# Lufthavnens kravspecifikation -> CheckIn & Booking

Case: Hvordan får folk tilmeldt sig til et fly og checker in?

## Opret en plan for håndtering af casen

Hvordan får vi undersøgt hvilke aktører der bliver berørt af systemet. Herunder identificering af primære, sekundære og perifere interessenter.

* Et værktøj til at udføre dette kunne være en interessentanalyse.
  + Så vi kan identificere, hvilke interessenter er gidsler, eksterne interessant og ressourcepersoner.

Hvilke usecases har de forskellige aktører?

* Hertil kan der udarbejdes use case diagrammer eller enkelte fully dressed usecases til at få overblik og nøje information om aktørernes brugsscenarier.

Hvordan afdækker endebrugernes behov?

* Behov / kravene skal afspejle værktøjer som *FURPS+* og *MOSCOW*.
* Functional prototyping ved hjælp af et værktøj som<https://balsamiq.com/>, til formål for at få afdækket funktionelle behov. Her menes publikummet skal være systemets formentlige endebrugere.
* Nødvendige behov (i relation til MOSCOW’s inddeling af krav) vil planlægges at blive afdækket ved første møde; eksempler kunne være et krav på performance.
* Ikke-funktionelle krav og nice-to-have krav kan identificeres gennem dialog, ved fremvisning af tidlige demoer fra sprints.

Er der hardware eller software snitflader der bliver berørt af systemet?

* Møde med repræsentanter fra de afdelinger (grupper) der står til ansvar for de snitflader.

Er der nogle snitflader hvori der forventes noget af det udviklede system, eller er der noget det udviklede system forventer af snitfladerne?

* Det udviklede systems forventes at kende til format, struktur og objekter med diverse forskellige systemer.
* Dette indføres via en **kommunikationskontrakt**. Som laves i konsensus med udviklerne både dette system og de berørte systemer.
  + Dette medfører at udviklingsprocessen skal afspejle en kontraktbaseret udvikling.

Findes der eksisterende systemer / hyldevare man kan gøre brug af?

* Drag inspiration fra potentielle konkurrenter til udarbejdelse af dette systems backend- og frontend software.

## Forstå, hvad det handler om (hvem er brugerne og hvordan bruger de det)

**Hvem er brugerne?**

Gennem ekstensiv og intensiv analyse, er det blevet konkluderet at dette systems endebrugere kan identificeres som “*flyrejsende*”.

**Hvordan bruger de det?**

Igennem benyttelsen af paperprototyping (<https://balsamiq.com/>), har vi dannet en fælles forståelse af endebrugernes forventning til systemet. Derudover er der blevet draget teknisk inspiration fra potentielle konkurrenter.

På basis af ovenstående, er der blevet konkluderet at systemet har behov for en **website applikation**. Der kan udstille grænseflader til *flyrejseoversigt*, *booking* og *checkin*.

På basis af den user journey der blev udviklet i tidligere analyse er det blevet konkluderet at der er ét systematisk flow for endebrugeren. Som kan udtrykkes gennem en række (ikke fully-dressed) use cases. Hvori deres relation til hinanden udtrykkes endvidere med “*preconditions*”.

### Use Case UC1: Booking af en flyrejse

**Scope:** Booking

**Primær Aktøre:** Flyrejsende

**Stakeholders and interests:**

* FIDS: Vil gerne modtage digital information ift. de givne flyrejser som bookingsystemet skal håndtere.

**Forudsætning(er):** Der er flyrejser, som aktørerne kan tilmelde sig til

**Primære Succes Scenarie:**

* Aktøren går ind på flyrejseoversigten
* Systemet henter data fra FIDS og udskriver det til Aktøren.
* Aktøren vælger en flyrejse
* Systemet videredirigerer aktøren til bookingsiden for den givne flyrejse.
* Aktøren udfylder information (Kuffertvægt, Navn, Adresse, Email, tlf. og By) og booker rejsen.
* Systemet udskriver billetinformationen til aktøren.

### Use Case UC2: Afbestilling af flyrejse

**Scope:** Booking

**Primær Aktøre:** Flyrejsende

**Forudsætning(er):** Aktøren har allerede gennemført UC1, men ikke UC3.

**Primære Succes Scenarie:**

* Aktøren går ind på afbestillingssiden.
* Aktøren udfylder information (Navn, Adresse, Email, Tlf., By og BookingId)
  + Informationerne: Navn, Adresse, Email, Tlf. og By skal stemme overens med dem der blev udfyldt da bookingen blev lavet.
* Aktøren godkender og systemet udskriver bekræftelse til vedkommende.

### Use Case UC3: CheckIn af en flyrejse

**Scope:** Checkin

**Primær Aktøre:** Den rejsende

**Forudsætning(er):** Kunden har booket en flyrejse

**Primære Succes Scenarie:**

* Kunden går ind på CheckIn siden og indtaster sit bookingsId
* Systemet checker kunden ind til den givne rejse.
* System udleverer et boardingpass med følgende information (FlyId, FlyafrejseDato, FlyDestination, KundensFuldeNavn, CheckInId, FlyAfrejse, SædeNummer) til kunden.

## Identificere de aspekter der skal afklares

* Hvordan skal systemet give kunderne besked omkring booking / checkin? (email - SMS?)
* Skal der login ift. afbestilling?
* Skal man kunne logge ind og se ens bookinger?
* Hvilke service skal der bruges til afregning?
* Hvordan skal GDPR overholdes?

## Identificere de største udfordringer

**Identificering af kompleksiteter & usikkerheder**

* Integration med betalingsudbydere
* Integrationer til systemer som dette udviklingsteam ikke har direkte adgang til (blackbox)
* Udviklingsteamet er uerfaren i at udvikle lufthavnssystemer.
* Kommunikation mellem diverse teams.
* Usikkerhed ift. indførsel af læringsmål i henhold til de forskellige fag.

## Identificere de områder, der skal arbejdes med

* Systemintegration
  + Asynkron system kommunikation i form af message broker infrastruktur (RabbitMQ)
* Database
  + Forskellige databaser
  + Replication strategy
  + Udføre normalisering
  + Gør brug af Views, Triggers, Functions, Stored procedures til at optimere database queries. Så der kun hentes relevante data ud af databasen.
* Store systemer
  + Microservice arkitektur implementation
    - Korrekt REST api implementering

Ift. Flyrejse

CheckIn og Booking

## Opret en liste med spørgsmål til projektlederen (AKA. underviser)

* Hvordan finder vi ud af hvilke message patterns vi skal bruge hvor?
* Hvilke ikke-funktionelle krav er der?
* Hvordan vil I verificere de funktionelle krav?
* Hvilke former for testing skal der benyttes?
* Dokumentation…
  + Hvor meget dokumentation skal der foretages?

Til udarbejdelse af backlog kan følgende checkliste bruges som udgangspunkt:

## Software Kravspecifikations checkliste:

* Clear
* Correct
* Consistent
* Coherent
* Comprehensible
* Modifiable
* Verifiable
* Prioritized
* Unambiguous
* Traceable
* Credible Source
* Quantifiable