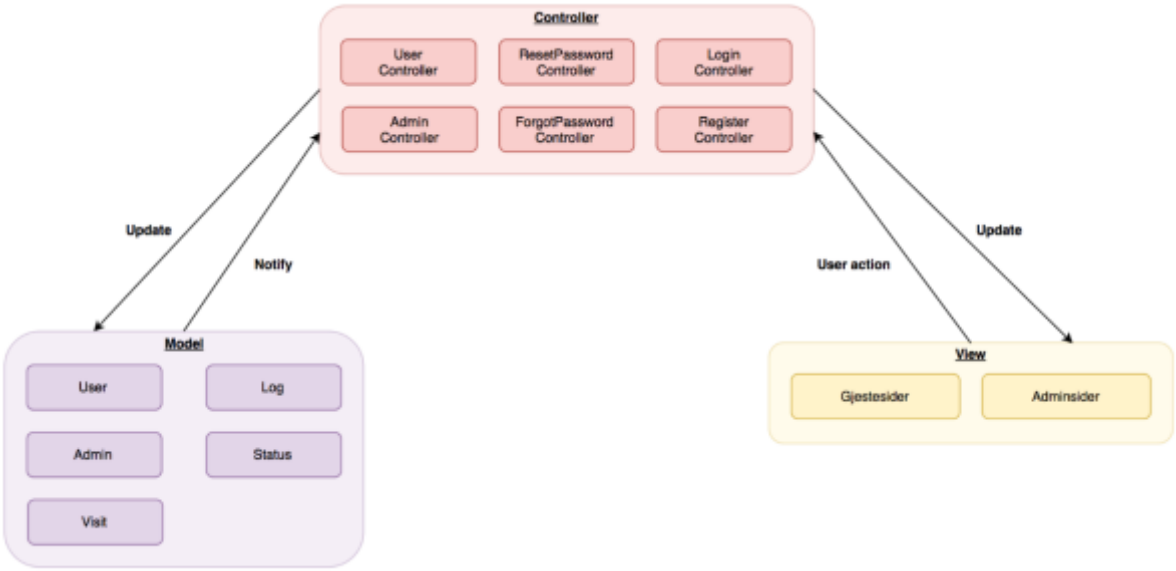
Use-case-diagrammer brukes innenfor systemutvikling og er en modell som representerer ulike handlinger og steg som utføres. Typisk er å definere interaksjonen mellom ulike aktører (roller) som brukere og systemer. En aktør kommuniserer med systemet via ett eller flere use cases. Det finnes to typer hovedaktører:

Primæraktøren har sine egne mål, dvs de initierer use-cases som oppfyller deres mål.
Sekundæraktører har ikke egne mål, men er nødvendige for å realisere målene til primæraktørene.

Dette use-caset er bygget opp på de funksjonelle kravene som er beskrevet i seksjonen om funksjonelle krav.

Vi har tre primæraktører; gjest, administrator og superadministrator, og en sekundæraktør; system som brukes for å oppnå gjestens og administratorens mål.
Her ser vi at gjesten som ankommer NC-Spectrum skal kunne registrere seg i systemet, sjekke inn eller sjekke ut som "besøkende" eller "ikke besøkende" og opprette møter. Dette vil registreres i systemet, og systemet vil håndtere nødvendige tiltak for å fullføre use-casene.
Administratoren vil ha et separat brukergrensesnitt fra gjestenes og har flere ulike valg og funksjoner som de kan interagere med. Superadministratoren vil ha litt annerledes brukergrensesnitt fra administratoren da superadministrator har et par tilleggs valg og funksjoner som de kan interagere med, siden de har høyere rettighet. Alle use-casene som administratorene og superadministrator ser og bruker er knyttet opp mot systemet som bevarer, håndterer og presenterer all relevant informasjon.
Det er også verdt å nevne at systemet vil ha en type "logg/historikk" som logger og lagrer all interaksjon som påføres systemet, som administratoren vil få representert via en databasetabell.

9.2 MVC (MODEL-VIEW-CONTROLLER)



MVC er et designmønster innenfor systemutvikling. Designmønsteret deler systemet i tre ulike komponenter slik at informasjonen som blir presentert og håndtert internt i systemet, er ulikt fra informasjonen som blir presentert til brukerne. Laravel bruker som nevnt MVC som utgangspunkt.

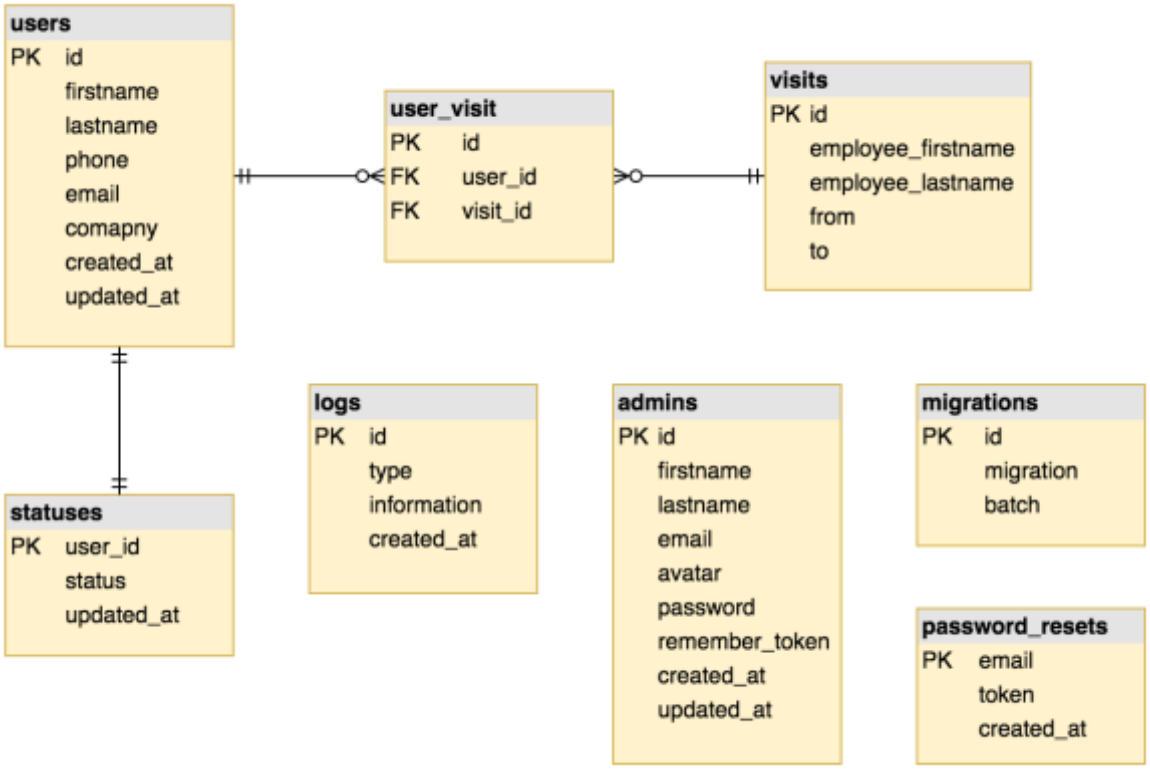
Komponentene i MVC:

- **Model** er en sentral komponent for designmønsteret. Den håndterer oppførselen til systemet og håndterer data, logikk og regler direkte.
- **View**-komponenten representerer det brukeren ser, som brukergrensesnitt eller grafer.
- **Controller** er komponenten som håndterer input og output mellom modell-komponenten og view-komponenten. Det vil si den komponenten som håndterer det interne, slik at det interne blir representert ulikt for brukerne, og vice versa.

Det er noen punkter som er verdt å nevne i MVC-modellen:
Modell-komponenten representerer de ulike klassene og modellene som vi kommer til å bruke i systemet. Her er det verdt å nevne at både ansatte og gjester vil ligge i "User" -modellen, og administratorer vil ha sin egen modell. Måten man skiller mellom ansatte og gjester gjøres ved bruk av bedriftsnavnet. Ansatte vil ha bedriftsnavnet "NC-Spectrum" og gjestene kan ha alt annet enn det. Mer om dette blir skrevet under punktet om ER-modell.

Controller-komponenten vil ha flere controllere, blant annet en for brukere og en for administratorer. Dette er for sikkerhetsmessig grunner, slik at brukere ikke eventuelt skal få tilgang til funksjoner som tilhører administratoren.

9.3 ER-MODEL (ENTITY-RELATIONSHIP MODEL)



ER-Modell, Entity-Relationship-Model beskriver domenekunnskap innenfor databasene. En ER-modell består av entitetstyper som representerer hovedklasser eller ting som er mest relevant og deres forhold til hverandre.

- **Admins**: En tabell som inneholder informasjon om administratorbrukerne i webløsningen. Det er kun disse brukerne som kan aksessere administratorløsningen.
- **Users:** En tabell som inneholder alle registrerte besøkende og ansatte. Besøkende og ansatte skilles ved bedriftsnavnet. Ansatte har "NC-Spectrum" som bedriftsnavn.
- **Visits:** En tabell som inneholder alle møter som har vært. Hver rad vil representere et møte med start- og sluttid samt den ansatte som har hatt ansvaret for møtet.
- **Uservisit:** Denne tabellen eksisterer som resultat av oppløsningen av mange-til-mange relasjonen mellom tabellene **users** og **visits**. Tabellen inneholder relasjoner mellom hvilke møter som besøkende har deltatt i.
- **Statuses:** En tabell som inneholder statusen til alle besøkende. Hver besøkende vil ha en relatert rad i denne tabellen som representerer om de er sjekket inn i møte eller ikke, samt med en kolonnen for når de sist var sjekket inn i et møte. Tidsstemplet kan også representerer når de var sist på besøk hos NC-Spectrum.
- **Logs:** En tabell som lagrer et innlegg av ulike handlinger som har blitt utført av administrator. Tabellen kan inneholde innlegg som hvem administrator slettet, endret eller opprettet osv.
- **Migrations:** Dette er en tabell som Laravel benytter seg av når den skal generere alle tabellene i databasen. Blant annet holder den styr på om tabellene som skal opprettes allerede finnes i databasen eller ikke.
- **Password_resets:** Tabellen kreves for at administratorbrukere skal kunne tilbakestille passordet sitt hvis de har glemt det. Hvis administratoren ber om å tilbakestille passordet vil mailadressen til administratoren bli lagret sammen med en nygenerert token. Tidsstemplet representerer når forespørselen ble gjort.