Marcus Galdal Tollefsen, Yunus Øzdemir, Ridwan Abukar, Abdala Ali, Gorgos fares Tammo

DriveMe

Software engineering HIø 2022

Innholdsfortegnelse

[Gruppemedlemmer 2](#_Toc120373655)

[Info om Applikasjonen: 3](#_Toc120373656)

[Personas 4](#_Toc120373657)

[Personas 1: 4](#_Toc120373658)

[Personas 2: 5](#_Toc120373659)

[Personas 3: 5](#_Toc120373660)

[Personas 4: 6](#_Toc120373661)

[Personas 5: 6](#_Toc120373662)

[User Stories 7](#_Toc120373663)

[User case: 7](#_Toc120373664)

[User case figur: 8](#_Toc120373665)

[Krav 9](#_Toc120373666)

[Generelle krav: 9](#_Toc120373667)

[Funksjonelle krav: 9](#_Toc120373668)

[Ikke funksjonelt krav: 9](#_Toc120373669)

[Sekvens diagram 10](#_Toc120373670)

[Sekvens diagram figur: 11](#_Toc120373671)

[Sekvens diagram figur 2: 12](#_Toc120373672)

[Bestilling av en bil 12](#_Toc120373673)

[Aktivitets diagram 13](#_Toc120373674)

[Låne ut bilen 13](#_Toc120373675)

[Aktivitets diagram figur låne ut bilen 13](#_Toc120373676)

[Innlogging 14](#_Toc120373677)

[Aktivitets diagram figur innlogging 14](#_Toc120373678)

[Dataflyt diagram: 15](#_Toc120373679)

[Problemstilling og domenet: 16](#_Toc120373680)

[Løsning til prosjektgruppen 16](#_Toc120373681)

[Prototype 17](#_Toc120373682)

[Veiledning for å åpne prosjektet (direkte metode) 17](#_Toc120373683)

[Veiledning for å sette opp Maven (i tilfelle 7.1 ikke fungerer) 19](#_Toc120373684)

[Litt om prototypen 22](#_Toc120373685)

[Prototypens layout og funksjon 24](#_Toc120373686)

[Testing ved bruk av Junit 31](#_Toc120373687)

# Gruppemedlemmer

# 

Yunus Øzdemir Ridwan Abukar

* [yunusao@hiof.no](mailto:yunusao@hiof.no) - [ridwanaa@hiof.no](mailto:ridwanaa@hiof.no)

Marcus Galdal Tollefsen Abdala Ali

* [marcusgt@hiof.no](mailto:marcusgt@hiof.no) - [abdalama@hiof.no](mailto:abdalama@hiof.no)

Gorgos Fares Tammo

* [gorgosft@hiof.no](mailto:gorgosft@hiof.no)

# Info om Applikasjonen:

Info om applikasjonen, det er en applikasjon som gjør det mulig for deg som ikke har en bil som gjerne ønsker få bruk for en bil i en kort periode til å låne en. Da har vi lansert en app der du som kunde kan lage deg en bruker og via bank id kan du betale for antall ønskende timer du ønsker å leie en bil. Da kan du velge mellom alle de tilgjengelige bilene som du ønsker og leie. Før du tar i bruk appen så må du ha gyldig førerkort, men også bank id, da bank id sjekker for deg om du har førerkort og er over 18år. I applikasjonen kan du velge mellom hva slags type bil som passer til din tur, om det skulle være SUV eller personbil.

Når du har laget deg bruker og har kommet helt fram til betalingen så er det enkelt og utføre prosessen etter det. Det er sånn at du får koordinater til bilens plassering, der du som kunde velger bilen som er nærmest deg da. Det er også sånn at bilene låses da opp enkelt med mobilen, døgnet rundt, og utgiftene kunden bruker regnes ut automatisk når du avslutter reisen i appen. For brukere som ønsker å registrere seg så befinner det mye informasjon i appen før man tar i bruk bilene og registeringen. Betalinger skjer etter turene er ferdig der kostnader som bom, parkering og drivstoff er med i regningen på slutten. Målgruppen blir jo da alle som er over 18, men er mest egnet for folk som trenger å låne en bil i mellomtida, så her er det stor målgruppe.

# Personas

Appen vår går ut på at folk som ønsker seg å låne bile eller ut låne bil til andre. Bruker kan søke etter biler til å låne så de kan gjøre

## Personas 1:



Elina Eriksen er en mor som jobber mye i hverdagen, hun har barn som for ikke mye tid med dem. Så hun prøver alltid å bruke så mye tid ho kan med barna i da helgene og i ferier. Så hun tenker å låne enn bil som er behagelige å sitte i til en stor Familie. Hun tenker å ta barna til tusenfryd og Bø sommerland.

## Personas 2:



Denne personen vil ha god bruk for applikasjonen der han behøver og besøke familien sin innimellom, leie bilene vil være kjekt for korte turer innimellom. James han er en mann som har nylig pensjonert seg, han tenker nå å låne en bil i mellomtida for besøk til barn og barnebarna sin sammen med sin kone, så dette ville være bra for han og kona å kunne låne en bil, det er enkelt der de kan gjøre alt via appen.

## Personas 3:



Ahmed er ung kar har en del jobb i byen, men han syns transportmidler tar lang tid, og han jobber som advokat så han må være i god tid på jobben sånn at han ikke blir forsinket, og han tenker å låne en bli som ikke koster mye og som kan få han til jobben

## Personas 4:



Ola er en ung man som her nettopp fått seg en jobb, han tenker å låne en bil som han kan bruke imellom tiden.

## Personas 5:



Adam er en man som liker å reise rundt i Norge, han tenker å låne en bil fordi han bil er i verksted. Han er interessert i bilen som han liker lage bil turer.

# User Stories

* Som kunde så vil jeg kunne logge inn i systemet
* Som kunde så vil jeg kunne se detaljer om bilen
* Som kunde så vil jeg kunne betale med vipps eller bank kort
* Som kunde så vil jeg kunne få bekreftelse på E-post eller på SMS
* Som kunde så vil kunne søke på forskjellige kategorier på bilen
* Som kunde så vil jeg kunne låne ut bilen min
* Som admin så vil jeg kunne

# User case:

Her er en USER CASE så hvis hva bruker og admin kan gjøre i vårt app. Det finnes ting som bruker kan ikke gjøre eller for ikke tilgang til med, men det kan gjøres av admin, for eksempel på det kan var at bruker ikke fått gitt godkjent på bilen, men det kan admin. Kunden kan logging og registrere seg i appen, kunden kan også se på spesifikasjoner av bilen altså detaljer om bilen det kan også admin se. Kunden skal kunne endre på eposten og passordet sitt med det kan ikke en admin gjøre for kunden det er kunden som har ansvar for det. Kunden skal kunne endre og avbestilte bestilling sitt bestilling selv men hvis kunde trenger hjelp så kan admin hjelpe kunde med det.

## User case figur:



# Krav

## Generelle krav:

### Funksjonelle krav:

1. Kunde skal kunne logging inn i systemet ved hjelp av en knapp.
2. Kunde skal kunne legge til en eller flere biler for utleie gjennom å registrere de i systemet.
3. Kunde skal få bekreftelse på e-post.
4. Kunde skal kunne logge ut ved å trykke på «logg ut» knappen.
5. Kunde skal kunne velge mellom betalings metoder.
6. Kunden skal kunne betale med vipps, ved å trykke på «Betale med vipps» knappen.
7. Kunden skal kunne betale med bankkort, ved å trykke på «Betale med bank kort» valgalternativet
8. Kunde skal kunne sjekke om bilen er ledig gjennom å trykke på knappen «sjekk om bilen er ledig»
9. Kunde skal kunne velge om h\*n vil låne bil ved å trykke på «lån bil» knappen
10. Kunde skal kunne velge om h\*n vil låne ut bil ved å trykke på «registrer bil» knappen

### Ikke funksjonelt krav:

1. Systemet må være raskt, systemet skal ikke bruke mere enn 5 sekunder på å laste inn den gjeldene siden
2. Systemet må være tilrettelagt for enhver, det skal ikke ta en ny kunde mere enn et minutt til å låne en bil.
3. Systemet må fungere på alle plattformer.
4. systemet må handtere 60 000 samtidige brukere
5. Sjekken om betalingen er fullført skal ikke ta mere enn 3-5 sekunder.

# Sekvens diagram

Dette sekvens diagram hvis hvordan appen vår fungerer, og hvordan den håndtere ting, altså den viser hvordan ting kommer til å kjøre. Hvis kunde logger inn med feil passord så for brukeren en melding som sier «prøv igjen». Med andre ord sekvens diagram viser hvordan data i et system samhandler i en prosess. Den viser data som går mellom brukeren og systemet og i hvilken rekkefølge de kjøres.

## Sekvens diagram figur:

Et bilde som inneholder bord

Automatisk generert beskrivelse

# Sekvens diagram figur 2:

## Bestilling av en bil

Her så er det laget et sekvens-diagram om brukere/kunder som skal bestille en låne bil, her så viser diagrammet prosessen fra å bestille en bil, til betalingen til bestillingen også er det i dette tilfelle appen/nettsiden som viser deg hvilke tilgjengelige biler det er og gir ut da informasjonen tilbake til kunden.

Figuren viser når kunden skal foreta seg bestilling av en lånebil. Systemet viser da prosessen kunden må foreta seg for å nå målet sitt, her er andre elementer på plass som selskapet. Til slutt mottar selskapet om ledige biler, som de da sender informasjon videre til kunden.



# Aktivitets diagram

## Låne ut bilen

Hvis Kunde vil låne ut bil sin det må vare lett for h\*n å gjøre det. Det kunden må gjøre er å logge seg inn systemet der etter trykke på knappen å låne ut bilen, der etter sette bilen i systemet, der etter så bli bilen sjekket om det er godt kjente for låning. Kunden kan ikke selv si at bilen er godkjent. Hvis bilen er godkjent så blir man sendt vider, men vis bilen er ikke godkjent så for kunden melding på det.

### Aktivitets diagram figur låne ut bilen



## Innlogging

Hvis Kunde vil logge seg inn i systemet så det skal være lett for brukeren å gjøre det. Det første kunde må gjøre er å skrive sitt brukernavn og passord, der etter bli brukernavn og passordet sjekket og hvis alt er som det skal så blir brukeren logget inn, hvis ikke så for brukeren er melding om h\*n vil endre passordet eller prøve på nytt.

### Aktivitets diagram figur innlogging



# Dataflyt diagram:

Denne figuren sjekker om brukeren er registrert og om det er ledige biler tilgjengelig, også sjekker den om brukeren har en gyldig bruker i appen, vis ja kan man gå videre. I denne figuren, så får vi vite om kunden har en gyldig bruker og om de kan få sjekke om det finnes ledige biler. Først og fremst så må alle kundene være over 18 år og ha gyldig legitimasjon, dette sjekkes når de lager seg en bruker. I dette tilfellet har de enten en bruker som viser at de har førerkort og har gyldig bruker, ellers så har de ikke gyldig førerkort eller ikke gyldig bruker. Figuren viser da tilslutt hvor mange biler som er tilgjengelige.



# Problemstilling og domenet:

Hva er mest lønnsomt for deg som har en bil, men ikke bruker den så ofte, og deg som ikke har en bil, men trenger å låne?

Da har vi kommet med en løsning, vi har lansert en app som gjør deg som har en bil kan låne ute en bil og deg som ikke har en bil, kan låne en bil. For deg som låner, så er det lønnsomt å låne en bil når du vil, billigere vis du ikke ønsker å ha en bil eller ikke tilgang til det innimellom. Da kommer appen (Driveme) inn og kan løse problemene dine, om du skal på et møte, jobb, besøk og andre ting man ønsker. Domenet, omgivelsene og miljøet er jo lagt for storbyer og folk som trenger en bil i en kort periode. Hvem vi henvender også til er folk som er over 18 år og har førerkort. Behovet rundt prosjektet, tilfredsstille behovet til kunden, ved å kunne låne en bil etter behovet deres.

# Løsning til prosjektgruppen

Det går ut på at den løser problemene i problemstilling da vi har lagd en prototype for kundene, den viser da hvordan appen skal fungere da vi har forskjellige funksjoner som er tilegnet forskjellige brukere. Som for eksempel om du har bil skal du benytte deg en knapp på som vises på skjermen din, eller om du ønsker å låne deg en bil. Dette kommer fram i applikasjonen og du vil også kunne velge mellom noen få forskjellige biler som du ønsker selv. Den håndterer de forskjellige elementene i problemstilling da appen løser problemene du har ved å da benytte deg ved lånebil når du måtte ønske deg. Du må også vøre over 18 år og ha førerkort for å benytte deg av tjenestene.

# Prototype

Protoypen vårt har blitt laget ved bruk av Java forms. Vi lagde den på plattformen IntelliJ IDEA. Java Forms gir brukeren en ganske lett GUI som man kan sette opp sitt program i. Hvis man vil jobbe videre med protoypen, eller vil bare lese kildekoden så kan man lett gjøre det ved bruk av IntelliJ. For å utføre testing så brukte vi «dependenciene»; JUnit Jupiter API (scope: test, version 5.7.2), JUnit Jupiter engine (scope: test, version 5.7.2), JUnit Jupiter params (scope: test, version 5.7.2) og til slutt maven-failsafe-plugin (versjon 2.22.2). Vi brukte også en siste dependancy kalt gson. Denne lot oss serialisere og deserialisere objekter til JSON så vi kunne skrive/lese til/fra fil. IntelliJ kan bli nedlastet her: <https://www.jetbrains.com/idea/download>.

## Veiledning for å åpne prosjektet (direkte metode)

1. Trykk «Open» på IntelliJ idea:



1. Finn frem prosjektmappen (kalt SEG):

Et bilde som inneholder tekst

Automatisk generert beskrivelse

1. Så lett som det så skal det være mulig å se på kildekoden/jobbe videre med protoypen:

Et bilde som inneholder tekst

Automatisk generert beskrivelse

## Veiledning for å sette opp Maven (i tilfelle 7.1 ikke fungerer)

1. Trykk «New Project» på IntelliJ.

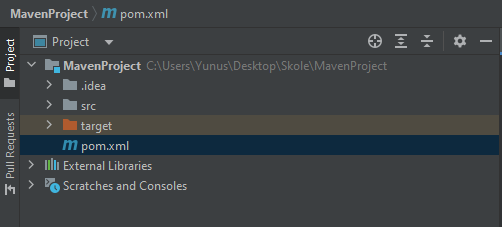


1. Her så skal du velge egendefinert navn og lokasjon. Deretter velger du Java som språk, og Maven som build system. Behold versjon 17 på openjdk. Trykk create når ferdig.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

1. Nå skal du over til pom.xml som du kan finne på listen til siden.

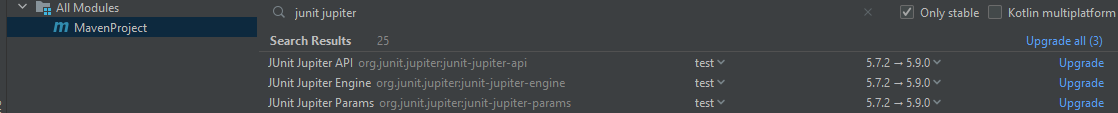


1. Du kan enten kopiere koden under og build prosjektet, eller manuelt legge til dependencies. Koden ligger på side 23. Ved å dobbel trykke «bildet» så får du muligheten til å kopiere (usikker på hvordan funksjonaliteten fungerer på pdf). Trykk dependencies som finnes nederst i vinduet. Her søker du på «junit jupiter».

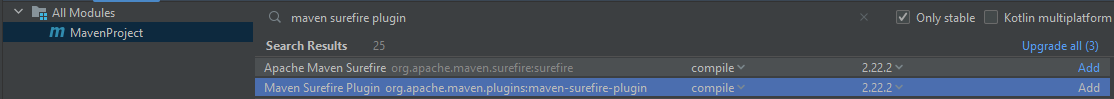
Et bilde som inneholder tekst, elektronikk, datamaskin, skjermbilde

Automatisk generert beskrivelse

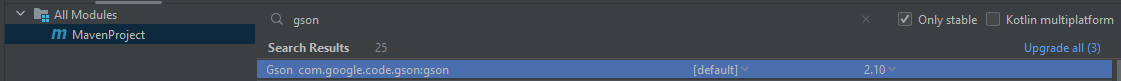
1. Bytt fra compile til test på; engine, api og params. Du skal i tillegg sette alle 3 versjonene til 5.7.2. Deretter kan du trykke add på hver av dem:



1. Nå søker du etter maven surefire plugin og legger til den:



1. Til slutt så søker du etter gson og legger til den:



1. Nå er alt klart. Hvis gson for en eller annen grunn ikke fungerer (slik som det gjorde med oss på starten), så kan du fikse det ved å gå over til «maven» helt til høyre i vinduet og trykke på «reload» knappen:

Et bilde som inneholder tekst

Automatisk generert beskrivelse

### Litt om prototypen

Protoypekoden er delt opp i 2 hovedmapper for kildekoden, og 2 egne javafiler for testing. Det var tidligere en egen mappe for Interfaces, som vi prøvde å sette opp funksjoner som vi kunne kalle til for å skrive til fil/lese fra fil. Dette fungerte dessverre ikke slik som vi ville, og har derfor blitt fjernet i sluttversjonen av protoypen. Endringene hvor vi brukte FileHandler og slikt kan finnes i git endringene. Vi fikk til eventuelt til å skrive til fil, men dessverre så fikk vi ikke til å lese fra fil. Dette kan finnes i versjon 3.4

Et bilde som inneholder tekst

Automatisk generert beskrivelse

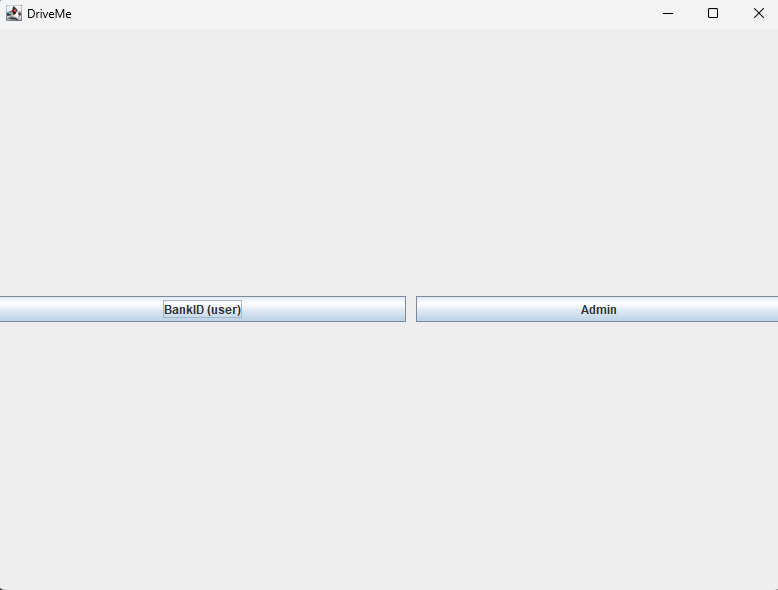
I forms mappen så kan vi finne både forms filen som ble brukt til å designe GUI-en. I tilegg så er javafilen som kontroller forms i samme sted. Denne javafilen kontrollerer alt fra hva som blir vist, når det blir vist og hva som skjer når vi trykker på knapper og samhandler med protoypen.

I modules mappen så kan vi se begge klassene som har blitt brukt; CarRegistration og CarRepository. Først så lages et CarRepository som blir brukt til å lagre bilene som blir laget ved bruk av CarRegistration.

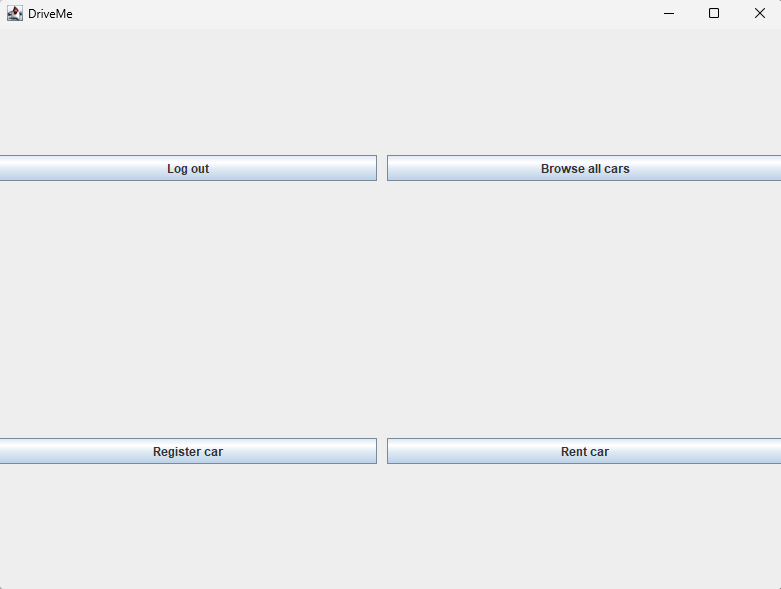


## Prototypens layout og funksjon

Når du først åpner prototypen så blir du velkommen av logg inn siden. Her har du muligheten til å logge inn som bruker (BankID), som gir deg muligheten til å registrer og låne bil, og admin, som gir deg muligheten til å redigere og slette biler fra listen.



Hvis du logger inn som bruker så får du muligheten til 4 funksjoner. Første er å logge ut (som tar deg tilbake til forrige siden), andre er å se alle biler som er registrert, tredje er å registrere en bil mens den siste er å kunne bestille en bil.



Vi starter først med «Browse all cars». Her så kan du se alle biler. Både de som er «rented» og de som ikke er «rented». Ved å trykke på main menu så går du alltid tilbake til innloggingssiden til brukeren.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

I register car panelet så har man 5 forskjellige valg. Ut ifra de 5, så har man et begrenset antall valg på 3 av dem. Disse 3 inkluderer antall «seats» man kan velge, hva slags «transmission type» det er på bilen, og hva slags «engine type» det er på bilen. Disse har allerede blitt fast bestemt av oss på hva vi aksepterer. Vi har gjort det umulig å kunne registrere biler uten registrerings nummer, men utenom det så er alt mulig. Man kan også velge om bilen er klar til å bli lånt ut med en gang, eller om du allerede har lånt den ut.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Et bilde som inneholder tekst

Automatisk generert beskrivelseEt bilde som inneholder tekst

Automatisk generert beskrivelseEt bilde som inneholder tekst

Automatisk generert beskrivelse

Til slutt så har vi «rent car» panelet. Her får du en liste med alle biler som er tilgjengelige. Du kan nå velge hvilken bil fra listen du vil låne bare ved å trykke på bilen, og deretter velge betalingsmåte.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Til slutt så har vi «admin» panelet. Når du først går inn i panelet så får du listen over alle biler i listen, både tilgjengelig og utilgjengelig. Her kan du lett trykke på en bil og slette den fra listen.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

Du har også muligheten til å redigere biler ved å trykke på en bil og trykke «edit car». Her har du muligheten til å bytte alt informasjon slik som du ønsker ettersom at du er admin.

Et bilde som inneholder tekst

Automatisk generert beskrivelse

# Testing ved bruk av Junit

**Tittel**:

Hvis du registrerer en bil uten å røre «rented» sjekkboksene så skal bilen automatisk være «not rented».

**Antagelser:**

Brukeren glemte å trykke på «not rented».

**Test steg:**

1. Bruker legger til bil uten å trykke på sjekkboksene.

**Forventet resultat:**

Bilen blir automatisk lagret som not rented.

**Junit test:**



**Tittel**:

Når du låner en bil så skal det bli faktisk bli lånt ut.

**Antagelser:**

Brukeren låner bilen.

**Test steg:**

1. Bruker låner bilen.

**Forventet resultat:**

Bilen blir lånt ut og kommer nå ikke opp på «availableCarsList».

**Junit test:**



**Tittel**:

Når du registrer en bil som du vil legge til «repository» så blir den faktisk lagt til «repository»-en.

**Antagelser:**

Brukeren registrerte bil.

**Test steg:**

1. Bruker registrerer bil.

**Forventet resultat:**

Bilen kommer inn i listen for bil oppbevaringsstedet.

**Junit test:**



**Tittel**:

Hvis admin fjerner bil fra «repository»-en så skal den bli fjernet.

**Antagelser:**

Admin vil fjerne en bil

**Test steg:**

1. Admin sletter bil fra «repository».

**Forventet resultat:**

Bilen blir slettet og befinner seg ikke lenger i «repository»

**Junit test:**



**Tittel**:

Sjekker om «repository» skrives til fil og leses opp. Sjekkes ved å se om «repository» er fortsatt samme klasse etter å ha blitt lest opp igjen

**Antagelser:**

Om fil skrivning og fil lesing fungerer riktig.

**Test steg:**

1. Blir gjort via backend.

**Forventet resultat:**

Listen blir skrevet ned og lest opp riktig, og sjekkes om listen er tom, og om listen er samme klasse som før den ble skrevet ned.

**Junit test:**



**Tittel**:

Bruker skal ikke kunne legge til 2 biler med samme registreringsnummer

**Antagelser:**

Bruker prøver å legge til 2 biler med samme registreringsnummer

**Test steg:**

1. Bruker lager 1 bil med registreringsnummer PG46886
2. Bruker prøver å registrere en bil til med registreringsnummer PG46886

**Forventet resultat:**

Bilen blir ikke lagt til og vi sjekker ved å se om bilen eksiterer i listen.

**Junit test:**



**Tittel**:

Bruker skal bare få opp tilgjengelige biler i listen når bruker skal låne bil

**Antagelser:**

Bruker vil låne ledig bil

**Test steg:**

1. Bruker åpner panelet for å låne bil og ser en ledig liste.

**Forventet resultat:**

Alle bilene som kommer opp er bare ledige biler. Sjekkes ved å som opptatte biler kommer frem i «getAllAvaialableCars» listen.

**Junit test:**



**Tittel**:

Alle biler, både tilgjengelig og utilgjengelig, blir vist når det trengs.

**Antagelser:**

Bruker/admin vil se alle bilene som har blitt registrert

**Test steg:**

1. Bruker/admin åpner listen for alle biler

**Forventet resultat:**

Alle biler, både tilgjengelig og utilgjengelig, blir vist.

**Junit test:**

