KRISTOFFER GJØSUND

Fritiden min brukes til å videreutvikle ferdighetene mine innenfor IT-kompetanse og filosofi.

Jeg holder meg aktiv med regelmessig trening, og trives med å oppleve nye eventyr med familie og venner.

Jeg motiveres av å utfordre meg selv og strekke meg etter nye mål. Jeg søker et faglig miljø hvor jeg kan bruke IT-forståelsen min og problemløsningsevne til å skape positive brukeropplevelser, samtidig som jeg fortsetter å styrke min tekniske kompetanse.

EKSTERNE LENKER

KONTAKTINFO

Oslo +47 46 65 29 38

krigjo25@outlook.com

LINKER TIL ATTESTER

- SosEnt Norge
- Get Academy
- Plantasjen AS
- Polyform AS
- Kløver kjøtt AS
- Ingridsglassverksted AS

UTDANNING & KURS

- **GetAcademy**
  IT-Utviklingsfaget 2024-2026
- **Harvard XCS50**
  Computer Science for Python Programming
- **Borgund VGS**
  Helse og oppvekstfag 2011-2014
  Bygg og anlegg 2010-2011

JEG MESTRER

- **Programmeringsspråk**
  TypeScript, SQL, Python, C#,
- **Rammeverk & CMS**
  Vue/Nuxt - Composition API, React

  Pydantic, FastAPI, Numpy, SQLAlchemy, Pandas, matplotlib, Reflex, pytest ReportLab

  ASP.NET Core, NUNIT, Entity, TinaCMS

VERKTØY / DEVOPS / DX:

- **Versjonskontroll**
  Conventional Commits, GitHub Actions,
- **Sky & Devops**
  Google Cloud, Skybase, Netlify, Docker, Linux,
- **KI, QA & UX**
  OpenAI API (GPT-4o), Google Gemini API, NUnit, pytest, vitest, **SASS,** WCAG, SEO Optimalisering TailwindCSS

RELEVANT ARBEIDSERFARING

EventHappenix - stealth startup
Techlead & System architect SEP, 2026 -

Arrangementsportal som samler arrangementer fra norske og internasjonale kilder.

- Utviklet brukergrensesnitt med React og TypeScript.
- Utviklet nettskrapere i **Python** for å hente og strukturere arrangementsdata.
- Arbeidet med sikkerhet, personvern og utrulling med Docker og Google Cloud.
- Bygget API-er og funksjoner for databehandling, deduplisering, søk og filtrering.
- Implementerte nyhetsbrev- og abonnementsfunksjoner med e-postbekreftelse.

GetAcademy
Fullstack Utvikler APR, 2025 - MAI, 2026

Godkjent fagskole med intensiv, yrkesrettet utdanning innen IT og programmering.

- Utviklet backend-tjenester og REST-API-er i **C#** og **.NET 8**, med tester i **NUnit**.
- Implementerte rollebasert tilgangskontroll og bidro til migrering av internplattformen til **TypeScript**.
- Deltok i pull request-gjennomganger og brukte LLM-verktøy som støtte til utvikling, feilsøking og dokumentasjon.

Konsulenttjenester & Open Source
lumina-sass JUNI 2026 -

Åpen kildekode-designsystem og stylingrammeverk for Dart Sass.

- Samlet designverdier i gjenbrukbare tokens og Dart Sass @use-moduler.
- Utviklet miksere for vanlige layouter og komponentstiler.
- La vekt på konsekvent UI, enklere vedlikehold og tilgjengelighet.

Sosial Innovasjon APR, 2025 - JUNI 2026

Nasjonal interesseorganisasjon for sosiale entreprenører innen inkludering og velferd.

- Utviklet en **Nuxt**- og **TypeScript** løsning med integrert innholdsstyringssystem og fokus på universell utforming.
- Dokumenterte arkitektur og funksjonalitet med README-filer, tester og diagrammer.
- Lagde brukerveiledning og ga opplæring i innholdsstyringssystemet for nye bidragsyter.

PORTE FØLJE

Mars, 2026

Presenteres av

Kristoffer Gjøsund

@krigjo25

krigjo25@outlook.com

https://krigjo25.no

Mars, 2026

INNHOLDS FORTEGNELSE

Dette er et utvalg av mine prosjekter som viser hvordan jeg kombinerer teknisk logikk med praktiske løsninger for å skape effektive brukeropplevelser.

TOOLS WEBAPP

WEB APPLICATIONS

EVENT HAPPENIX

GETNEWS

INTRANET

KARRIERE-KI

LIBRARY MANAGEMENT

PORTFOLIO

SOSIAL INNOVASJON

event Happenix

Hensikten med dette prosjektet var å utvikle en plattform som automatisk samler inn, fjerner duplikater fra og presenterer arrangementer fra hele verden, og sender dem som nyhetsbrev.

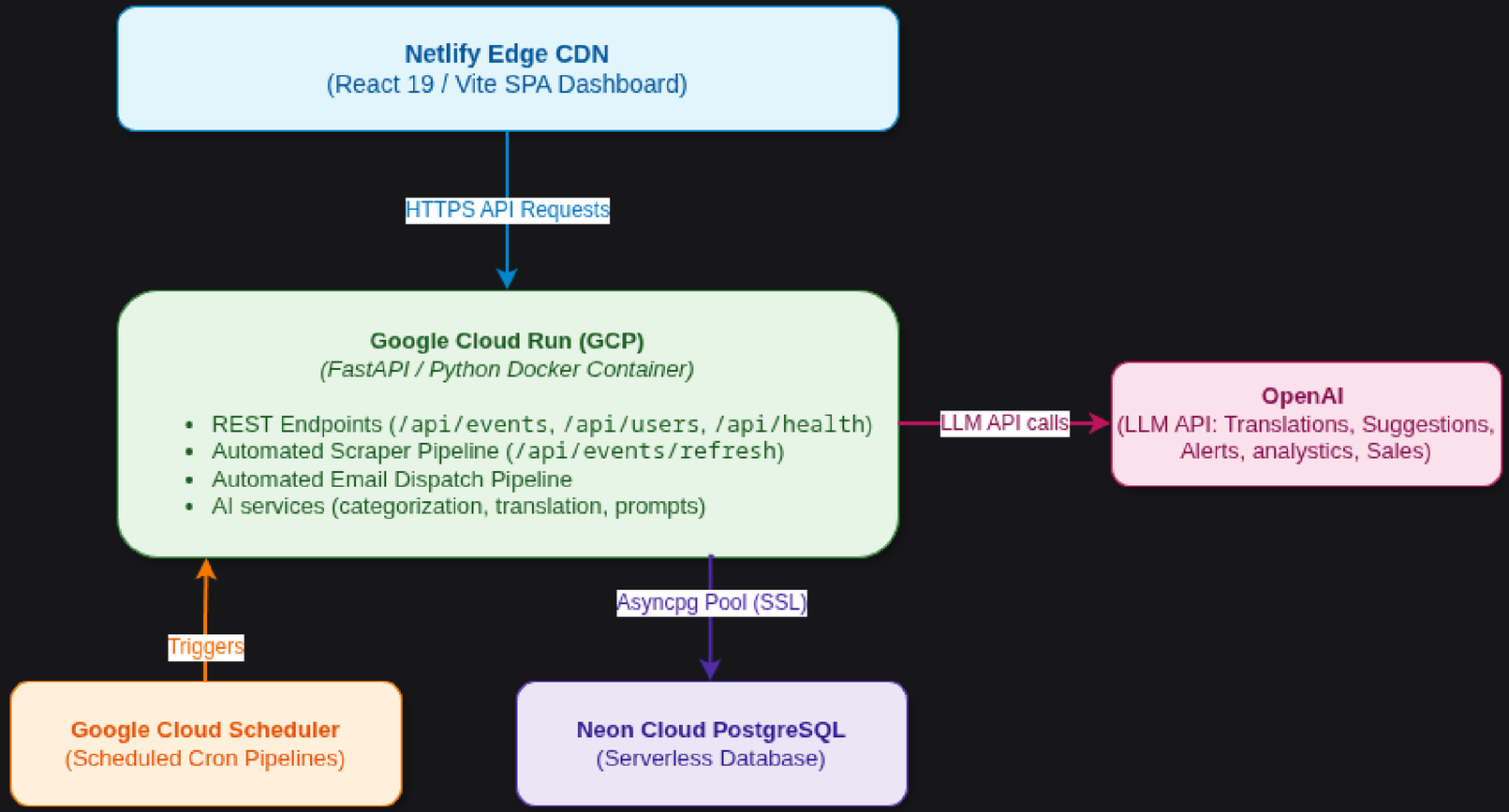
Jeg utviklet backend med **FastAPI** i **Python**, hvor **asynkrone** scrapere henter data fra ulike kilder. Dataene lagres i **PostgreSQL** med **SQLAlchemy** og **Alembic**.

KI-tjenester via **OpenAI** og **Gemini** kategoriserer og oversetter arrangementer, og lager vekstråd og salgspitcher. Jeg har videreutviklet konkret LLM-prompting med egne promptmaler per oppgave, regelbasert fallback og månedlig tokenbudsjett.

Frontend er bygget med **React 19**, **TypeScript**, **Vite** og **Sass**, med **semantisk HTML** og mål om **WCAG 2.2 AA**.

For å ivareta **sikkerhet** og **GDPR** bruker jeg API-nøkkel på **administrative endepunkter**, **hemmeligheter i miljøvariabler**, **CORS**, **XSS-sanitering**, **cookiefri anonymisert sporing** og **GDPR-sletting**.

Tjenesten kjører på **GCP Run**, **NeonDB** og **Netlify**, med **CI**/**CD** i **GitHub Actions**.



Demo

Prosjekt GETNEWS

April - Mai 2025

GetNews er et modulbasert abonnementssystem utviklet i **C#** og **.NET 8** som administrerer livssyklusen til nyhetsbrevabonnenter gjennom registrering, token-verifisering og avmelding.

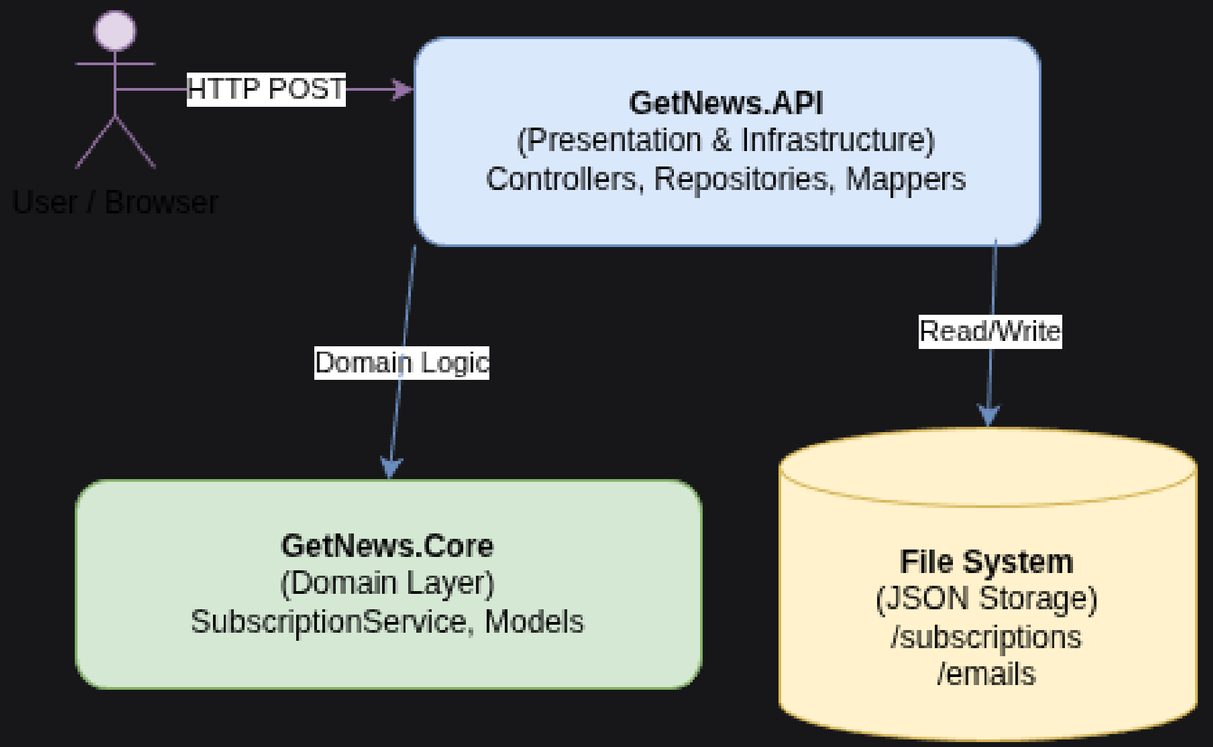
Backend er strukturert etter rene arkitekturprinsipper med et skarpt skille mellom **forretningslogikk** og **I/O**. Domenekjernen i GetNews.Core opererer som en deterministisk tilstandsmaskin basert på Result-mønsteret, helt isolert fra sideeffekter. Persistens og varsling håndteres i API-laget via filbaserte adaptere (SubscriptionFileRepository og DummyEmailService) som serialiserer JSON-data lokalt.

Brukerflaten er implementert som en ren klientside-applikasjon i **HTML** og **JavaScript** levert via Kestrel-statiske filer, med en reaktiv tilstandsmodell og asynkrone API-kall.

Sikkerhet enivaretas gjennom syntaktisk **e-postvalidering**, unike **GUID-verifiseringstokens** for å forhindre uautorisert registrering, samt eksplisitt tilstandskontroll ved avmelding.

Enhetstester er realisert med **NUnit** og **Coverlet** for isolert validering av domenetilstander og e-postregler under lokal utførelse.

I prosjektet bidro jeg med automatiserte enhetstester i **NUnit** og **Coverlet** for isolert testing av e-postvalidering, feilkoder og tilstandsoverganger for å sikre stabilitet.



Kildekode

Prosjekt Intranet

November - Desember 2025

Intranet er en fullstack utdanningsplattform utviklet for Get Academy for å sikre og distribuere kursopplegg, pensum og timeplaner basert på studenters og ansattes tilgangsnivå.

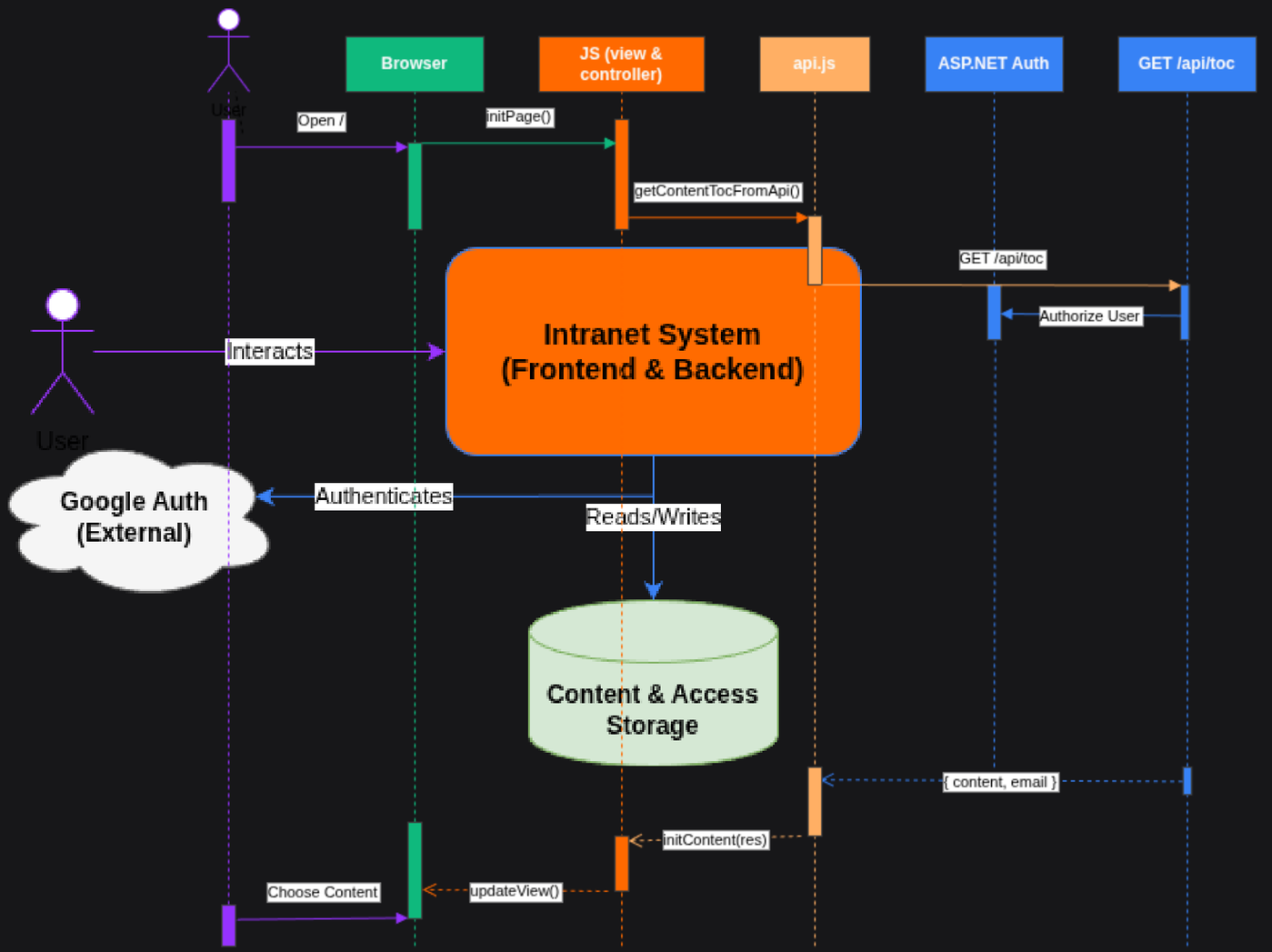
Backend er utviklet i **C#** og **.NET 8** med **Minimal API** og **Clean Architecture**. Forretningslogikk og datatilgang er adskilt via Dependency Inversion og strategimønsteret. Systemet benytter en to-trinns rollebasert tilgangskontroll med wildcard-matching mot JSON-strukturer og mapper dynamisk innholdsfortegnelse direkte fra CSV til Markdown-artikler.

Brukergrensesnittet er en modulær klientside-applikasjon i HTML, CSS og **JavaScript**, strukturert med MVC-arkitektur og asynkron henting av sikret innhold.

Sikkerhet er ivaretatt med **Google OAuth 2.0** for identitetshåndtering, cookie-basert økthåndtering, samt autorisasjonsvalidering mot systemets **RBAC-motor** før dokumenter returneres.

Kvalitetssikring er implementert med automatiserte enhetstester via **NUnit** og **Moq** for isolert testing av tilgangsregler, orkestrert med Makefile for bygg og formatering.

Jeg bidro med å opprette automatiserte enhetstester med NUnit og Moq for å verifisere tilgangsreglene og sikre robust drift, samt Makefile for lokal bygging og kodeformatering.



Sekvensdiagrammet viser hvordan systemet håndterer en brukerforespørsel – fra siden lastes, til brukeren autoriseres og riktig innhold hentes og vises.

Library Management

Mars, 2025

Comprehensive Library Management Using MariaDB

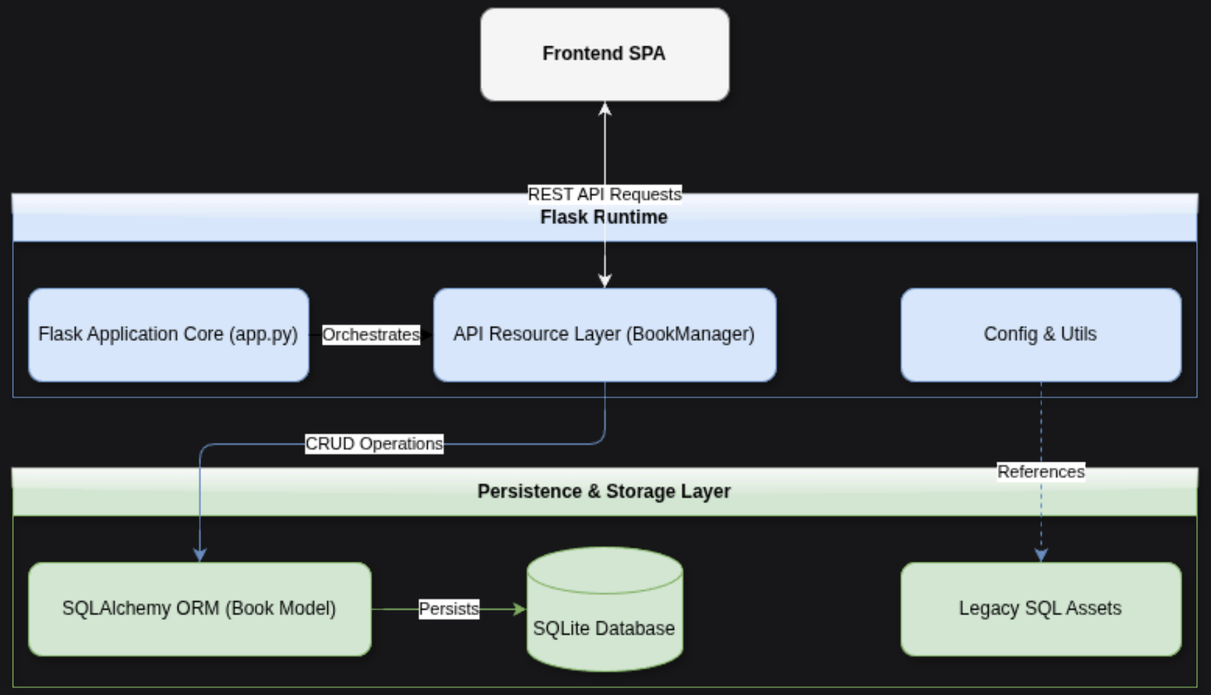
LibSys er et fullstack biblioteksystem utviklet for å administrere bokkataloger, forfatterrelasjoner og utlånsprosesser med fokus på dataintegritet og brukervennlighet.

Jeg utviklet backend som et **REST-API** i **Python** og **Flask** med modulære **MethodView**-endepunkter. Databaselaget er modellert med **SQLAlchemy** og **SQLite**, hvor transaksjonssikkerhet og **Pydantic v2** sikrer streng skjemavalidering og typekonvertering ved opprettelse, redigering og sletting av bøker og forfattere.

Frontend er bygget som en moderne énsidesapplikasjon (SPA) med **Vue 3**, **TypeScript**, **Pinia** og **Vite**. Brukergrensesnittet benytter **Sass** og **Bootstrap** for responsiv layout, med **typesikre API-kall** via **Axios** og sentralisert tilstandshåndtering for sanntidsoppdatering av katalogen.

Sikkerhet og sporbarhet er ivaretatt med **CORS-policy**, **miljøvariabler** for konfigurasjon og detaljert logging via et dedikert overvåkingslag som logger **feiltilstander**.

Koden er kvalitetssikret med **Pytest** og **Pytest-cov** for backend samt **Vitest** for frontend, med streng typesjekk via **Pyright** og **ESLint**.



Kildekode

Januar-, 2026

Prosjekt Karriere-ki

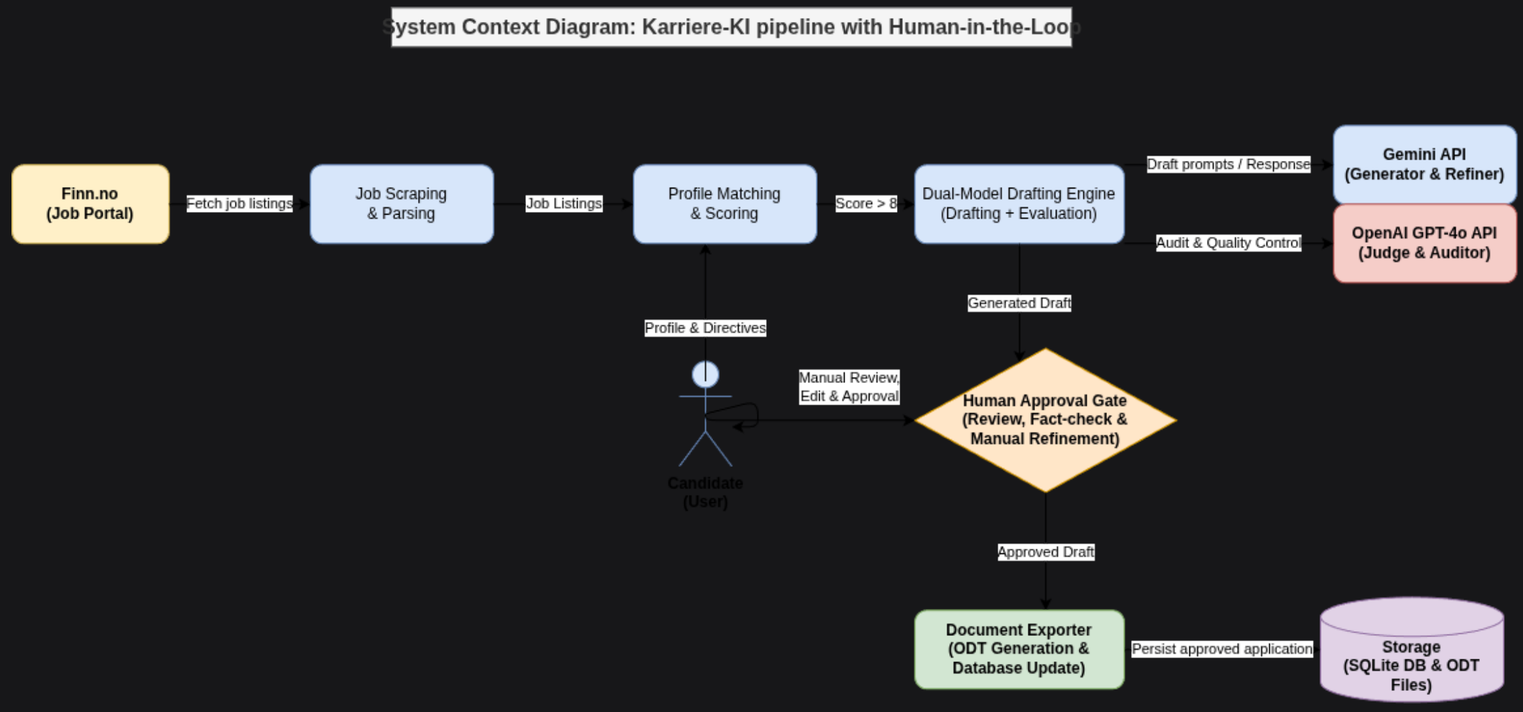
KI - prosjekt

Karriere-KI er et autonomt kommandolinjeverktøy i Python utviklet i samarbeid for å automatisere overvåking av stillingsportaler, kandidatmatching og søknadsskriving.

Backend benytter **asynkron web-scraping** med **httpx** mot portaler som FINN.no og NAV. Dataene lagres i en lokal, **SQLite**-database via **aiosqlite** for å sikre **fullt personvern** og skjerming av kandidatdata uten ekstern sporing. Koden er strukturert med full **typeannotering** og statisk **typekontroll** via **Pylance** for å hindre kjøretidsavvik.

Det jeg bidro med til prosjektet var å utvikle den helhetlige AI-orkestreringen og søknadsmotoren. Jeg opprettet en to-modell arkitektur som kombinerer **Gemini Flash** - for poengsetting mot lokale Markdown profiler med **GPT-6-luna** for iterativ søknadsgenerering, styrt av modulære promptkontrakter som sikrer deterministiske resultater.

Jeg automatiserte tester i **Pytest** for å verifisere promptkontrakter, scraperlogikk og databaseintegritet, samt integrasjon mot ODT-dokumentgenerering via odfpy.



KI presentasjon

Prosjekt portfolio

Automatisert Porteføljе-økosystem

Aug, 2024 -

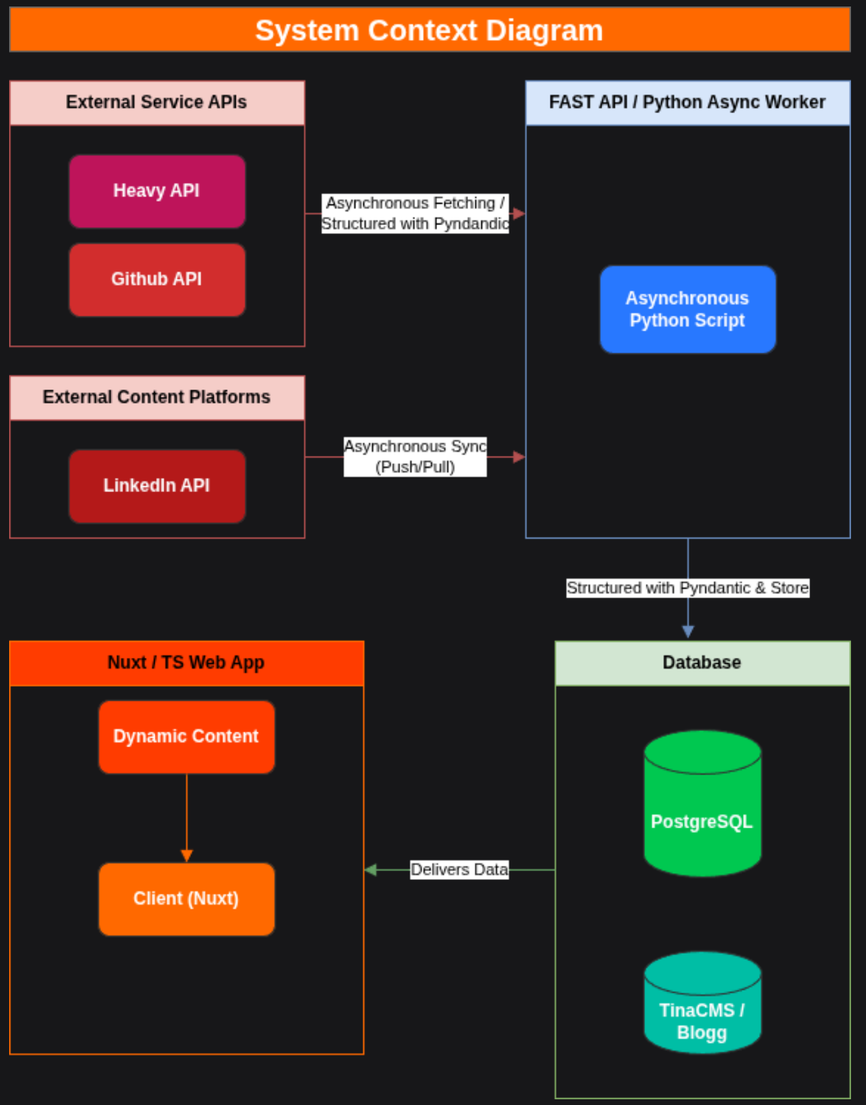
Webapp-Portfolio er en fullstack produksjonsplattform utviklet for å automatisere presentasjon, prosjektovervåking og innholdsdistribusjon for egen teknisk portefølje på krigjo25.no.

Backend er bygget som et **asynkront REST-API** i **Python** og **FastAPI** med **Uvicorn**. Databaselaget er modellert med **asynkron SQLAlchemy** og **Alembic**-migreringer mot **PostgreSQL** og **SQLite**, hvor **Pydantic v2** sikrer streng skjemavalidering og serialisering ved innhenting av eksterne utviklerdata og prosjektmetrikker.

Frontend er en hybrid SSR og SSG-applikasjon i **Nuxt 4** og **TypeScript**, **Nuxt Content** for Git-støttet innholdsforvaltning.
Brukerflaten er bygget med designtokens fra lumina-sass og validert mot **WCAG 2.2 AA** og **AAA** med **html-validate** og **vitest-axe**.

Sikkerhet og personvern er ivaretatt med **cookiefri sporing**, **CORS-regler** og hemmelighetshåndtering gjennom miljøvariabler.

Drift og leveranse er fullautomatisert med **Docker**, **GitHub Actions CI/CD** for testing med **Pytest** og **Vitest**, **semantisk versjonering** og produksjonsdistribusjon til **Google Cloud Run** og **Netlify**.



Kildekode **Demo**

Sosial Innovasjon

Web-app med Nuxt & TinaCMS

Mai, 2025 - Juni, 2026

Prosjektet er en fullstack formidlingsplattform utviklet i samarbeid for organisasjonen SosEnt Norge på sosent.no for å presentere medlemmer, nyheter og arrangementer.

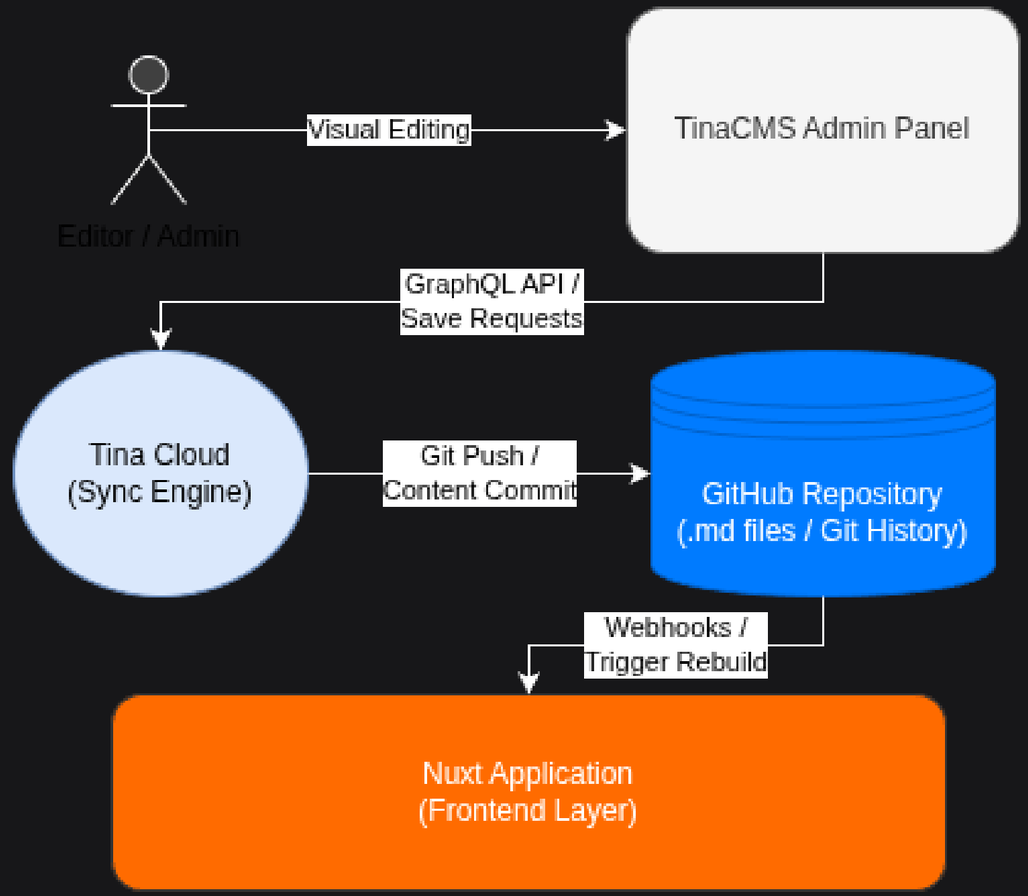
Plattformen benytter en **serverløs**, **Git-basert CMS-arkitektur** hvor innhold lagres som Markdown i GitHub og indekseres statisk ved bygging via **Nuxt Content**, helt uten behov for en ekstern database.

Det jeg bidro med til prosjektet var å opprette frontend-arkitekturen i **Nuxt 4** og **TypeScript** med **TinaCMS** for innholdsredigering, utvikle det responsive brukergrensesnittet med lumina-sass for **streng universell utforming** (**WCAG 2.2 AA** og **AAA**), automatisere skript for rensing av utgåtte arrangementer, samt sette opp automatiserte testsuiter i **Vitest**, **vitest-axe** og **html-validate** kombinert med flerspors **CI/CD i GitHub Actions** og **semantisk versjonering**.

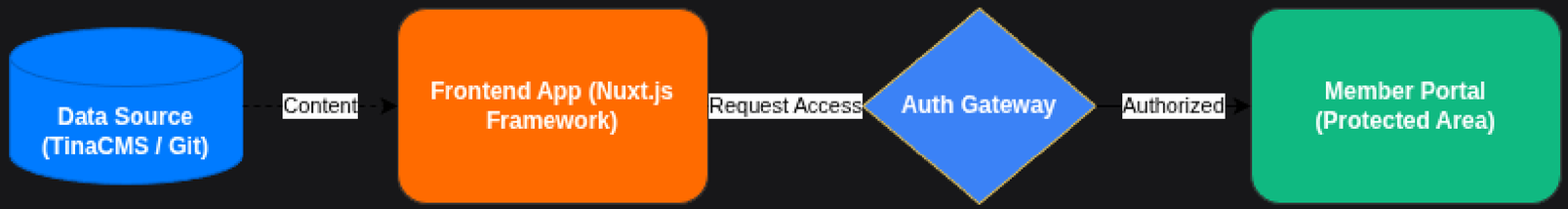
Sikkerhet og personvern ivaretas med **prinsippet om minste privilegium** i tilgangsstyringen mot eksterne ressurser.

Innholdsflyt fra redaktør til publisert nettside via **TinaCMS** og **Github**

Headless CMS Workflow



Tilgangsstyring som skiller åpent innhold fra beskyttet medlemsarkiv



Demo

TOOLS

LUMINA-SASS SIMPLIFIED LOGGER

lumina-sass

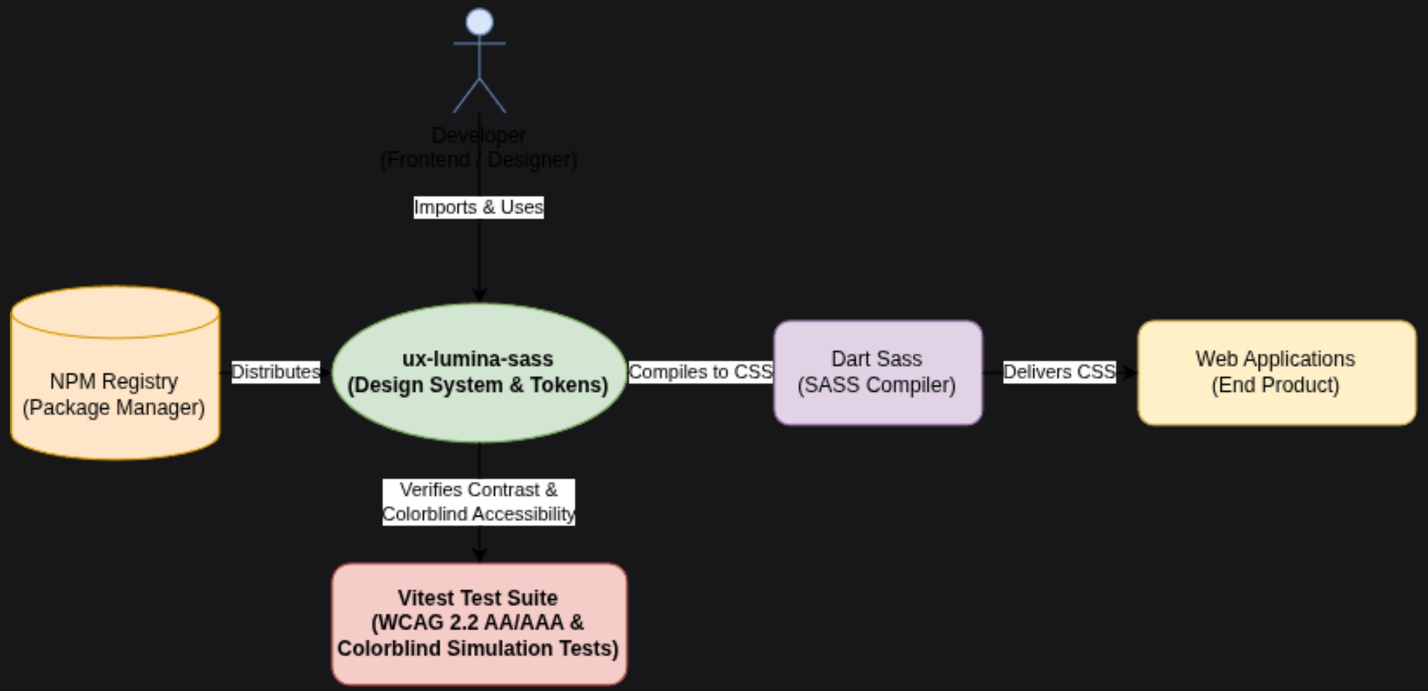
Open Source project

Lumina-Sass er et modulært designsystem og åpen kildekode-stilbibliotek distribuert som en offentlig NPM-pakke for å standardisere brukergrensesnitt og universell utforming på tvers av webprosjekter.

Pakken er bygget med moderne **Sass-moduler** og **TypeScript**, organisert i uavhengige moduler for fargetokens, kartstrukturer, matematiske hjelpefunksjoner og fleksible rutenett. Systemet eksporterer sub-path-moduler som gjør det mulig å importere spesifikke verktøy isolert for optimal byggestørrelse.

Det jeg bidro med til prosjektet som hovedutvikler var å designe og implementere hele biblioteket, inkludert matematiske algoritmer for beregning av fargekontraster i tråd med **WCAG 2.2 AA** og **AAA**, strukturere pakke-eksportene i package.json, etablere enhetstester med **sass-true** og **Vitest**, samt automatisere flerspors **CI/CD**-arbeidsflyter i GitHub Actions for syntakskontroll, semantisk versjonering og automatisk publisering til NPM Registry.

Sikkerhet og pakkekvalitet sikres gjennom **streng linting** med **Stylelint** og fjerning av avhengighetsadvarsler i Dart Sass.

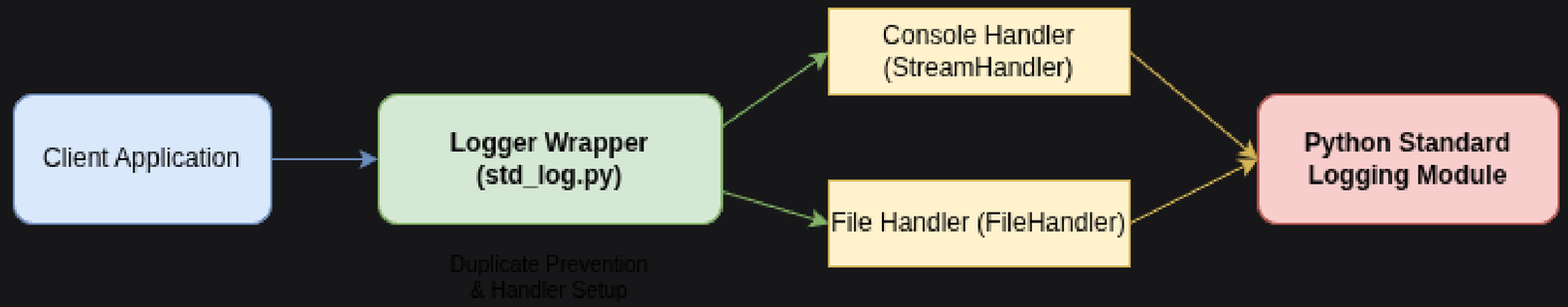


Kildekode

Simplified Logger

Januar, 2026 -

Pypl project



Prosjektet er et åpen kildekode-loggingsbibliotek distribuert som en offentlig PyPI-pakke for å forenkle, standardisere og berike logging og feilsøking i Python-systemer.

Biblioteket pakker inn Pythons standard logging-modul i en strukturert og modulær arkitektur med støtte for farget konsolloppsett, tids- og størrelsesbasert filrotering, samt integrasjon mot structlog og serilog for maskinlesbar loggutskrift med minimal ytelsespåvirkning.

Det jeg bidro med til prosjektet som hovedutvikler var å arkitektere og programmere hele loggingspakken, implementere egne metodedekoratører og watchers for automatisk overvåking av funksjonskall, opprette fullstendige typestubber (.pyi) og py.typed for streng typesikkerhet i Mypy, etablere automatiserte enhetstester i **Pytest**, samt sette opp automatiserte GitHub Actions for bygging og publisering til PyPI.

Kodekvalitet og sikkerhet ivaretas gjennom **streng statisk typekontroll** med **Mypy**, **Pylint** og deterministisk formatering med Black.

Kildekode