Høgskulen på Vestlandet

Fakultet for teknologi, miljø- og samfunnsvitenskap

Institutt for datateknologi, elektroteknologi og realfag

Bildekvalitetsdetektor for dekk på kjøretøy i bevegelse

Image Quality Detector for tires on vehicles in motion

Kravdokumentasjon

Versjon <2.0>

REVISJONSHISTORIE

Dato	Versjon	Beskrivelse	Forfatter
<25/02/24>	<1.0>	1. iter av kravdokument	Bjørn Ellingsen Håkon Lervåg Øyvind Holter
<13/04/24>	<2.0>	2. iter av kravdokumentasjon. Gjort en del oppdateringer	Bjørn Ellingsen Håkon Lervåg Øyvind Holter

INNHOLDSFORTEGNELSE

1 INNLEDNING

Dette dokumentet er utformet med det formål å gi en omfattende beskrivelse av de funksjonelle kravene som ble stilt i visjonsdokumentet knyttet til vårt system. Dokumentet inneholder et brukstilfellediagram, domenemodell, wireframe samt en HTML-prototype. Disse kravene er avgjørende for å sikre at systemet fungerer som forventet og leverer de nødvendige tjenestene til Counting Hero.

2 FUNKSJONALITET

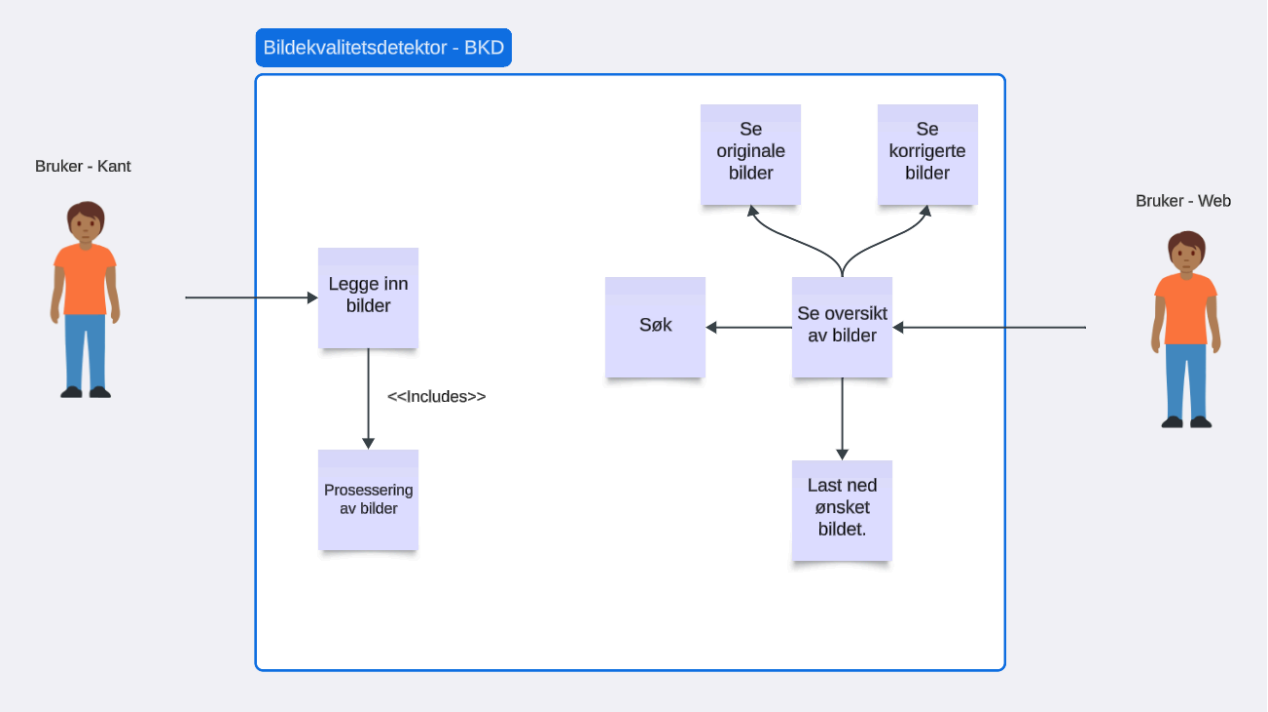
For å tydeliggjøre systemets opprinnelige krav på en mer visuell og forståelig måte, utarbeidet vi et brukstilfellediagram basert på de funksjonelle kravene som er beskrevet i kapittel 5 i visjonsdokumentet. Figur 2.1 viser en helhetlig representasjon av vårt opprinnelige system, både på web og “kant”. Brukstilfellebeskrivelsene for figur 2.1 er dokumentert i tabellene fra 2.1 til 2.6.

Som figur 2.1 viser, har brukeren i kant mulighet til å "Legge inn bilder". Etter at bildene er lagt inn, blir de prosessert av vårt system. Prosesseringen av bildene har visse krav som den skal oppfylle. Den skal:

- Kunne ta imot bilder av bildekk for deteksjon.
- Kunne detektere om et bilde med “motion blur”, dårlig lys og urent kamera.
- Kunne korrigere bilder med “motion blur”
- Lagre bilder i lokal lagring.

Som figur 2.1 viser, har brukeren på nettet muligheten til å "Se oversikt av bilder" på nettsiden. Den interaktive delen av prosjektet er nettsiden som brukere kan benytte for å se bildene som er blitt analysert for dårlig lys, "motion blur" og urent kamera. Nettsiden har visse krav som den må oppfylle:

- Oversikt over alle instanser av kjøretøy med relevant informasjon.
- Brukerdefinert søking etter ID, dato, klokkeslett, sted eller flagg.
- Kolonne med oversikt over originale bilder
- Kolonne med oversikt over korrigerte bilder
- Knapp for nedlasting av bilder.



Figur 2.1: Brukstilfellediagram

Tabell 2.1 Brukstilfellebeskrivelse: Kant

Navn:	Kant
Aktører:	Bruker
Hensikt/Målsetting:	Bruker skal kunne legge inn en bilder til analyse.
Normalflyt:	1. *Bruker legger inn bilder.* 2. *Prosessering av bilder kjøres.*
Unntaksflyt [#1]:	1. Ingen bilder - ingen resultat.

Tabell 2.2 Brukstilfellebeskrivelse: Web - Se oversikt av bilder

Navn:	Web - Se oversikt av bilder.
Aktører:	Bruker
Hensikt/Målsetting:	Brukere skal få en oversikt over alle bildene med relevant informasjon.
Normalflyt:	1. *Åpner nettside og blir møtt med en oversiktlig nettside med ulike filter.*

Tabell 2.3 Brukstilfellebeskrivelse: Web - Søk

Navn:	Web - Søk.
Aktører:	Bruker
Hensikt/Målsetting:	For å få en god oversikt på en nettside med mye informasjon må vi ha søkefunksjonalitet.

Normalflyt:	*1. Trykker på søkeknappen øverst på nettsiden*

Tabell 2.4 Brukstilfellebeskrivelse: Web - Se originale bilder

Navn:	Web - Se originale bilder.
Aktører:	Bruker
Hensikt/Målsetting:	Filosofien er at brukeren skal ha tilgjengelighet til alle ressurser. Derfor har vi inkludert originalbilder slik at de er enkle å få tilgang til dersom de skulle være nødvendige.
Normalflyt:	*1. Nettsiden viser de originale bildene*

Tabell 2.5 Brukstilfellebeskrivelse: Web - Se korrigerte bilder

Navn:	Web - Se korrigerte bilder.
Aktører:	Bruker
Hensikt/Målsetting:	Brukere skal ha enkel tilgang til de korrigerte bildene.
Normalflyt:	*1. Nettsiden viser de korrigerte bildene*
Unntaksflyt [#1]:	1. Bildene behøvde ingen korrigering - ingen korrigerte bilder.

Tabell 2.6 Brukstilfellebeskrivelse: Web - Last ned bilder

Navn:	Web - Last ned bilder.
Aktører:	Bruker
Hensikt/Målsetting:	Dersom brukere ønsker å laste ned bildene er det mulig.
Normalflyt:	*1. Trykker på ikonet for "Last ned".*

3 DOMENEMODELL

Vi har konstruert en domenemodell for det opprinnelige systemet. Modellen er avgjørende for å forstå hvordan domenene samhandler og skal gi en overordnet forståelse for hvordan systemet fungerer. Domenemodellen er basert på de funksjonelle kravene som ble presentert i kapittel 5 av visjonsdokumentet. Vår initielle løsningsidé hadde en del problemer som vi identifiserte utover prosjektet. Vi hadde dermed behov for å avgrense og gjøre endringer for å komme frem til den endelige problemstillingen. Disse endringene fører til at domenemodellen har noen forskjeller fra den endelige løsningen.

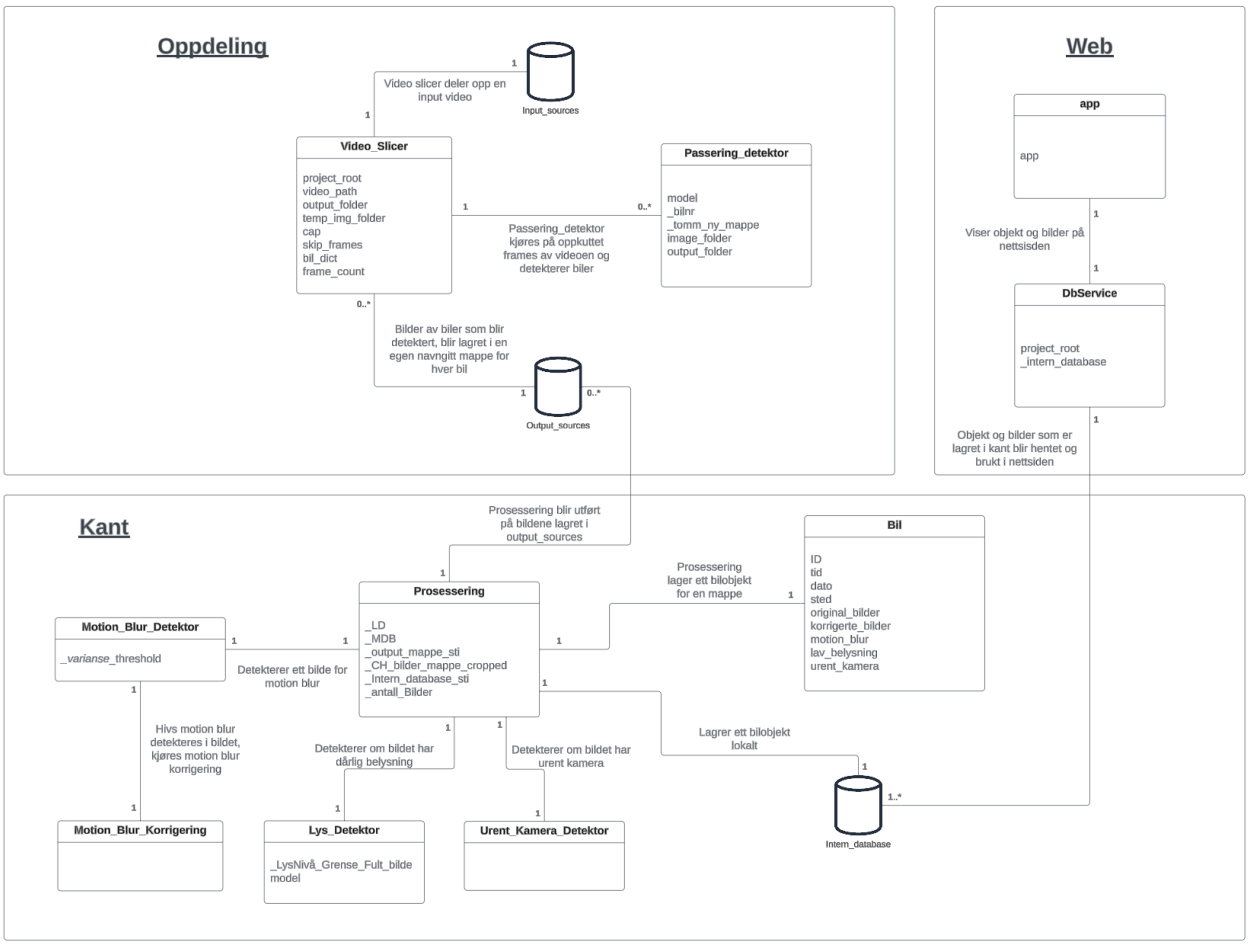
Ved utformingen av systemet utarbeidet vi en grunnleggende plan for utviklingen. Vi trengte en bedre forståelse av problemet vårt og hva som krevdes for å løse det. Derfor opprettet vi spesifikke klasser for videosegmentering. I figur 3.1 inkluderer dette en "Video_Slicer" som kunne segmentere videomaterialet til individuelle bilder, og en "Passering_detektor" som kunne avgjøre om et bilde inneholdt et kjøretøy eller ikke.

På dette stadiet i utviklingen brukte vi ikke de bildene av dekk som vår endelige løsning benytter, da vi ennå ikke hadde fått tilgang til disse. Vi søkte på nett og fant en video [1] med passende lysforhold som et utgangspunkt for å skaffe bilder av kjøretøy å jobbe med. Formålet med å bruke en oppkuttet video og utvikle disse klassene var å parallellisere arbeidet vårt for å øke effektiviteten i utviklingsprosessen. På dette tidspunktet hadde vi satt opp prosjektet vårt slik at vi kunne begynne å utvikle nettsiden med midlertidige bilder, samtidig som vi begynte å arbeide med deteksjonen på bildene som vi kuttet opp fra denne testvideoen. I den endelige koden for løsningen kan disse klassene fremdeles kjøres med en testvideo som et alternativ til bildene vi fikk fra Counting Hero. I kapittel 8 av systemdokumentasjonen har vi forklaring til hvordan man gjør dette.

Klassene "Video_Slicer" og "Passering_detektor" er laget for oppkutting av en video for testing samt effektivisere utviklingen. På grunn av dette er disse klassene ikke relevante for vår endelige løsning og er derfor ikke inkludert som en del av klassediagrammet i systemdokumentasjonen.

Domenemodellen som vist i figur 3.1 inkluderer "Motion_Blur_Korrigering" og "Urent_Kamera_Detektor" i henhold til de funksjonelle kravene som ble satt. Disse funksjonelle kravene ble avgrenset senere i prosjektperioden. "Motion_Blur_Korrigering" og "Urent_Kamera_Detektor" er derfor ikke en del av den endelige løsningen. Endringene som ble gjort for å avgrense oppgaven blir forklart nærmere i kapittel 2.2 i rapporten.

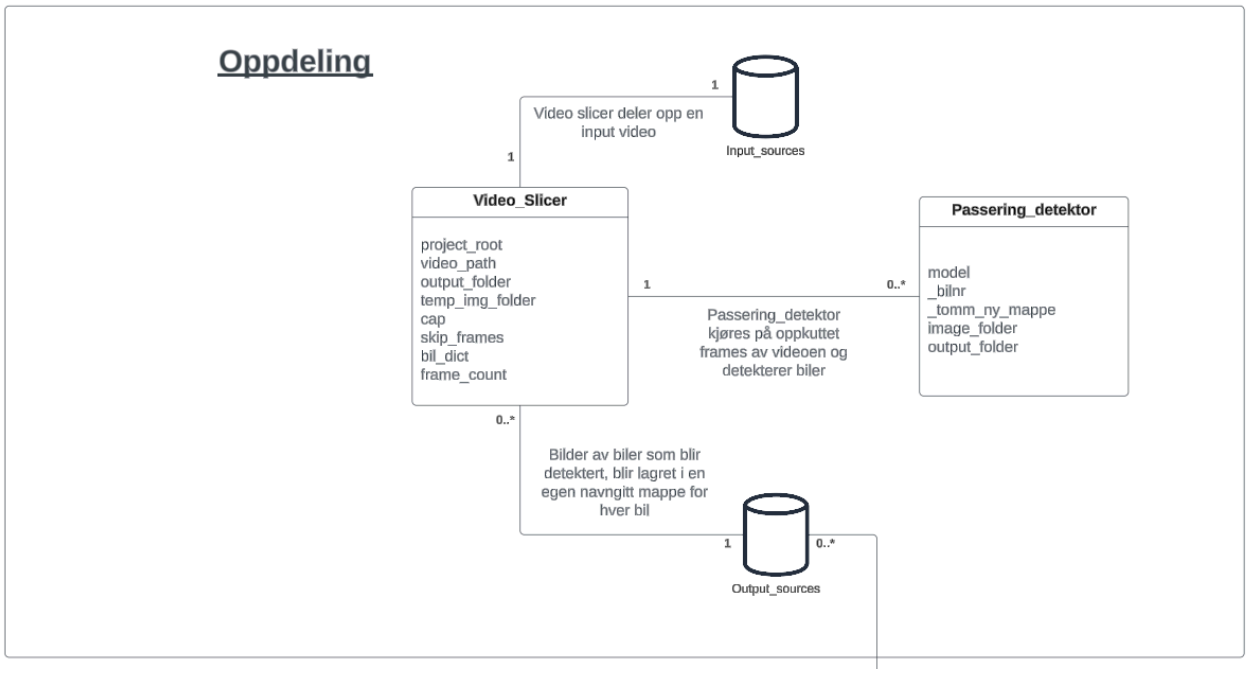
Domenemodellen i figur 3.1 består av 3 deler for å gjøre den mer oversiktlig: Oppdeling, Kant, og Web.



Figur 3.1 Domenemodell [2]

3.1 Klasser for oppdeling av videoer:

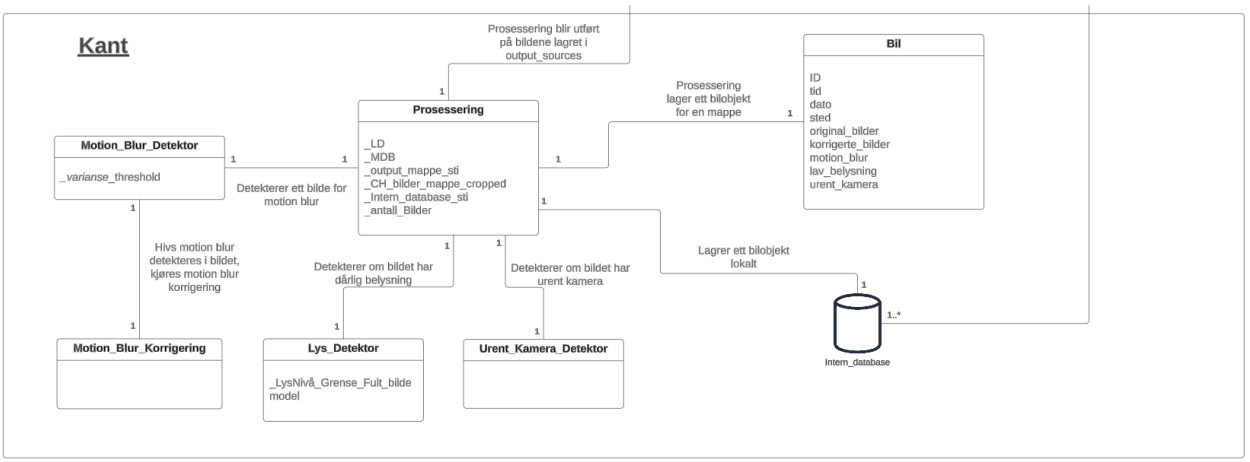
Figur 3.2 illustrerer hvordan vi la videon som vi brukte inn i "Input_Sources". Klassen "Video_Slicer" kunne segmentere videomaterialet til individuelle bilder, og "Passering_detektor" kunne avgjøre om et bilde inneholdt et kjøretøy eller ikke. Til slutt ble de ferdig oppkuttet bildene av kjøretøy lagret i “Output_sources” mappen.



Figur 3.2 Domenemodell - Klasser for oppdeling av videoer[2]

3.2 Kant

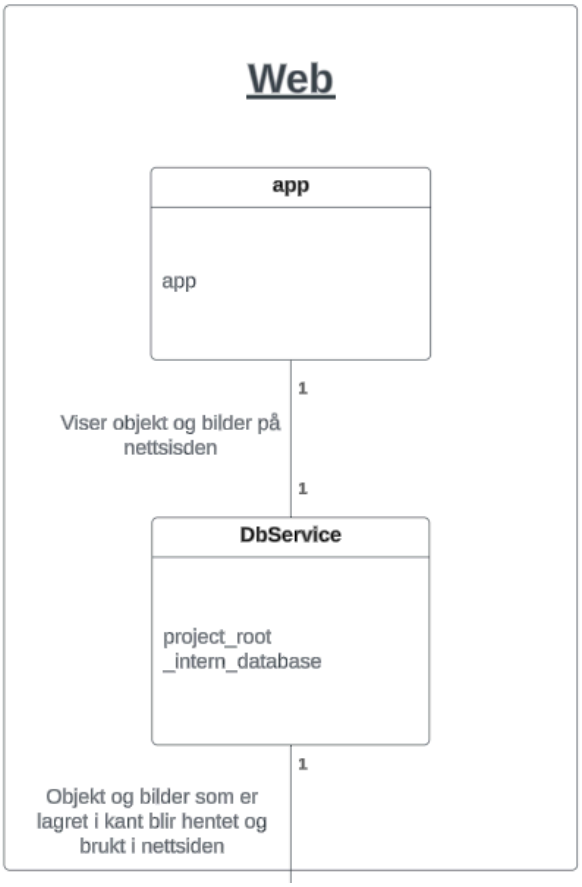
Fra figur 4.3 ser vi hvordan klassen “Prosessering” sjekker bildene i mappen “Output_Sources” for feil med bildekvaliteten. I den opprinnelige planen for systemet skulle “Prosessering” kjøre 3 detektorer. Disse detektorene var “Motion_Blur_Detektor”, “Urent_Kamera_Detektor” og “Lys_Detektor”. Planen på dette tidspunktet var også å implementere “Motion_Blur_Korrigering”. Etter at bildene var prosessert skulle klassen lagre bilobjektene lokalt i en mappe som het “Intern_database”. Med prosesseringen av bildene gjennomført gjenstod det å vise frem bilobjektene med de tilhørende bildene på nettsiden.



Figur 3.3 Domenemodell - Kant [2]

3.3 Web

For webutvikling har gruppen benyttet Flask [3]. Standarden for applikasjoner som benytter dette rammeverket er å ha en klasse med navnet "app", slik figur 3.4 viser. "DbService" fungerer som backend for nettsiden og utfører funksjoner for å hente bildene fra der de er lagret, slik at de kan vises på nettsiden.



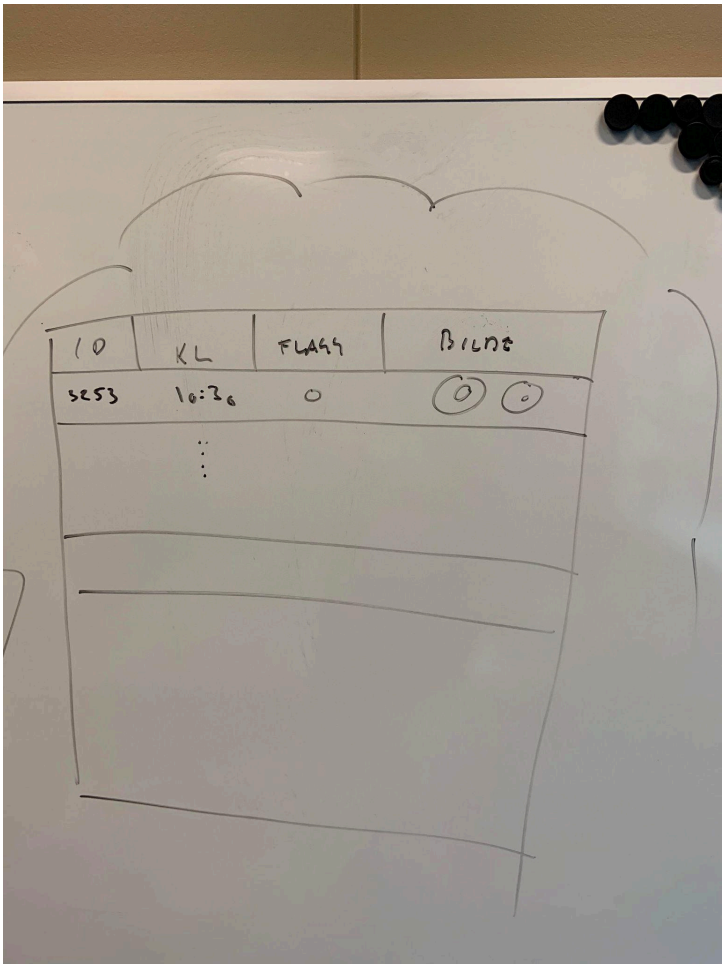
Figur 3.4 Domenemodell - Web [2]

4 PROTOTYPER

For å gi en mer konkret ide om hvordan brukergrensesnittet til systemet vil se ut, har vi også utviklet wireframes og HTML-prototyper.

4.1 Wireframes

Figur 4.1 viser første iterasjon fra tidlig i planleggingsfasen for hvordan nettsiden til systemet kan se ut. Figuren er et bilde av en enkel tegning tatt fra et tidlig møte sammen med oppdragsgiver. Vi fikk tilbakemeldinger fra oppdragsgiver om hva som ønskes å vises av relevant informasjon som et utgangspunkt for hvordan nettsiden skulle se ut. I figur 4.1 kan man se kolonner for relevant informasjon som vi ønsker å vise frem på nettsiden. Fra venstre har vi ID, klokkeslett, flagg for markering av feil med bildekvaliteten, og bildet som er blitt sjekket i kolonnen helt til høyre.



Figur 4.1 Wireframe av nettsiden

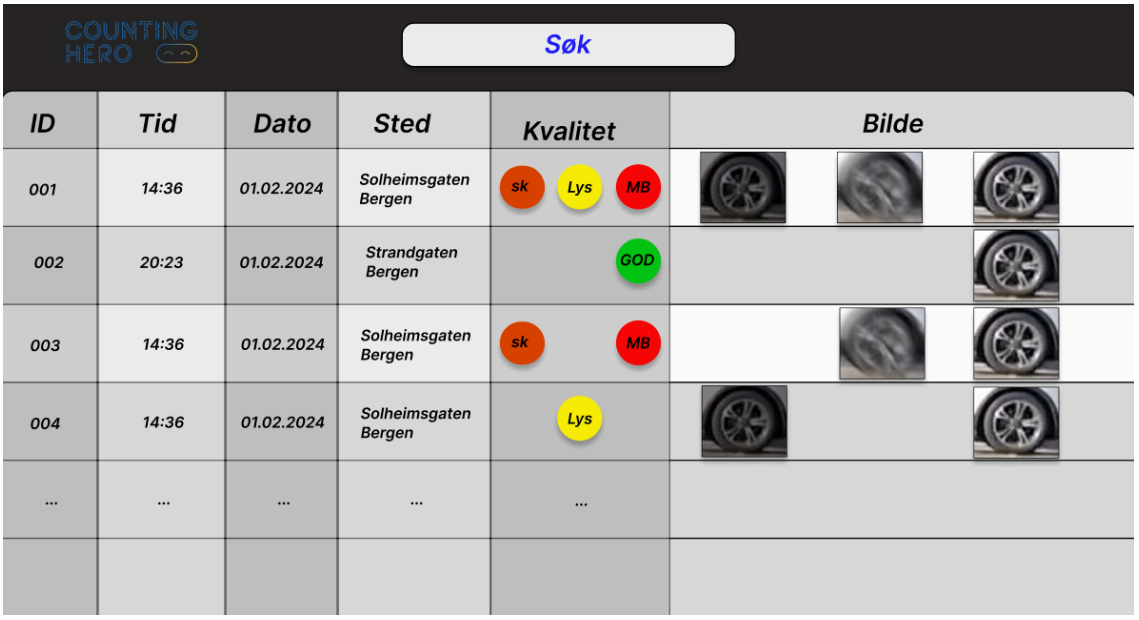
4.2 HTML-prototyper

HTML-prototypen av nettsiden, slik den vises i figur 4.2, er designet for å presentere relevant informasjon knyttet til deteksjon av bildekvalitet. Prototypen er designet i Figma [4]. Figma er en skybasert plattform for design og prototyping av brukergrensesnitt. HTML-prototypen tar utgangspunkt i de funksjonelle kravene i visjonsdokumentet kapittel 5 og wireframe av nettsiden fra forrige kapittel.

Figur 4.2 viser informasjonen som blir vist frem på nettsiden. Dette inkluderer identifikasjonsnummer (ID), tidspunkt, dato og sted for bildet som er tatt. Disse dataene kan søkes etter i søkefeltet for å gjøre det enklere for brukerne å navigere på nettsiden.

En sentral funksjon i denne prototypen er definisjonen av bildekvalitet. Disse indikatorene gir en visuell representasjon av det som er oppdaget i bildet, og markerer eventuelle feil eller mangler som kan påvirke bildekvaliteten. Indikatorene er "GOD" for god bildekvalitet, "SK" for skittent/urent kamera, "Lys" for dårlig belysning, og "MB" for "motion blur".

Videre viser prototypen både det originale bildet og det prosesserte bildet. Det prosesserte bildet inkluderer eventuelle korrigeringer som er gjort for å forbedre bildet.



Figur 4.2 HTML-prototype av nettsiden

5 REFERANSER

[1] N. Sanchar, Sandeide bussterminal, Fyllingsdalen, Bergen, Norge. *Car passing by in a Highway - Royalty Free Stock Video / (Copyright Free) Download.* (Jun. 04, 2018). Hentet Feb. 05, 2024. [Online video]. Tilgjengelig: https://www.youtube.com/watch?v=K6xsEng2PhU&ab_channel=NauloSanchar

[2] Linnæus University. UML Domain Model [Online]. Tilgjengelig: https://coursepress.lnu.se/courses/object-oriented-analysis-and-design/02-theory/domain_class_diagram (Hentet 15. Februar 2024)

[3] Flask, (2010). Flask Documentation [Online]. Tilgjengelig: https://flask.palletsprojects.com/en/3.0.x/

[4] Figma, (2024). Figma: The Collaborative Interface Design Tool [Online]. Tilgjengelig: https://www.figma.com/