SmartFridge

# Projektformulering

**Fridge app**



**Køleskab**

**Lokal DB**

**Ekstern DB**

**Computer**

**Smartphone**

**Tablet**

**Web app**

**Figur 1** Skitsering af systemet SmartFridge

Formålet med dette projekt er at udvikle et system, som tillader registrering af varer i køleskab via en grafisk brugergrænseflade på en lokal skærm. Lagring af disse oplysninger sker i en lokal database, der synkroniseres med en ekstern database, som kan tilgås via et web-interface.

Systemet, som er relativt simpelt, byder desuden på rig mulighed for udvidelse, i form af inkorporering af ekstra funktioner. Eksempelvis oplysning om ernæringsværdier, opskrifter baseret på eksisterende ingredienser og eksisterende/fremtidige tilbud på manglende varer.

Systemet skal ses som en ekstern tilføjelse til eksisterende køleskabe, og ikke i første omgang som en indbygget feature i nye modeller. Dette sikrer at systemet kan tilbydes til en bredere målgruppe end køberne af ”high end”-køleskabe.

Visionen er at tilbyde brugeren et hurtigt og effektivt overblik over køleskabets indhold, til lettelse i en hverdag, hvor man ikke altid har en opdateret indkøbsliste inden for rækkevidde.

Brugeren kan være privatpersoner såvel som industrielle køkkener og catering-virksomheder, som ønsker overblik over indholdet i køleskabet.

# Projektafgrænsning

Udførslen af projektet er med fokus på softwareudvikling, så problemer som ikke er software-relaterede, vil ikke blive forsøgt løst. Dette inkluderer:

* Strømforsyning til enheden som kører Fridge app’en.
  + Kabling
  + Evt. batterilevetid
* Udvikling af enhed til kørsel af Fridge app’en.
  + Lenovo Yoga 2 Pro[[1]](#footnote-1) benyttes som platform for Fridge app.

# Systembeskrivelse

Systemet skal kunne assistere Bruger, ved tilbyde opretholdelse af en liste over hvilke varer, der er i køleskabet. Der kan tilføjes og fjernes varer i takt med indkøb og forbrug, samt opretholdes en liste over ønskede varer i køleskabet. Systemet vil desuden bestå af en lokal og en ekstern database, hvori de oprettede lister gemmes, så listerne også kan tilgås fra en web-applikation.

Når Bruger starter systemet, vil den lokale database synkronisere med den eksterne database, for at sikre at begge databaser er opdateret og stemmer overens. Herefter vil den lokale og den eksterne database synkronisere jævnligt, så længe systemet er i brug.

Når systemet er startet, vil det være muligt for Bruger at tilføje varer til listen over varer i køleskabet, på indkøbslisten eller i standardbeholdningen, hvorpå Bruger får eksisterende varetyper som forslag, eller kan tilføje nye varetyper selv.

Det er muligt at se varerne på de forskellige lister. I denne forbindelse vil det være muligt at redigere mængden af varerne, i takt med at de fjernes.

Systemet vil påminde Bruger når køleskabet indeholder en mindre kvantitet af en vare, end der er angivet på listen over standard-varer, og det vil være muligt for Bruger at se hvad der mangler i køleskabet, samt en indkøbsliste, der som minimum indeholder de manglende varer.

# Kravspecifikation

## **Figur 2** Aktørdiagram over SmartFridge Aktører

**Figur 2** viser use case-diagrammet med alle aktører og deres forhold til systemet SmartFridge.

## Aktørbeskrivelse

### Bruger

|  |  |
| --- | --- |
| **Aktørnavn:** | Bruger |
| **Alternativt navn:** |  |
| **Type:** | Primær |
| **Beskrivelse:** | Bruger er den primære bruger af systemet. Bruger interagerer med systemet gennem en brugergrænseflade. |

### Ekstern Database

|  |  |
| --- | --- |
| **Aktørnavn:** | Ekstern Database |
| **Alternativt navn:** | Online Database |
| **Type:** | Sekundær |
| **Beskrivelse:** | Den Eksterne Database er en database, der ligger på en webserver. Den har kun forbindelse til det lokale system når der synkroniseres (se Use Case 7), og fungerer som direkte database for web-app’en. Databasen får tilføjet, fjernet og redigeret data ved synkronisering, eller manipulationen kan foregå fra systemets web app. |

## Termliste

### Fridge app

Fridge app er den lokale del af systemet, og dækker over den lokale brugergrænseflade, samt den lokale database.

### Web app

Web app er den eksterne del af systemet, og dækker over websitet.

### Kernefunktionalitet

Begrebet dækker over de mest basale funktionaliteter, som gør at systemet er sammenhængende og brugbart. Disse funktionaliteter er repræsenteret ved use cases 1-5.

### Standard-varer

Begrebet dækker over en række varer, som Bruger altid ønsker at have i sin varebeholdning. Varerne tilføjes til en liste på lige fod med andre lister i systemet.

Kaldes også ”standard-beholdning”.

## Use cases

For alle use cases gælder det, at i enhver undermenu, findes der altid mulighed for at vende tilbage til hovedmenuen.



**Figur 3** Use case-diagram over SmartFridge

### UC 1: Se varer

Use case 1 har til formål at lade brugeren får overblik over enten varebeholdningen, indkøbslisten eller listen over standard-varer.

|  |  |
| --- | --- |
| **Use case nr./navn** | **1 – Se varer** |
| **Mål** | At få frembragt varerne på en liste. |
| **Initialisering** | Bruger |
| **Aktører** | * Bruger |
| **Samtidige forekomster** | 1 |
| **Forudsætninger** | UC1: Opstart af applikation |
| **Resultat** | Indholdet på den valgte liste vises på skærmen. |
| **Hovedscenarie** | 1. Bruger trykker på knappen ”I køleskab”.   **[Alternativt flow 1.a: Bruger trykker på ”Indkøbsliste”]**  **[Alternativt flow 1.b: Bruger trykker på ”Standard-varer”]**   1. En liste over nuværende varer i køleskabet, samt mængden af disse, vises på skærmen.   **[Undtagelse 2.a: Listen over varer i køleskabet er tom]** |
| **Alternativt flow** | **1.a Bruger trykker på ”Indkøbsliste”.**  1.a.1 En liste over nuværende varer på indkøbslisten, samt mængden af disse, vises på skærmen.  **[Undtagelse 1.a.1.a: Indkøbslisten er tom]**  **1.b Bruger trykker på ”Standard-varer”.**  1.b.1 En liste over nuværende standard-varer, samt mængden af disse, vises på skærmen.  **[Undtagelse 1.b.1.a: Listen over standard-varer er tom]** |
| **Undtagelser** | **2.a Listen over varer i køleskabet er tom.**  2.a.1 En besked om at listen over varer i køleskabet er tom, vises på skærmen, og UC afsluttes.  **1.a.1.a Indkøbslisten er tom.**  1.a.1.a.1 En besked om at indkøbslisten er tom, vises på skærmen, og UC afsluttes.  **1.b.1.a Listen over standard-varer er tom.**  1.b.1.a.1 En besked om at listen over standard-varer er tom, vises på skærmen, og UC afsluttes. |

### UC 2: Tilføj vare

Use case 2 har til formål at lade brugeren tilføje varer til enten varebeholdningen, indkøbslisten eller listen over standard-varer.

Felterne ”Antal” og ”Volumen/vægt” udfyldes automatisk med standardværdier, når varetypen er valgt, men kan ændres efter behov.

|  |  |
| --- | --- |
| **Use case nr./navn** | **2 - Tilføj vare** |
| **Mål** | At tilføje en vare til en liste. |
| **Initialisering** | Bruger |
| **Aktører** | * Bruger * GUI |
| **Samtidige forekomster** | 1 |
| **Forudsætninger** | UC1: Se varer |
| **Resultat** | Varen er tilføjet til en liste. |
| **Hovedscenarie** | 1. Bruger trykker på ”Tilføj”. 2. Varetype vælges.   **[Undtagelse 2.a: Varetypen eksisterer ikke]**   1. Antal vælges. 2. Volumen/vægt vælges. 3. Bruger trykker på ’Tilføj og afslut,’ og varen tilføjes til listen.   **[Alternativt flow 5.a: Bruger trykker på "Tilføj"]**   1. Der returneres til listen. |
| **Alternativt flow** | **5.a Bruger trykker på "Tilføj".**  5.a.1 Varen gemmes, og der fortsættes fra punkt 2. |
| **Undtagelser** | **2.a Varetypen eksisterer ikke.**  2.a.1 Bruger indtaster den ønskede vare, og fortsætter fra punkt 3. |

### UC 3: Rediger vare

Use case 3 har til formål at lade brugeren redigere varer på enten varebeholdningen, indkøbslisten eller listen over standard-varer.

|  |  |
| --- | --- |
| **Use case nr./navn** | **3 - Rediger vare** |
| **Mål** | At redigere en vare på en den nuværende liste. |
| **Initialisering** | Bruger |
| **Aktører** | * Bruger * GUI |
| **Samtidige forekomster** | 1 |
| **Forudsætninger** | UC1: Se varer |
| **Resultat** | Varen er blevet redigeret. |
| **Hovedscenarie** | 1. Bruger trykker på ”Rediger”. 2. Bruger retter vareinformation.   **[Alternativt flow 2.a: Bruger ændrer Varetype] [Alternativt flow 2.b: Bruger ændrer Antal] [Alternativt flow 2.c: Bruger ændrer Volumen/Vægt] [Alternativt flow 2.d: Bruger ændrer Enhed] [Alternativt flow 2.e: Bruger ændrer intet]**   1. Bruger trykker på ”Gem” og ændringerne gemmes i varen.   **[Alternativt flow 3.a: Bruger trykker på "Annuller"]**   1. Der returneres til listen. |
| **Alternativt flow** | **2.a Bruger ændrer Varetype** 2.a.1 Bruger ændrer varens antal, og der returneres til punkt 2. **2.b Bruger ændrer Antal** 2.b.1 Bruger ændrer varens antal, og der returneres til punkt 2. **2.c Bruger ændrer Volumen/Vægt** 2.c.1 Bruger ændre varens volumen/vægt, og der returneres til punkt 2. **2.d Bruger ændrer Enhed** 2.d.1 Bruger ændre varens enhed, og der returneres til punkt 2. **2.e Bruger ændrer intet** 2.e.1 Bruger ændrer ikke noget, og der fortsættes fra punkt 3. **3.a Bruger trykker på "Annuller"** 3.a.1 Ændringer gemmes ikke, og der fortsættes fra punkt 4. |

### UC 4: Rediger vare

Use case 4 har til formål at lade brugeren fjerne varer på enten varebeholdningen, indkøbslisten eller listen over standard-varer.

|  |  |
| --- | --- |
| **Use case nr./navn** | **4 – Fjern vare** |
| **Mål** | At fjerne en vare fra en beholdning |
| **Initialisering** | Bruger |
| **Aktører** | * Bruger, GUI, Database |
| **Samtidige forekomster** | 1 |
| **Forudsætninger** | UC1: Se varer |
| **Resultat** | Varen er fjernet fra beholdning |
| **Hovedscenarie** | 1. Bruger trykker på ”Fjern”-Ikonet ud for en eksisterende vare 2. Varen fjernes fra GUI og database |

## Kommende Use Cases

### Kernefunktionalitet

#### Synkroniser til ekstern database

Applikationens lokale database synkroniseres med en ekstern database. Synkroniseringen initieres ved både systemstart og systemafslutning, og foretages desuden løbende under drift.

I applikationens hovedmenu findes en knap, som kan initiere en synkronisering mellem databaserne. Denne knap indikerer også hvorvidt der er foretaget ændringer på den lokale database, som endnu ikke er synkroniseret med den eksterne.

#### Fjern vare

**Prækondition:** Use case 1 (Se varer).

Ved siden af varen, findes en knap, som Bruger trykker på, og varen forsvinder fra listen. Bruger trykker på knappen for alle varer som ønskes fjernet.

Hvis Bruger, i stedet for ”I køleskab”, vælger en af de to andre knapper, vil samme type liste fremkomme, med de samme muligheder. Fjernes en vare, som forekommer på standard-listen, fra indkøbslisten, vil forskellen mellem standard-listen og varebeholdningen stadig figurere. Ønskes varer fra standard-listen fjernet, fjernes disse fra listen ”Standard-varer”.

### Udvidelser

Følgende udvidelser vil senere blive opdelt efter MoSCoW[[2]](#footnote-2)-metoden. De vil blive implementeret efter prioritet, efter systemets kernefunktionaliteter er implementeret.

#### Sidste type vare fjernes

Når Bruger fjerner den sidste type af en vare, kommer applikationen med en notifikation, for at gøre opmærksom på dette.

#### Gammel vare fjernes

Hvis holdbarhedsdatoen er overskredet ift. den dato som Bruger har angivet for en vare, notificeres Bruger. Notifikationen fjernes når Bruger indikerer til systemet at varen er fjernet, eller holdbarhedsdatoen på varen er ændret til en fremtidig dato.

#### Tilføj opbevaringssted

Bruger vælger tilføj opbevaringssted, og giver denne et navn. Bruger kan nu vælge dette sted som opbevaringsplads, når en ny vare tilføjes eller redigeres.

#### Log in

Når Bruger vil tilgå web-app'en, skal der først logges ind. Når bruger er logget ind, kan egen del af den eksterne database tilgås, og de sædvanlige funktioner vil være tilgængelige.

#### Importer indkøbsliste

Når Bruger har handlet ind, kan alle varer på indskøbslisten med ét tryk overføres til varebeholdningen.

#### Scan vare

En stregkodescanner tilsluttes systemet, og varer kan scannes. Varerne tilføjes til en valgfri varebeholdningsliste.

#### Vis ernæringsværdier

Bruger kan få oplyst ernæringsværdier for de enkelte varer.

#### Find opskrift

Bruger kan finde opskrifter baseret på de tilgængelige varer. Applikationen kan ud fra den valgte opskrift danne en indkøbsliste, med evt. manglende varer.

#### Valg af tema

Bruger får mulighed for at skifte grafisk tema på applikationen.

#### Juster temperaturalarm

Et termometer, som kan kommunikere med Fridge app’en, lægges i køleskabet, og Bruger sætter en max.- og en min.-temperatur. Kommer temperaturen uden for de satte værdier, advares Bruger.

#### Find tilbud

Bruger vælger ”Find tilbud” på den færdige indkøbsliste. Applikationen finder nu selv de supermarkeder hvor der er tilbud på de varer, der er på indkøbslisten. De butikker hvor alle nødvendige varer sammenlagt kan købes billigst, kommer først.

## Ikke-funktionelle krav

Alle krav er specificeret ud fra Lenovo Yoga 2 Pro[[3]](#footnote-3) som platform.

1. **System**
   1. Mulighederne i Fridge app og web app skal være de samme.
2. **Databaser**
   1. Den lokale og den eksterne database skal automatisk synkroniseres hvert 10. minut.
   2. I tilfælde af konflikter ved synkronisering, overskriver de nyest tilføjede data de ældste.
3. **Fridge app**
   1. Ved opstart og nedluk, forsøges synkronisering mellem den lokale og den eksterne database.
   2. Ændringer af data lagres straks i den lokale database.
   3. En knap/et ikon på skærmen skal indikere status for synkronisering.
      1. Synkroniseret
      2. Ikke synkroniseret
      3. Ingen forbindelse
   4. Responstiden for navigation må maksimalt være to sekunder.
   5. Skal kunne anvendes uden internetforbindelse.
4. **Web app**
   1. Ændringer af data lagres straks i den eksterne database.

# Accepttest

## Use cases

Prækondition: Systemet er tændt (UC1), og mindst én vare er tilføjet til listen ”I køleskabet” (UC4).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC1: *Se varer*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 1:**  Bruger trykker på knappen ”Se varer”. | Bruger trykker på knappen ”Se varer”, og tester visuelt om rette menu fremkommer. | Et nyt vindue, med oversigt over de forskellige lister, vises på skærmen. |  |  |
| **Punkt 2-3:**  Bruger trykker på knappen ”I køleskab”. | Bruger trykker på knappen ”I køleskab”, og tester visuelt om den korrekt liste fremkommer. | En liste over nuværende varer i køleskabet, samt mængden af disse, vises på skærmen. |  |  |

Prækondition: Systemet er tændt (UC1), og mindst én vare er tilføjet til hver af listerne ”Indkøbsliste” og ”Standard-varer” (UC4).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC1: *Alternativt flow*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 2.a:**  Bruger trykker på ”Indkøbsliste”. | Bruger trykker på knappen ”Indkøbsliste”, og tester visuelt om den korrekt liste fremkommer. | En liste over nuværende varer på indkøbslisten, samt mængden af disse, vises på skærmen. |  |  |
| **Punkt 2.b:**  Bruger trykker på ”Standard-varer”. | Bruger trykker på knappen ”Standard-varer”, og tester visuelt om den korrekt liste fremkommer. | En liste over nuværende standard-varer, samt mængden af disse, vises på skærmen. |  |  |

Prækondition: Systemet er tændt (UC1), og alle lister er tomme.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC1: *Undtagelser*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 3.a:**  Listen over varer i køleskabet er tom. | Bruger trykker på knappen ”I køleskab”, og tester visuelt om den korrekt meddelelse fremkommer. | En besked om at listen over varer i køleskabet er tom, vises på skærmen. |  |  |
| **Punkt 2.a.1.a:**  Indkøbslisten er tom. | Bruger trykker på knappen ”Indkøbsliste”, og tester visuelt om den korrekt meddelelse fremkommer. | En besked om at indkøbslisten er tom, vises på skærmen. |  |  |
| **Punkt 2.b.1.a:**  Listen over standard-varer er tom. | Bruger trykker på knappen ”Standard-varer”, og tester visuelt om den korrekt meddelelse fremkommer. | En besked om at listen over standard-varer er tom, vises på skærmen. |  |  |

Prækondition: Systemet er tændt (UC1), og en liste over varer vises på skærmen (UC3).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC2: *Tilføj Vare*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 1:**  Bruger trykker på ”Tilføj”. | Bruger trykker på knappen "Tilføj", og tester visuelt om det korrekte vindue fremkommer. | Et vindue med valgmuligheder vises på skærmen. |  |  |
| **Punkt 2:**  Varetype vælges. | Bruger vælger varetypen "Mælk". | Den valgte varetype markeres som valgt. |  |  |
| **Punkt 3:**  Antal vælges. | Bruger vælger mængden 1 stk. | Mængden 1 er valgt. |  |  |
| **Punkt 4:**  Volumen/vægt vælges | Bruger vælger 1 liter. | Volumenen "1 liter" er valgt. |  |  |
| **Punkt 5-6:**  Bruger trykker på "Tilføj og afslut", og varen tilføjes til listen. | Bruger trykker på "Tilføj og afslut", og tester visuelt om varen er tilføjet til listen. | Den tilføjede varer figurerer på den viste liste. |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC2: *Alternativt flow*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 5.a:**  Bruger trykker på "Tilføj" | Bruger trykker på "Tilføj", og tester visuelt om det korrekte vindue fremkommer. | Et vindue med valgmuligheder vises på skærmen. |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **UC2: *Undtagelser*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 2.a:**  Varetypen eksisterer ikke. | Bruger indtaster en varetype som ikke findes i systemet, og tester visuelt om varen blive stående. | Den nye varetype er valgt. |  |  |

## Ikke-funktionelle krav

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1: *System*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 1.1:**  Mulighederne i Fridge app og web app skal være de samme. | Det testes visuelt om de respektive accepttests for de funktionelle krav, også kan udføres på web-app’en. | De samme muligheder er tilgængelige på web-app’en, som på Fridge-app’en. |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **2: *Databaser*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 2.1:**  Den lokale og den eksterne database skal automatisk synkroniseres hvert 10. minut. | UC2 udføres, hvorefter der tages tid, og efter 10 minutter, åbnes web-app’en, hvorefter det testes visuelt om varen er tilføjet. | Den tilføjede vare er nu synlig gennem web-app’en. |  |  |
| **Punkt 2.2.1:**  I tilfælde af konflikter ved synkronisering, overskriver de nyest tilføjede data de ældste. | UC2 udføres først på Fridge app, hvor antallet sættes til 1. Herefter udføres UC2 for samme vare på web app, hvor antallet sættes til 2. Til sidst udføres UC5, og antallet af varen testes visuelt på begge apps. | Antallet af varer er 2. |  |  |
| **Punkt 2.2.2:**  I tilfælde af konflikter ved synkronisering, overskriver de nyest tilføjede data de ældste. | Ovenstående test udføres igen, men tilføjelserne udføres i omvendt rækkefølge. Antallet af varer testes visuelt på begge apps. | Antallet af varer er 1. |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **3: *Fridge app*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 3.1.1:**  Ved opstart og nedluk, forsøges synkronisering mellem den lokale og den eksterne database. | Først UC5, og herefter UC2 udføres på Fridge app, hvorefter systemet lukkes. Herefter testes visuelt på web-app’en, om varen er tilføjet. | Varen er tilføjet. |  |  |
| **Punkt 3.1.2:**  Ved opstart og nedluk, forsøges synkronisering mellem den lokale og den eksterne database. | UC2 udføres på web app, hvorefter Fridge app startes. Herefter testes visuelt på Fridge-app’en, om varen er tilføjet. | Varen er tilføjet. |  |  |
| **Punkt 3.2:**  Ændringer af data lagres straks i den lokale database. | Først UC5, og herefter UC2 udføres på Fridge app, hvorefter der testes visuelt, at varen er tilføjet. | Varen er tilføjet. |  |  |
| **Punkt 3.3.1:**  En knap/et ikon på skærmen skal indikere status for synkronisering. | UC5 udføres, og det testes visuelt, i hovedmenuen, om et ikon på skærmen indikerer at der er synkroniseret. | Et ikon indikerer at der er synkroniseret. |  |  |
| **Punkt 3.3.2:**  En knap/et ikon på skærmen skal indikere status for synkronisering. | Først UC5, og herefter UC2 udføres, og det testes visuelt, i hovedmenuen, om et ikon på skærmen indikerer at der ikke er synkroniseret. | Et ikon indikerer at der ikke er synkroniseret. |  |  |
| **Punkt 3.3.3:**  En knap/et ikon på skærmen skal indikere status for synkronisering. | Internetforbindelsen afbrydes, UC5 udføres, og det testes visuelt, i hovedmenuen, om et ikon på skærmen indikerer at der ingen forbindelse er. | Et ikon indikerer at der ingen forbindelse er til den eksterne database. |  |  |
| **Punkt 3.4:**  Responstiden for skift af kontekst i menuen må maksimalt være to sekunder. | Der trykkes på ”Se varer”, og tiden fra trykket til skift af kontekst måles. | Tidsmålingen overstiger ikke to sekunder. |  |  |
| **Punkt 3.4:**  Skal kunne anvendes uden internetforbindelse. | Internetforbindelsen afbrydes, UC2 udføres, og det testes visuelt, om varen tilføjes. | Varen er tilføjet. |  |  |
| **4: *Web app*** | **Test** | **Forventet resultat** | **Resultat** | **Godkendt/ kommentar** |
| **Punkt 4.1:**  Ændringer af data lagres straks i den eksterne database. | UC2 udføres på web app, hvorefter der testes visuelt, at varen er tilføjet. | Varen er tilføjet. |  |  |

1. <http://www.wupti.com/file/file?fileId=266584> (tilføjes i endelig rapport som bilag) [↑](#footnote-ref-1)
2. <http://en.wikipedia.org/wiki/MoSCoW_method> [↑](#footnote-ref-2)
3. <http://www.wupti.com/file/file?fileId=266584> (tilføjes i endelig rapport som bilag) [↑](#footnote-ref-3)