Det første vi gjorde ved starten av prosjektet var å se gjennom lignende prosjektarbeid som var gjort tidligere. Her fant vi mye forskjellig med varierte implementasjoner og mål. I vårt prosjekt krevde vi OpenCV til deteksjon, video framfor individuelle bilder, datasett med fartsgrenseskilt, og rask nok maskinlæringsmodell til å kunne kjøre på mobiltelefon. Følgende prosjekter ble brukt til inspirasjon:

- https://github.com/kenshiro-o/CarND-Traffic-Sign-Classifier-Project
- https://drive.google.com/drive/folders/1BjKon6kIaPNvLJ8uRYxROiM_x93OodLu
- https://github.com/vamsiramakrishnan/TrafficSignRecognition
- https://github.com/jacobssy/Traffic_Sign_detection
- https://github.com/hoanglehaithanh/Traffic-Sign-Detection
- https://www.geeksforgeeks.org/opencv-and-keras-traffic-sign-classification-for-self-driving-car/

Etter at vi hadde gått gjennom de relevante prosjektene vi fant på nett, satte vi oss ned og diskuterte for å bli enige om hvilken strategi vi skulle bruke til skiltdeteksjon og klassifisering. Her gikk det også mye tid til å lese oss opp om forskjellige deteksjonsstrategier og maskinlæringsmodeller, for å finne ut hva som passet best til vårt formål.

Det gikk også en god del tid til oppsett av miljø for å teste noen av prosjektene vi fant på nett. Grunnen til at vi gjorde dette var for å se hvilke løsninger som fungerte best, og hvilke som var raskest i praksis. Vi satte blant annet opp miljø med cuda, anaconda og wsl. Etter å ha lest oss opp og testet mye forskjellig, valgte vi til slutt å bruke SVM som modell for klassifiseringen fordi den er rask og relativt robust. For detekteringen fant vi mye bra inspirasjon i prosjektene vi fant på nett, og bestemte oss for å kjøre en kombinasjon av flere deteksjonsstrategier, hvor fokuset lå på å balansere riktig detektering og høy ytelse.

Da vi følte vi hadde en god aning om hvordan vi skulle løse oppgaven, satte vi opp et prosjekt i git, og startet med å prøve oss fram. Det gikk ganske kjapt å få på plass detekteringen, i og med at vi fant mye open source kode som dekket dette, men vi måtte fortsatt bruke en del tid på å sette oss inn i, og forstå de forskjellige strategiene. Etter mye testing med forskjellige kombinasjoner av strategier, hadde vi kommet fram til en som syntes å fungere godt.

Da vi hadde detektering så og si på plass, begynte vi å se på SVM-modellen. Her var det også mye å finne på nett, men de fleste implementasjoner fungerte svært dårlig da vi prøvde å kopiere direkte for å se hvordan resultatet ble. Vi bestemte oss for en implementasjon, og begynte å sette oss godt inn i hvordan den fungerte sånn at vi skulle kunne forbedre den så mye som mulig.

Vi brukte et datasett fra nettet med tyske fartsgrenseskilt, som er nesten like med norske. De eneste skiltene som mangler er 90- og 110-skilt. Datasettet pakket vi ut, og lagret bildene på .png-format i stedet. Dette gjorde vi gjennom python sin pickle.load-funksjon på "train"-fila. signnames.csv inneholdt hvilke skilttyper som korresponderte med hvilke "labels" i train-fila. En label vil si et fasitsvar på hva bildet representerer (et av skiltene i datasettet). train-fila hadde to "kolonner" med bilde og label. Hver av bildene i train-fila hadde da en korresponderende label i samme rad. Dette gjorde at vi kunne pakke ut hver klasse av bilder til hver sin mappe i datasettet. Dette fungerte til slutt, og vi brukte en video til å teste på. Deretter filma vi flere videoer til og fra skolen, i tillegg til litt ekstra kjøring for å få inn nok testdata.

Resultatet var lovende, men langt ifra optimalt, og vi skjønte at vi kom til å måtte bruke mye tid på å tweake detekteringen og modellen, men dette rakk vi ikke å begynne noe særlig med disse ukene.