



UNIVERSITETET I AGDER

DAT215

Prosjekt Rapport
Rusta Vrak Bilutleige

Student:

Martin ENGEN

Veileder:

Arne WIKLUND

Ved

Institutt for IKT

Fakultet for Teknologi og Realfag

2. desember 2016

Obligatorisk egenerklæring/gruppeerklæring

Den enkelte student er selv ansvarlig for å sette seg inn i hva som er lovlige hjelpemidler, retningslinjer for bruk av disse og regler om kildebruk. Erklæringen skal bevisstgjøre studentene på deres ansvar og hvilke konsekvenser fusk kan medføre. Manglende erklæring fritar ikke studentene fra sitt ansvar.

1.	Jeg/vi erklærer herved at min/vår besvarelse er mitt/vårt eget arbeid, og at jeg/vi ikke har brukt andre kilder eller har mottatt annen hjelp enn det som er nevnt i besvarelsen.	<input checked="" type="checkbox"/>
2.	Jeg/vi erklærer videre at denne besvarelsen: <ul style="list-style-type: none"> - ikke har vært brukt til annen eksamen ved annen avdeling/universitet/høgskole innenlands eller utenlands. - ikke refererer til andres arbeid uten at det er oppgitt. - ikke refererer til eget tidligere arbeid uten at det er oppgitt. - har alle referansene oppgitt i litteraturlisten. - ikke er en kopi, duplikat eller avskrift av andres arbeid eller besvarelse. 	<input checked="" type="checkbox"/>
3.	Jeg/vi er kjent med at brudd på ovennevnte er å <u>betrakte som fusk</u> og kan medføre annullering av eksamen og utestengelse fra universiteter og høyskoler i Norge, jf. Universitets- og høyskoleloven §§4-7 og 4-8 og Forskrift om eksamen §§ 31.	<input checked="" type="checkbox"/>
4.	Jeg/vi er kjent med at alle innleverte oppgaver kan bli plagiatkontrollert.	<input checked="" type="checkbox"/>
5.	Jeg/vi er kjent med at Universitetet i Agder vil behandle alle saker hvor det forligger mistanke om fusk etter høyskolens retningslinjer for behandling av saker om fusk.	<input checked="" type="checkbox"/>
6.	Jeg/vi har satt oss inn i regler og retningslinjer i bruk av kilder og referanser på biblioteket sine nettsider.	<input checked="" type="checkbox"/>

Publiseringsavtale

Fullmakt til elektronisk publisering av oppgaven

Forfatter(ne) har opphavsrett til oppgaven. Det betyr blant annet enerett til å gjøre verket tilgjengelig for allmennheten (Åndsverkloven. §2).

Alle oppgaver som fyller kriteriene vil bli registrert og publisert i Brage Aura og på UiA sine nettsider med forfatter(ne)s godkjennelse.

Oppgaver som er unntatt offentlighet eller taushetsbelagt/konfidensiell vil ikke bli publisert.

Jeg/vi gir herved Universitetet i Agder en vederlagsfri rett til å gjøre oppgaven tilgjengelig for elektronisk publisering:

☒ JA ☐ NEI

Er oppgaven båndlagt (konfidensiell)?

☐ JA ☒ NEI

(Båndleggingsavtale må fylles ut)

- Hvis ja:

Kan oppgaven publiseres når båndleggingsperioden er over?

☐ JA ☐ NEI

Er oppgaven unntatt offentlighet?

☐ JA ☒ NEI

(inneholder taushetsbelagt informasjon. Jfr. Offl. §13/Fvl. §13)

Sammendrag

Dette prosjektet har gått ut på å utvikle en nettside for bedriften Rusta Vrak Bilutleige. Denne nettsiden skal fungere som et bestillingsverktøy for kunder som ønsker å leie bil, og for at bedriften skal ha et verktøy for å kunne ha oversikt over alle utleier til enhver tid. I dag har Rusta Vrak Bilutleige en enkel informasjonsnettside uten noen form for funksjonalitet, og har derfor et ønske om å kunne tilby sine kunder et alternativt digitalt bestillingsverktøy.

I tillegg til selve utviklingen, skulle også prosjektet publiseres og gjøres tilgjengelig på internett. Til dette formålet har det blitt benyttet en metode for publisering i skytjenesten til Google, på plattformen Google App Engine. Ved å benytte en slik publiseringsmetode kunne det settes av mer tid til selve utviklingen av nettsiden.

Innhold

Sammendrag	iii
Innhold	iv
List of Figures	vi
1 Innledning	1
1.1 Bakgrunn	1
1.2 Problemdefinisjon	1
1.2.1 Oppgaven	1
1.2.2 Krav	2
1.3 Litteratur Studie	2
1.4 Problemløsning	3
1.4.1 Prosjekt Plan	3
1.5 Arbeidsmetode	4
1.6 Rapportens Struktur	5
2 Oversikt over valgte Teknologier	6
2.1 Publisering	6
2.1.1 Platform as a Service	6
2.1.2 Fordeler ved PaaS	6
2.1.3 Valg av PaaS Leverandør	7
2.2 Back-end	8
2.2.1 Python	8
2.2.2 Django	8
2.3 Front-end	9
2.3.1 Bootstrap	9
2.3.2 JavaScript Biblioteker	9
2.4 Verktøy	10
3 Løsning	11
3.1 Krav	11
3.1.1 Kundens Workflow	11
3.2 Designspesifikasjoner	12
3.3 Resultat	13
3.3.1 Forsiden	13
3.3.2 Liste over Biler	14
3.3.3 Individuelle Biler	15

3.3.4	Kontaktskjema	16
3.3.5	Bekreftelse	17
3.4	Administrasjon	18
3.4.1	Reservasjoner i Administrasjonsside	19
3.5	Database	20
3.6	Priser og Rabatt	21
3.7	Testing og Validering	22
3.7.1	Django Debugging	22
3.7.2	Django System Check	22
4	Diskusjon	23
5	Konklusjon	24

Figurer

1.1	Iterativ Utviklings Diagram	4
2.1	Django Oversikt	8
3.1	Forsiden til Nettsiden	13
3.2	Liste over Biler	14
3.3	Instillinger for Søk	14
3.4	Individuelle Biler	15
3.5	Kontaktskjema	16
3.6	Ordre Bekreftelse	17
3.7	Forside i Administrasjons Side	18
3.8	Administrasjonsside - Oversikt over reserverasjoner	19
3.9	Ny reservasjon fra administrasjonsside	19
3.10	Tabellene i Database	20
3.11	Utleiepris Diagram	21

Kapittel 1

Innlending

1.1 Bakgrunn

Rusta Vrak Bilutleie er et lite firma basert i Førde og Jølster i Sogn og Fjordane. Målet til bedriften er å kunne tilby utleie av greie og fullt funksjonelle biler til en rimelig pris. Bedriften ble startet i 2013, og blir drevet av verkstedmesteren Stein Olav Erikstad. Rusta Vrak Biltuleie fungerer som et sideprosjekt, men har gode muligheter for vekst i fremtiden, og dersom et lite bilutleie firma skal være kompetitivt på markedet, er det viktig å kunne tilby digitale bestillingsmuligheter på lik linje som de større og mer etablerte. Derfor har Rusta Vrak Bilutleie uttrykt et ønske om å få en ny nettside knyttet til bedriften.

1.2 Problemdefinisjon

Rusta Vrak Bilutleie har i dag en nettside basert på tjenesten Blogspot [1], dette er en enkel informasjonsside som oppgir informasjon om bilene, priser og kontaktinformasjon. Deretter må kunden ta kontakt enten vha. telefon eller epost for å foreta selve bestilling. Da firmaet blir drevet av én person som et sideprosjekt, kan det være vanskelig å alltid være tilgjengelig, og samtidig ha en oversikt over hvilke biler som er ledige til enhver tid. Dette kan bli forbedret ved å gjøre nettsiden til firmaet mer interaktiv både for kunde og bedrift.

1.2.1 Oppgaven

Lag en nettside for bilutleie firmaet Rusta Vrak Bilutleie. Denne nettsiden skal kunne benyttes av kunder som et bestillingsverktøy. Dette inkl. mulighet til å se hvilke biler som er tilgjengelig for leie, når de individuelle biler er ledige, og foreta en reservasjon gjennom internett.

Nettsiden skal også kunne benyttes av firmaet for å få en oversikt over bilene som er klar for utleie, samt gi en oversikt over kunder som leier og som har leid bil tidligere. Det må være mulig å legge til, fjerne og redigere bilene som ligger for utleie forløpende.

1.2.2 Krav

I samarbeid med firmaet ble det utarbeidet en liste over krav som var viktig å få med i produktet. Denne kravlisten ble videre brukt som grunnmuren av planleggingsfasen av prosjektet.

Lett å bruke. Nettsiden skal være enkel å bruke, og skal helst være så intuitiv som mulig. Det betyr at den ikke skal for komplisert i bruk.

Skalering mellom skjermstørrelser. Gjøre nettsiden like god å bruke på smartphone som på PCer.

Selvstendig og Billig. Firmaet er lite, og derfor skal nettsiden kunne være så selvstendig som mulig. Dvs. arbeid med drift og opprettholdning skal være minimal. Utover dette burde hosting av nettsiden også være rimelig.

Håndere alle reservasjoner. Nettsiden skal både fungere som et bestillingsverktøy for kunder, og som et oversiktsverktøy for bedriften. Derfor skal kunder kunne benytte nettsiden for reservasjoner, og ansatte skal ha et administrasjons område hvor det er mulig å legge til nye reservasjoner manuelt.

Fordel å leie i lange perioder. Det skal være en fordel å leie over lengre perioder. Systemet må ta hensyn til dette og det skal implementeres en funksjon for å kalkulere prisen i forhold til lengden på leieperiode.

1.3 Litteratur Studie

Det aller meste av studie mot denne oppgaven har foregått på internett. Dette er hovedsakelig bruk av dokumentasjonen til de forskjellige biblioteker, rammeverker og plattformer. Utover dette har det blitt benyttet noen bøker som har fungert som oppslagsverk gjennom hele prosessen.

Bøker

- Code Complete 2. Edition (Steve McConnell)
- Programming Google App Engine with Python (Dan Sanderson)
- HTML and CSS Design and Build Websites (Jon Duckett)

1.4 Problemløsning

1.4.1 Prosjekt Plan

Det første som ble gjort mot denne oppgaven var å lage en plan over alle funksjoner og arbeidsoppgaver som må med i det endelige produktet, og legge disse inn i kategorier basert på viktighet. Kategoriene går fra 1 til 5, og planen var å gå stegvis gjennom denne listen.

1. Oppsett og Installasjon

- (a) **Lokalt Utviklingsmiljø** - Installere Python, oppsett av virtualenv, installere og integrere JetBrains PyCharm, installere rammeverket Django.
- (b) **Lokal Database** - Installasjon og opprette kobling mellom Django prosjekt og lokal MySQL database.
- (c) **Google Cloud Platform** - Installering og oppsett av kobling mellom lokalt prosjekt og Google Cloud Platform. Dette inkluderer både hosting og database.

2. Første Steg i Utviklingen

- (a) **Database Tabeller** - Planlegging og oppretting av database modeller og tabellene.
- (b) **Forside** - Enkel forside.
- (c) **Admin** - Mulighet til å legge til / slette biler fra opprettet database.
- (d) **Generere Liste over Biler fra Database** - Basert på biltype.
- (e) **Bil Reservasjon** - Mulighet å legge inn en reservasjon av bil som går mellom to datoer.

3. Videreutvikling

- (a) **Bilder** - Mulighet for å kunne vise bilde av bilene. Både som thumbnail og alternativt galleri.
- (b) **Kalender** - Gjør det mulig for kunde å ha visuell oversikt når den spesifikke bilen er ledig for utleie.
- (c) **Epost Bekreftelse** - Send en email til kunde som reserverer. Denne skal inneholde nyttig informasjon om reservasjonen.
- (d) **Reservasjon Håndtering** - Legg til funksjonalitet som stopper en reservasjon fra å krasje med annen allerede eksisterende reservasjon.
- (e) **Søkefunksjon** - Finne ledige biler på dato.

4. Ekstra Funksjonalitet

- (a) **Pris Funksjon** - Det skal være en fordel å leie over lengre perioder. Derfor må en funksjon kunne dele ut riktig mengde rabatt over hvor mange dager som er i leieperioden. Videre skal denne kunne runde til nærmeste 5 for å slippe unødvendig småpenger.
- (b) **PDF** - Mulighet å laste ned PDF med bestillingsdetaljer.
- (c) **Filtrering av Bil Listen** - F.eks. kun biler som har automat.
- (d) **Konfigurering av Admin** - Søk bil på skiltnr. Oversikt over alle reservasjoner og kunde informasjon.

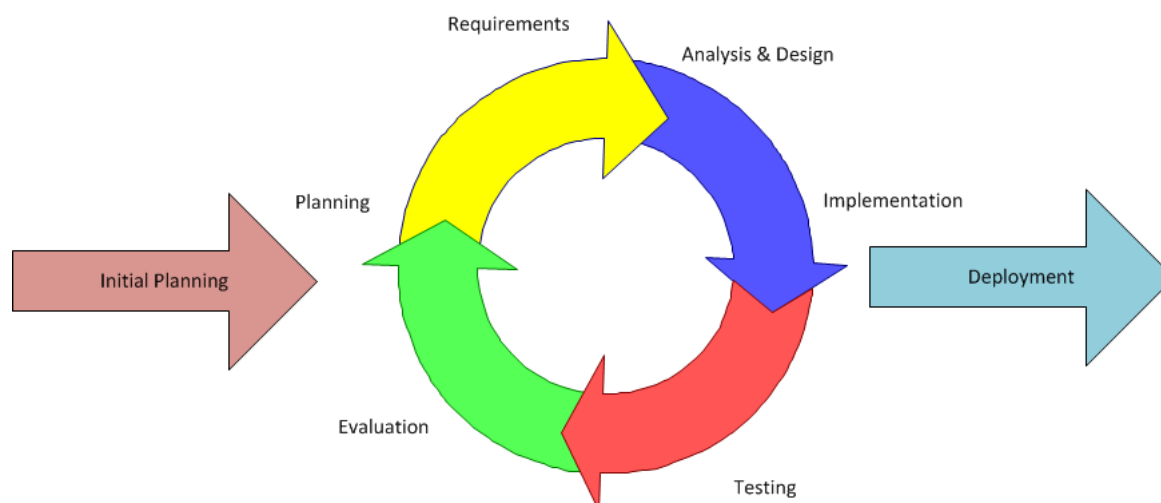
5. Dersom tid

- (a) **Google Calendar** - Legg til informasjon i en Google Calendar for enkel oversikt over når biler hentes og leveres.

1.5 Arbeidsmetode

Utviklingen av dette prosjektet har benyttet en iterativ og inkrementell arbeidsmetode. Denne metoden går ut på å gjennomføre utviklingen stegvis, man begynner utviklingen med et skjelett, som man vil videre legge til litt funksjonalitet, og deretter komme tilbake og legge til mer på toppen av dette igjen. Ved å benytte en slik prosess, blir det mulig å utvikle flere områder på nettsiden samtidig uten at det nødvendigvis krasjer med hverandre i prosessen.

En iterativ metode gjør det mulig å splitte et større prosjekt i mindre, mer håndterlige biter. Planlegging, design, utvikling av kode og testing blir gjennomført i gjentatte sykluser [2]. Dermed kan man følge figur 1.1 for prosessen gjennom hele utviklingen.



FIGUR 1.1: Oversikt over trinnene per syklus i iterativ utvikling. Figur hentet fra [3].

1.6 Rapportens Struktur

Denne rapporten har blitt delt inn i 5 kapitler.

Første kapittel introduserer bakgrunn og problemet som skal løses. Her finner man en kravliste utarbeidet i samarbeid med bedriften, og en planlagt fremgangsmåte for å kunne møte disse kravene. Deretter finner man kapitlet som beskriver de valgte teknologiene, og presenterer en liten oversikt over disse. Etter dette kommer kapitlet om selve løsningen, og her vil det endelige resultatet av prosjektet komme frem. Man får her et overblikk over hvordan nettsiden fungerer, både fra et brukerperspektiv, og et administrerende perspektiv. Så finner man et kapittel som diskuterer oppgaven, her drøftes det litt om hvorvidt målet har blitt møtt, og litt om fremtidig arbeid. Til slutt finner man konklusjonen for prosjektet.

Alle relevante vedlegg som vises til i rapporten finner man på de siste sidene.

Kapittel 2

Oversikt over valgte Teknologier

2.1 Publisering

Ett av kravene til prosjektet var at nettsiden skal være så selvstendig som mulig. Det skal være minimalt med nedetid, serveren må kunne oppdateres uten manuelt inngrep og backup av database må foregå automatisk. Samtidig som å oppfylle dette burde selve hostingen være så billig som mulig. Derfor ble det noe moderne konseptet av Platform as a Service benyttet.

2.1.1 Platform as a Service

En Platform as a Service (PaaS) er et utviklings og distribusjons miljø basert i en skytjeneste [4]. En PaaS leverandør har som ansvar å levere og opprettholde en utviklings og distribusjons plattform for sine kunder. De har videre ansvaret på områdene rundt konfigurering, oppsett av servere og sikkerhet på server siden. Det er populært å referere til PaaS leverandører som IT-avdelingen til produktet, som gjør at utvikleren kan fokusere mer på selve utviklingen enn oppsett av infrastrukturen som ligger bak [5, s. 10].

2.1.2 Fordeler ved PaaS

Spare Tid

Utviklingen foregår direkte mot arkitekturen til den valgte skytjenesten, derfor vil man unngå forarbeidet rundt oppsett og installasjon av servere. Dersom en server hos PaaS leverandøren skulle gå offline, vil dette bli ordnet fortløpende av leverandøren, uten at man som kunde trenger å gripe inn [5, s. 9].

Billig

Samtidig som å levere mye i pakken, kan publisering vha. PaaS være billig. Dette kommer av at de aller fleste leverandørene bruker dynamisk ressursallokering. Det betyr at ressursene allokert til et prosjekt vil følge samme kurve som behovet. Ved lite eller ingen trafikk vil et prosjekt gå i dvalemodus og bli der helt til en forespørsel blir sendt til serveren. Man betaler kun for de ressurser som blir benyttet.

2.1.3 Valg av PaaS Leverandør

PaaS er fortsatt et moderne konsept, og man har mange muligheter ved valg av leverandør. Det er en enorm jobb å foreta en sammenlikning over de relevante valgene man har, derfor har jeg valgt å gå ut ifra et studie som jeg foretok under et bachelor prosjekt våren 2016 [5, s. 18]. Her ble det utført en sammenlikning av 3 av de største PaaS leverandørene: Google Cloud Platform, Microsoft Azure og Amazon Web Services.

Resultat

Alle de tre nevnte leverandørene har et bredt utvalgt av muligheter på sine løsninger. Alle tilbyr et løfte av server tilgjengelighet på 99,95% av tiden. De har gode tilkoblingsmuligheter i Europa og prisen er basert på tid og ressurser.

Men det er spesielt ett område de har unike løfter; Gratis Kvoter.

Microsoft Azure leverer kun gratis kvoter i form av trial account. Her får man ressurser for 200\$ som man kan benytte som man selv bestemmer, i løpet av de første 30 dager etter Trial Account blir opprettet.

Amazon Web Services kommer utstyrt med 1 år 'free tier', som inneholder 5GB lagringsplass, 20 000 GET requests og 2 000 Put Requests.

Google Cloud Platform derimot tilbyr daglige gratis kvoter. Dette inkluderer bl.a. 28 Frontend Instance timer¹, 1GB plass for lagring av logger, 1GB data inn og ut.

Ved å bruke Google Cloud Platforms gratis kvoter, vil man kunne ha hosting av mindre trafikkerte nettsider bortimot gratis, og man behøver kun betale for databasen.

Valg: Google Cloud Platform

¹Google Cloud Platforms 'frontend instance' blir opprettet ved at en applikasjon får en forespørsel, og denne instance vil være tilgjengelig for bruk de neste 15 minutter. Alle nye forespørsler i disse neste 15 minutter vil benytte samme frontend instance

2.2 Back-end

Med dette prosjektet ville jeg videreutvikle min kompetanse innen bruken av web-applikasjons rammeverket Django, basert på programmeringsspråket Python.

2.2.1 Python

Python er et skriptingspråk utviklet av nederlandske Guido van Rossum. Den tidligste versjonen kom frem i 1996 og det har stadig vært i utvikling siden. Python har et formål om å gjøre kode leselig og gjenbrukbar, og har et mindre fokus på ren hastighet. Utover dette har Python et fokus på å gjøre utviklingen, og debuggings fasene av prosjekter så raskt som mulig [5, s. 11].

2.2.2 Django

Django er et web-applikasjons rammeverk som kommer levert med en 'batterier inkludert' filosofi. Dette betyr at vanlig funksjonalitet som ofte blir benyttet i en web sammenheng blir levert med i grunnpakken til Django. Dette inkluderer bruker-system, URL routing, template system, administrasjons system og database object-relational mapper (ORM) [6].

På figur 2.1 kan man se en oversikt over hvordan Django håndterer et HTML request.

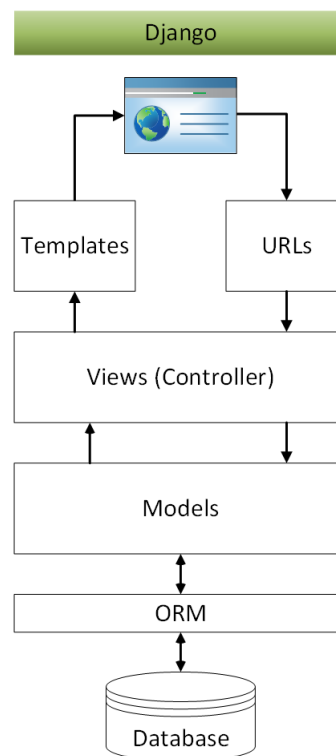
Først vil den tyde hvilke URL som kommer inn. Django inneholder et register over tilgjengelige URL som kan benyttes vha. regular expressions. Den vil sjekke dette registeret og finne hvilken funksjon (view) som tilhører den innkommende URL.

Den korresponderende funksjon i view vil nå håndtere data fra HTML requestet, dette innebærer å sjekke om det er et POST eller GET request, og hente ut eventuell data med henyn til dette.

Data vil bli hentet vha. Djangos Modeller, som er et objekt orientert metode å fremstille den tilkoblede databasen. Når det er klart hvilke data som skal hentes, vil Djangos ORM konvertere Djangos Database API til SQL Query. Den data som resulterer fra spørringen vil bli returnert tilbake til view.

Til slutt vil view samle data og generere den tilhørende template (HTML), som videre sendes tilbake til browseren.

FIGUR 2.1: Stegene Django kjører gjennom ved et HTML request.



2.3 Front-end

2.3.1 Bootstrap

Bootstrap ble først laget av en designer og utvikler hos Twitter i 2010, og har fått stort følge siden. Det er i dag ett av de største og mest populære front-end rammeverkene i verden. Da det er så mye brukt, vil det da være et større følge aktive brukere på nettsider som stackoverflow.com, som videre gjør det mer attraktivt å bruke.

Skalerbart

Bootstrap kommer utstyrt med et Grid system. Dette systemet er basert på å dele opp skjermen inn seksjoner, basert på skjermstørrelsen. Dette gir muligheten av å kunne spesifisere hvordan et element skal oppføre seg i forhold til skjermstørrelsen, dette resulterer i en nettside som fungerer like godt på smartphones, tablets og PCer.

Komponenter

Rammeverket kommer med en hel rekke komponenter som kan benyttes. Dette gjelder bl.a. en rekke valg innen navigasjonsmenyer, input bokser, knapper etc. Disse komponentene er videre fullt mulig å redigere vha. CSS².

2.3.2 JavaScript Biblioteker

JQuery. JQuery har blitt brukt som hovedbibliotek bak all JavaScript mot nettsiden. Det er et såkalt «write less, do more» bibliotek som kommer utstyrt mange gode widgets (deriblant Kalender). Dette biblioteket tar mange vanlige oppgaver om vanligvis krever mange linjer JavaScript, og pakker disse inn i metoder man kan benytte i en enkelt linje kode i prosjektet [7].

Fullcalendar. Fullcalendar har blitt benyttet for å kunne fremstille når bilene er ledig for utleie. Dette er et åpent kildekode bibliotek som kan brukes for å fremstille 'events' visuelt i form av en kalender.

²Cascading Style Sheets. Håndterer hvordan HTML elementer vises på skjermen. Kontrollerer layout på en ryddig metode.

2.4 Verktøy

GitHub ble benyttet som versjons kontroll underveis i utviklingen. For hver nye feature som skulle introduseres, ble dette utviklet på en egen branch. Når utviklingen av en ny feature fungerte som planlagt, skulle den testes grundig før den ble pushet over på Master branch.

Google App Engine SDK. Google Cloud Platform kommer utstyrt med en SDK når man skal arbeide mot Google App Engine. Denne SDK inneholder et lokalt utviklingsmiljø som kjører på samme infrastruktur som det benyttet ved hostingen. Man får også publisering og administrasjons muligheter direkte vha. enkel GUI³. Denne har hovedsakelig blitt brukt som et test område underveis i utviklingen, samt. Publisering av nyere versjoner etter hvert som nettsiden har blitt oppdatert.

JetBrains PyCharm har blitt brukt som IDE⁴. Denne inneholder en bred rekke hjelpemidler for å gjøre utviklingsprosessen smidigere. Den støtter både utvikling av Django prosjekter og html/css/javascript. I tillegg har den et godt system for autocomplete av kode, og vil generere hint for forbedringer av bl.a. navn på funksjoner, indentering og sørger for at all kode benytter samme navn konvensjon.

LateX. Under rapportskrivningen har typesettingssystemet LateX blitt brukt får å generere dokumentet. Dette systemet hjelper med å skape en god utforming av hele dokumentet, som gir bedre muligheter for å fokusere på selve skrivingen. LateX gir et såkalt WYSIWYM (What You See Is What You Mean), som beskriver en form for skriving hvor det resulterende dokument bedre representerer den faktiske informasjonen man arbeider med.

³Grafisk brukergrensesnitt. Forenkler prosessen ved å jobbe med vinduer og grafiske fremstillinger.

⁴Integrated Development Environment. Utviklerverktoy, inneholder ofte en editor, compiler og debugger. [8]

Kapittel 3

Løsning

3.1 Krav

For å se kravlisten av prosjektet viser jeg til kapittel [1.2.2](#) for å se hvilke krav som ble utarbeidet i samarbeid med bedriften, og kapittel [1.4.1](#) for å se den mer detaljerte liste over alle featurelist fremgangsmåten som ble fulgt.

Ut av kravene ble et workflow utarbeidet, dette omhandler bestillingsprosessen fra kundens perspektiv.

3.1.1 Kundens Workflow

Steg 1

Forside

Man starter med å se på forsiden av nettsiden. Denne skal inneholde generell informasjon om bedriften, litt om hvordan leien fungerer, priser, vilkår osv. Utover dette skal det her være 2 muligheter for brukeren å gå videre i reservasjonsprosessen:

Søke på Dato Her skal systemet finne ut hvilke biler som er ledige i en gitt leieperiode, og returnere disse.

Se Biler Man kan se alle bilene innenfor en gitt kategori, eller alle biler i systemet.

Steg 2

Liste over biler

Uavhengig av fremgangsmåten brukeren har valgt, vil man nå presenteres med en liste over biler. Listen inneholder generell informasjon om hver av bilene, samt bilde og døgnpris i leien. Utifra denne listen velger brukeren å gå videre på en av bilene.

Steg 3

Spesifikk bil

Nå befinner brukeren seg på siden til den bestemte bilen. Her skal informasjon om bilen og eventuelle bilder presenteres. Dersom brukeren ikke søkte på dato i Steg 1, må det her sjekkes når bilen er ledig for utleie. Dette skal visualiseres i form av en kalender.

Videre skal brukeren fylle inn dato i leieperioden (Dersom man søker på Dato i Steg 1, skal dette bli fylt inn automatisk), så gå videre.

Steg 4

Kontakt Informasjon

Det siste steget av reservasjonsprosessen. Her skal brukeren fylle inn kontakt informasjon, og kunne se informasjon om reservasjonen og den valgte bil.

Når informasjonen er fylt inn trykker brukeren på Fullfør Bestilling, som skal videreføre til en bekreftelses side om hele bestillingen. Nå skal også en epost bli sendt til brukeren som inneholder informasjon om bestillingen.

Steg 5

Last ned informasjon som PDF (Valgfritt)

Brukeren skal ha mulighet til å laste ned ordre bekreftelse som PDF.

3.2 Designspesifikasjoner

Det ble i hovedsak satt 2 krav til design for nettsiden. Den skal skalere godt mellom skjermstørrelser, og den skal være så oversiktlig og lett å bruke som mulig.

Skalering. For å oppnå god skalering mellom skjermstørrelser ble rammeverket Bootstrap benyttet. Ved å bruke Bootstrap vil man jobbe med Grid systemet som lar deg konfigurere hvordan den skal takle plassering av elementer på små og store skjermer.

Oversiktlig. Det er litt vanskeligere å tilfredsstille kravet om at det skal være oversiktlig da dette er noe mer subjektivt, og meninger vil variere fra bruker til bruker. Men det har blitt utarbeidet en løsning med så lite støy som mulig. Dvs. alt av innhold på nettsiden skulle ha en grunn til å være der.

3.3 Resultat

3.3.1 Forsiden

På forsiden av nettsiden (Figur 3.1) finner man de to elementene for å finne frem til en bil for utleie. Her har man valget av å enten søke på dato i leieperioden, eller man kan få en liste over biler. Ved å søke på dato har man også valget om å spesifisere hvilke biltyper som skal være med i søket.

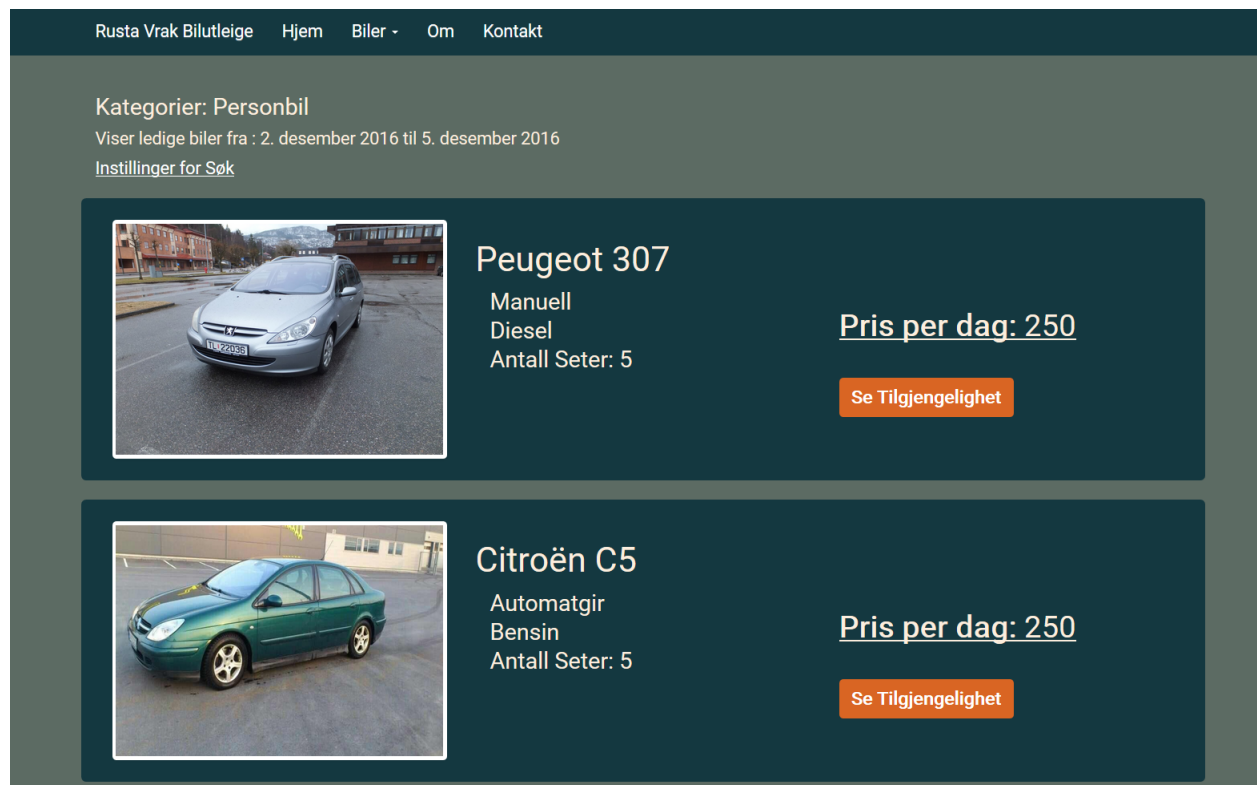
Utover dette har vi ytterligere informasjon i form av seksjoner, her finner man vilkår, generell informasjon, kontakt informasjon og om oss.

The screenshot shows the homepage of 'Rusta Vrak Bilutleie'. At the top is a dark navigation bar with links: 'Rusta Vrak Bilutleie', 'Hjem', 'Biler', 'Om', and 'Kontakt'. Below this is a header section with the title 'Rusta Vrak Bilutleie' and the text 'Du finn oss i Førde og Jølster'. The main content area has a dark background and features a search section titled 'Søk etter ledig bil'. This section includes two date pickers labeled 'Ønsket Hentes' and 'Ønsket Levering', a 'Biltype' section with radio buttons for 'Personbil', 'Varebil', and 'Kombinertbil', and an orange 'Søk' button. Below the search section is a section titled 'Se alle biler' which displays three car images in a grid. The first image is a silver hatchback labeled 'Personbil'. The second image is a silver van labeled 'Varebil'. The third image is a dark grey station wagon labeled 'Kombinertbil'.

FIGUR 3.1: Forsiden til nettsiden. Her kan man se de to valgene kunden kan benytte for å finne frem til bil. Enten vha. Søk etter ledig bil i form av leieperiode, eller liste over alle biler basert på biltype.

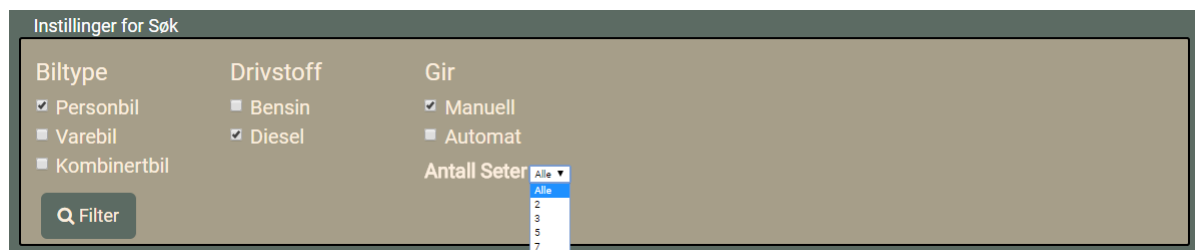
3.3.2 Liste over Biler

Denne siden viser alle biler som ble søkt fra forsiden, se figur 3.2. Dersom et det ble gjort et søk på dato, vil alle bilene i listen være ledig i den gitte periode. Hver bil har et hovedbilde, som kan forstørres ved å klikke på bildet, og det blir oppgitt informasjon om hver bil som type girkasse, drivstoff, antall seter, diverse tilbehør og leiepris.



FIGUR 3.2: En liste over alle biler fra det oppgitte søk. Hver bil viser generell informasjon og har en knapp for å gå videre til den spesifikke bilen.

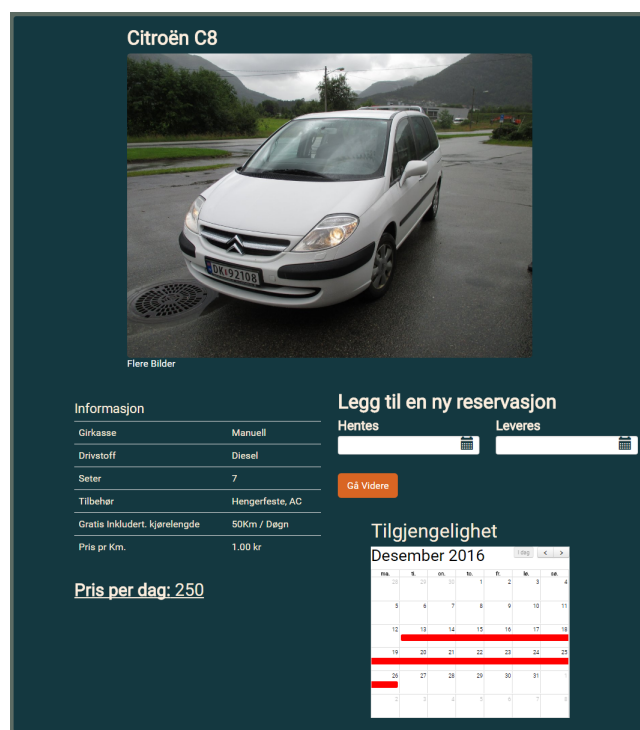
Denne listen kan filtreres videre ved å trykke på «Innstillinger for Søk» (Figur 3.3). Her kan man velge å spesifisere ytterligere hvilke egenskaper en bil skal ha, dette inkl. valg av biltyper, type girkasse, drivstofftype og antall seter. Ved å trykke på filter knappen vil serveren generere en ny liste som oppfyller de oppgitte kravene i instillingene.



FIGUR 3.3: En filtreringsmeny for å kunne spesifisere ytterligere hvilke biler som skal bli oppgitt i listen.

3.3.3 Individuelle Biler

Siden som fremstiller de individuelle bilene er identisk for alle biler, og viser bilde (og eventuelt galleri dersom flere bilder er oppgitt), informasjon om bilen, input områder for å foreta en reservasjon og en kalender som fremstiller når bilen er ledig for utleie.



FIGUR 3.4: Siden til de individuelle bilene. Her vil man få oppgitt generell informasjon om bilen, bilder, funksjonalitet for å reservere bilen og en kalender som visuelt fremstiller når bilen er ledig for utleie.

Når det opprettes en ny reservasjon, vil systemet først sjekke om den oppgitte perioden er en gyldig, de sjekker som blir gjort er som følger:

1. Hente dato er satt før leveringsdato.
2. Leieperioden er større enn 0 og mindre enn 31 (maks leieperiode man kan opprette på egenhånd er 30 dager).
3. Leieperioden ikke satt tilbake i tid.
4. Leieperioden for den spesifikke bilen er ledig (Se vedlegg D for koden bak denne prosessen).

Dersom alle kravene er oppfylt, vil brukeren omdirigeres videre til neste steg i reservasjons prosessen.

3.3.4 Kontaktskjema

Etter brukeren har valgt en bil, og funnet en ledig periode for utleie må det til slutt fylles ut et kontaktskjema (Figur 3.5). Her må det fylles ut enkel informasjon om kunden: epost, fornavn, etternavn og telefonnummer. Man har også et valgfritt tekstområde for eventuell ekstra informasjon man ønsker å oppgi, eller om det skulle være noen spesielle omstendigheter ved leien.

I tillegg får man oppgitt informasjon om valgt leieperiode, pris, og den valgte bilen.

Kontakt Informasjon

For å fullføre reservasjonen, vennligst fyll ut det følgende skjema. Husk å dobbelsjekk informasjon av reservasjonen før du går videre.

Epost adresse

Fornavn

Etternavn

Telefonnummer

Annen Informasjon

Fullfør bestilling

Informasjon

Reservasjonen

Hente Dato	29. desember 2016
Leverers Dato	7. januar 2017
Antall Dager	9
Total pris	1855

Citroën C8

Girkasse	Manuell
Drivstoff	Diesel
Seter	7
Tilbehør	Hengerfeste, AC
Gratis inkl. kjørelengde	450km
Pris pr Km. (Dersom gratis inkl. lengde overstiges)	1.00 kr

FIGUR 3.5: Det siste steget i bestillingsprosessen. Her må man fylle inn kontaktskjemaet. Man har også her muligheten til å se alle detaljer omgående reservasjonen.


3.3.5 Bekreftelse

Når bestillingen er vellykket og er blitt lagret til databasen, blir brukeren videresendt til en bekref-
telses side (Figur 3.6). Her finner man all informasjon om reservasjonen, og man har et valg om å
laste ned en PDF som inneholder samme informasjon (Vedlegg A). Utover dette blir det også sendt
en epost til brukeren, som også inneholder all informasjon om reservasjonen.

Reservasjon er registrert

Informasjon om bestillingen

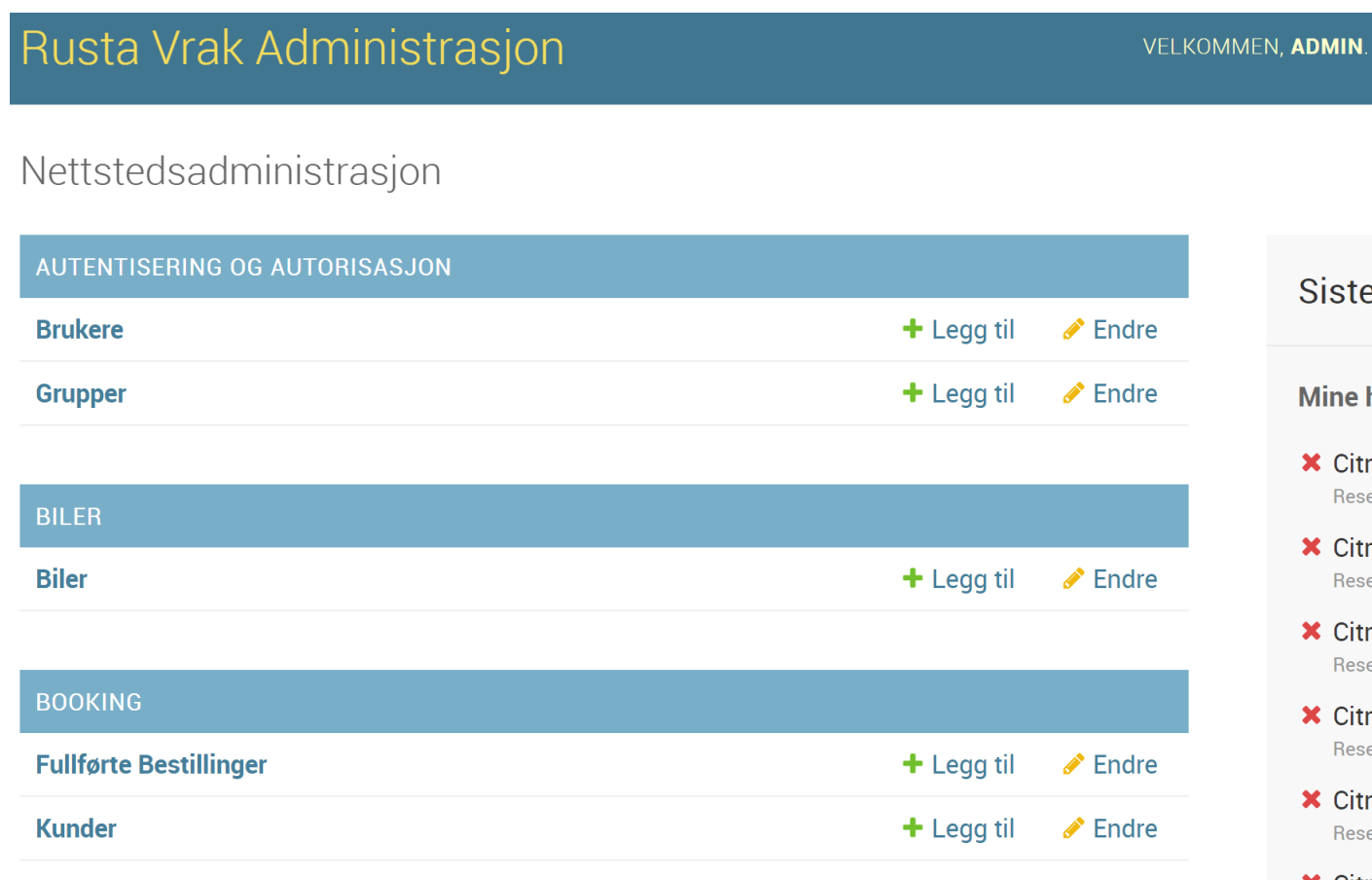
Reservasjonen	
Reservasjon Nr.	41
Bil	Citroën C8, 7 seter, Diesel
Hentes Dato	29. desember 2016
Levers Dato	7. januar 2017
Gratis inkl. kjørelengde	450 Km
Pris	1855
Bestiller	
Navn	Martin Engen
Epost	enepost@email.com
Telefon	12332112

 Last ned kvittering på PDF

FIGUR 3.6: Det siste steget i bestillingsprosessen. Her må man fylle inn kontaktskjemaet. Man har også her muligheten til å se alle detaljer omgående reservasjonen.

3.4 Administrasjon

Administrasjonssiden skal hovedsakelig benyttes av bedriften for å kunne holde styr over sine kunder, biler og bestillinger. På figur 3.7 kan man se et skjermbilde av forsiden av admin siden. Denne admin siden blir generert av Django og reflekterer de konfigurasjonene som blir gjort i koden. Her velges bl.a. hvilke tabeller som er interessante å ha i admin siden, hvilke elementer som skal ligge i listen, hva som er relevant i søke funksjoner osv. Her har man også muligheten til å opprette nye admin kontoer og velge hvilke rettigheter de skal kunne ha på nettsiden.



FIGUR 3.7: Hvordan administrasjonssiden ser ut. Denne har blitt generert og konfigurert med web-applikasjons rammeverket Django.

3.4.1 Reservasjoner i Administrasjonsside

På figur 3.8 kan man se listen over alle ferdig reservasjoner. Her kan man se informasjon om kunden, hvilken bil som er reservert, bestillings, hente og leveringsdato, og den totale prisen for reservasjonen. For å kunne filtrere ut spesifikke reservasjoner kan man bruke søke funksjonen. Denne godtar søk på skiltnummer, bilmerke og modell, kundens etternavn og epost. Søket blir gjort å en såkalt «Contains» metode, som betyr at dersom man f.eks. gjør et søk på 'TV', vil reservasjonen med en bil med skiltnummer 'TV65282' bli med i resultatet.

Velg Reservasjon du ønsker å endre LEGG TIL NY RESERVASJON +

Q

Handling: 0 av 8 valgt

<input type="checkbox"/>	BIL	BESTILLINGS DATO	FRA DATO	TIL DATO	KUNDE	KUNDE EPOST	KUNDE TLF	PRIS
<input type="checkbox"/>	CITROËN JUMPY	26. november 2016	26. november 2016	22. desember 2016	Kimi Raikkonen	user05@email.com	22225555	2485
<input type="checkbox"/>	CITROËN JUMPY	26. november 2016	12. desember 2016	29. desember 2016	Ayrton Senna	user04@gmail.com	12345678	2385
<input type="checkbox"/>	OPEL ZAFIRA	26. november 2016	1. desember 2016	31. desember 2016	Fernando Alonso	user03@gmail.com	87654321	2500
<input type="checkbox"/>	WOLKSVAGEN SHARAN	26. november 2016	6. desember 2016	9. desember 2016	Nico Rosberg	user02@email.com	11112222	750
<input type="checkbox"/>	WOLKSVAGEN SHARAN	26. november 2016	1. desember 2016	4. desember 2016	Frank Sinatra	user01@email.com	55556666	750
<input type="checkbox"/>	PEUGEOT 307	25. november 2016	6. desember 2016	9. desember 2016	Martin Engen	Nitrax92@gmail.com	46966993	750
<input type="checkbox"/>	CITROËN JUMPY	19. november 2016	22. november 2016	26. november 2016	Martin Engen	Martin.Engen@outlook.com	5646544	1000
<input type="checkbox"/>	PEUGEOT 406	19. november 2016	20. november 2016	21. november 2016	sondre engen	email@adr.com	48883881	250

8 Fullførte Bestillinger

FIGUR 3.8: Liste over alle reservasjoner registrert på nettsiden.

Legge til ny reservasjon

Dersom en kunde ikke ønsker å reservere gjennom nettsiden, kan ansatte også gjøre dette manuelt gjennom administrasjonssiden. Dette er en enkel prosedyre, man velger å legge til en ny reservasjon, fyller ut vinduet (Figur 3.9) og lagrer dette.

Legg til ny Reservasjon

Car:

Customer:

Ekstra informasjon:

1: Pending, 2: Approved, 3: Declined:

Hente Dato:
Merk: Du er 1 time foran server-tid.

Leverings Dato:
Merk: Du er 1 time foran server-tid.

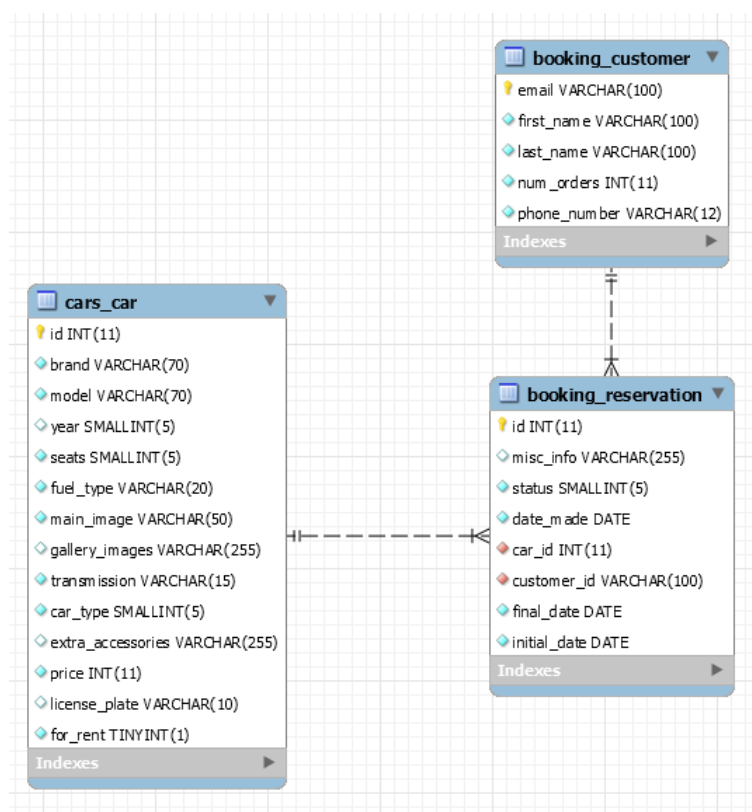
FIGUR 3.9: Hvordan man legger inn en ny reservasjon direkte fra administrasjonssiden.

3.5 Database

Dette prosjektet benytter Django, og dermed vil dette rammeverket ta hånd om mye av arbeidet mot databasen. Django kommer utstyrt med en ORM (Object Relational Mapper), som håndterer overgangen fra Python klasser til MySQL tabeller.

Figur 3.10 viser de tabellene som er direkte knyttet til biler, kunder og reservasjoner på nettsiden. Her har det blitt utviklet et enkelt system, da det ikke var behov for noe mer utbredt. En mulig forbedring ville vært å splitte Car tabellen opp i flere små tabeller, men per i dag trenger man all informasjon om bilen hver gang den hentes fra databasen, så en refaktorering har ikke vært en prioritet.

For å se et komplett bilde av alle tabeller i databasen viser jeg til vedlegg B.



FIGUR 3.10: Tabellene i databasen direkte knyttet til bilene, kundene og reservasjonene.

3.6 Priser og Rabatt

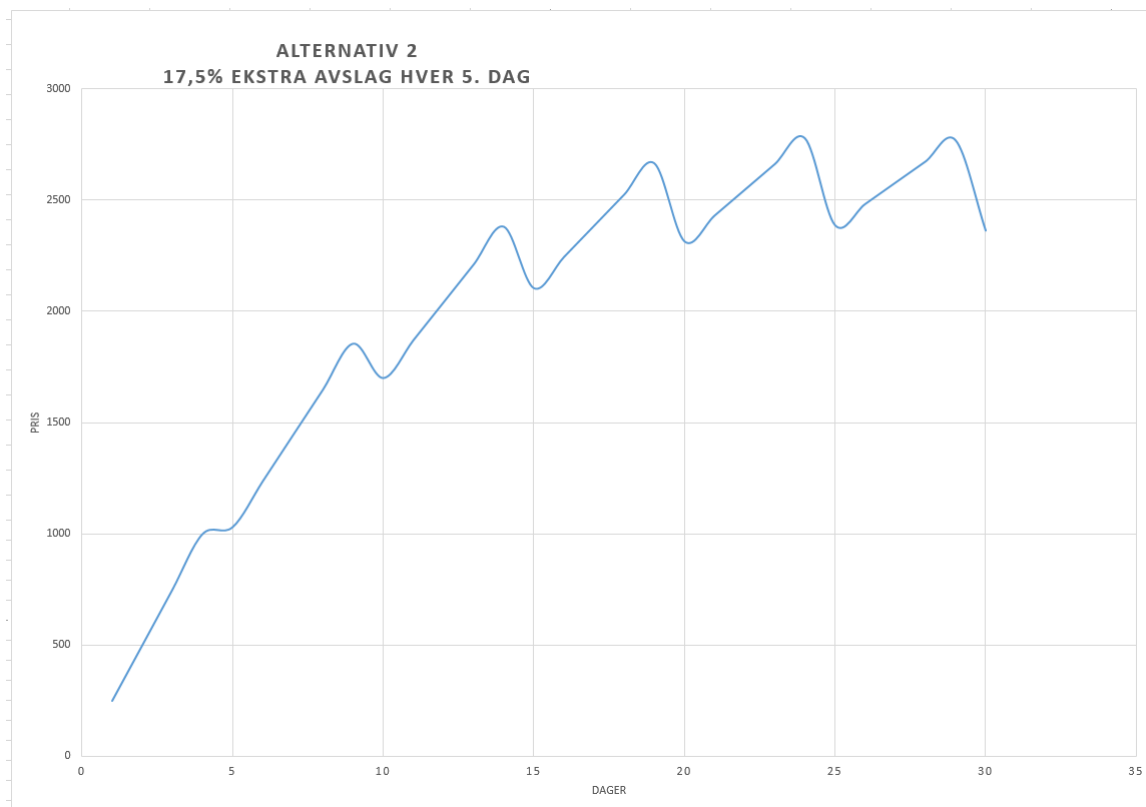
Rusta Vrak Bilutleige ønsker at når en kunde først skal leie en bil, skal det være så lenge som mulig for å slippe å hente og levere biler mer enn nødvendig. Derfor vil bedriften gi kunder gode rabatter som øker i forhold til lengde på leieperiode. Leieprisen av de aller fleste bilene ligger på 250kr pr. døgn, og bedriften tilbyr utleie på disse bilene i 30 dager for kun 2500kr. Dette tilsvarer en prisreduksjon på 66.6%. For å kunne implementere et liknende system på nettsiden ble det laget en rekke løsningsforslag i Excel, dette finner man vedlagt i vedlegg C.

Valgte Løsning

Den valgte løsningen går ut på å gi kunden 17.5% ekstra avslag hver 5. dag i leieperioden. Se tabell 3.1 for en oversikt over hvordan pris avslaget stiger. Figur 3.11 viser hvordan prisantydningen blir på biler som har en døgnpris på 250kr. Dette vil kunne bidra til at en kunde leier litt ekstra kun for å få ekstra avslag på prisen. Koden bak den implementerte priskalkuatoren finner man i vedlegg D.

TABELL 3.1: Rabatt man oppnår ved leieperioder

Antall Dager	0-4	5-9	10-14	15-19	20-24	25-29	30
Rabatt	0%	17.5%	31.9375%	43.849%	53.675%	61.782%	66.66%



FIGUR 3.11: Oversikt over rabatt som tilføres i forhold til hvor lenge en bil blir utleid.

3.7 Testing og Validering

Det har blitt benyttet en iterativ arbeidsmetode som beskrevet tidligere i kapittel 1.5. Denne metoden tilsier at testing skal og må foregå underveis i utviklingen for hver syklus. Dette har blitt gjennomført vha. Pythons logging bibliotek, Djangos debuggings modus og Djangos system check.

3.7.1 Django Debugging

En av hovedegenskapene til debuggingsmodus er at man vil få detaljerte error sider. Dersom en applikasjon kjører seg fast vil man få oppgitt en side med detaljert traceback, data om miljøet og eventuelle SQL query som blir utført i sammenheng med krasjet [9].

Dersom man slår av debuggingsmodus vil applikasjonen redirigere til en standard error side (404, 400 etc.) ved feil.

3.7.2 Django System Check

Kommandoen ‘python manage.py check’ kjører en bred sjekk av hele systemet [10]. Denne vil gå grundig gjennom følgende deler av prosjektet:

Modeller. Går gjennom alle database modeller og sjekker om modeller, og felter er som de skal være, og reflekterer tabellene i databasen.

Admin. Ser på alle egendefinert Admin funksjoner og oversiktstabeller. Først og fremst sjekker at alle tabeller inneholder virkelig data, og at de grunnleggende admin funksjoner (Legge til, slette, redigere) er funksjonelle.

Kompatibilitet. Denne vil se at koden benytter Djangos anbefalte funksjoner, og vil gi advarsel dersom en nyere versjon av Django har forandringer, og hva som bør endres for å gjøre koden kompatibel med nyere django versjoner.

Sikkerhet. Det blir utført en noe begrenset, men en såkalt ‘low-hanging-fruit’ sjekklister. Dette er bl.a. en sjekk om at alle POST requests inneholder en Django spesifikk csrf token (Cross-site request forgery).

Templates. En gjennomgang av alle HTML som finnes til Templates mappene. Her blir det gjort en sjekk for å se om alle Tags ¹ blir benyttet på korrekt måte.

¹Man kan utføre logikk i HTML i form av Tags. Disse kan f.eks. bruke variable, enkle IF setninger og løkker. Tags blir evaluert når den blir generert av serveren før den sendes til brukeren i form av ren HTML [11].

Kapittel 4

Diskusjon

Målet

Hovedmålet til dette prosjektet var å utvikle en nettside for bedriften Rusta Vrak Bilutleige. Denne skulle fungerer som et bestillingsverktøy for kunder, og som et oversiktsverktøy for bedriften. Etter min mening har dette målet blitt møtt, og prosjektet er klart til å bli tatt i bruk. Prosjektet skulle også publiseres, og dette har blitt gjennomført vha. Google App Engine. Jeg hadde tidligere erfaring med publisering på denne tjenesten, så prosessen her var velkjent. Prosjektet er å finne på følgende URL: <https://rusta-vrak.appspot.com/>

Fremtidig Arbeid

Domene. Rusta Vrak har et eget domene, men det ligger fortsatt tilkoblet den gamle nettsiden. Dette må konfigureres for å kunne kjøre direkte til den nye nettsiden.

Admin Reservasjoner. Det er fortsatt et problem som gjenstår å bli løst. I dag kan en admin registrere reservasjoner som overlapper med allerede eksisterende reservasjoner, uansett bil og kunde. Dette kommer av at admin systemet ikke er direkte knyttet til samme system brukere benytter for å legge inn nye reservasjoner, og dette burde bli tatt hånd om relativt snart for å unngå uheldige dobbelreservasjoner. Systemet fungerer godt når det legges inn nye reservasjoner fra perspektiv fra kunde, så derfor må ansatte som skal legge inn en ny reservasjon god kontroll på hvilke dato og biler som er ledige i en gitt periode.

Kapittel 5

Konklusjon

Formålet med denne oppgaven var å utvikle en ny nettside for bedriften Rusta Vrak Bilutleige. Denne nettsiden skulle fungere som en alternativ bestillingsmetode for kunder, og hadde noen krav om at den skulle skalere godt over forskjellige skjermstørrelser, være lett og oversiktlig i bruk. Bedriften skulle også kunne benytte denne nettsiden som et oversiktsverktøy over sine biler, kunder og reserverasjoner. Etter endt oppgaveperiode fyller nettsiden alle disse kravene og behovene, og produktet er klar til bruk.

Bibliografi

- [1] Rusta vrak bilutleige. URL <http://rustavrak.blogspot.no/>. [Online; besøkt 02-Desember-2016].
- [2] URL <https://www.inflectra.com/GraphicsViewer.aspx?url=Methodologies/Waterfall.xml&name=wordml://03000005.png>. [Online; besøkt 14-November-2016].
- [3] URL <https://www.inflectra.com/Methodologies/Waterfall.aspx>. [Online; besøkt 23-November-2016].
- [4] "what is paas?". URL <https://azure.microsoft.com/nb-no/overview/what-is-paas/>. [Online; besøkt 23-November-2016].
- [5] Martin Engen. Bachelor rapport vår 2016 - dfs treningsdagbok. Technical report, Universitetet I Agder, 2016.
- [6] full stack python, django", . URL <https://www.fullstackpython.com/django.html>. [Online; besøkt 23-November-2016].
- [7] jquery introduction. URL http://www.w3schools.com/jquery/jquery_intro.asp. [Online; besøkt 02-Desember-2016].
- [8] Margaret Rouse. integrated development environment (ide). URL <http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment>. [Online; besøkt 02-Desember-2016].
- [9] Debug, . URL <https://docs.djangoproject.com/en/1.10/ref/settings/#debug>. [Online; besøkt 02-Desember-2016].
- [10] system check framework". URL <https://docs.djangoproject.com/en/1.10/ref/checks/>. [Online; besøkt 22-November-2016].
- [11] the django template language", . URL <https://docs.djangoproject.com/en/1.9/topics/templates/#the-django-template-language>. [Online; besøkt 22-November-2016].

Vedlegg

Rusta Vrak Bilutleige Ordre Bekreftelse

Reservasjonen

Reservasjon Nr	39
Hente Dato:	01.02.2017
Leveres Dato:	03.02.2017

Bilen

Citroën C8	
Drivstoff	Diesel
Antall Seter	7
Girkasse	Manuell

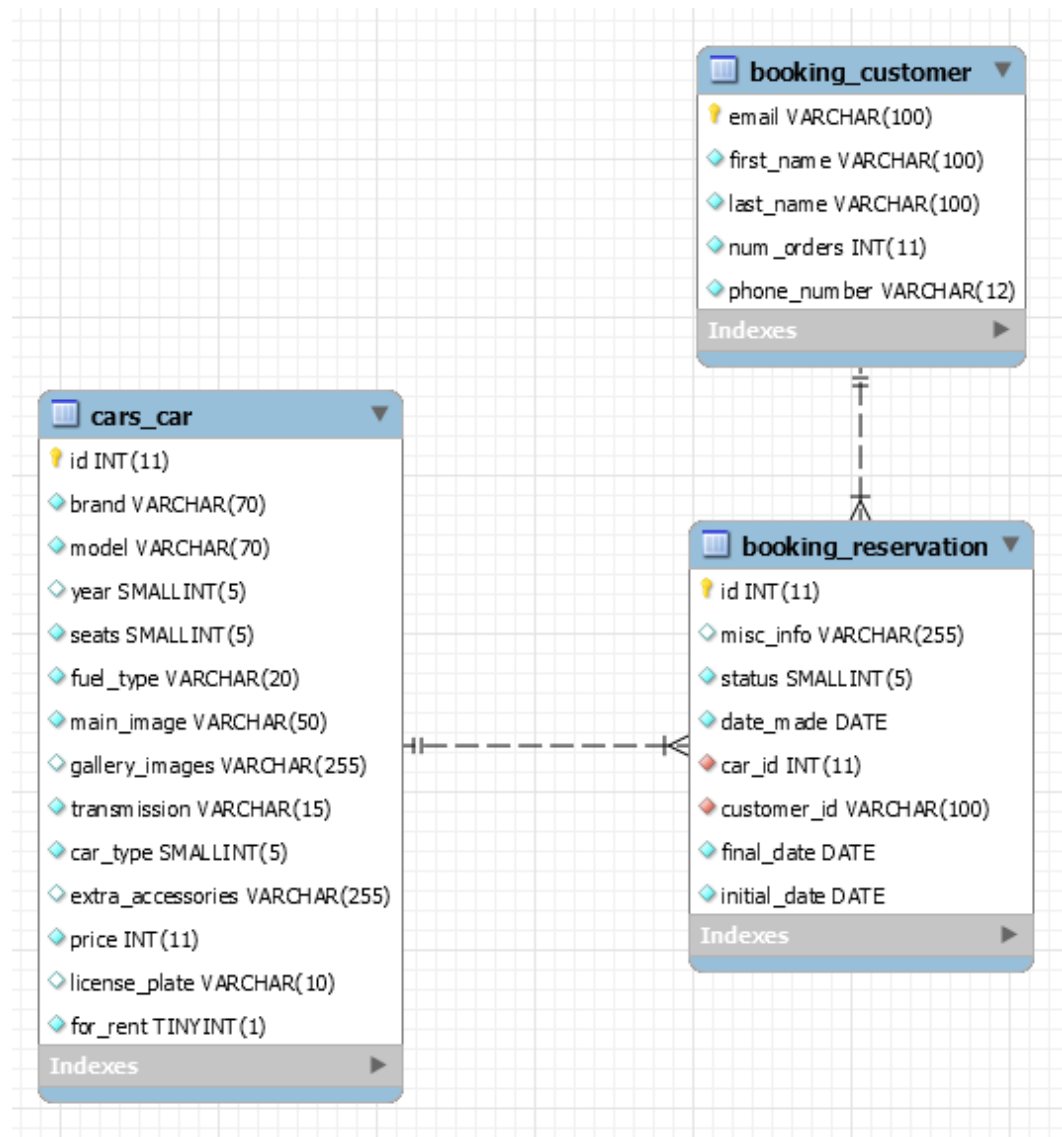
Kunde Opplysninger

Navn	Martin Engen
Epost	Nitrax92@gmail.com
Telefon Nummer	46966993

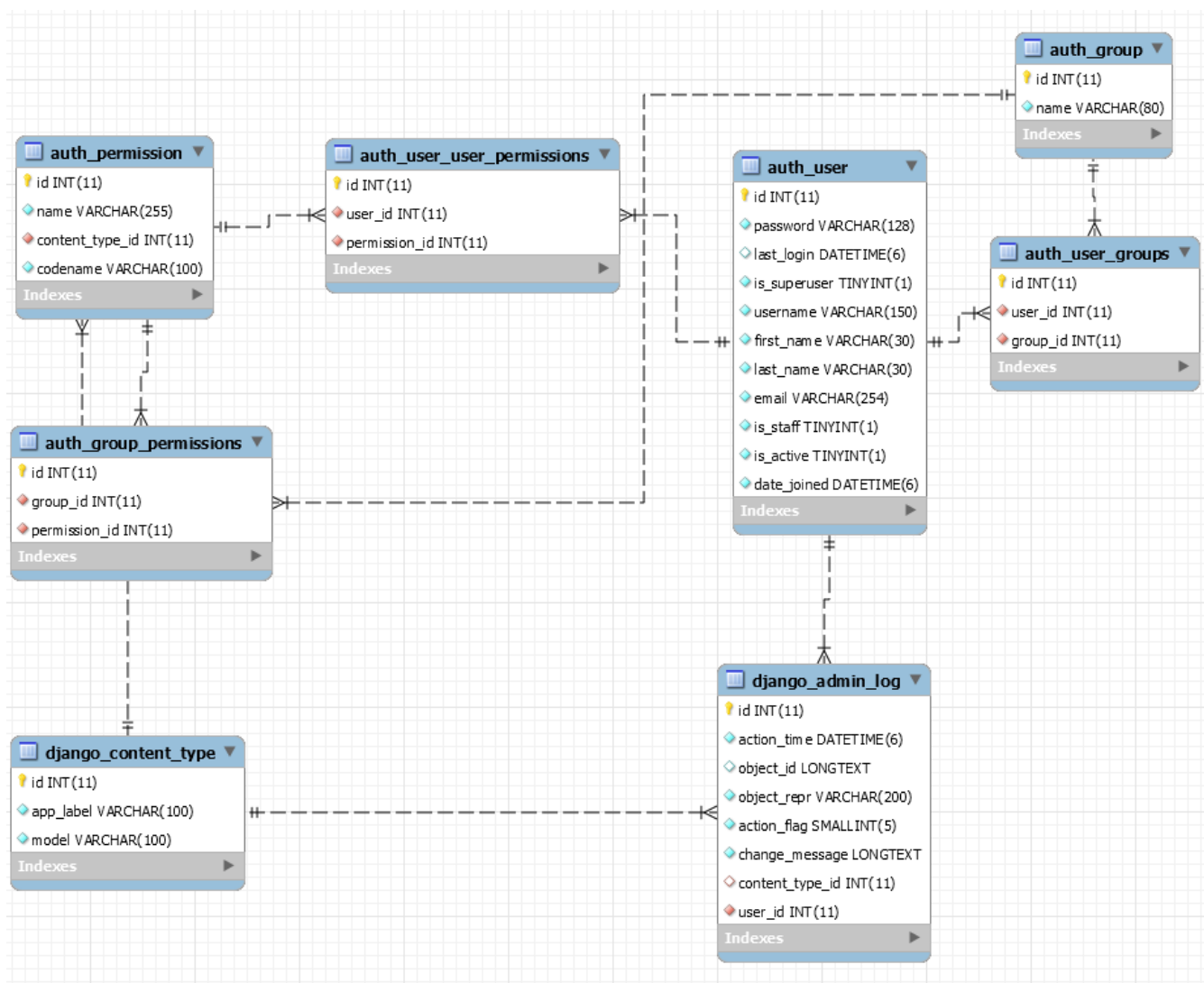
Kontakt Informasjon

Epost adresse: rusta.vrak@gmail.com
Telefon +47 400 49 489

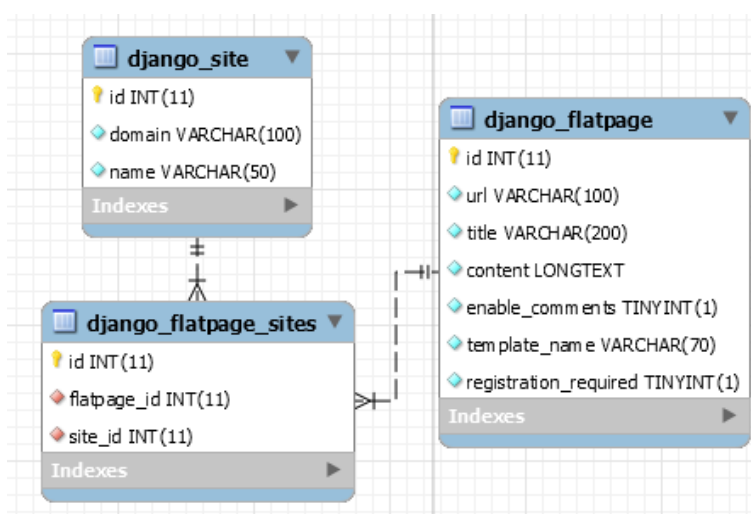
Vedlegg C



Tabeller Direkte knyttet til Bilene, Reservasjonen og Kundene til bedriften.

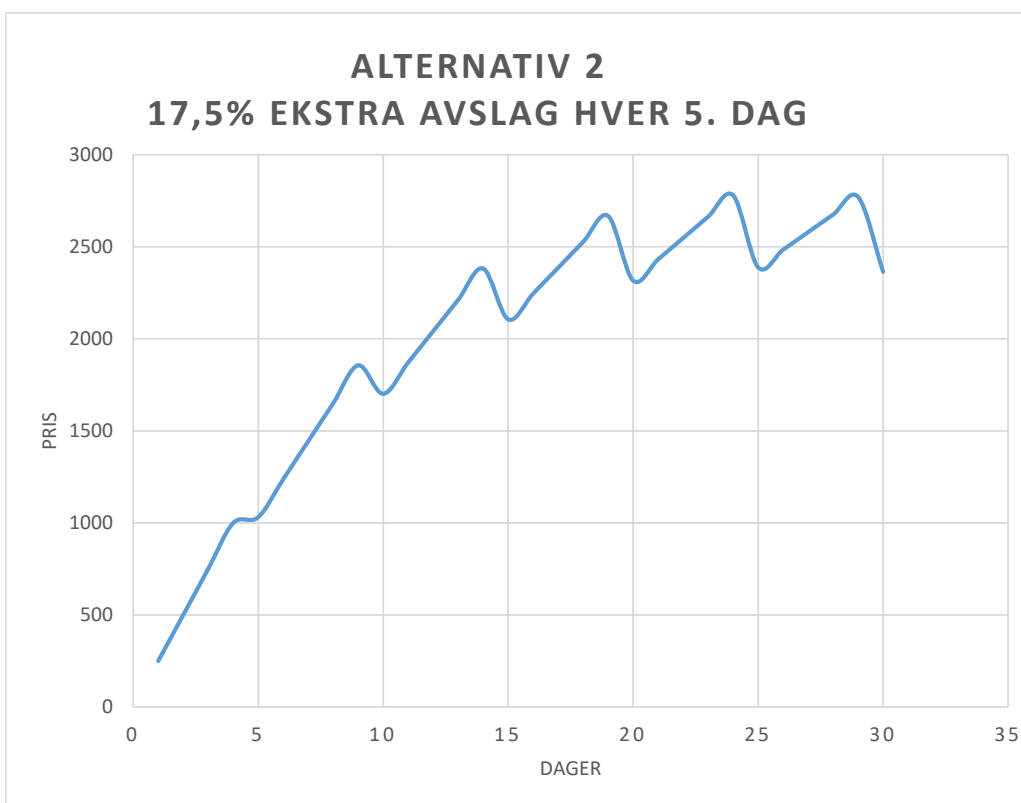
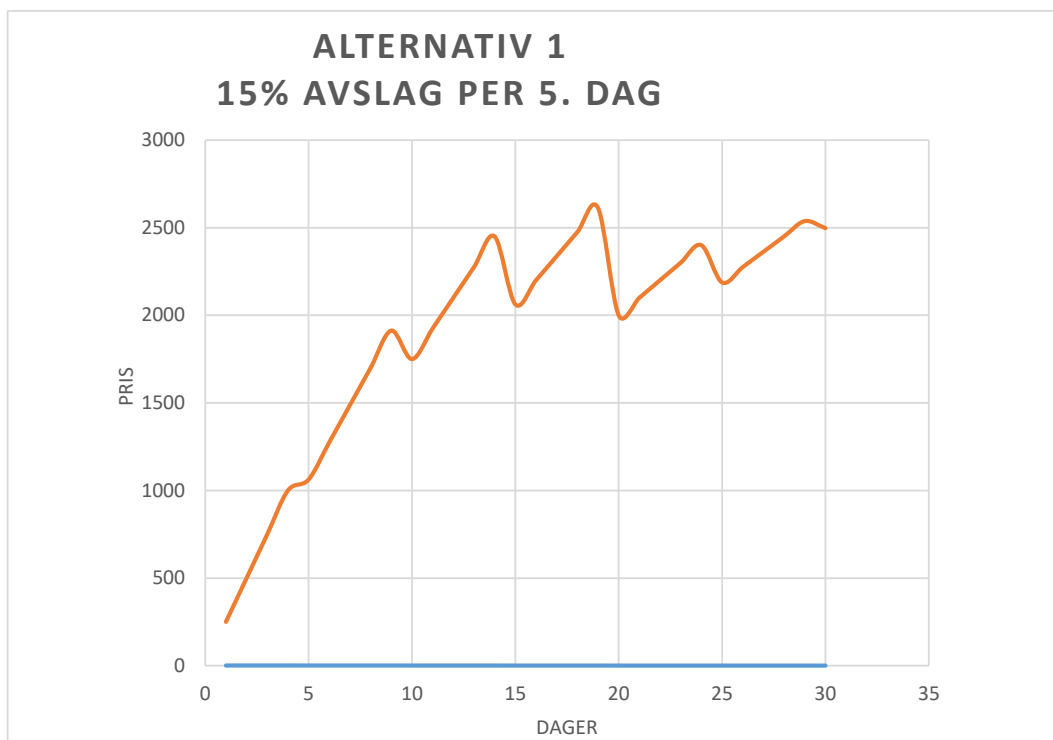


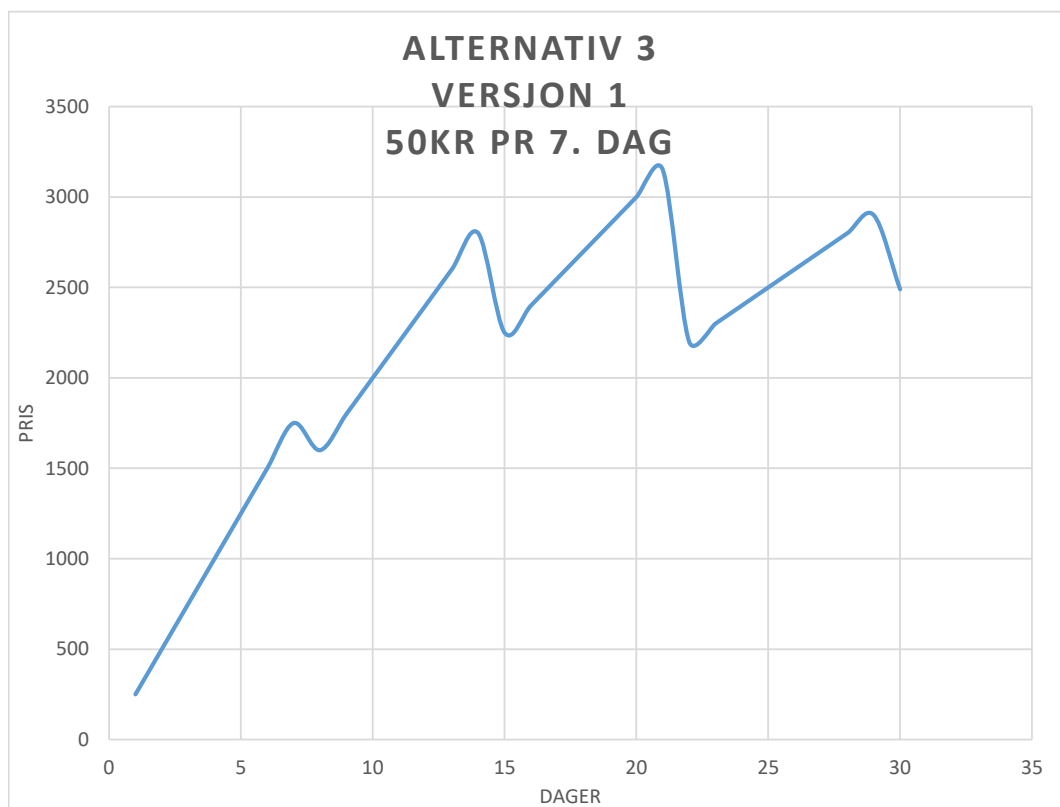
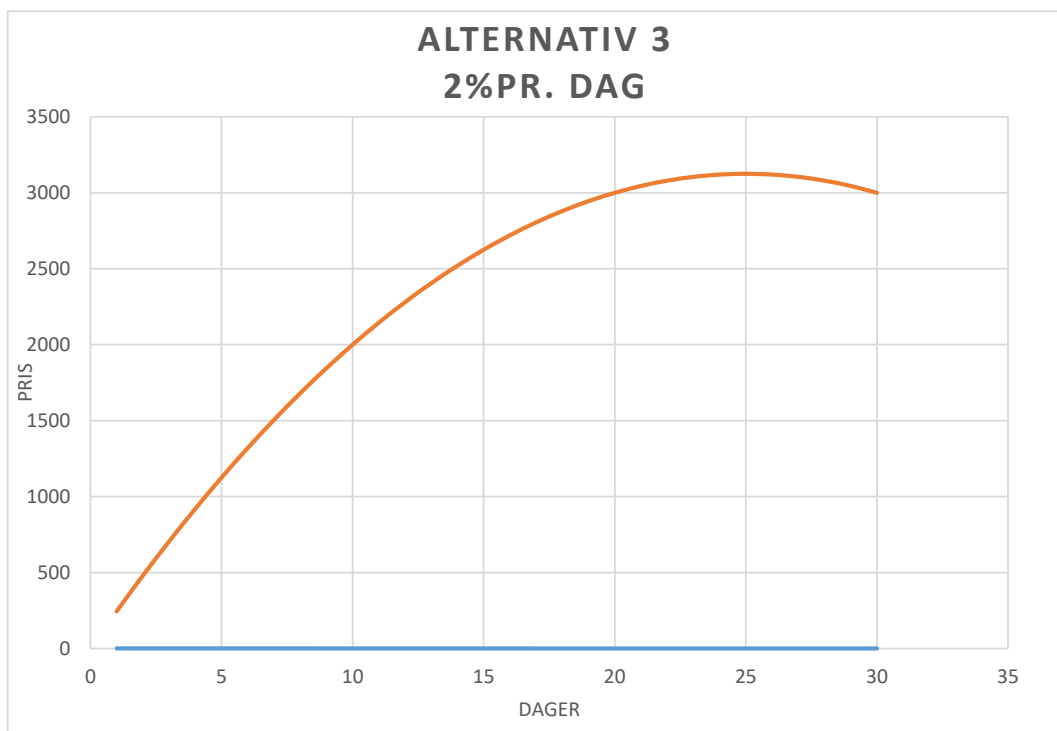
Tabellene som inneholder informasjon om brukersystemet. Det samme systemet brukes for admin brukere, som for normale brukere til nettsiden. Pr. i dag har ikke et brukersystem blitt koblet til da det ikke har vært behov.

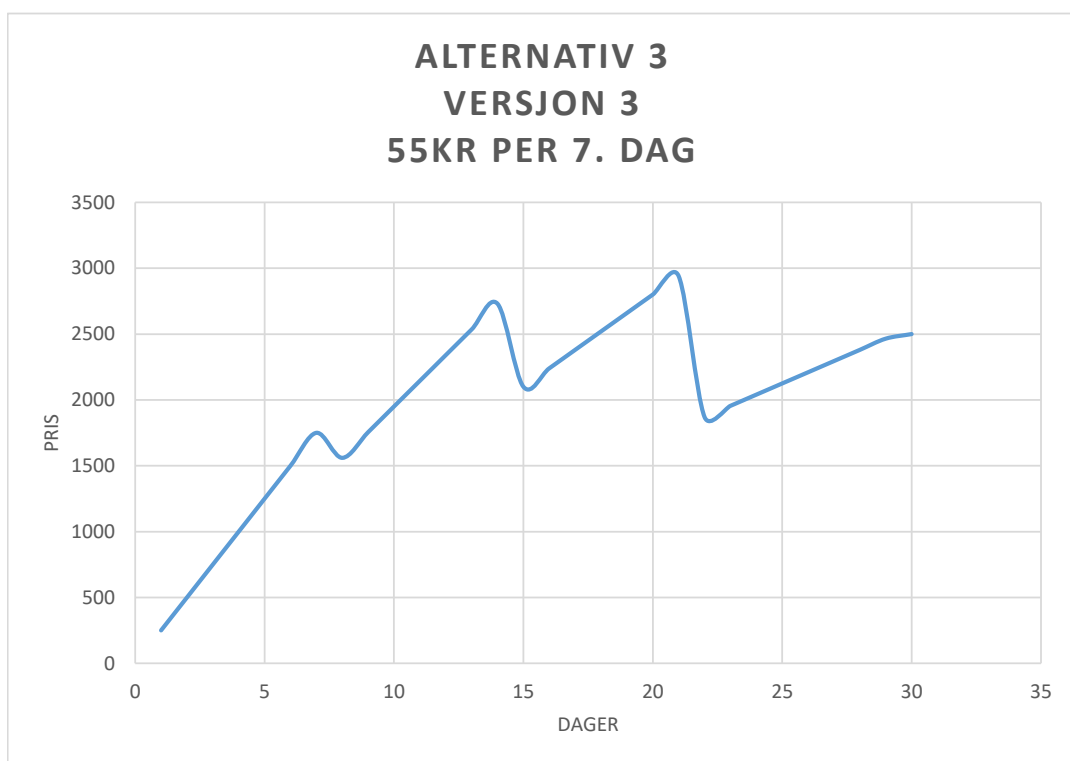
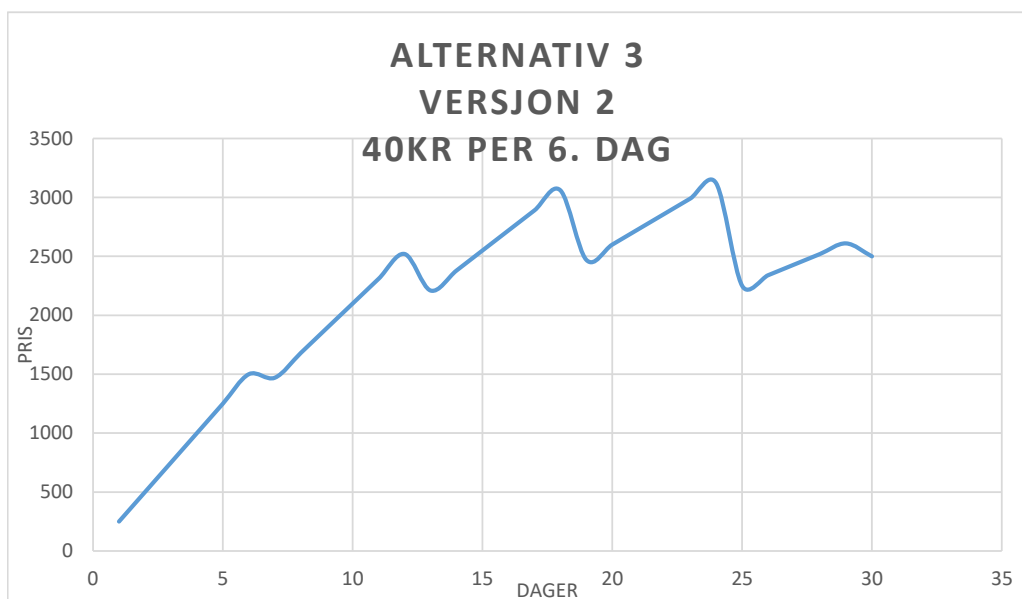


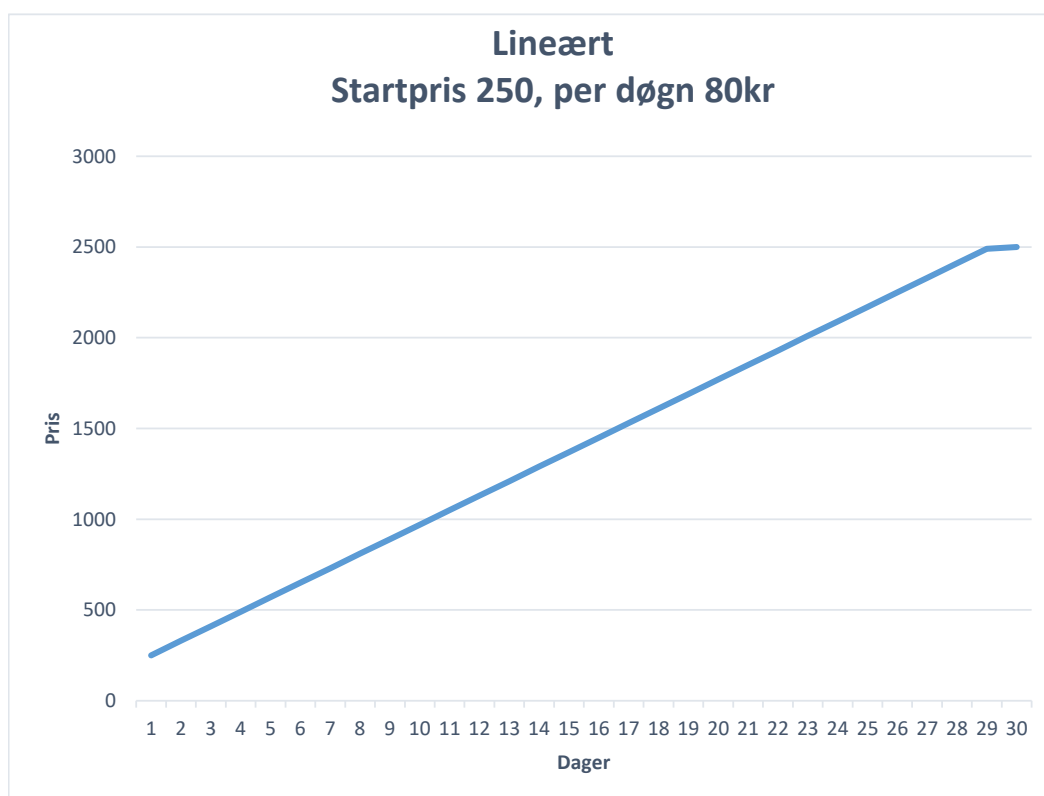
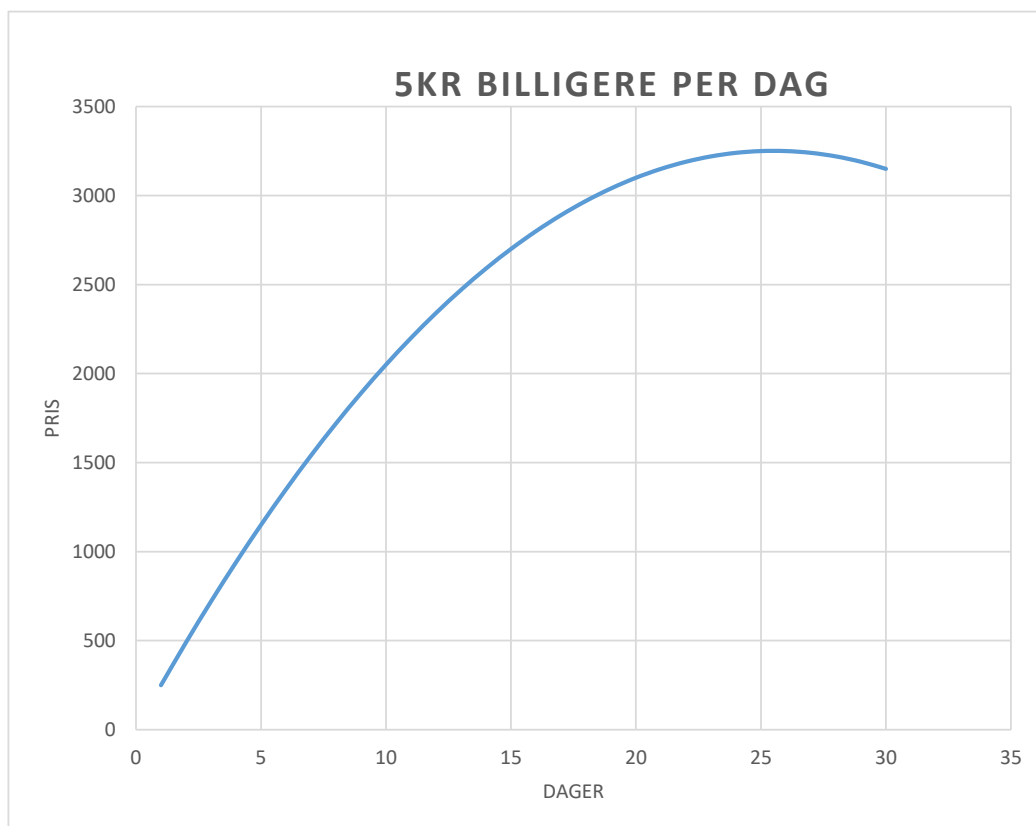
Flatpages. Dette systemet lar det være mulig å opprette dynamiske sider av nettsiden uten å redigere HTML filen for å redigere innholdet. Dette systemet ble brukt tidlig i prosjektsperioden, men ble slått av grunnet lite behov for denne type innhold.

Rusta Vrak Bilutleige Pris Forslag









Ledige Biler

Find_available_cars funksjonen håndterer systemet for å finne kun de biler som er ledige mellom 2 datoer (initial_date og final_date). Den vil kun sjekke på de biler som blir sendt inn som parameter.

Initial_booking_overlap og final_booking_overlap leter etter biler som overlapper. Disse blir lagt til i en liste, og disse bilene blir fjernet fra den liste over biler som blir sendt inn i parameter.

Kun de ledige biler blir returnert.

```
def find_available_cars(initial_date, final_date, cars):
    # All reservations of the cars
    # Using car__in to only get 1 of each car.
    car_res = Reservation.objects.filter(car__in=cars)

    # Checking if the dates input is between or same as any dates already reserved.
    initial_overlap = car_res.filter(Q(initial_date__lte=initial_date) & Q(final_date__gte=initial_date))
    final_overlap = car_res.filter(Q(initial_date__lte=final_date) & Q(final_date__gte=final_date))

    available_cars = cars.exclude(id__in=initial_overlap.values("car_id")).exclude(id__in=final_overlap.values("car_id"))

    return available_cars
```

Pris Kalkulator

Figuren under viser koden bak den implementerte priskalkulatoren.

Her ser vi at det er kun leieperioder mellom 1 til 29 dager som blir trenger bli kalkulert.

```
def price_calculator(days, start_price):
    # Round to closest 5
    def round_function(x, base=5):
        return int(base * round(float(x) / base))

    if 0 < days <= 29:
        # Initialize variables
        discount_calculated = 1          # Initialize to 0% discount.
        amount_of_discounts = days / 5  # Number of discount chunks.
        discount = 0.825                # 17.5% discount for each chunk

        # Iterate through the chunks, adding a 17.5% discount to the existing discount.
        for i in range(0, amount_of_discounts):
            discount_calculated *= discount

        final_price = round_function(start_price*days*discount_calculated)

    else:
        # 30 days rented, final price is 2500 for the normal priced cars.
        if start_price == 250:
            final_price = 2500

        # Pricier cars will be a bit more expensive for a 30 day rental.
        else:
            final_price = 2750

    return final_price
```