**Bidrag til hovedrapport**

Kode/app:

Vi har vært avhengig av å bruke Google API til flere deler av vår webapplikasjon. Jeg har kodet vår autocomplete funksjon (basert på Google) slik at brukeren får opp forslag til start og stopp lokasjon når man fyller inn tekstfeltene. Deretter vil Google hente koordinatene basert på valgte steder.

Men hovedsakelig benytter vi oss av Vegvesenets tjeneste Visveg, og deres API. Visveg benytter den europeiske standarden ERST89 som måles i meter, i motsetning til Google som er basert på den amerikanske WGS84 som bruker grader. Man må også ta hensyn til «sone», for eksempel regnes norsk sone som 33. Min oppgave var derfor å lage en konverter som ligger mellom Google og Visveg. Denne konverteren får inn koordinatene fra Google, konverter dem til europeisk standard, og sender de nye dataene til Visveg i en JSON-string.

Vi har undersøkt om det er mulig å benytte andre api’er enn de vi allerede bruker. Finnes det andre systemer som kombinerer de ulike funksjonalitetene vi ser etter? Det hadde selvsagt vært enklere og mer oversiktlig å kun benytte seg av et api. Dessverre finnes det ingen system tilgjengelig som både kan gi god informasjon om avstander, bomstasjoner (med priser) og kjøreretning. Så lenge vi vil benytte Googles autocomplete-funksjon samt ha nødvendig informasjon om bomstasjoner må vi kombinere disse. 1881 sitt system for veibeskrivelser innehar alle nødvendige data vi trenger. Men de tilbyr ikke noe api for veibeskrivelser, kun for personsøk.

Et problem som dukker opp når vi kombinerer Google og Visveg er at kjøreretning på kartet kan vises forskjellig fra system til system. Det vil si at informasjonen om avstand og bompenger vi får fra Visveg vil stemme med det brukeren skriver inn, men ikke nødvendigvis med kjøreretningen på Google-kartet. For eksempel kan en bruker ha kjørt fra Oslo til Kirkenes langs E6, mens Google vil vise korteste distanse, altså gjennom Sverige og Finland. Google tilbyr ingen mulighet for restriksjoner i forhold til landegrenser. På den andre siden støtter ikke Visveg kjøreretninger utenom Norge. *Google støtter heller ikke andre karttjenester kombinert med deres autocomplete-funksjon.*

Vi har diskutert om vi virkelig har behov for en karttjeneste i vår app. Vi skal ikke levere et system som støtter veibeskrivelser, men et system som skal behandle reiser etter at de har funnet sted. Kartet er ment som et hjelpemiddel for å bekrefte reiseruten som ble valgt da brukeren gjennomførte sin reise. Kan denne informasjonen vises på andre måter? I stedet for å hente informasjon om valgt reiserute fra Google, kan vi heller vise den i tekstformat fra Visveg. For eksempel: Bruker A kjørte fra Oslo til Trondheim, E6 til Hamar, om Østerdalen, og videre på E6 frem til destinasjonen. Jeg vil argumentere for at denne informasjonen best vises punktvis i tekstformat fremfor en stiplet linje på et kart.

Vi har i det siste begynt å flytte noe av logikken over på server. Det er for at systemet skal være mer vedlikeholdbart. For eksempel i forhold til gjenbruk slik at man slipper å lage samme komponent flere ganger. Men også for å unngå «cross-domain», altså at brukerens nettleser som ligger på vår server oppretter kontakt med en annen server for å innhente informasjon. I stedet skal vår server innhente informasjonen fra andre steder, og så vise dataene til brukeren i nettleseren. Brukeren skal kun være på vår server hele tiden. Andre fordeler er at man løser «interoperability problems», altså at forskjellige systemer på forskjellige plattformer kan kommunisere med hverandre.

Jeg er en av to som har som oppgave å se hvordan dette kan løses. En mulighet vi nå jobber med er å sette opp en web service som oppretter kontakt med vegvesenets server. Det vil si at nettleseren sender en ajax-request til vår web service som så sender en forespørsel til visveg. Når denne svarer får web servicen et svar tilbake (i vårt tilfelle JSON), som så sender denne tilbake til frontend-scriptet som viser dataene til brukeren.

Jeg har både sett på hvordan man kan gjøre dette i Microsoft sitt ASP.Net rammeverk som gjør jobben med å snakke med en webserver mindre krevende, men også hvordan dette kan la seg gjøre fra en ren html-side. Microsoft sitt ASP.Net er ment å gjøre jobben med å lage en dynamisk webside enklere. Det vil si at det å vedlikeholde websiden (innholdet) skal være mer overkommelig. Fo eksempel hvis man vil inkludere en ny side (web forms/ aspx) eller redigere innholdet i en meny. Men samtidig vil arbeidet med å «snakke» med en web service vært enklere via web forms enn i ren html. Microsoft introduserte muligheten for at backend-koden ligger i aspx.cs-filer (.cs hvis man bruker C#), mens statisk kode ligger i aspx-filene. Det vil si at all kode som skal behandle funksjoner basert på en brukers valg på siden behandles av filene med aspx.cs-endelse.

Alle behandlede og kompilerte filer i ASP.Net lagres i ulike kataloger som er tilgjengelig for alle sider på nettstedet. Det inkluderer også web services. Alle filer (inkludert WSDL- filer som bestemmer hvordan kommunikasjonen skal foregå) som refererer til en web service ligger i en egen katalog slik at denne kan nås enkelt i koden fra alle sider (web forms).

Innhenting av informasjon:

Den informasjonen jeg har trengt for å kunne fullføre arbeidet jeg har blitt satt til, har blitt innhentet hovedsakelig fra nett. Det er særlig arbeidet med å forstå koordinater og de forskjellige standardene jeg har jobbet med.

Jeg organiserte også et møte med foreleser Per Bisseberg ved HIOF for å få flere synspunkt på vårt arbeid, samt mer informasjon om hvordan ASP.Net applikasjoner med web service virker.

* Stackoverflow: Dette nettsamfunnet har jeg benyttet til alle deler av arbeidet.
* Youtube: Her finner man mange gode tutorials. Særlig har jeg lett etter informasjon på hvordan man bruker en web service i Visual Studio.
* Google Developers: Fremgangsmåte for å bruke Googles autocomplete-funksjon (og Google Maps).

Bidrag til forprosjektrapport:

Jeg skrev de delene som omhandlet bakgrunnen for prosjektet, formål vi har og informasjon om oppdragsgiver.