Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

Tolking av PDF-notesett og gjenkjenning av stemmer.

Systemdokumentasjon

Versjon 1.1

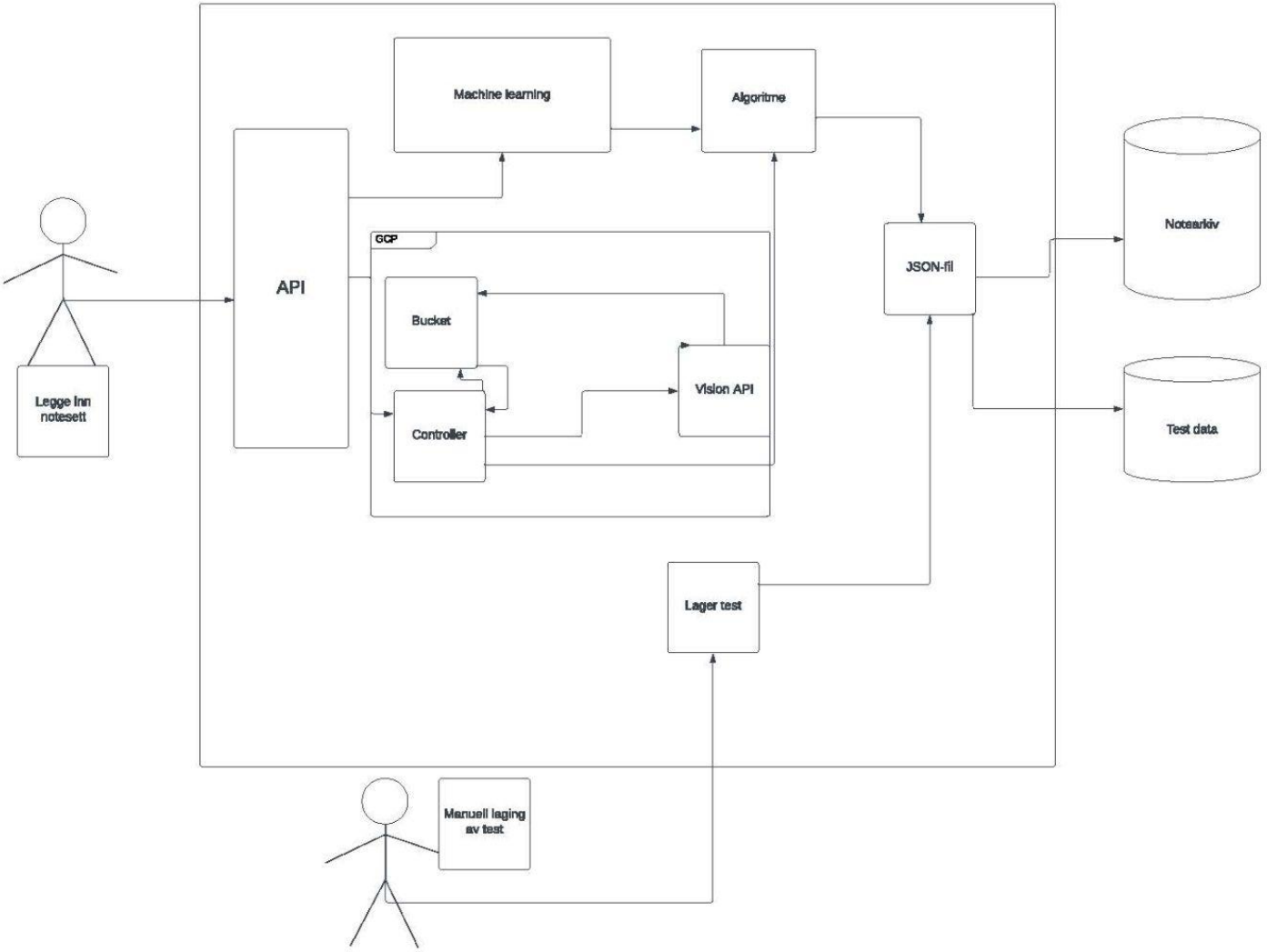
Dokumentet er basert på Systemdokumentasjon utarbeidet ved NTNU. Revisjon og tilpasninger til bruk ved IDER, DATA-INF utført av Carsten Gunnar Helgesen, Svein-Ivar Lillehaug og Per Christian Engdal. Dokumentet finnes også i engelsk utgave.

INNHOLDSFORTEGNELSE

1 INNLEDNING

Systemdokumentet skal være en oversikt over det endelige produktet. Detaljer rundt valg skal være i hovedrapporten. Dette dokumentet har flere mangler ettersom det skjer flere justeringer som vil være klare innen slutten av iterasjonen.

2 Arkitektur



Figur 1. Arkitektur figur

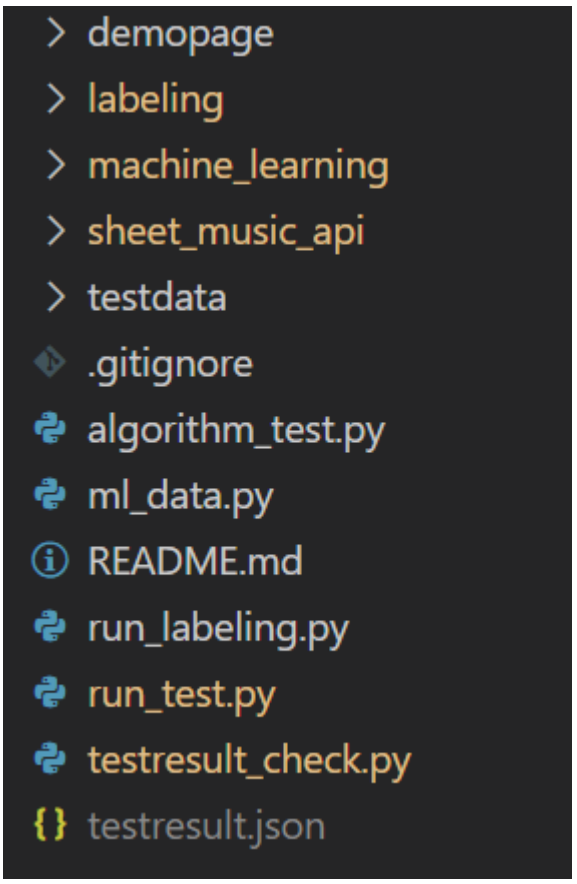
Flyten på arkitekturen

1. En bruker velger et notesett som settes inn i API-et
2. Blir sendt til GCP domainet og blir kjørt igjennom Maskin learning
    a. GCP, tar in PDF-filen. I controller klasssen
    b. sendes til Vision API for å lese
    c. Lagres i bucket
    d. sendes fra vision til Bucket for å oppdatere der
    e. bucket levere til controller
    f. controller sender videre til algoritmen
3. Maskin learning
    a. Tar inn pdf og leser på hvilken sider er hva
    b. vurdere om stemmer går over flere sider
    c. sender til algoritmen
4. Algortimen dinner
    a. stemmer
    b. tittel
    c. komponist
    d. arrangør
5. sendes til json format
6. lagres i database

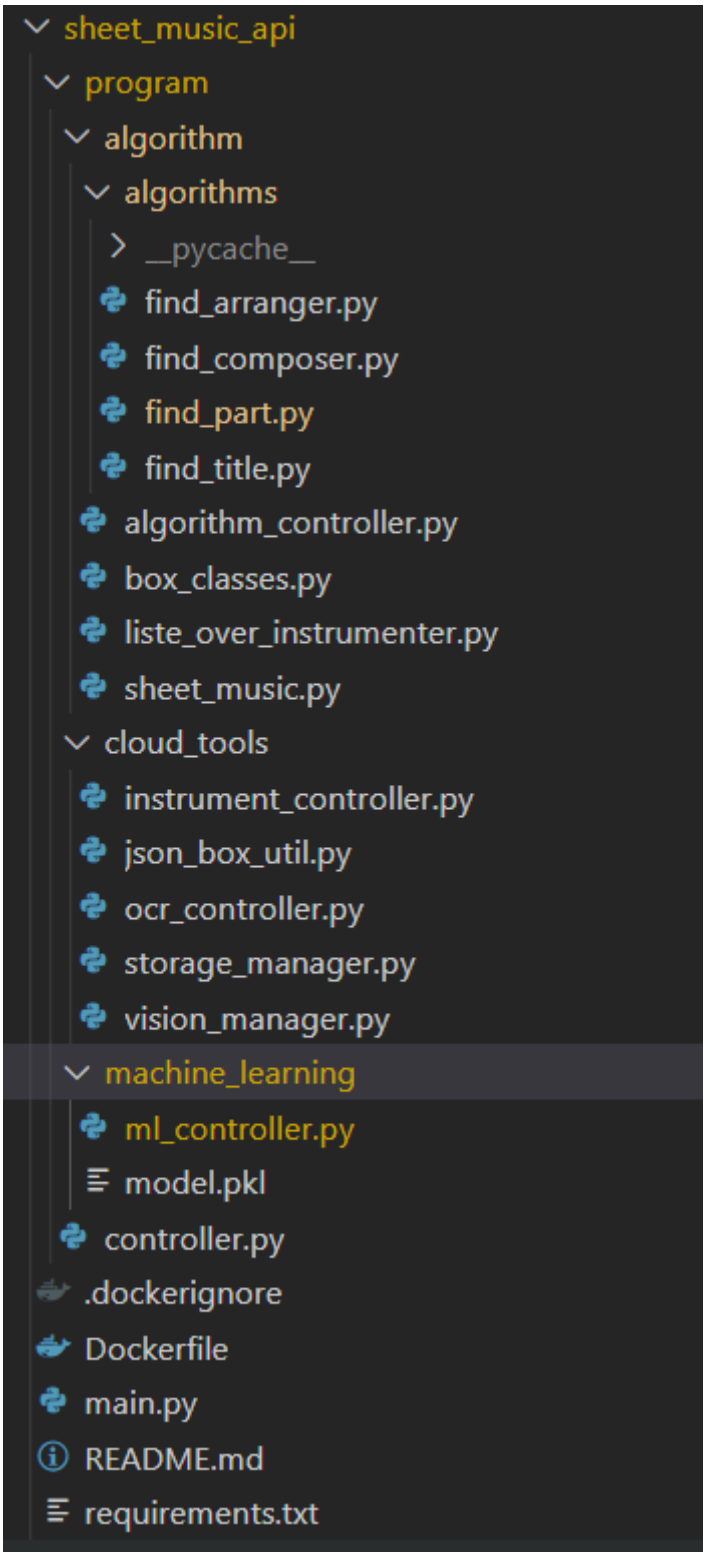
Flyten på test metoden

1. Lager test manuelt ved bruk av en applikasjon
2. formatere det som JSON
3. lagrer det i test databasen vår

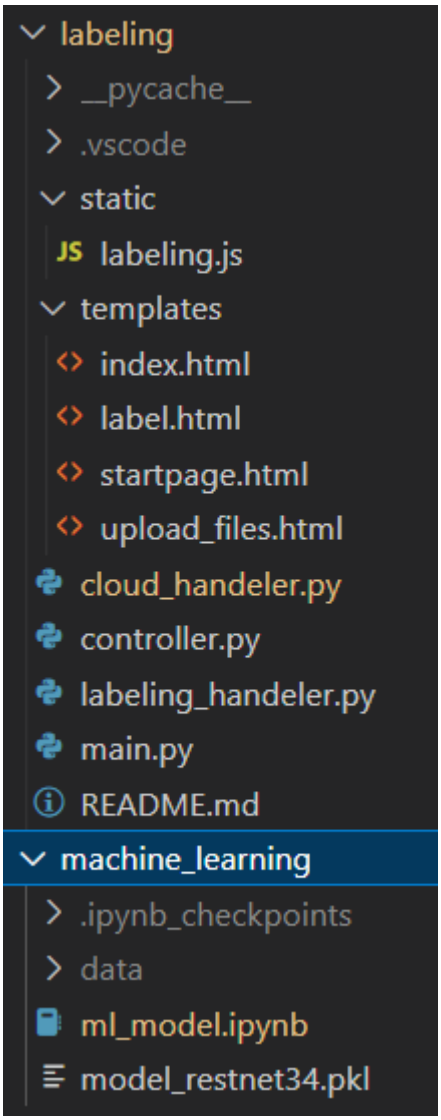
3 PROSJEKTSTRUKTUR



Figur 2. Overordnet struktur på prosjektet

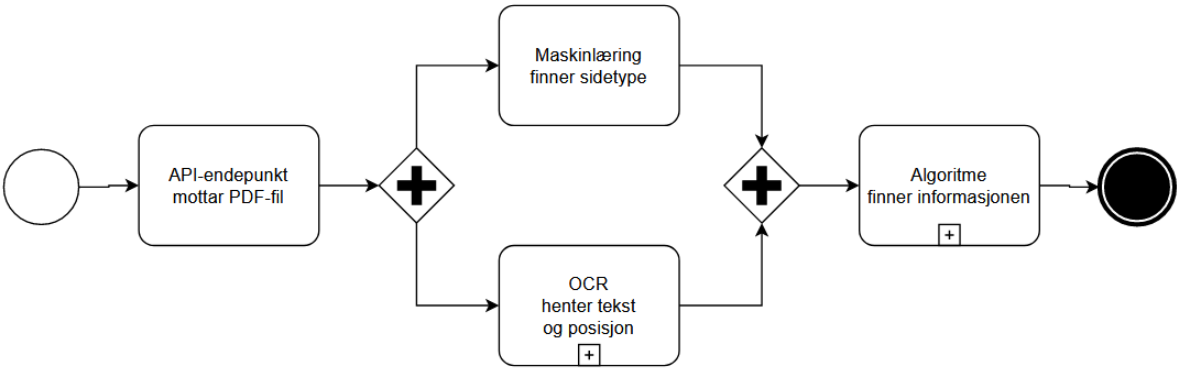


Figur 3. Struktur til kildekoden til API-et

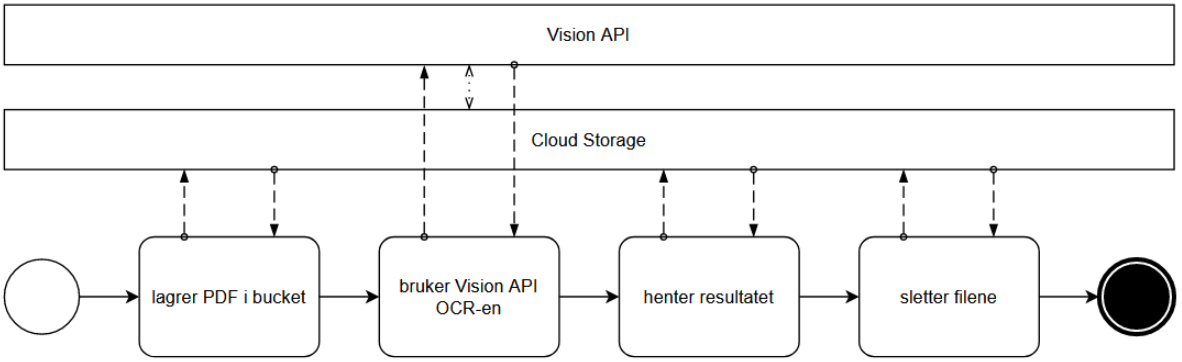


Figur 4. Struktur til Labelingvertøyet og delen som brukes for å trene modellen

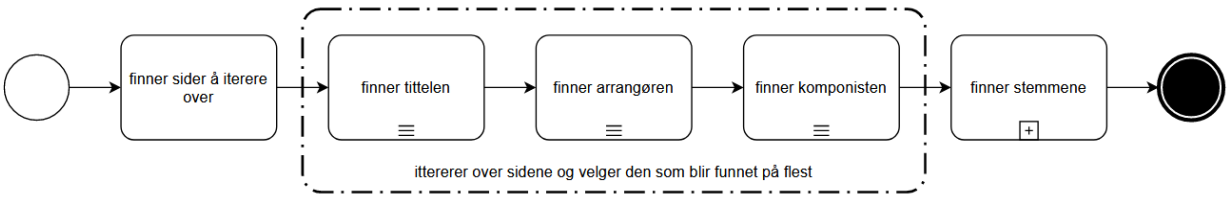
4 PROGRAMFLYT



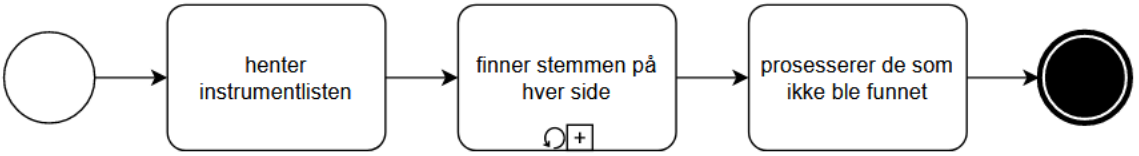
Figur 5. Overordnet prosess til programmet.



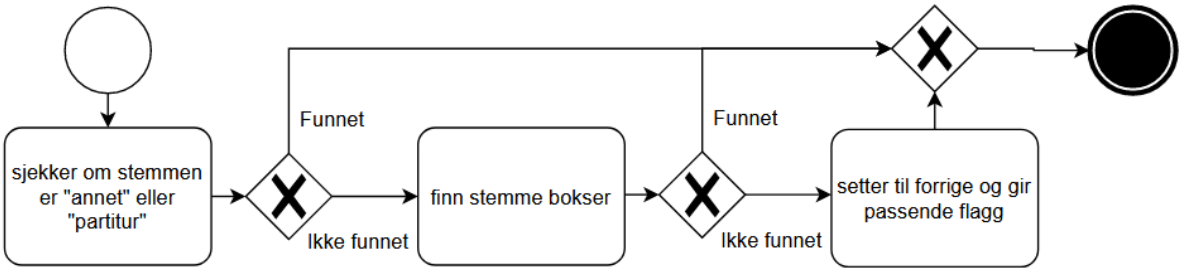
Figur 6. Subprossesen "OCR henter tekst og prosisjon" til figur 5



Figur 7. Subprossesen "Algoritme finner informasjon" til figur 5



Figur 8. Subprossesen "finner stemmen" til figur 7



Figur 9. Subprossesen "finner stemmen på hver side" til figur 8

5 DATABASEMODELL

For å bruke Vision API må PDF-filen lagres i en Storage Bucket som en del av prosessen. Resultatet av OCR (optisk tegngjenkjenning) genereres også og lagres i denne Bucketen. Imidlertid er det ikke behov for å beholde disse filene etterpå, så de slettes før resultatet sendes videre til algoritmen (*Detect text in files (PDF/TIFF)* | *Cloud Vision API*, ingen dato; *Cloud Storage*, ingen dato).

For å unngå konflikter når flere filer lastes opp samtidig, genereres det unike navn for både inngangs- og utgangsfiler. Disse navnene baseres på tid og en hash-verdi av filen. Alle filene plasseres under "t/[unikt navn]/" -mappen. Hvis det oppstår en feil som forhindrer sletting av filene, er det implementert regler som sikrer at filer som ligger under "t/"-mappen og er eldre enn en dag, slettes.

Årsaken til at filene må plasseres under "t/"-mappen er at det også opprettes en instrumentliste-fil som er nødvendig i algoritmen. Denne filen skal ikke slettes, og den er den eneste filen i Storage Bucketen som ikke plasseres i "t/"-mappen.

6 Server-tjenester

Bruker Cloud Run for å hoste applikasjonen. Bruker Flask-rammeverket (*Quickstart: Deploy a Python service to Cloud Run | Cloud Run Documentation*, ingen dato).

6.1 Tolking

```
POST /
```

```
curl -X POST -H "Content-Type: application/pdf" --data-binary
"@[path/to/data.pdf]" [url]/
```

Bytt *[path/to/data.pdf]* ut med banen til PDF-en du ønsker å hente informasjon fra.

Den aksepterer POST request til "/" med body som er en pdf-fil og content-type "application/pdf". Denne vil returnere informationen som er hentet som en json-fil. Formatet på json filen er:

```
{
    "pages": {
        "1": {
            "part": "[navnet på stemmen]"
        },
        "2": {
            "part": "[navnet på stemmen]",
            "flag": "[informasjon om den uthentede stemmen]"
        },
        ...
    },
    "title": "[tittelen]",
  "composer": "[komponisten]",
  "arranger": "[arrangøren]",
  "number_of_pages": [totalt antall sider]
}
```

Ved partitur settes "part" til "score". Ved forside, infoside eller andre sider som ikke er notesider, settes den til "other". Eller settes "part" til teksten som er funnet.

Det finnes 6 forskjellige flagg:

- Expected same as previous
    - Alt er ok! Den har ikke funnet stemmen, men forventer at det ikke er den første siden av en stemme, så den velger stemmen til den forrige siden.
- Position based part

    - Fant ikke et instrument fra instrumentalisten, men forventet at det skulle bli en ny stemme. Finner stemmen basert på posisjonen til andre stemmer.
- Position based, previous part
    - Samme som "Expected same as previous", bare at den originale stemmen er posisjonsbasert.
- Expected new part
    - Stor usikkerhet. Forventer en ny stemme, men klarer ikke å finne den. Velger det samme som forrige stemme.
- Unknown first page
    - Stor usikkerhet. Hvis den ikke finner stemmen, og ikke tror det er "partitur" eller "annen"-side, setter stemmen til "other" og får dette flagget
- Unknown previous part
    - Stor usikkerhet. Hvis den forventer at det ikke er en ny stemme og forrige side er "Expected new part", "Unknown first page" eller "Unknown previous part", settes dette

6.2 Instrumentliste

Den andre funksjonen er knyttet til en instrumentliste som brukes under tolkningen av teksten.

```
GET /instrument_list
```

```
curl [url]/instrument_list
```

GET request vil gi deg lista over instrumenter.

```
POST /instrument_list
```

```
curl -X POST -d "[instrumentet]" [url]/instrument_list
```

Bytt *[instrument]* ut med instrumentet du ønsker å legge til. Eks. *fløyte*.

POST legger til instrumentet (kan kun sende en av gangen). Returnerer lista hvis det blir lagt til.

```
DELETE /instrument_list
```

```
curl -X DELETE -d "[instrumentet]" [url]/instrument_list
```

Bytt *[instrument]* ut med instrumentet du ønsker å fjerne til. Eks. *fløyte*.

DELETE fjerner instrumentet (kan kun slette en av gangen). Returnerer lista hvis det blir fjernet.

6.3 Ping

```
GET /ping
```

```
curl [url]/ping
```

Returnerer "pong" om serveren kjører.

7 INSTALLASJON OG KJØRING

Google Cloud brukes som plattform for installasjon og kjøring av løsningen. Løsningen utnytter flere produkter fra Google Cloud, inkludert Cloud Run, Cloud Storage og Vision API. Nedenfor følger en beskrivelse av de viktigste eksterne avhengighetene og programvarebibliotekene som brukes, samt en installasjonsveiledning og instruksjoner for å kjøre løsningen.

Eksterne avhengigheter og programvarebiblioteker:

1. Flask: Flask er et populært web-rammeverk for Python. Det brukes til å utvikle den webbaserte API-løsningen. Flask gjør det enkelt å håndtere forespørsler og responser, samt bygge ruter for API-endepunkter (*Welcome to Flask — Flask Documentation (2.3.x)*, ingen dato).
2. Google Cloud SDK: Google Cloud SDK er et sett med kommandolinjeverktøy som lar utviklere administrere og samhandle med Google Cloud-plattformen. Det brukes til å konfigurere og administrere Cloud Run-tjenesten, samt samhandle med Cloud Storage og Vision API (*Cloud SDK - Libraries and Command Line Tools*, ingen dato).
3. Google Cloud Storage-bibliotek: Dette biblioteket gir et enkelt grensesnitt for å samhandle med Cloud Storage-tjenesten fra Python. Det brukes til å laste opp og lagre bilder som sendes til API-et (LLC, ingen dato a).
4. Google Cloud Vision-bibliotek: Dette biblioteket gir funksjonalitet for bildeanalyse ved hjelp av Vision API fra Google Cloud. Det brukes til å behandle de opplastede bildene og ekstrahere informasjon som ansiktsdeteksjon, objektgjenkjenning og tekstgjenkjenning (LLC, ingen dato b).

API-et er laget for å kunne kjøre på Google Cloud Run. For å deploye en ny versjon kan kommandoen "gcloud run deploy" kjøres. Selve deployeringsprosessen kan leses mer om her.

Produktet er installert hos oppdragsgiver som en del av prosjektet.

8 DOKUMENTASJON AV KILDEKODE

I prosjektet har vi inkludert enkle dokumentasjonsmetoder som en del av arbeidet. Dette er gjort ved å opprette README.md-filer som gir en oversiktlig beskrivelse av hvordan programmet fungerer. For å gi ytterligere forklaringer og innsikt i kildekoden, har vi benyttet oss av kommentarer som beskriver hva ulike deler av koden gjør, samt hvilke typer input de aksepterer.

En annen nyttig praksis vi har implementert er bruken av “typing”. Selv om Python i utgangspunktet ikke er et “typed” språk, har vi inkludert denne funksjonaliteten for å dra nytte av flere fordeler som fører til en mer oversiktlig kodebase. Ved å inkludere typer i kildekoden, blir det lettere å forstå hvilke typer variabler og verdier som forventes som input til ulike funksjoner. Dette kan bidra til å redusere feil og misforståelser i utviklingsprosessen (*Typing Python Libraries — typing documentation*, ingen dato).

Samlet sett bidrar tillegg av enkel dokumentasjon i form av README.md-filer, bruk av kommentarer og inkludering av "typing" til å gjøre koden mer forståelig og oversiktlig for alle som jobber med prosjektet.

9 KONTINUERLIG INTEGRASJON OG TESTING

I et typisk kontinuerlig utviklingsprosjekt vil utviklere ofte gjøre hyppige endringer i kildekoden og ønske å deployere og teste disse endringene så raskt som mulig. Google Cloud Run og Google Cloud Build er to verktøy som kan brukes i denne prosessen for å sikre rask og pålitelig utvikling (*Cloud Run: Container to production in seconds*, ingen dato; *Cloud Build Serverless CI/CD Platform*, ingen dato).

Google Cloud Run er en tjeneste som gjør det mulig å kjøre stateless konteinere som en enkelt funksjon, noe som betyr at det er veldig raskt å deployere og skalerer automatisk etter behov. Google Cloud Build er et verktøy for kontinuerlig integrasjon og distribusjon (CI/CD) som gjør det mulig å bygge og deployere programvare automatisk (*Cloud Run: Container to production in seconds*, ingen dato; *Cloud Build Serverless CI/CD Platform*, ingen dato).

Testing i programmet innebærer bruk av ulike verktøy og filer for å evaluere funksjonaliteten og påliteligheten til programmet. En viktig del av testingen er bruken av filen "run_test.py", som genererer en ny fil kalt "testresult.json". Denne filen inneholder en samling av forventede data og dataene produsert av modellen. For å analysere disse dataene, kan man bruke "testresult_check.py"-filen, som gir en enkel analyse av resultatene.

En annen viktig fil for testing er "algorithm_test.py", som tester algoritmen ved hjelp av en spesifikk fil. Denne testen sikrer at algoritmen fungerer som forventet i en normal flyt situasjon.

10 KILDEKODE OG DEMOSIDE

10.1 Kildekode

Kildekoden er konfidensiell. Det blir gitt tilgang til koden gjennom en GitHub-bruker. Tilgangen fjernes etter sensur.

URL: https://github.com/591337/BachelorProject_PDF_Notes

Brukernavn: hvlNotesett

Passord: Exit0-Reappoint-Bluish

Tofaktor-autentiserings-QR-kode:

10.2 Demoside

I prosjektet er det laget en demoside. Denne siden er ment for å gjøre det mulig å demonstrere hvordan API-et fungerer. Demosiden er laget på en slik måte at API-et ikke blir eksponert. Her finnes det også to eksempelfiler.

URL: https://demopage-zhoxnvhm7a-ew.a.run.app/

11 REFERANSER

Cloud Build Serverless CI/CD Platform (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/build (hentet: 21 May 2023).

Cloud Run: Container to production in seconds (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/run (hentet: 18 May 2023).
Cloud SDK - Libraries and Command Line Tools (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/sdk (hentet: 20 May 2023).
Cloud Storage (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/storage (hentet: 20 May 2023).
Detect text in files (PDF/TIFF) | Cloud Vision API (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/vision/docs/pdf (hentet: 10 May 2023).
LLC, G. (ingen dato a) 'google-cloud-storage: Google Cloud Storage API client library'. tilgjengelig: https://github.com/googleapis/python-storage (hentet: 20 May 2023).
LLC, G. (ingen dato b) 'google-cloud-vision: Google Cloud Vision API client library'. tilgjengelig: https://github.com/googleapis/python-vision (hentet: 20 May 2023).
Quickstart: Deploy a Python service to Cloud Run | Cloud Run Documentation (ingen dato) *Google Cloud*. tilgjengelig: https://cloud.google.com/run/docs/quickstarts/build-and-deploy/deploy-python-service (hentet: 18 May 2023).
Typing Python Libraries — typing documentation (ingen dato). tilgjengelig: https://typing.readthedocs.io/en/latest/source/libraries.html# (hentet: 20 May 2023).
Welcome to Flask — Flask Documentation (2.3.x) (ingen dato). tilgjengelig: https://flask.palletsprojects.com/en/2.3.x/# (hentet: 19 May 2023).

Figurliste