# Produkt rapport

# Vaskeriet

Mathias Wriedt Kamp

Marius Møller

# Titelblad

## Vaskeriet produktrapport

| | |
|---|---|
| Uddannelse: | Datateknikker med speciale i programmering |
| Hovedforløb: | 5. Hovedforløb |
| Titel på projektet: | Vaskeriet |
| Projektperiode: | Fra mandag d. 27/02/2023 til fredag d. 31/03/2023 |
| Vejledere: | Camilla Mai Ryskjær - faglærer ZBC Ringsted |
| Udarbejdet af: | Mathias Wriedt Kamp, Marius Møller |

_____

Mathias Wriedt Kamp

_____

Marius Møller

_____

Camilla Mai Ryskjær

| | |
|---|---|
| Antal normalsider: | 8 |
| Afleveringsdato: | 23/03/2023 |

Mathias Wriedt Kamp

Marius Møller

# Indholdsfortegnelse

Mathias Wriedt Kamp

Marius Møller

Mathias Wriedt Kamp

Marius Møller

# Kravspecifikation

## 1. Introduktion

Dette dokument indeholder alle kendte krav, både funktionelle og ikke funktionelle krav som er stillet af brugere samt udlejere der har med vaskerier at gøre.

Private lejere og udlejere føler sig snydt i forhold til elpriserne når der skal betales for brug af vaskemaskiner og tørretumblere. I nogle tilfælde så bliver der betalt for meget fordi den variable elpris har været lav i den periode, og i andre tilfælde så skal udlejeren spise resten af regningen da privatlejerens husleje ikke dækker hele elregningen.

Udlejere har gjort sig nogle tanker om hvordan de kan løse problemet med for meget / for lidt betaling for el. De har blandt andet gjort sig nogle tanker om forudbetaling for el. og hermed også en stigning af huslejen. Men da udlejere maksimalt må hævehuslejen med 4% hvert år, så er det i rigtigt mange tilfælde ikke nok til at dække for betalingen af el.

Udlejere ønsker derfor muligheden for en bedre form for forudbetaling af el når der bliver brugt vaskemaskiner og tørretumblere hos deres lejere. Forudbetalingen skal tage højde for hvilket tidspunkt på dagen der bliver vasket / tørret, og beregne prisen ude fra tidspunktets elpris. Dermed vil man få en mere reel pris for el og både udlejere og lejere bliver tilfredse.

Der er et ønske om at man skal kunne lave en online booking af en vaskemaskine / tørretumbler med prædefineret vaske / tørreprogrammer. Onlinebookingen skal tage højde for elprisen i den givende periode som der bliver valgt, som skal bruges til forudbetalingen.

### 1.1 Formål med kravspecifikationen

Formålet med denne kravspecifikation er at definere de krav der måtte være til det nye system. Det forventes at kravene løbene testes, samt at kravspecifikationen løbene opdateres.

### 1.2 Definitioner, akronymer og forkortelser

**Fully-dressed:** en use-case med veldefineret forløbsbeskrivelse.

**Maskine**: referer til en vaskemaskine eller en tørretumbler.

## 2. System

Dette system består af et online bookingsystem der samarbejder med elprisenligenu.dk som benyttes til at hente de nyeste elpriser både vest og øst for Storebælt. Derudover vil der blive eftermonteret et modul til maskinen som består af en RFID-skanner, et kommunikationsmodul samt et valideringsmodul. Systemets hjemmeside kommunikerer direkte med et WEB API som håndterer alle interaktioner med systemet.
Systemet får opdateret dets elpriser klokken 13:10 hver dag da elprisenligenu.dk modtager morgendagens elpriser klokken 13 dagen forinden. opdateringen

**RFID-skanner** har til formål at modtage en RFID-chips unikke id og sende det videre til kommunikationsmodulet som sender id'et videre til WEB API 'et til validering for om det er det samme RFID som har booket maskinen.

**WEB API** har til formål at håndtere alle former for interaktioner med systemet. Interaktioner fra hjemmesiden og kommunikationsmodulet.

**Database** har til formål at håndtere data fra hjemmesiden, kommunikationsmodulet og apiklienten som skal hente nye elpriser hver dag.

**Kommunikationsmodul** har til formål at skabe forbindelse mellem vaskemaskinemodulet og WEB API 'et.

## 3. Funktionalitet

| **Use Case navn** | **Login til webapplikation** |
|---|---|
| **Id** | 1 |
| **Version** | 1.0 |
| **Beskrivelse** | Denne use-case beskriver hvordan en bruger logger ind i webapplikationen |
| **Problemstillingen** | En bruger skal have mulighed for at logge ind i webapplikationen |
| **Scope** | Webapplikationen |
| **Aktør(er)** | Bruger |
| **Stakeholder og Interesser** | Brugeren: Vil logge ind i webapplikationen |
| **Prækonditioner** | Brugeren er oprettet i systemet<br>Brugeren kender sit brugernavn og password |
| **Postkonditioner** | Brugerens brugernavn og password er korrekt<br>Brugeren er logget ind i webapplikationen |
| **Success forløb** | 1. Brugeren indtaster sit brugernavn og adgangskode og klikker på "Log ind" knappen.<br>2. Systemet validerer brugerens oplysninger og logger brugeren ind i webapplikationen.<br>3. Brugeren omdirigeres til webapplikationens startside |
| **Alternativt forløb** | 1.1 (Alt 1 forkert brugernavn eller password)<br>Brugeren har indtastet forkert brugernavn eller password.<br>2.1 (Alt 1 forkert brugernavn eller password)<br>systemet validerer at brugernavnet og passwordet ikke |

| | |
|---|---|
| | matcher en bruger i systemet. Brugeren modtager en fejlbesked om at brugernavnet eller passwordet er forkert. |
| **Udvidelsesmuligheder** | 1. Hvis brugeren ikke kan huske sit brugernavn eller adgangskode, kan brugeren klikke på "Glemt adgangskode" som skal sende en besked til administratoren af systemet om at passwordet skal nulstilles.<br>2. Hvis brugeren har indtastet sit brugernavn eller password forkert 3 gange låses brugeren hvis den eksisterer. |
| **Ikke funktionelle krav** | 1. Systemet skal have en maksimal svartid på 5 sekunder, når brugeren klikker på login<br>2. brugerens brugernavn og adgangskode gemmes i browserens adgangskoder |
| **Åbne problemer** | ingen |

| **Use Case navn** | **Opret booking** |
|---|---|
| **Id** | 2 |
| **Version** | 1.0 |
| **Beskrivelse** | Denne use case beskriver, hvordan en bruger kan oprette en booking i webapplikationen. |
| **Problemstillingen** | En bruger skal have mulighed for at oprette en booking som enten går på at de skal vasketøj eller tørre tøj. |
| **Scope** | Webapplikationen. |
| **Aktør(er)** | En bruger |

| | |
|---|---|
| **Stakeholder og Interesser** | Bruger – Ønsker at oprette en booking for at vaske / tørre tøj |
| **Prækonditioner** | 1. brugeren er logget ind i webapplikationen<br>2. brugeren har navigeret til ”Booking” siden<br>3. brugeren har klikket på ”opret booking”<br>4. Der er mindst 1 tid ledig |
| **Postkonditioner** | 1. brugeren har oprettet en booking i webapplikationen og har sikret sig adgang til en ledig tid. |
| **Success forløb** | 1. Brugeren navigerer til ”Booking” siden på webapplikationen<br>2. Systemet viser en oversigt maskiner<br>3. Brugeren vælger en maskine<br>4. Systemet viser en oversigt over programmer tilknyttet til den valgte maskine<br>5. Brugeren vælger et program<br>6. Systemet viser en oversigt over ledige tider<br>7. Brugeren vælger en ledig tid<br>8. Brugeren klikker på ”opret booking”<br>9. Systemet validerer bookingens information<br>10. Systemet viser en besked om at bookingen er oprettet |
| **Alternativt forløb** | 2.1 Systemet giver en fejlbesked om at der ikke er nogle maskiner<br><br>4.1 Systemet giver en fejlbesked om at der ikke er nogle programmer<br><br>6.1 Systemet giver en fejlbesked om at der ikke er nogle ledige tider<br><br>10.1 Systemet giver besked om at bookingen ikke blev oprettet. |
| **Udvidelsesmuligheder** | 1. hvis brugeren ønsker at aflyse sin booking, kan brugeren gøre det fra brugerens bookingoversigt. |

| | |
|---|---|
| | 2. hvis brugeren ønsker at ændre sin booking, kan brugeren gøre det fra brugerens bookingoversigt.<br>3. brugeren kan modtage en mail eller sms-bekræftelse på at bookingen er registreret. |
| **Ikke funktionelle krav** | 1. systemet skal have en maksimal svartid på 5 sekunder, når en bruger opretter en booking.<br>2. systemet skal kunne håndtere mindst 5 samtidige brugere der er i gang med at oprette bookinger. |
| **Åbne problemer** | Ingen. |

Mathias Wriedt Kamp

Marius Møller

| **Use Case navn** | **Skan RFID-kort og start maskine** |
|---|---|
| **Id** | 3 |
| **Version** | 1.0 |
| **Beskrivelse** | Denne use-case beskriver, hvordan en bruger kan skanne sit RFID-kort og starte en maskine hvis skanningstidspunktet stemmer overens med den booking brugeren har oprettet. |
| **Problemstillingen** | En bruger ønsker at skanne sit RFID-kort og starte en maskine. |
| **Scope** | RFID-læser, maskine. |
| **Aktør(er)** | Bruger: den bruger der ønsker at starte en maskine ved at skanne sit RFID-kort og dermed tage sin booking i brug.<br>System: Vaskeriets system, der validerer brugerens RFID-kort og bookingen |
| **Stakeholder og Interesser** | Bruger: Ønsker at benytte en maskine via et RFID-kort |
| **Prækonditioner** | 1. brugeren har oprettet en booking i systemet.<br>2. maskinen er ledig på det ønskede tidspunkt for bookingen<br>3. brugeren har et gyldigt RFID-kort som er bundet op på brugeren der har oprettet bookingen |
| **Postkonditioner** | 1. Brugeren kan starte maskinen efter skan af RFID-kort. |
| **Success forløb** | 1. Brugeren står foran den ønskede maskine som er ledig og scanner sit RFID-kort på maskinens kortlæser.<br>2. Systemet validerer, at kortet er gyldigt og at brugeren er bundet op på en eksisterende booking<br>3. Systemet validerer, at skanningstidspunktet er inden for 30 minutter før bookingen, er sat til at starte. |

| | |
|---|---|
| | 4. systemet viser en besked på maskinens display om at maskinen kan benyttes. |
| **Alternativt forløb** | 1.1. Brugeren scanner et ugyldigt RFID-kort: Hvis RFID-kortet ikke er gyldigt eller ikke er tilknyttet nogen booking, vil systemet vise en fejlmeddelelse og nægte brugeren adgang til at benytte maskinen.<br>1.2. Brugeren scanner et gyldigt RFID-kort uden for booking-vinduet: systemet viser en fejlmeddelelse og nægter adgang til at brugeren kan starte maskinen. |
| **Udvidelsesmuligheder** | 1a. Hvis kortet er ugyldigt eller ikke tilknyttet til nogen bruger, vises en fejlbesked på displayet og afviser adgang til maskinen.<br><br>1b. Hvis der ikke er nogen eksisterende booking tilknyttet til brugerens RFID-kort vises en fejlbesked på displayet og afviser adgang til maskinen.<br><br>3a. Hvis brugeren skanner sit RFID-kort på et forkert tidspunkt for bookingen, viser systemet en fejlbesked på displayet og afviser adgang til maskinen.<br><br>4.a hvis maskinen ikke starter, giver systemet en fejlbesked på displayet. |
| **Ikke funktionelle krav** | 1. systemet skal give besked efter 5 sekunder fra brugeren har skannet sit RFID-kort |
| **Åbne problemer** | Ingen. |

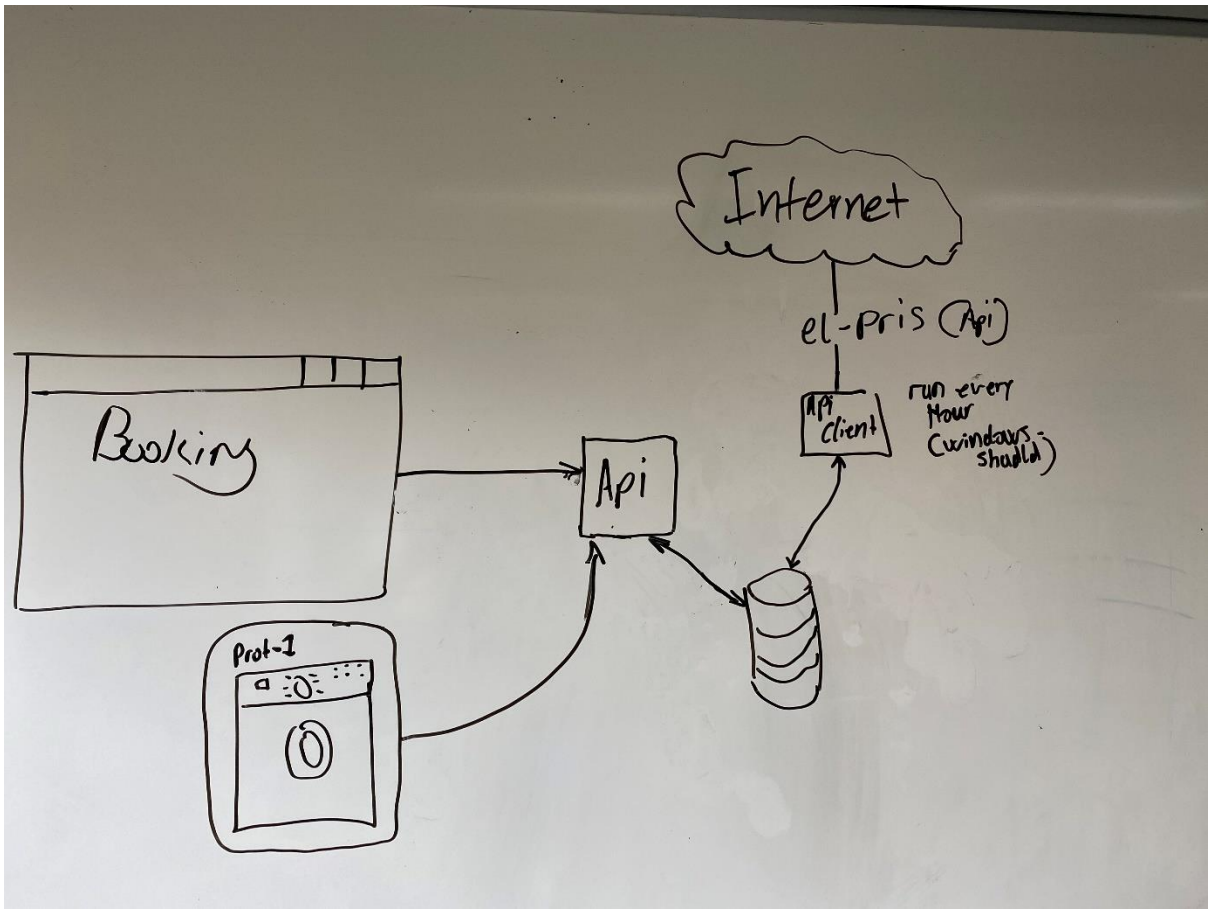
Mathias Wriedt Kamp

Marius Møller

# Teknisk produkt dokumentation

## Rigt billede

[Bilag\System documentation\Rich picture.jpg]



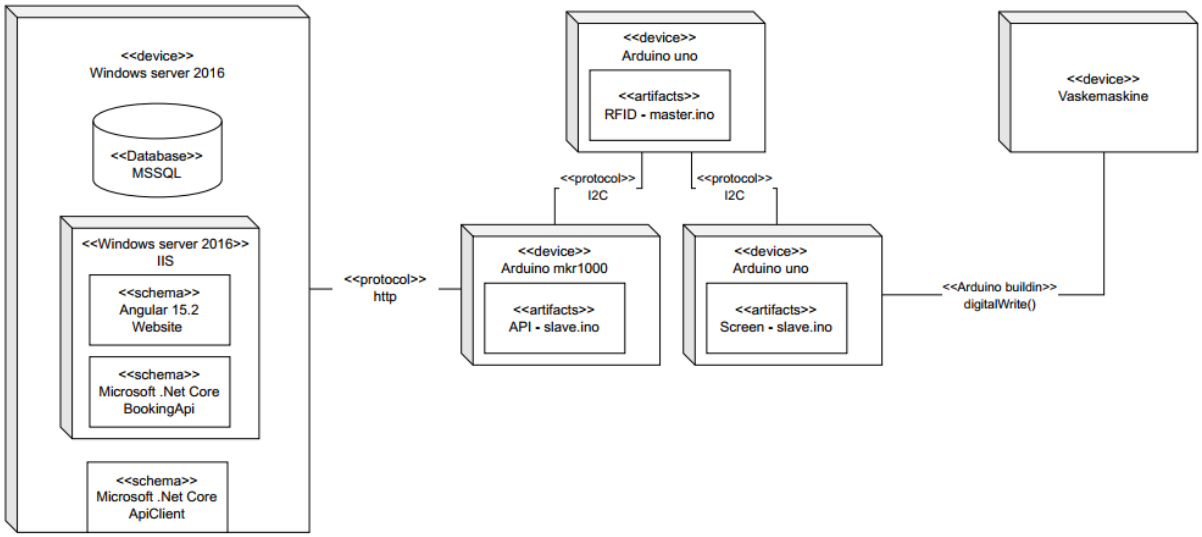

*Figur 1 Rigt billede*

Mathias Wriedt Kamp

Marius Møller

## Deployment diagram

[Bilag\System documentation\deployment diagram.pdf]



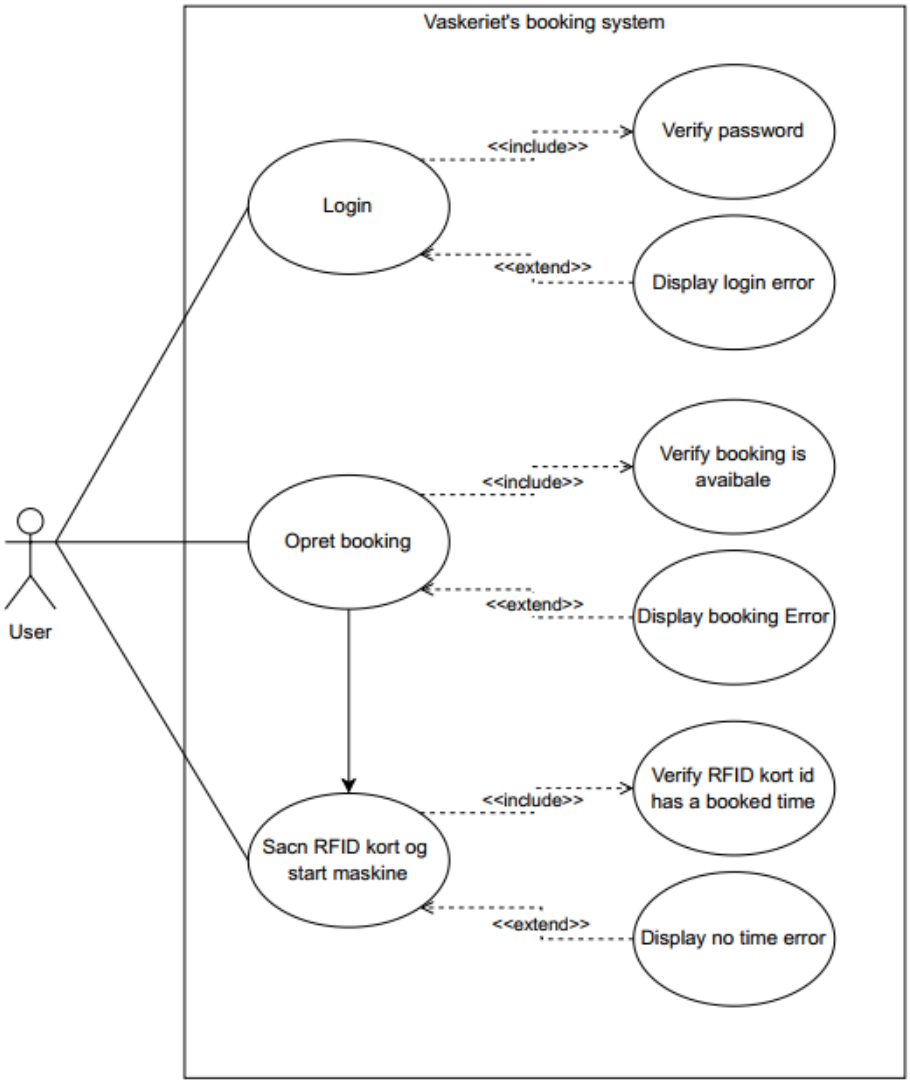

*Figur 2 Deployment diagram.*

Mathias Wriedt Kamp

Marius Møller

## Use-case diagram

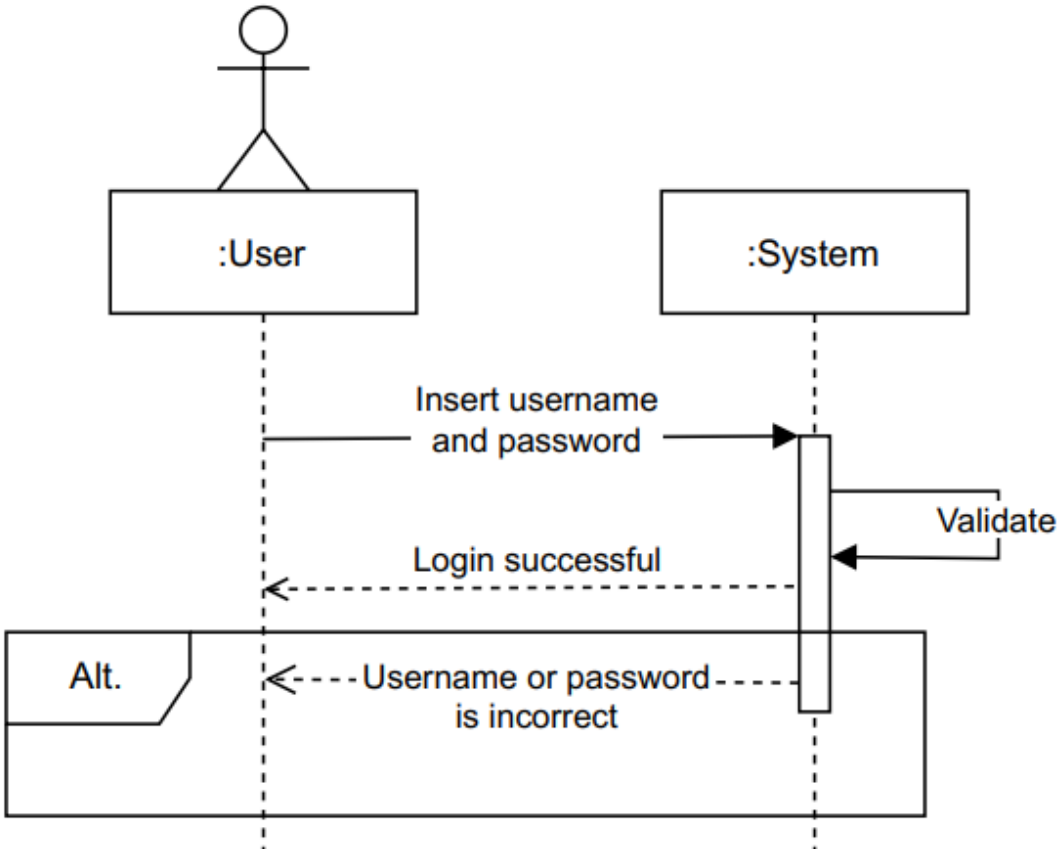
[Bilag\Use-case diagram.pdf]




*Figur 3 use-case diagram.*

## Login – SSD

Bilag\System documentation\SSD and SD diagram\1) Login\Login - SSD diagram.pdf



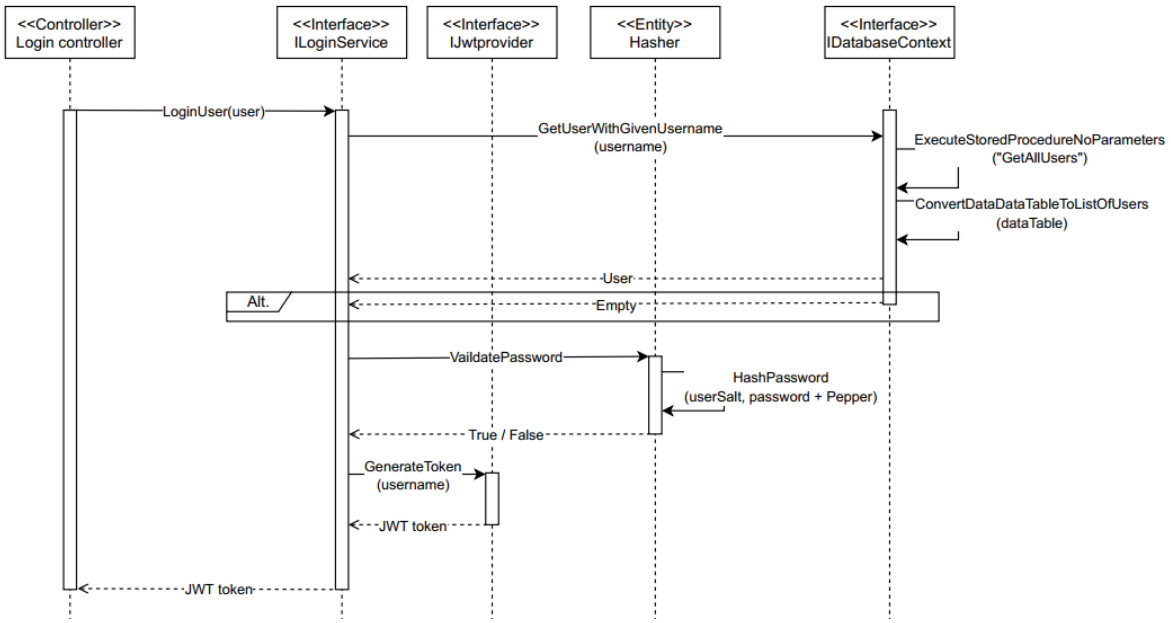

*Figur 4 Login - SSD*

Mathias Wriedt Kamp

Marius Møller

## Login – SD

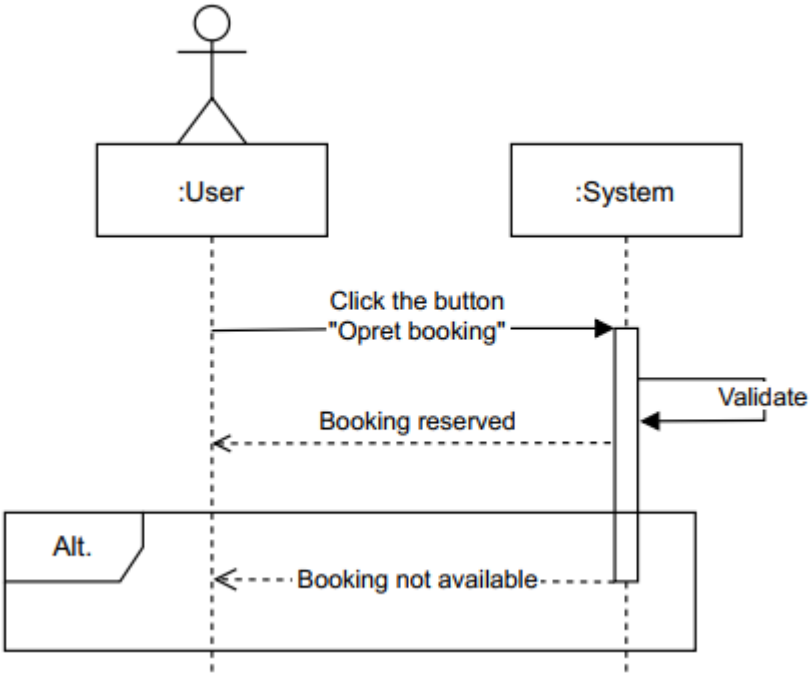
[Bilag\System documentation\SSD and SD diagram\1) Login\Login - SD diagram.pdf]




*Figur 5 Login-SD*

## Create booking – SSD

Bilag\System documentation\SSD and SD diagram\2) Create booking\Create Booking - SSD diagram.pdf



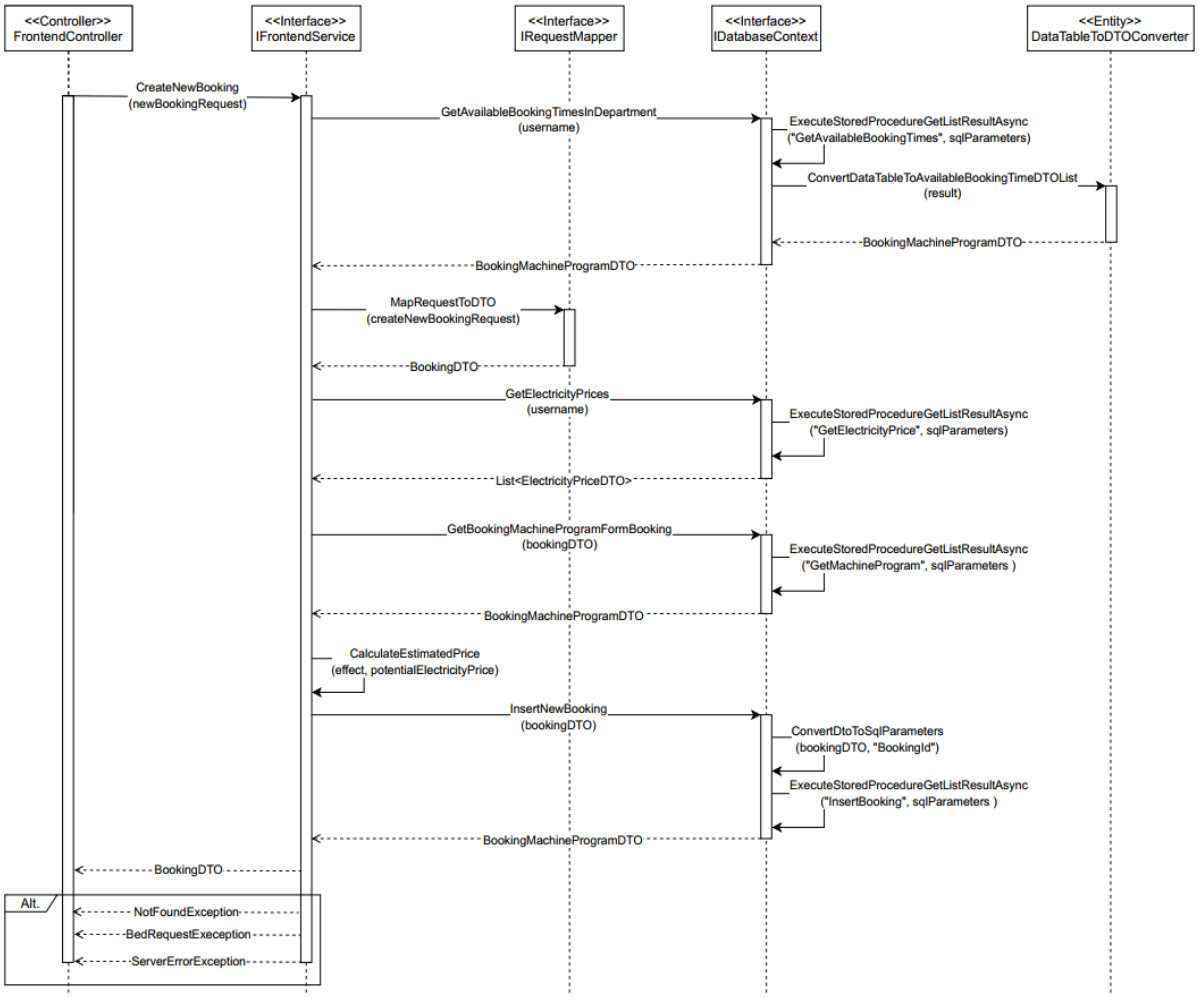

*Figur 6 Create booking – SSD.*

Mathias Wriedt Kamp

Marius Møller

## Create booking – SD

[Bilag\System documentation\SSD and SD diagram\2) Create booking\Opret booking - SD diagram.pdf]



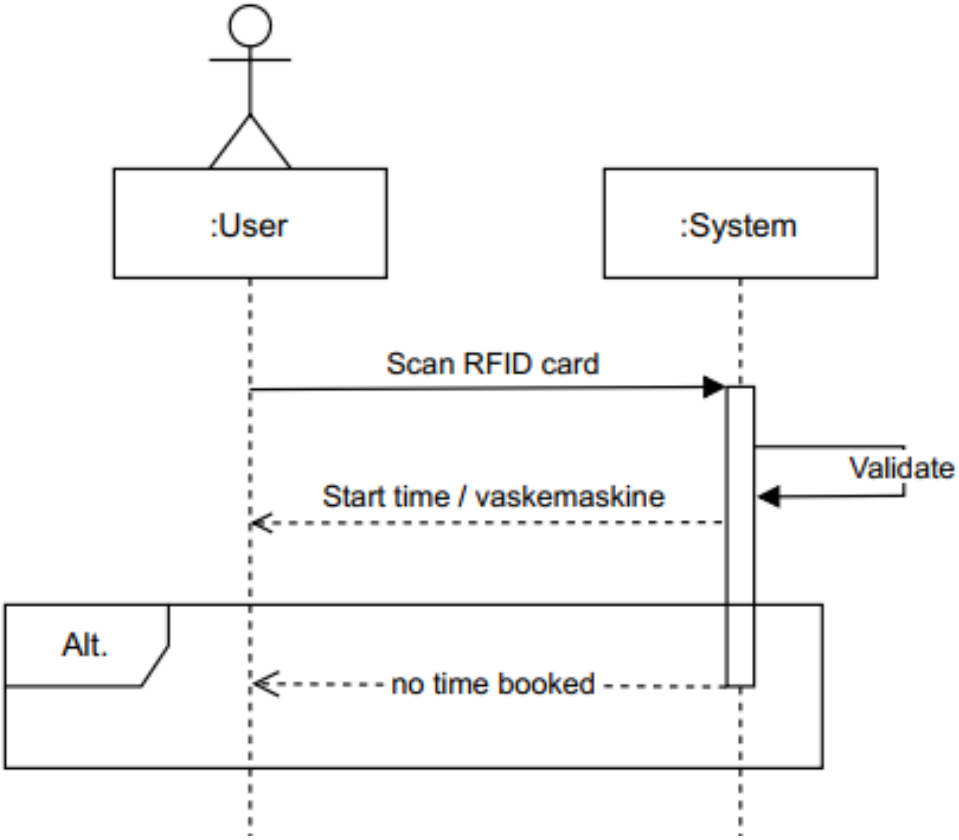

*Figur 7 Create booking – SD.*

Mathias Wriedt Kamp

Marius Møller

## Scan rfid-card – SSD

Bilag\System documentation\SSD and SD diagram\3) Scan RFID card\Scan RFID card - SSD diagram.pdf



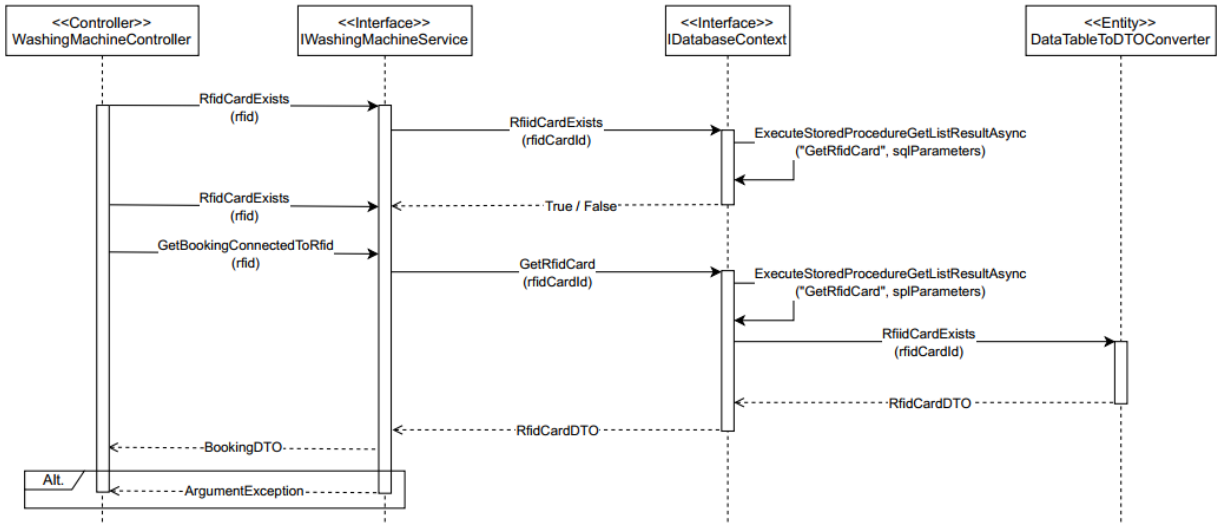

*Figur 8 Scan Rfid-card – SSD.*

Mathias Wriedt Kamp

Marius Møller

## Scan Rfid-card – SD

Bilag\System documentation\SSD and SD diagram\3) Scan RFID card\Scan RFID card - SD diagram.pdf



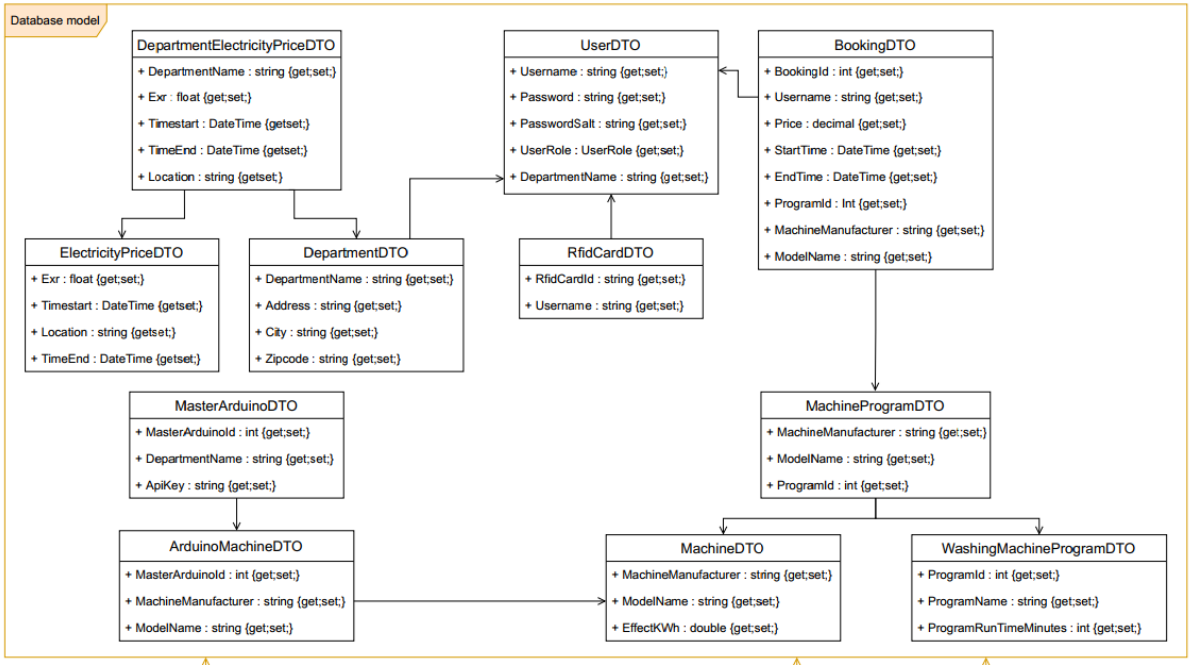

*Figur 9 Scan Rfid-card SD*

Mathias Wriedt Kamp

Marius Møller

## Booking Api – Class diagram

Det fulde klassediagram kan ses her [Bilag\diagrammer\Class diagram\api booking Class diagram.pdf]



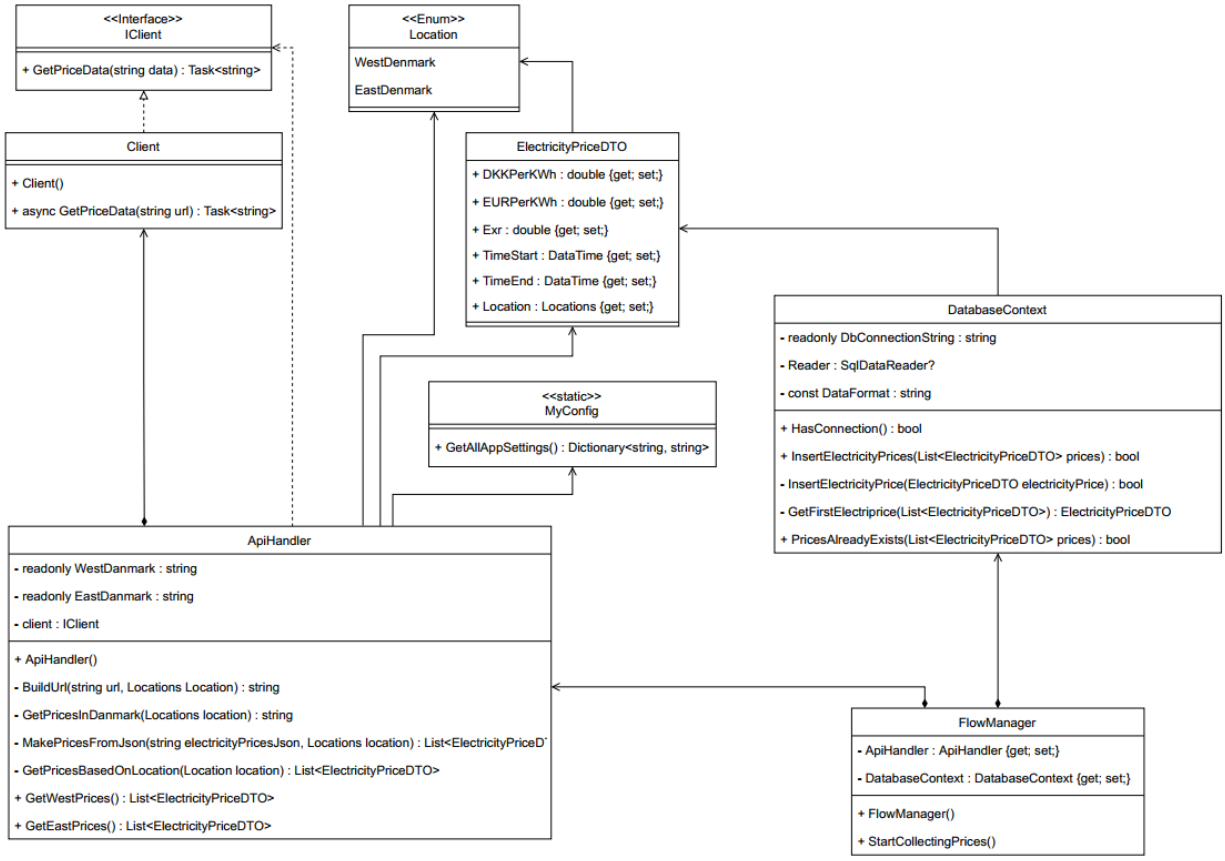

*Figur 10 Booking API Class diagram*

Mathias Wriedt Kamp

Marius Møller

## Api client – class diagram

[Bilag\System documentation\ApiClient\Api client Class diagram.pdf]



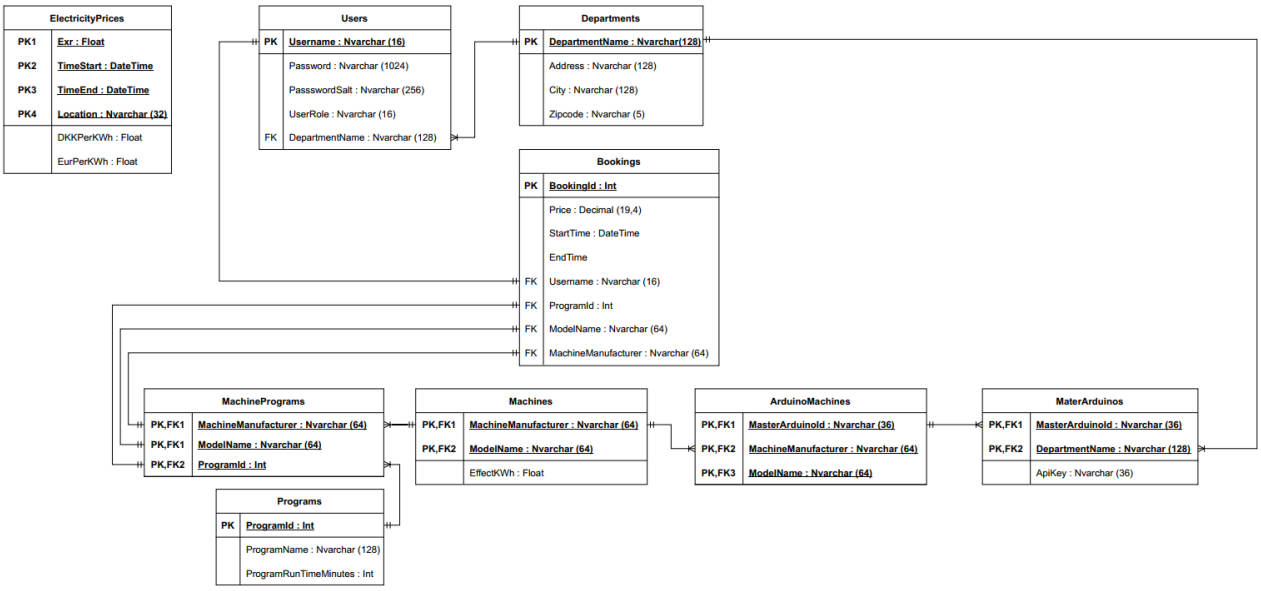

*Figur 11 Api client - class diagram*

Mathias Wriedt Kamp

Marius Møller

## Database – Er diagram

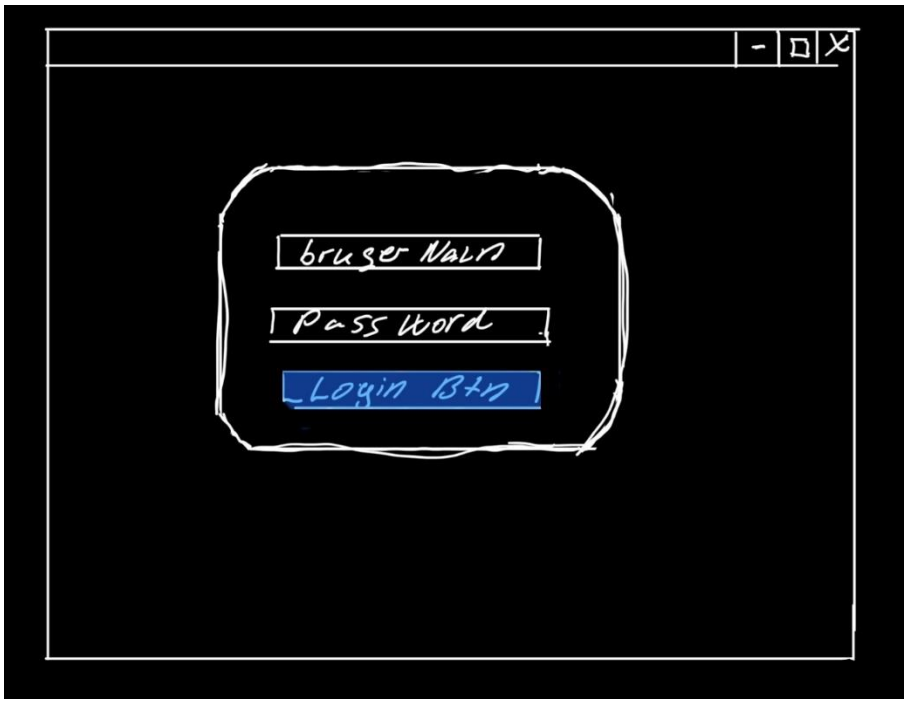
[Bilag\System documentation\Database\Er-diagram\Er_diagram_VaskeriDb.pdf]




*Figur 12 Er - diagram over databasen*

## Webapplikation – wireframe Login

Bilag\System documentation\Webapplikation\Wireframe\Login.png



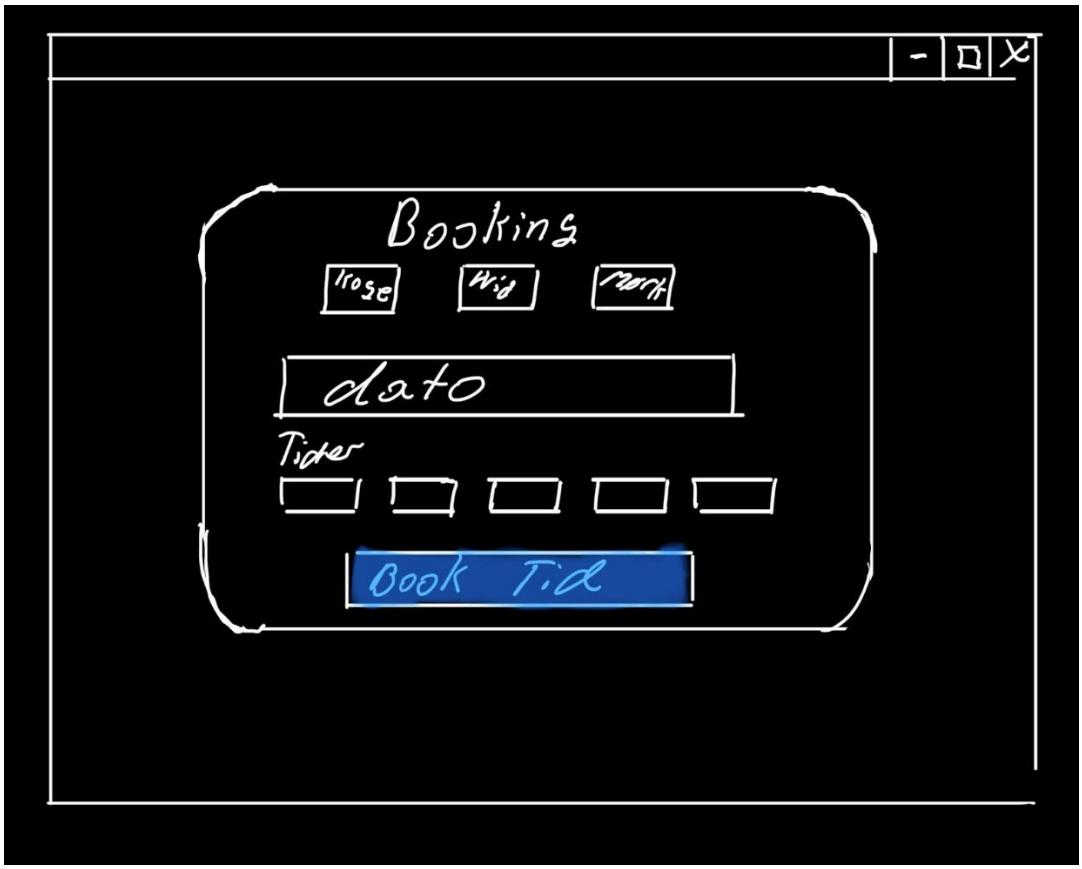

*Figur 13 Webapplikation wireframe af login siden*

Mathias Wriedt Kamp

Marius Møller

## Webapplikation – Wireframe Create booking

Bilag\System documentation\Webapplikation\Wireframe\Create booking.png




*Figur 14 Webapplikation Wireframe af create booking siden*

Mathias Wriedt Kamp

Marius Møller

## Webapplikation – Wireframe Check bookings

Bilag\Wireframe\check bookings.jpg

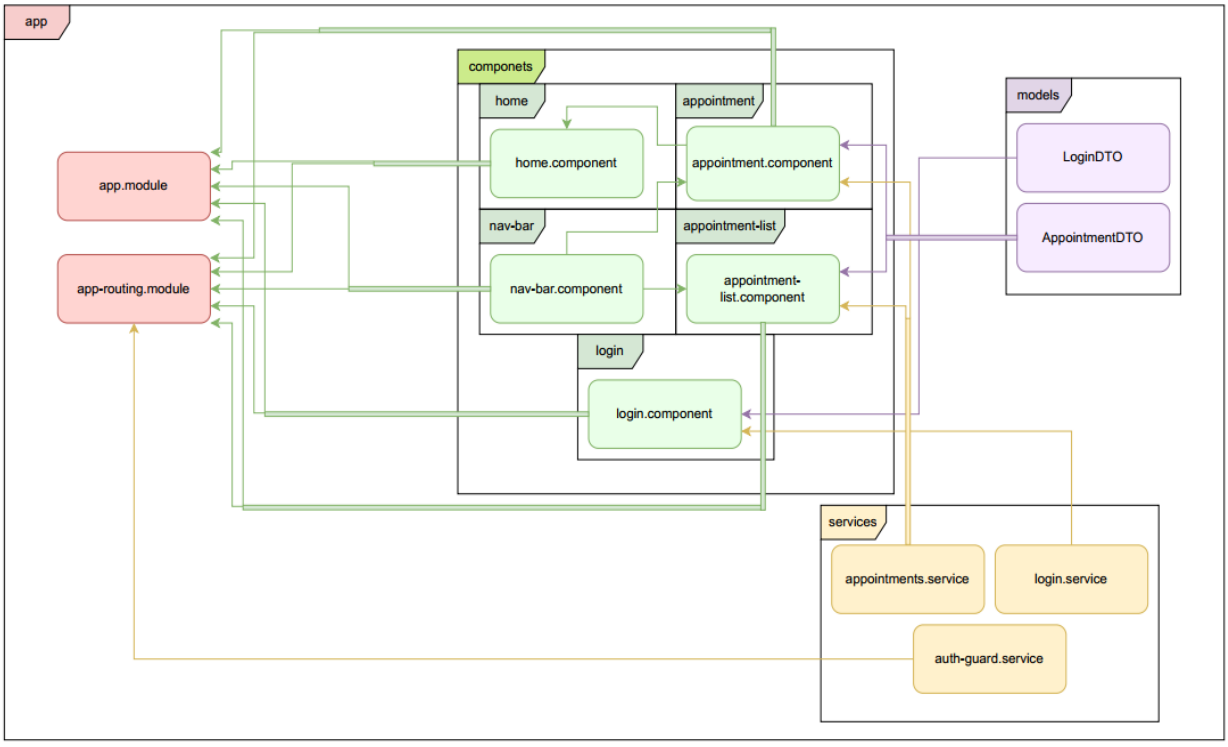

*Figur 15 Webapplikation - Wireframe over check bookings.*

Mathias Wriedt Kamp

Marius Møller

## Angular webapplikation – component diagram

Bilag\diagrammer\Component diagram\Angular component diagram.pdf



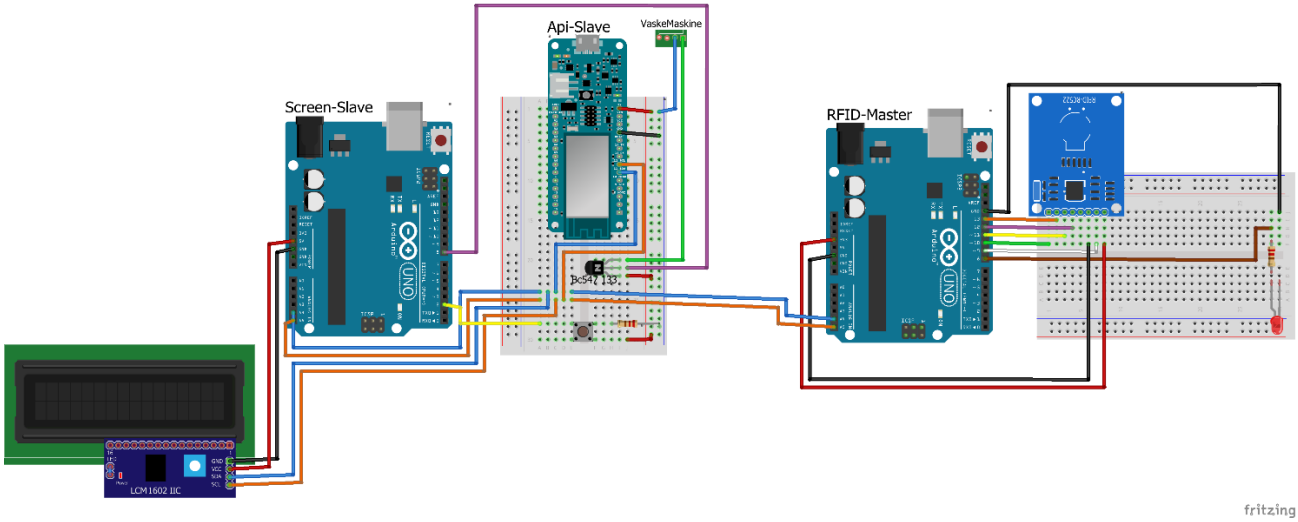

*Figur 16 Component diagram over Angular webapplikation.*

Mathias Wriedt Kamp

Marius Møller

## Arduino – Circuit diagram

Bilag\System documentation\Arduino\Circuit diagram\Arduino circuit dirgram.png



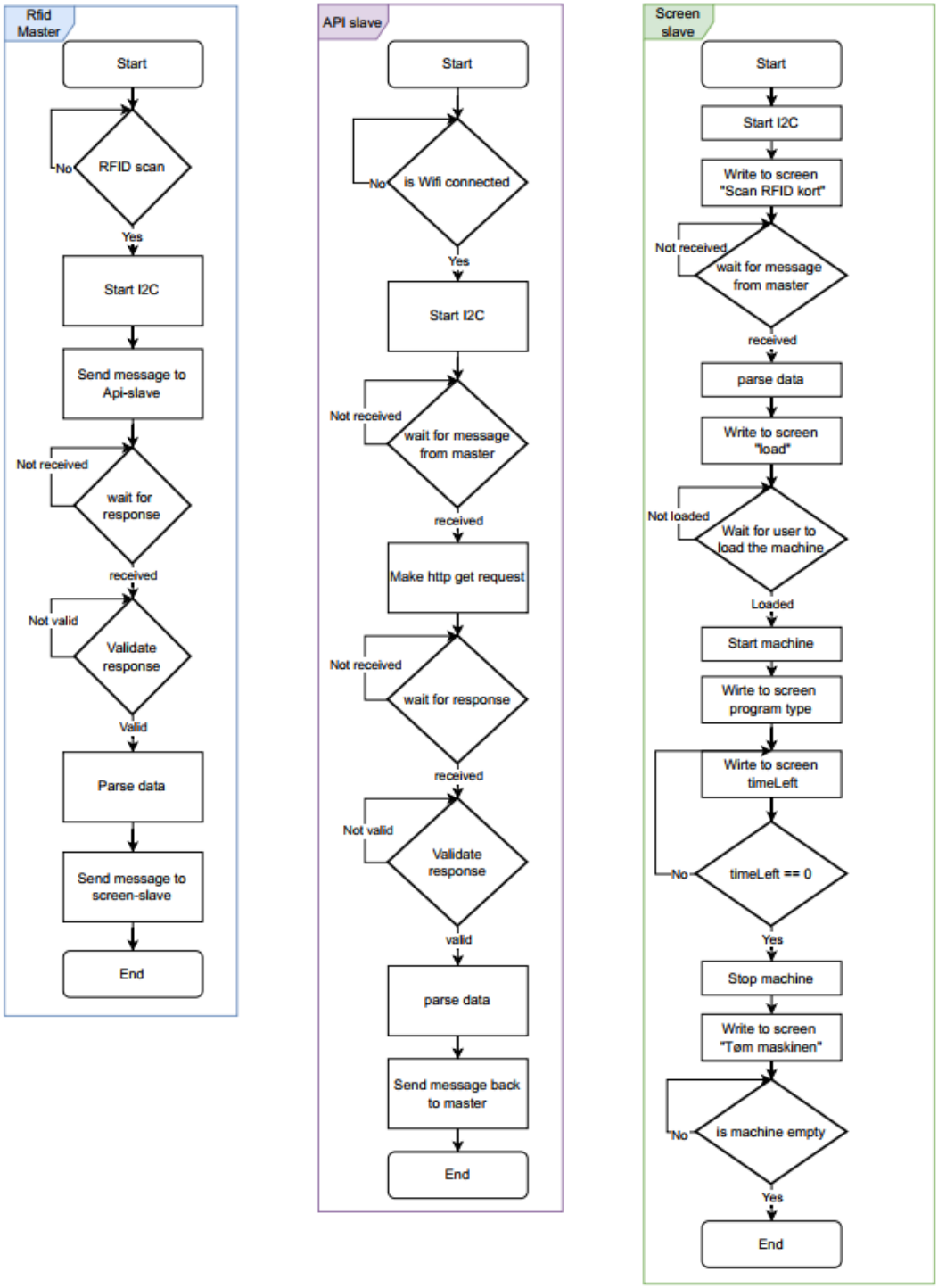

*Figur 17 Arduino - Circuit diagram*

Mathias Wriedt Kamp

Marius Møller

## Arduino – Flow diagram

[Bilag\diagrammer\Flow diagram\arduino flow diagram.pdf]



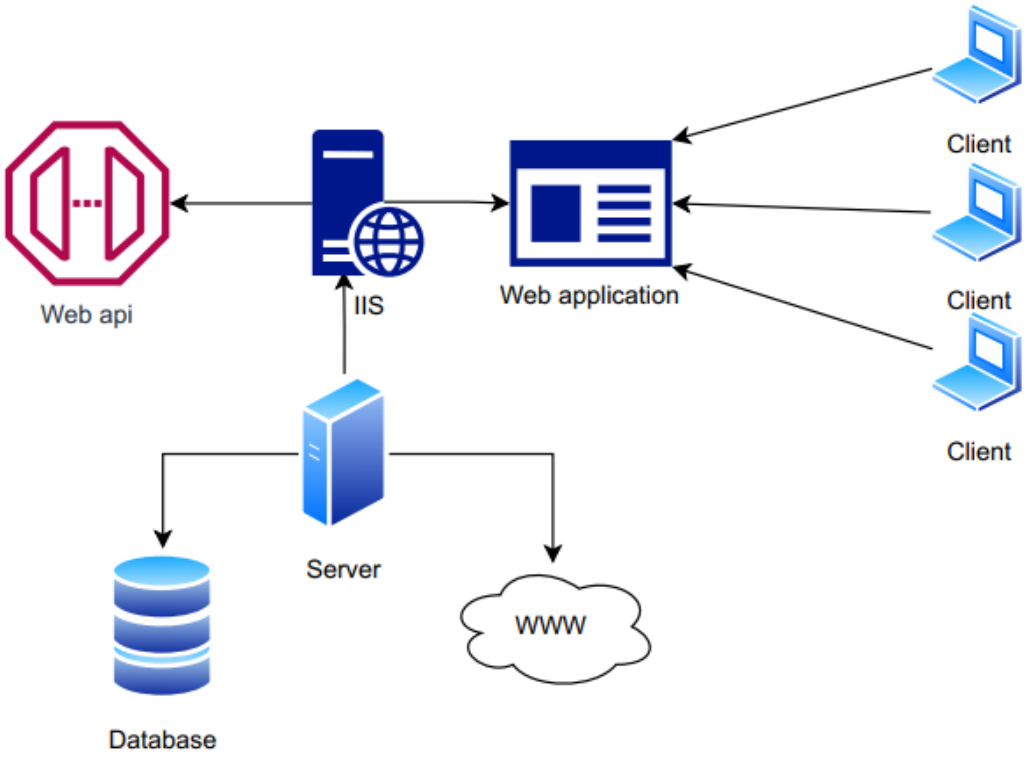

*Figur 18 Arduino - Flow diagram*

Mathias Wriedt Kamp

Marius Møller

# Topologier

[Bilag\System documentation\Topology\Topology.pdf]




*Figur 19 Topologi i systemet*

Mathias Wriedt Kamp

Marius Møller

# Testrapport

Til denne testrapport har vi haft en testperson til at udføre UAT på vores use-cases. Resultaterne er beskrevet i denne rapport sammen med use-casenes unittests.

## Login use-case id 1

### Introduktion

formålet med denne test er at validere, at login-funktionen fungerer korrekt og sikre adgang til applikationen hvis brugeren indtaster korrekt brugernavn og password.
Til at teste dette har vi fået en testperson til at udføre en UAT.

### Testresultater

| UAT: Login (use-case id 1)<br>Test person navn: testperson1 | | | |
|---|---|---|---|
| Accept kriterie | Resultat | | Kommentar |
| | Ja | Nej | |
| Indtast brugernavn og password (tester, password) og tryk på login | | | |
| Hvis det lykkedes at logge ind, forventes det at du bliver omdirigeret til bookingskærmen. Bliver du det? | X | | |
| Tryk på ”log ud” | | | |
| Hvis det lykkedes at logge ud, forventes det at du bliver omdirigeret til login siden. Bliver du det? | X | | Når man holder musen over ”log out” så er den en cursor |
| Indtast brugernavn og password (tester1, password) og tryk på login og tryk på login | | | |
| Hvis brugeren ikke eksisterer, forventes det at du får en fejlbesked med teksten ”Incorrect username or password”. Får du det? | X | | |

*Table 1 UAT over use-case id 1*

Mathias Wriedt Kamp

Marius Møller

## Unittest – use case 1

Denne unittest laver samme øvelse som i figur 20 & 21. Den logger ind med brugernavnet og passwordet (tester, password) hvis brugeren og passwordet er korrekt bliver der returneret en JWT (JSON web Token) som indikerer at brugeren er logget ind.

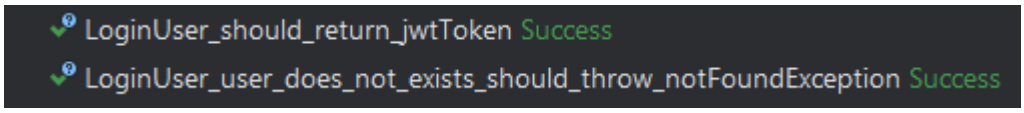
## Unittest login med korrekt brugernavn og password

```
[Fact]
public async Task LoginUser_should_return_jwtToken()
{
    var loginUserRequest = new LoginUserRequest("tester", "password");

    await TestDataInserter.InsertTestUser(LoginService);

    var actual = await LoginService.LoginUser(loginUserRequest);

    Assert.StartsWith("Bearer ", actual);
    Assert.True(actual.StartsWith("Bearer ") && actual.Length > 10);
}
```

## Unittest login med forkert brugernavn

```
[Fact]
public async Task
LoginUser_user_does_not_exists_should_throw_notFoundException()
{
    // Arrange
    var loginRequest = new LoginUserRequest("tester1", "password");

    // Actual
    var actual = await Assert.ThrowsAsync<NotFoundException>(async () =>
await LoginService.LoginUser(loginRequest));

    // Expected
    var expected = new NotFoundException("Username or password is
incorrect");

    // Assert
    Assert.Equal(expected.Message, actual.Message);
}
```

## Unittest resultater




*Figur 20 Unittest login bruger*

## Konklusion

Brugeren har indtastet brugernavn og password (tester, password) og har klikket på log-inknappen. Derefter bliver brugeren omdirigeret til startskærmen og kan benytte applikationen.

Det ses også på unittesten af funktionen login at den virker som den skal. Testen opfylder kravene for test-case 1.

## Opret booking use-case id 2

### Introduktion

formålet med denne test er at validere, at opret booking funktionen fungerer korrekt og at brugeren kan oprette en booking og at der vises de rigtige fejlbeskeder hvis der er mangler til at kunne lave en booking.

### Testresultater

| UAT: Opret booking (use-case id 2)<br>Test person navn: testperson1 | | | |
|---|---|---|---|
| Accept kriterie | Resultat | | Kommentar |
| | Ja | Nej | |
| Indtast brugernavn og password (tester, password) og tryk på login | | | |
| Hvis det lykkedes at logge ind, forventes det at du bliver omdirigeret til bookingskærmen. Bliver du det? | X | | |
| Udvælg en maskine | | | |
| Hvis der er nogle programmer tilknyttet til den valgte maskine, forventes det at der bliver vist hvilke programmer der kan vælges. Gør der det? | X | | |
| Udvælg et program | | | |
| Hvis det er valgt et program, forventes det at systemet viser hvilke ledige tider der kan vælges. Gør der det? | X | | |
| Udvælg en tid og tryk på ”opret booking” | | | |
| Hvis den valgte tid stadigvæk er tilgængelig, forventes det at systemet giver en besked på at bookingen er oprettet. Får du den besked? | X | | |
| Lav bookinger på de resterende ledige tider | | | |
| Hvis der ikke er nogle ledige tider tilbage, forventes det at systemet giver besked på at der ikke er nogle ledige tider. Får du den besked? | X | | |
| Udvælg en maskine uden program (giv besked til os ved dette step så vi kan slette programmerne tilknyttet til maskinen) | | | |
| Hvis der er valgt en maskine uden tilknyttede programmer, forventes det at systemet giver en fejlbesked om at der ikke er nogle programmer. Får du den? | X | | |

*Table 2 UAT opret booking (use-case id 2)*

Mathias Wriedt Kamp

Marius Møller

# Unittest opret booking

## Unittest opret booking uden gyldigt program

```
[Fact]
public async Task
CreateNewBooking_with_No_program_should_throw_notFoundException()
{
    // Arrange
    // programId 230 does not exists in the database
    var bookingRequest = new CreateNewBookingRequest
    {
        Username = "tester",
        MachineManufacturer = "Tester",
        ModelName = "Test model",
        ProgramId = 230
    };
    // insert a machine
    await TestDataInserter.InsertTestMachine(DatabaseContext);
    // insert programs
    await TestDataInserter.InsertTestProgram(DatabaseContext);

    // map machine and programs together
    await TestDataInserter.InsertTestMachineProgram(DatabaseContext);
    // insert a department
    await TestDataInserter.InsertTestDepartment(DatabaseContext);
    // insert times to be booked
    await TestDataInserter.InsertAvailableBookingTimes(DatabaseContext);
    // insert electricityPrices
    await TestDataInserter.InsertElectricityPrices(DatabaseContext);
    // clear taken bookings
    await
TestDataInserter.UpdateAllBookingTimesToBeAvailableInTestDepartment(Databas
eContext);
    // get list of booking times
    var availableBookingTimes = await
FrontendService.GetAvailableBookingTimes(bookingRequest.Username);
    // take the first time
    bookingRequest.StartTime = availableBookingTimes.First().StartTime;

    // create booking with programId = 230 which does not exists in the
database
    var actual =
        await Assert.ThrowsAsync<NotFoundException>(() =>
FrontendService.CreateNewBooking(bookingRequest));

    // Assert
    Assert.Contains("program is not presented", actual.Message);
}
```

Mathias Wriedt Kamp

Marius Møller

## Unittest ingen ledige tider

```
[Fact]
public async Task
CreateNewBooking_with_No_AvailableBookingTimes_should_throw_notFoundExcepti
on()
{
    // Arrange
    var bookingRequest = TestDataCreator.GetTestBookingRequest();

    // Available booking times is determined from the department that the
user is in. therefore we set a username that is not valid
    bookingRequest.Username = "does not exists";

    var actual =
        await Assert.ThrowsAsync<NotFoundException>(() =>
FrontendService.CreateNewBooking(bookingRequest));

    Assert.Contains("no available booking times", actual.Message);
}
```

Mathias Wriedt Kamp

Marius Møller

## Unittest alt er godt booking oprettes

```
[Fact]
public async Task CreateNewBooking_Should_Return_BookingDTO()
{
    // Arrange
    var bookingRequest = TestDataCreator.GetTestBookingRequest();

    // insert available booking times
    await TestDataInserter.InsertAvailableBookingTimes(DatabaseContext);

    // clear all taken booking times
    await
TestDataInserter.UpdateAllBookingTimesToBeAvailableInTestDepartment(Databas
eContext);
    // get list of available booking times
    var availableBookingTimes = await
FrontendService.GetAvailableBookingTimes(bookingRequest.Username);
    // take the first time
    var firstBookingTime = availableBookingTimes.First();

    // update the bookings startTime to be the first available booking
time. (simulate that the booking is created to be started at the same time
as the available one)
    bookingRequest.StartTime = firstBookingTime.StartTime;

    // insert test machine
    await TestDataInserter.InsertTestMachine(DatabaseContext);
    // insert programs
    await TestDataInserter.InsertTestProgram(DatabaseContext);
    // map machines and programs
    await TestDataInserter.InsertTestMachineProgram(DatabaseContext);
    // insert a test department
    await TestDataInserter.InsertTestDepartment(DatabaseContext);

    // Actual should be a booking
    var actual = await FrontendService.CreateNewBooking(bookingRequest);

    // Assert
    Assert.True(actual.BookingId > 0);
}
```

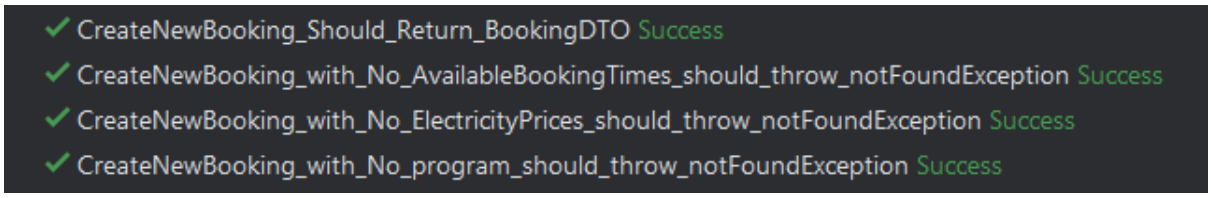
Mathias Wriedt Kamp

Marius Møller

## Unittest ingen elektricitets pris til rådighed

```
[Fact]
public async Task
CreateNewBooking_with_No_ElectricityPrices_should_throw_notFoundException()
{
    // Arrange
    var bookingRequest = TestDataCreator.GetTestBookingRequest();

    bookingRequest.Username = "tester";
    // set the booking to be started at today's date at 18 o'clock
    bookingRequest.StartTime =
        new DateTime(DateTime.Now.Date.Year, DateTime.Now.Date.Month,
DateTime.Now.Date.Day, 18, 0, 0);

    // get list of electricityPrices
    var prices = await
DatabaseContext.GetElectricityPrices(bookingRequest.Username);

    // find the electricityPrice that starts on today's date 18 o'clock
    var priceToDeleteFromDb = prices.FirstOrDefault(price =>
        price.TimeStart.Date == bookingRequest.StartTime.Date &&
        price.TimeStart.Hour == bookingRequest.StartTime.Hour);

    // insert available booking times
    await TestDataInserter.InsertAvailableBookingTimes(DatabaseContext);

    // delete the electricityPrice that the booking is supposed to use
    await TestDataInserter.DeleteElectricityPrice(DatabaseContext,
priceToDeleteFromDb);

    // Actual should throw a notFoundException, telling that the
electricityPrice is not presented
    var actual =
        await Assert.ThrowsAsync<NotFoundException>(() =>
FrontendService.CreateNewBooking(bookingRequest));

    Assert.Contains("electricityPrice is not presented", actual.Message);
}
```

## Unittest resultater




*Figur 21 unittest resultater over opret booking*

## Konklusion

Som det kan ses i UAT og unittests så virker opret booking efter hensigten. Der bliver givet fejl hvis der ikke er et program til maskinen, der bliver givet fejl hvis der ikke er flere ledige tider.

Mathias Wriedt Kamp

Marius Møller

## Skan RFID-kort og start maskine

### Introduktion

Formålet med denne test er at validere at ”Skan RFID-kort” funktionen virker som den skal og at brugeren kan skanne sit RFID-kort for at kunne starte maskinen.

### Testresultater

| UAT: Skan RFID-kort og start maskine (use-case id 3) | | | |
|---|---|---|---|
| Accept kriterie | Resultat | | Kommentar |
| | Ja | Nej | |
| Skan RFID-kort (kort nummer 1) på RFID-kortlæseren | | | |
| Hvis RIFD-kortet er gyldigt, og brugeren tilknyttet til kortet har oprettet en booking som skal starte indenfor skanningstidspunktet, forventes det at systemet skriver ”load” på displayet. Gør det det? | X | | |
| Tryk på den blå knap for at loade maskinen med tøj, og starte maskinen | | | |
| Hvis maskinen er loaded og der er trykket på den blå knap, forventes det at systemet starter maskinen. Gør den det? | X | | |
| Når maskinen er færdig, står der ”Tom maskinen” på displayet, tryk på den blå knap igen for at indikere at maskinen er tømt. | | | |
| Hvis maskinen er færdig og det er trykket på den blå knap, forventes det at systemet skriver ”Skan RFID-kort” på displayet igen. Gør den det? | X | | |

*Figur 22 UAT use-case Skan RFID-kort og start maskine*

Mathias Wriedt Kamp

Marius Møller

## Unittest – Skan RFID-kort og start maskine

```
[Fact]
public async Task GetBookingConnectedToRfid_should_return_bookingDTO()
{
    // Arrange
    var testBooking = TestDataCreator.GetTestBooking();

    // Insert available booking times
    await TestDataInserter.InsertAvailableBookingTimes(DatabaseContext);

    // clear all taken times. (as we only have 5 times per day, they all
gets taken during tests)
    await
TestDataInserter.UpdateAllBookingTimesToBeAvailableInTestDepartment(Databas
eContext);
    // get available booking times
    var availableBookingTimes = await
TestDataInserter.GetAvailableBookingTimes(DatabaseContext);

    // take the first time from the list
    var firstAvailableTime = availableBookingTimes.First();

    // update the test bookings start time to be the first available start
time
    testBooking.StartTime = firstAvailableTime.StartTime;
    // update the test bookings end time to be the first available end time
    testBooking.EndTime = firstAvailableTime.EndTime;

    // simulate that the user is going to scan his RFID-card 5 minutes
after the booking is created
    var scannedTime = firstAvailableTime.StartTime.AddMinutes(5);

    // insert RFID card to database
    await TestDataInserter.InsertTestRfidCard(DatabaseContext);
    // get the RFID-card from database
    var testRfidCard = TestDataCreator.GetTestRfidCard();

    // create the test booking
    var bookingCreated = await
TestDataInserter.InsertTestBooking(DatabaseContext, testBooking);

    // set the booking Id on the available time
    firstAvailableTime.bookingId = bookingCreated.BookingId;

    // update database with the available time. Which makes it taken by the
created booking
    await TestDataInserter.UpdateAvailableBookingTimes(DatabaseContext,
firstAvailableTime);

    // get the booking from RFID-card
    var actual = await
WashingMachineService.GetBookingConnectedToRfid(testRfidCard.RfidCardId,
scannedTime);

    // Assert
    Assert.True(actual.BookingId > 1);
    Assert.True(actual.BookingId  == bookingCreated.BookingId);
}
```

Mathias Wriedt Kamp

Marius Møller

## Unittest – Skan RFID kort – program til at skrive på displayet

```
[Fact]
public async Task GetBookingProgram_should_return_ProgramResultDTO()
{
    // Arrange
    var testBooking = TestDataCreator.GetTestBooking();
    // Insert a test booking to database
    var testBookingCreated = await
TestDataInserter.InsertTestBooking(DatabaseContext, testBooking);

    // Actual the booking program
    var actual = await
WashingMachineService.GetBookingProgram(testBookingCreated);

    // Assert
    Assert.True(testBookingCreated.MachineManufacturer ==
actual.MachineManufacturer);
    Assert.True(testBookingCreated.ModelName == actual.ModelName);
    Assert.True(!actual.Equals(default(ProgramResultDTO)));
}
```

## Unittest – Skan RFID kort – RFID kort eksisterer

```
[Fact]
public async Task RfidCardExists_should_return_true()
{
    // Arrange
    var rfidTest = TestDataCreator.GetTestRfidCard();

    // insert a valid RFID-card
    await TestDataInserter.InsertTestRfidCard(DatabaseContext);

    // Actual get the RFID-card
    var actual = await
WashingMachineService.RfidCardExists(rfidTest.RfidCardId);

    // Assert
    Assert.True(actual);
}
```

Mathias Wriedt Kamp

Marius Møller

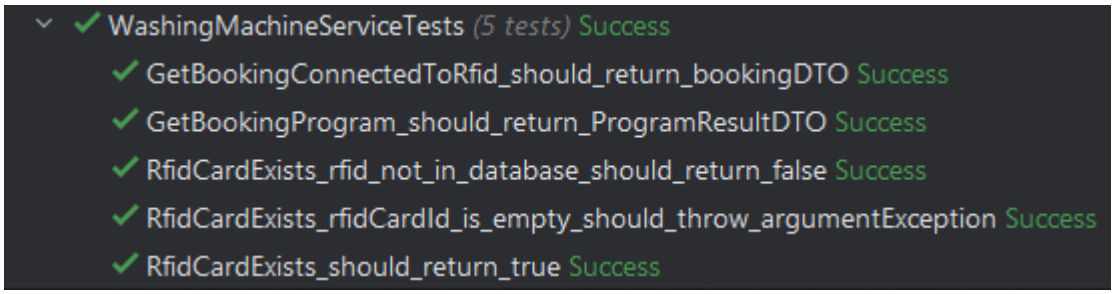
## Unittest – Skan RFID kort – RFID kort eksisterer ikke

```
[Fact]
public async Task RfidCardExists_rfid_not_in_database_should_return_false()
{
    // Arrange
    var rfidTest = TestDataCreator.GetTestRfidCard();
    // RfidCardId that does not exists in the database
    rfidTest.RfidCardId = "does not exist";

    // insert the correct RFID-Card
    await TestDataInserter.InsertTestRfidCard(DatabaseContext);

    // Actual check if the rfidCard exists
    var actual = await
WashingMachineService.RfidCardExists(rfidTest.RfidCardId);

    // Assert
    Assert.False(actual);
}
```

## Unittest – Skan RFID kort – RFIDid er ikke udfyldt

```
[Fact]
public async Task
RfidCardExists_rfidCardId_is_empty_should_throw_argumentException()
{
    // Arrange
    var rfidTest = TestDataCreator.GetTestRfidCard();
    // Empty RfidCardId to simulate that the user didnt parse the RFIDCard
to the method
    rfidTest.RfidCardId = "";

    // insert a valid RfidCard
    await TestDataInserter.InsertTestRfidCard(DatabaseContext);

    // Actual, check if rfidTest exists in the databae
    var actual =
        await Assert.ThrowsAsync<ArgumentException>(async () => await
WashingMachineService.RfidCardExists(rfidTest.RfidCardId));

    // Assert
    Assert.Contains("cannot be null or empty", actual.Message);
}
```

### Unittest resultater




*Figur 23 unittest resultater af Skan RFID kort*

### Konklusion

Som det kan ses i UAT og unittests så virker Skan RFID-kort efter hensigten. Der bliver givet fejl hvis der mangler et RFID-kortId og hvis RFID-kortet er skannet inden for bookingens starttid, så returneres bookingen.

## Bilag

Alle dokumenter, billeder og diagrammer har en reference til det oprindelige dokument og hvor det kan findes.