PRO203

SMIDIG PROSJEKT

PROSJEKTRESULTAT

Høyskolen Kristiania

Gruppenummer: 58

Gruppemedlemmer:
Ayanle Mohamed - 549
Barkhad Isse - 486
Mikkel Villard - 460
Christoffer Heien - 32
Tobias Kløfta - 197
Kristian Melgård - 628
Johannes Evensen - 293

1.0 Beskrivelse av løsningen

Den tekniske løsningen skal beskrives både funksjonelt, visuelt og teknisk. Dersom løsningen er tilgjengelig, skal det beskrives hvordan man finner og får tilgang til den.

Det skal også eventuelt forklares hvordan løsningen kan settes opp og kjøres basert på vedlagt kildekode, inkludert beskrivelse av krav til utviklingsmiljø.

Prosjektet inkluderer en README som beskriver hvordan løsningen settes opp og kjøres lokalt, inkludert krav, databaseoppsett, oppstart av backend og frontend, samt nødvendige API-headere for test og demonstrasjon.

Førstegangsregistrering og etablering av tilgang

Selv om løsningen vi leverer er en MVP, har vi brukt tid på å tenke grundig gjennom hvordan førstegangsregistrering og oppstart av systemet burde fungere i praksis. Dette er ikke funksjonalitet som er ferdig implementert i kode, men en bevisst del av systemlogikken og designet slik vi mener det burde vært løst i et reelt barnehagesystem. Førstegangsregistreringen henger tett sammen med både informasjonsflyt, rollebasert tilgang og GDPR-arbeidet som er beskrevet senere i rapporten.

Utgangspunktet vårt er at barnehagen allerede sitter på nødvendig grunndata om barn, foresatte og ansatte. I stedet for å legge ansvar og valg over på sluttbrukerne, er løsningen lagt opp slik at barnehagen gjør et kontrollert oppsett i forkant. Ved starten av et nytt barnehageår registrerer barnehageleder barna, knytter foresatte til riktig barn, og legger inn ansatte med tilhørende avdelinger. Når dette er gjort, har systemet all informasjonen som trengs for å tildele riktige roller og tilganger senere.

Når en foresatt eller ansatt tar i bruk løsningen for første gang, er tanken at systemet skal kunne identifisere brukeren basert på kontaktinformasjon som allerede er registrert av barnehagen. Brukeren skal derfor ikke selv måtte velge barn, avdeling eller rolle. Disse koblingene er allerede definert i systemet, og tilgangen etableres automatisk. Etter førstegangsoppsettet fungerer løsningen som en helt vanlig app, der brukeren logger inn og umiddelbart får tilgang til riktig informasjon og funksjonalitet.

Denne måten å løse førstegangsregistrering på går hånd i hånd med resten av arkitekturen og logikken i systemet. Den støtter rollebasert tilgang ved at alle tilganger er forhåndsdefinert og kontrollert, og den reduserer risiko for feilbruk ved at brukere ikke kan koble seg til feil barn eller få for bred tilgang. Samtidig gir dette en enklere brukeropplevelse, der fokus ligger på bruk av løsningen i hverdagen, ikke på oppsett og valg.

Løsningen ivaretar også GDPR på en bedre måte enn alternative tilnærminger der brukere selv søker opp eller velger barn og roller. Ved å basere førstegangsregistreringen på

eksisterende grunndata, sikres det at det kun gis tilgang til informasjon brukeren faktisk har behov for. Dette støtter prinsippene om dataminimering og innebygd personvern, og bidrar til å redusere både feil og unødvendig eksponering av personopplysninger.

Selv om denne førstegangsregistreringen ikke er ferdig kodet i MVP-en, er den et viktig designelement i løsningen og viser hvordan systemet er tenkt brukt i praksis. Sammen med informasjonsflyt, RBAC og GDPR-arbeidet som er dokumentert i rapporten og vedleggene, gir dette et helhetlig bilde av hvordan løsningen kunne fungert som et realistisk og sikkert system i reell drift.

Førstegangsregistrering, brukerlogikk og rollebasert tilgang er beskrevet i større detalj i vedlegget *«Brukerregistrering og RBAC»*.

GDPR

Løsningen er utviklet med personvern som et grunnleggende premiss, og GDPR har hatt direkte innvirkning på både funksjonalitet, arkitektur og databehandling. Med utgangspunkt i Frostbyte AS-caset, der dagens Excel-baserte løsning mangler tilgangsstyring og sporbarhet, har målet vært å etablere en strukturert og dokumentert håndtering av personopplysninger om barn, foresatte og ansatte.

Et sentralt resultat av GDPR-arbeidet er en konsekvent tilnærming til dataminimering. Løsningen behandler kun opplysninger som er nødvendige for å registrere og vise barns tilstedeværelse, som navn, avdeling og inn- og utsjekkstatus. Det lagres ikke fødselsnummer, bilder eller andre unødvendige personopplysninger. Helseopplysninger, som allergier, er vurdert særskilt som sensitive data etter artikkel 9 og er dokumentert som et avgrenset og kontrollert unntak der barnets sikkerhet tilsier det.

Tilgang til personopplysninger er styrt gjennom rollebasert tilgangskontroll implementert i backend. Foreldre har kun tilgang til informasjon om egne barn, ansatte har tilgang basert på sitt arbeidsbehov, og administrative roller har systemtilgang uten innsyn i sensitiv informasjon. Denne tilnærmingen støtter prinsippet om innebygd personvern og sikrer at tilgang vurderes før data returneres, uavhengig av frontend.

GDPR har også påvirket hvordan løsningen håndterer feil og logging. Det er definert klare retningslinjer for trygg respons, der brukere kun mottar generiske feilmeldinger og aldri får informasjon som kan avsløre interne detaljer eller eksistensen av data de ikke har tilgang til. Logging er begrenset til tekniske metadata som er nødvendige for drift og feilsøking, og personidentifiserende informasjon logges ikke. Dette bidrar til å ivareta kravene til integritet, konfidensialitet og ansvarlighet.

Samlet sett har GDPR-arbeidet ført til at løsningen fremstår som en tydelig forbedring sammenlignet med utgangspunktet i Frostbyte AS-caset. Selv om løsningen leveres som en MVP, viser den hvordan sentrale personvernprinsipper kan omsettes til konkrete tekniske og arkitektoniske valg. Dokumentasjonen som følger løsningen gir samtidig et godt grunnlag for videreutvikling mot en produksjonsklar og fullverdig GDPR-etterlevelse.

Funksjonell beskrivelse av løsningen

Den utviklede løsningen er en digital innsjekkings- og informasjonsløsning for barnehager, med mål om å erstatte dagens praksis med delte Excel-ark. Løsningen er utformet med særlig vekt på enkelhet i bruk, tydelig rollefordeling og ivaretakelse av personvern. Funksjonelt er løsningen bygget rundt behovene til to hovedbrukergrupper: foresatte og ansatte i barnehagen. Foresatte får tilgang til en personlig visning der de kun kan se og registrere informasjon knyttet til egne barn, som inn- og utsjekk samt enkel meldingsutveksling med barnehagen. Ansatte får på sin side tilgang til en mer operativ oversikt, der de kan se barn fordelt på avdelinger, følge med på inn og utsjekkstatus, og håndtere relevant informasjon i løpet av barnehagedagen. All funksjonalitet er bevisst avgrenset for å redusere kompleksitet og risiko for feilbruk, samtidig som løsningen dekker kjernebehovene som ble identifisert i innsiktsfasen.

Visuell utforming og brukeropplevelse

Den visuelle utformingen av løsningen tar utgangspunkt i et mobil og nettbrettvennlig grensesnitt, inspirert av eksisterende applikasjoner som brukerne allerede er kjent med. Navigasjonsflyten er lineær og forutsigbar, der brukeren beveger seg fra innlogging til oversikt, videre til barnedetaljer og deretter til konkrete handlinger som inn eller utsjekk. Designet er utviklet i Figma og brukt aktivt som et verktøy både for konseptutvikling, brukertesting og som referanse under den tekniske implementasjonen. Visuelt er det lagt vekt på rolig fargebruk, tydelig typografi og begrenset informasjonsmengde per skjerm, for å støtte prinsippet om “privacy by default” og redusere kognitiv belastning for brukeren.

Teknisk arkitektur og løsningsstruktur

Teknisk er løsningen bygget som en todelt arkitektur med en backend og en enkel frontend. Backend er utviklet i Python ved bruk av FastAPI, og eksponerer et REST-basert API som håndterer all forretningslogikk, datatilgang og tilgangsstyring. API-et er strukturert med tydelige endepunkter for håndtering av barn, foresatte, ansatte, avdelinger og registrering av inn og utsjekk. For å sikre korrekt tilgang til data er det implementert rollebasert tilgangskontroll (RBAC), der foresatte kun får tilgang til egne barn, ansatte får tilgang til barn i tilknyttede avdelinger, og administrative roller har utvidede rettigheter. Denne tilnærmingen er valgt for å møte både funksjonelle krav og krav knyttet til personvern og GDPR.

Datagrunnlag, sikkerhet og personvern

Datagrunnlaget er i første fase løst med en enkel JSON-basert MVP-løsning for rask utvikling og testing, før det er lagt til rette for bruk av en relasjonsdatabase i MySQL. Databasestrukturen er definert i et eget schema som beskriver relasjoner mellom brukere, barn, avdelinger og hendelser som inn- og utsjekk. Overgangen fra JSON til database reflekterer en bevisst prioritering av læring og iterasjon tidlig i prosjektet, før mer robuste og skalerbare løsninger ble tatt i bruk. Backend inneholder også felles feilhåndtering og logging, der tekniske hendelser loggføres uten å eksponere sensitiv informasjon til sluttbruker.

Frontend og samspill med backend

Frontend er implementert som en simpel løsning med HTML og JavaScript, og fungerer primært som et grensesnitt for å demonstrere og teste samspillet mellom bruker og API. Frontenden er tett knyttet til designet i Figma og speiler de viktigste brukerflytene i løsningen. Språkvalg er implementert og lagres lokalt, slik at brukeren får et konsistent grensesnitt mellom økter. Selv om løsningen ikke er ment som et ferdig produksjonsklart produkt, gir den et realistisk bilde av hvordan løsningen kan fungere i praksis.

Oppsett og kjøring av løsningen i utviklingsmiljø

For å sette opp og kjøre løsningen lokalt kreves et utviklingsmiljø med Python, FastAPI og MySQL. Backend konfigureres ved hjelp av miljøvariabler definert i en .env-fil, blant annet for tilkobling til databasen. Databasen opprettes ved å kjøre det vedlagte SQL-schemaet, som etablerer nødvendige tabeller og relasjoner. Backend startes deretter ved bruk av Uvicorn, og kan i utviklingsfasen kjøres med såkalt mock-autentisering, der rolle og brukeridentitet sendes som headere i API-kall. Dette har gjort det mulig å teste tilgangsstyring og funksjonalitet uten å implementere full autentiseringsløsning i tidlig fase. Frontenden kan kjøres som statiske filer ved hjelp av en enkel lokal webserver.
Dersom man støter på trøbbel med kjøring av løsningen gir README-filen i kildekoden en samlet og oversiktlig veiledning for oppsett, konfigurasjon og kjøring av løsningen lokalt.

2.0 Leveransetest / Demo med annen gruppe

I stedet for å gjennomføre en tradisjonell leveransetest valgte vi å gjøre en praktisk demo av løsningen vår. Under denne demoen viste vi de viktigste funksjonene i applikasjonen og gikk gjennom sentrale deler av kildekoden sammen med en annen gruppe i emnet Smidig Prosjekt. Demoen gjorde det mulig å få konkrete tilbakemeldinger både på funksjonalitet, tekniske valg og helhetlig flyt i løsningen.

Gruppen vi samarbeidet med var gruppe Istanbul, med Øyvind Glimsdal Bakke som kontaktperson. Basert på demoen stilte vi noen spørsmål som gruppen besvarte.

Hva var bra med løsningen?

Svar:
“Leveransen viser at sentral funksjonalitet fungerer som forventet. Det er mulig å hente lister over avdelinger og barn, utføre inn- og utkryssing ved bruk av gyldige headere og ID-er, vise foreldreinformasjon i barnedetaljer og lagre kommentarer. Data hentes fra en MySQL-database som er initialisert via schema og seed-data, noe som gir en realistisk

demonstrasjon av systemets kjerneflyt og samspillet mellom frontend, backend og database.”

Hvilke hindringer eller mangler hadde løsningen?

Svar:
“Løsningen har noen mangler. Statusen «kommer senere» er ikke ferdig implementert og lagres ikke i databasen. API-ene er heller ikke helt like i hvordan de returnerer data, og feilhåndteringen er ikke tilpasset et ferdig produksjonssystem med strenge GDPR-krav. I tillegg må brukeren gå til innstillingssiden og laste appen på nytt for å bytte språk, noe som gjør brukeropplevelsen litt dårligere. Vi skjønner at dette er mer enn MVP, og det virker som dere har kontroll på teorien, så eventuelle videre sprinter hadde dere nok klart å videre skalere til feks react!”

Hva ga demoen oss av læring videre?

Demoen gjorde det tydelig hvilke deler av løsningen som fungerer godt, og hvilke områder som bør prioriteres videre. Spesielt ble det klart at kjerneflyten i systemet er på plass, men at det gjenstår arbeid knyttet til konsistens i API-responser, bedre feilhåndtering og en mer sømløs brukeropplevelse. Tilbakemeldingene bekreftet også at arkitekturen er et godt utgangspunkt for videre utvikling, selv om enkelte funksjoner fortsatt mangler før løsningen kan regnes som komplett.

Resultat av leveranse (Egenvurdering)

Hva har vi levert i forhold til hva som var “definition of done” fra sprint 1 og sprint 2.

Har vi kommet i mål med frontend?

Frontend-løsningen vurderes som ferdig på MVP- og demonstrasjonsnivå, men ikke produksjonsklar. Koden viser at løsningen har tydelig separerte brukerflyter for foresatte og ansatte, med egne sider for innlogging, oversikt, detaljvisning, statusbekreftelse og innstillinger. Navigasjon, tilbakeflyt og utlogging fungerer konsistent, og grensesnittet gir en realistisk fremstilling av hvordan løsningen er ment å brukes i praksis.

Frontend benytter faktiske API-kall mot backend gjennom JavaScript-kode som er strukturert i flere filer, med tydelig ansvar per funksjon og visning. Disse filene håndterer blant annet uthenting av barn, innsjekk, utsjekk og kommentering via relevante endepunkter. Rolleinformasjon sendes eksplisitt via HTTP-headere, noe som muliggjør simulering av ulike brukere og roller i leveransetesten. Dette viser at frontend ikke kun er en visuell prototype, men en fungerende klient mot backend.

Samtidig avdekker leveransetesten klare avgrensninger:

- autentisering er simulert og ikke sikker
- frontend håndterer i liten grad komplekse feilsituasjoner
- sikkerhet og tilgang er fullt ut delegert til backend

Frontend oppfyller dermed **Definition of Done for Sprint 2**, som var å kunne demonstrere løsning og brukerflyt, men er ikke ferdig i betydningen robust og produksjonsklar applikasjon.

Har vi kommet i mål med backend?

Backend-løsningen når målet for en demonstrasjons- og MVP-versjon, men har klare mangler før den kan regnes som produksjonsklar.

Funksjonelt dekket

Backend er implementert med FastAPI og benytter MySQL som database via db_service.py. Løsningen eksponerer endepunkter for håndtering av avdelinger, barnelister, barnedetaljer (inkludert foreldreinformasjon), innsjekk, utsjekk og kommentarer. Tilgangsstyring er løst gjennom rollebasert tilgangskontroll (RBAC), der rolle- og kontekstinformasjon sendes via HTTP-headere (X-Role, X-Department, X-Parent-Id, X-User-Id) og valideres mot data i databasen.

Databaseskjema og seed-data er definert i schema.sql, noe som gjør det mulig for andre å sette opp databasen (barnehage_db) og kjøre løsningen lokalt. CORS er aktivert, og frontend kommuniserer med backend gjennom faktiske API-kall, noe som bekrefter at backend fungerer som en reell tjeneste og ikke kun som en mock.

Avgrensninger og risiko

Samtidig avdekker leveransen flere tydelige avgrensninger:

- Det finnes ingen standardisert responsstruktur (for eksempel { success, data, error }), og endepunktene returnerer enten rå Pydantic-modeller eller standard FastAPI-feil.
- Exception-håndtering er i hovedsak FastAPI sin standard, noe som kan føre til at valideringsfeil og ufangede feil eksponerer tekniske detaljer, med potensielle konsekvenser for sikkerhet og personvern (GDPR).
- Autentisering er simulert via HTTP-headere, og det er ikke implementert ekte innlogging, token-basert autentisering eller sesjonshåndtering.
- Funksjonalitet som er merket som “kommer senere” mangler egne modeller og endepunkter. Status støtter kun checked_in, checked_out og not_arrived, basert på innsjekk- og utsjekklogikken.

- Kommentarer lagres sammen med innsjekk-/utsjekk-data, og det finnes ingen separat meldingsmodell eller dedikert visning.

- Logging, overvåkning og sikkerhetsherding er ikke implementert, inkludert tiltak som rate limiting, tydelig input-sanitiseringspolicy og redigering av sensitive felt i logger.

Status opp mot Definition of Done

Backend oppfyller Definition of Done for Sprint 2 og demonstrasjon, ved at den leverer nødvendige data og handlinger til frontend og kan settes opp direkte fra schema.sql. Derimot oppfyller den ikke kravene til en produksjonsklar løsning. For å nå dette nivået kreves blant annet styrket autentisering og autorisasjon, standardisert feilhåndtering og responsstruktur, utvidet statusmodell, bedre GDPR-tilpasning og etablering av grunnleggende operasjonelle kontroller.

Kort oppsummert kan backend demonstrere en full brukerflyt med ekte database og fungerende API-er, men krever ytterligere arbeid for å bli robust, sikker og egnet for produksjonsbruk.

-Hva har vi klart av GDPR? Hva må evt endres dersom vi har en sprint til?

GDPR-arbeidet i prosjektet er levert som en kombinasjon av omfattende dokumentasjon og delvis teknisk implementasjon. Dokumentasjonen viser at personvern har vært et gjennomgående vurderingstema, og at sentrale GDPR-prinsipper er identifisert, konkretisert og i stor grad omsatt til tekniske valg i løsningen.

I den implementerte koden ser vi at flere grunnleggende personvernmekanismer er på plass. Rollebasert tilgang håndheves konsekvent i backend, slik at foresatte kun får tilgang til egne barn, og ansatte kun får tilgang til barn i egen avdeling. Tilgang vurderes serverside før data returneres, noe som støtter prinsippet om innebygd personvern. Videre lagres inn- og utsjekk-hendelser med tidspunkt og teknisk bruker-ID, noe som gir nødvendig sporbarhet uten å lagre unødvendige personopplysninger.

Dataminimering er tydelig dokumentert gjennom felt- og rollematriser, og prinsippet er delvis realisert i kode. Samtidig viser leveransetesten at enkelte endepunkter alltid returnerer detaljer om foresatte, som navn og telefonnummer. Uten et eksplisitt filtreringslag på responsnivå innebærer dette at dataminimering i praksis er avhengig av korrekt bruk av rolle og endepunkt, og ikke håndheves konsekvent som en egen teknisk mekanisme. Dette representerer et viktig forbedringspunkt.

Logging og trygg respons er grundig beskrevet i dokumentasjonen, med klare retningslinjer for hvilke data som skal og ikke skal logges. I den faktiske implementasjonen er disse prinsippene kun delvis realisert. Det finnes ingen sentral mekanisme som garanterer at persondata aldri logges, og feilhåndtering er i stor grad basert på standard unntakshåndtering. Dette skaper et gap mellom dokumentert måltilstand og ferdig løsning, særlig sett opp mot kravene i GDPR artikkel 5 og 32.

Forbedringer og prioriteringer i en kommende sprint

Dersom prosjektet hadde hatt en ekstra sprint, ville fokus i stor grad vært rettet mot å redusere gapet mellom dokumentert personvernambisjon og teknisk implementasjon. Følgende tiltak vurderes som særlig viktige:

- Implementere et eksplisitt filtreringslag for API-responser, slik at dataminimering håndheves konsekvent uavhengig av endepunkt og rolle.
- Etablere en sentral responswrapper og standardisert API-responskontrakt for å sikre forutsigbar og trygg respons.
- Innføre GDPR-vennlig feilhåndtering med generiske feilmeldinger og tydelig skille mellom bruker- og systemnivå.
- Implementere ryddig og sentralisert logging som teknisk sikrer at persondata ikke logges utilsiktet.
- Fullføre funksjonalitet for statusfeltet “kommer senere” med eget endepunkt og persistering i databasen.
- Videreutvikle frontend med ferdig meldingsvisning og mer robust språkvalg uten behov for full reload.

Disse forbedringene er allerede godt forankret i eksisterende dokumentasjon og representerer naturlige neste steg for å gjøre løsningen mer robust, konsistent og bedre tilpasset kravene i GDPR

3.0 Egen vurdering av resultat

Gruppen skal gjøre en egen vurdering av resultatet. Det skal beskrives hva gruppen er fornøyd med, og hva som eventuelt kunne vært bedre. Denne delen skal vise gruppens innsikt og forståelse av både prosess og sluttresultat.

Sett i etterkant er gruppen gjennomgående fornøyd med det resultatet vi har produsert, både når det gjelder selve løsningen og måten vi har arbeidet oss frem til den på. Vi opplever at sluttresultatet i stor grad svarer på problemstillingen vi tok utgangspunkt i, og at løsningen dekker de viktigste behovene som ble identifisert gjennom innsiktsarbeid og tidlig

konseptutvikling. Spesielt er vi tilfredse med at vi har klart å utvikle en løsning som er funksjonelt avgrenset, forståelig for brukeren og samtidig teknisk gjennomtenkt, til tross for begrenset tid og varierende teknisk erfaring i gruppen.

Et område vi er særlig fornøyde med, er hvordan vi har satt personvern og tilgangsstyring i sentrum for både design og teknisk implementasjon. Gjennom hele prosjektet har GDPR vært en førende ramme for beslutninger, og dette gjenspeiles tydelig i løsningen gjennom prinsipper som “privacy by default” og rollebasert tilgangskontroll (RBAC). Vi har bevisst valgt å begrense synlighet av data basert på brukerrollen, slik at foresatte kun får tilgang til egne barn, mens ansatte kun ser informasjon som er relevant for deres avdeling. Dette har gitt oss en dypere forståelse av hvordan juridiske og etiske krav påvirker tekniske valg, og hvordan slike hensyn må integreres tidlig i utviklingsprosessen, ikke legges til i etterkant.

Vi er også fornøyde med den tekniske arkitekturen i løsningen, særlig backend-strukturen og API-designet. Valget av FastAPI har gitt oss en tydelig og ryddig kodebase, og arbeidet med RBAC, feilhåndtering og logging har bidratt til et mer profesjonelt og robust sluttresultat enn vi opprinnelig hadde forventet. Samtidig har overgangen fra en enkel JSON-basert MVP til en relasjonsdatabase i MySQL gitt oss verdifull erfaring med iterativ utvikling og teknisk modning av en løsning over tid. Denne prosessen har styrket vår forståelse av hvordan smidig utvikling kan brukes til å balansere læring, fremdrift og kvalitet.

Når det gjelder forbedringspotensial, ser vi i etterkant at frontendløsningen kunne vært videreutviklet både teknisk og visuelt. Selv om den nåværende HTML- og JavaScript-baserte frontenden fungerer godt som et demonstrasjonsverktøy og støtter de viktigste brukerflytene, opplever vi at den i mindre grad gir en autentisk “app-følelse”. Dersom vi skulle gjennomført prosjektet på nytt, eller hatt mer tid til videreutvikling, ville vi i større grad valgt å fokusere på et moderne frontend-rammeverk som React. Dette ville gitt bedre støtte for komponentbasert utvikling, mer dynamiske grensesnitt og en brukeropplevelse som i større grad samsvarer med forventningene brukere har til moderne applikasjoner.

Samtidig ser vi at valgene vi tok underveis var fornuftige sett i lys av prosjektets rammer og læringsmål. Prioriteringen av funksjonalitet, sikkerhet og forståelse fremfor avansert frontend-teknologi har gjort at vi har kunnet levere et helhetlig og konsistent resultat. Prosessen har gitt oss økt innsikt i hvordan tekniske, organisatoriske og brukerrelaterte hensyn henger sammen i smidige prosjekter, og hvordan kompromisser er en naturlig del av utviklingsarbeid.

Avslutningsvis vurderer vi prosjektet som svært lærerikt. Vi sitter igjen med en løsning vi er stolte av, og med en tydeligere forståelse av både styrker og svakheter i vårt eget arbeid. Denne refleksjonen gir et godt grunnlag for videre faglig utvikling, både innen smidig metode, teknisk utvikling og tverrfaglig samarbeid.

4.0 Vedlegg

Vedleggene skal inneholde selve løsningen, for eksempel kildekode og andre relevante filer. Leveransetesten skal legges ved i sin helhet. I tillegg skal andre artefakter som er produsert

underveis i prosjektet legges ved og beskrives, for eksempel skisser, analyser, prototyper, brukerundersøkelser og kommunikasjonsstrategier.

Vedlegg A - Kildekode til løsningen

Vedlegget inneholder prosjektets komplette kildekode for både frontend og backend, inkludert statisk frontend bygget med HTML, CSS og JavaScript, samt backend-løsning med API, databaseoppsett og rollebasert tilgangskontroll. Frontend viser hvordan brukergrensesnittet er strukturert og hvordan løsningen kommuniserer med backend gjennom API-kall.

Informasjon om oppsett og kjøring av løsningen er beskrevet i prosjektets README-fil.

Formålet med vedlegget er å dokumentere den tekniske løsningen, gjøre arbeidet etterprøvbart, og vise hvordan løsningen kan settes opp og kjøres basert på vedlagt kildekode.

Vedlegg B - Brukerundersøkelsen

Vedlegget beskriver gjennomføringen av brukerundersøkelsen, inkludert hvilke roller som ble testet, hvilke oppgaver brukerne utførte og hvordan testen ble gjennomført. Samlede tilbakemeldinger og hovedfunn er oppsummert, sammen med eventuelle justeringer gjort på bakgrunn av funnene.

Formålet med vedlegget er å dokumentere brukerinnsikt og hvordan denne påvirket løsningen.

Vedlegget kommer i Excel-format.

Vedlegg C - Figma prototype

Vedlegget inneholder prosjektets mobilapp-prototype. Brukergrensesnittet er bygget opp med rollebasert tilgang for ansatte og foresatte, støtte for flere språk, navigasjon mellom sentrale skjermer, samt håndtering av statusoppdateringer, notater og tidsstempler.

Formålet med vedlegget er å dokumentere den tekniske løsningen, gjøre arbeidet etterprøvbart, og vise hvordan brukergrensesnitt og funksjonalitet er implementert.

Vedlegg D - Dokumentasjon fra designsprinten

Link til mirobrett står i prosjektrapporten. Ligger som vedlegg der.

Vedlegget inneholder dokumentasjon fra designsprinten som ble gjennomført i Miro over to dager. Materialet viser arbeidet fra problemdefinisjon og brukerinnsikt til konseptutvikling, løsningsvalg og storyboarding av prototypen.

Dokumentasjonen omfatter blant annet langsiktig mål, sprintspørsmål, HKV-spørsmål, ekspertinnspill, gjennomgang av eksisterende løsninger og kartlegging av brukerreiser for både foreldre og ansatte. Videre inneholder vedlegget konseptskisser, avstemninger, valg av hovedkonsept og et detaljert storyboard som viser flyt mellom sentrale skjermer.

Fokuset i designsprinten var inn og utkryssing av barn, personvern og brukervennlighet for ulike brukergrupper. Materialet viser hvordan disse hensynene påvirket designvalg og prioriteringer i prosjektets tidlige fase.

Formålet med vedlegget er å dokumentere den kreative og strategiske prosessen bak løsningsvalget, og vise hvordan teamet systematisk utforsket problemstillingen og kom frem til en testbar prototype.

Vedlegg E - GDPR og personvern

Vedlegget inneholder prosjektets samlede dokumentasjon knyttet til personvern og GDPR. Dokumentene beskriver vurderinger av behandlingsgrunnlag, dataminimering, rollebasert tilgangskontroll, trygg respons, logging, feilhåndtering og identitetshåndtering, samt hvordan disse prinsippene er tenkt ivaretatt i løsningen.

Dokumentasjonen er utarbeidet gjennom prosjektets sprinter og viser både analyser, tekniske vurderinger og planlagte tiltak. Vedlegget gir et helhetlig bilde av hvordan GDPR har vært brukt som rammeverk for design og utvikling av løsningen.

Formålet med vedlegget er å dokumentere personvernarbeidet i prosjektet og gjøre vurderingene etterprøvbare i tråd med kravene i oppgaven.

Vedlegg F - Datamodellering og databasestruktur

Vedlegget inneholder prosjektets samlede dokumentasjon knyttet til datamodellering og databasestruktur. Dokumentene beskriver utforming av den relasjonelle datamodellen, inkludert identifisering av sentrale entiteter, relasjoner, kardinalitet og valg av primær- og fremmednøkler. Videre redegjøres det for normalisering, valg av datatyper og strukturering av tabeller for å sikre dataintegritet, konsistens og skalerbarhet.

Dokumentasjonen er utarbeidet gjennom prosjektets utviklingsfase og reflekterer en iterativ prosess hvor datamodellen har blitt justert i takt med funksjonelle krav og tekniske vurderinger. Vedlegget inneholder både konseptuelle modeller (ER-diagrammer) og logiske databasedesign, samt begrunnelser for sentrale designvalg.

Vedlegg G - Backend

Vedlegget inneholder prosjektets samlede dokumentasjon knyttet til backend-løsningen. Dokumentene beskriver hvordan backend er bygget opp, inkludert arkitektur, API-struktur, datatilgang og samspillet mellom frontend, backend og database. Videre forklares hvordan sentrale funksjoner som henting av barn og avdelinger, statusendringer og logging er tenkt løst.

Formålet med vedlegget er å gi en tydelig og etterprøvbar oversikt over backend-arbeidet i prosjektet, samt dokumentere hvordan løsningen er lagt til rette for videre utvikling, forbedringer og integrasjon med rollebasert tilgang og autentisering i senere sprinter.

Vedlegg H - Brukerregistrering og RBAC

Vedlegget inneholder prosjektets dokumentasjon knyttet til rollebasert tilgangskontroll (RBAC), informasjonsflyt og brukerregistrering. Dokumentene beskriver hvordan systemet er designet for å sikre at foresatte kun får tilgang til egne barn, og at ansatte kun ser informasjon som er relevant for deres rolle og avdeling.

Videre forklares hvordan barnehagens årlige grunnoppsett legger til rette for korrekt kobling mellom barn, foresatte og ansatte, samt hvordan førstegangsregistrering er tenkt gjennomført ved hjelp av forhåndsregistrerte opplysninger og identitetsbekreftelse. Selv om deler av denne flyten ikke er fullt implementert i MVP-en, er den dokumentert som en del av løsningens design.

Formålet med vedlegget er å vise hvordan tilgangsstyring, brukervennlighet og personvern er ivaretatt i løsningen, samt å synliggjøre avgrensninger og videre utviklingsmuligheter knyttet til RBAC og sikker brukerhåndtering.

5.0 Kilder

Personopplysningsloven. (2018). *Lov om behandling av personopplysninger* (LOV-2018-06-15-38). Lovdata. https://lovdata.no/dokument/NL/lov/2018-06-15-38/KAPITTEL_gdpr