Bagpacker

Grupp 9

Kravdokument

V. 4.0

2019-06-02

Dokumenthistorik

Datum	Version	Beskrivning	Författare
190323	**1.0**	Formulerat funktionella och kvalitativa krav	Ekaterina Korotetskaya
190325	**1.1**	Tillägg: Funktionella krav för webb-applikationen	Christina Knepper
190327	**1.2**	Tillägg: Funktionella krav för android	Johan Wingren
190327	**1.3**	Syfte, referenslista samt ordlista är adderat	Sebastian Tyrling
190328	**1.4**	Specifikation av funktionella krav för webb-applikationen.	Christina Knepper
190408	**2.1**	Specifikation av funktionella krav för webb-applikationen och databasen	Christina Knepper Iris Brinkborg
190414	**2.2**	Ändrat dokumentet efter granskningens input. Samt fyllt på ordlista.	Sebastian Tyrling
190417	**2.3**	Ändrat ordlistans utseende, korrekturläst efter ord samt lagt till orden i listan.	Sebastian Tyrling
190417	**3.0**	Implementering av granskningskommentarer	Sebastian Tyrling
190509	**3.1**	Tillägg: funktionella krav för android (generera packlistan)	Ekaterina Korotetskaya
190510	**3.2**	Formaterat dokument och justerat ordlistan	Johan Wingren
190511	**3.3**	Tillägg: Kvalitativa krav	Ekaterina Korotetskaya
190530	**3.4**	Ändringar enligt granskningen efter RS3	Sebastian Tyrling
190531	**3.5**	Sammanfattning av krav	Sebastian Tyrling
190602	**4.0**	Korrekturläst inför sista inlämningen	Sebastian Tyrling

Innehåll

Kravdokument

Syfte

Detta dokument är till för att dokumentera och specificera krav för produkten projektgruppen kommer att utveckla. Det kommer att finnas funktionella, icke-funktionella, kvalitativa krav som dels visar vad produkten kommer att innehålla. Kraven kommer utgå ifrån MoSCoW-modellen[1]. Detta kommer bygga upp vilka av projektgruppens krav som ska implementeras i vilken ordning och på så sätt påverka hur arbetet utformas.

Ordlista:

app	Här: android-applikation
autofyll	En funktion som hjälper användaren att hitta rätt/fylla i ett textfält.
Bottle	Är ett micro ramverk för python
cork	Är en del av bottle som används för att bygga skapa konto och logga in funktioner
databas	En lagringsplats för den data som används för att populera packlistorna. Webbservern hämtar data från databasen och sänder den vidare till webb- och androidapplikationen när dessa skickar HTTP-förfrågningar. Sparade, lösenordsskyddade packlistor lagras även i databasen.
etikett	Ett identifierande adjektiv som användaren uppger för att sätta en prägel på resan, vilket i sin tur påverkar filtreringen av packabler till packlistan.
modifiera	Samlingsord för vad användaren kan göra med listan (t.ex. lägga till/ta bort packabler)
packabel ~n, pl. ~er	Objekt som finns i packlistan. Varje föremål har ett antal olika parametrar, t.ex. namn, vikt, kategori, temperatur
webbapplikation	Programvara som exekveras på en webbserver och som användaren kan interagera med via en webbläsare.
webbserver	Mjukvaran som hanterar HTTP-förfrågningar från webbapplikationen samt kommunikationen med databasen.

Referenser

[1] Duncan Haughey “Moscow Method”
Project Smart, (ingen tid hittad), [Online]
Tillgänglig:https://www.projectsmart.co.uk/moscow-method.php
[Hämtad: 27 mars 2019]

Funktionella krav

Kraven för projektet kommer att delas upp i två delar: en för appen och en för webbsidan.
Kraven för appen i sin tur delas upp i kategorier: användarkrav och systemkrav.
Varje krav har ett namn som består av några bokstäver och siffror:

- den första bokstaven är F (funktionella krav) eller IF (icke-funktionella krav).
- den andra bokstaven är A (användarkrav) eller S (systemkrav).
- funktionella krav har prioriterats med hjälp av MoSCoW-modellen[1] där:
    - M står för must have
    - S står för should have
    - C står för could have
    - W står för would not

1. Webbapplikation

1.1 Användarinput

F.A.M.1.1.1 - Ange resmål
Användaren ska ha möjlighet att ange resmål.
Det behövs för att applikationen ska kunna ta reda på vädret på resmålet.

F.S.M.1.1.2 - Acceptera viss typ av inmatning
Det accepteras bara bokstäver som inmatning, ej numerisk datainmatning.
Vid felaktig inmatning visas ett felmeddelande.
I fältet för resmål anger användaren namnet på en stad och därför är siffror eller andra specialtecken ej aktuella.
F.S.M.1.1.3 - Autofyll
Användaren ska få en lista med förslag på resmål under inmatningen.
Autocomplete-funktionen förebygger felaktig inmatning. Om det finns flera destinationer med samma namn får användaren hjälp att välja “rätt” destination genom att information om land / region visas.

F.A.M.1.1.4 - Ange resans datum
Användaren ska kunna ange avrese- och returresedatum.
Datum behövs för att kunna beräkna resans längd samt hämta aktuell eller historisk väderinformation.

F.A.M.1.1.5 - Ange planerade aktiviteter
Användaren ska kunna ange planerade aktiviteter under resan. Det ska vara möjligt att välja flera olika alternativ.
Parameter som används för att skräddarsy packlistan.

F.A.M.1.1.6 - Aktiviteter
Användaren kommer att kunna välja mellan 8 aktiviteter (friluftsliv, sightseeing, vintersport, kultur, träning/motion, nattliv, sol och bad och annat).

F.A.C.1.1.7 - Ange etikett (ej implementerat, se sammanfattning av krav)
Användaren ska ha möjlighet välja bland olika etiketter (t.ex. miljövänlig, lyxig, romantisk etc.) för att bestämma typ av resa.

F.A.C.1.1.8 - Etiketter (ej implementerat, se sammanfattning av krav)
Användaren ska ha möjlighet att kunna välja mellan 6 olika etiketter (miljövänlig, lyxig, romantisk, blabla, blabla och blabla)

F.A.M.1.1.9 - Ange boende
Användaren ska kunna ange typ av boende. Det ska vara möjligt att välja flera olika alternativ.
Parameter som används för att skräddarsy packlistan.

F.A.M.1.1.10 - Boende
Användaren ska ha möjlighet att välja mellan 8 olika boende (hotell, tält, lägenhet, husbil-vagn, hos kompisar, stuga, hostel och annat).

F.A.M.1.1.11 - Ange färdmedel
Användaren kunna ange färdmedel. Det ska vara möjligt att välja flera olika alternativ.
Parameter som används för att skräddarsy packlistan.

F.A.M.1.1.12 - Färdmedel
Användaren ska ha möjlighet att välja mellan 8 olika färdmedel (flyg, färja, bil, cykel, buss, motorcykel, tåg och annat).

F.A.C.1.1.13 - Ange medresenärer och ålder
(ej implementerat, se sammanfattning av krav)
Ifall användaren ansvarar för någon annans packning (t.ex. barn) ska medresenär och dess ålder kunna anges. Det ska vara möjligt att lägga till flera medresenärer.

F.A.M.1.1.14 - Ange packlistans titel
(till viss del implementerat, se sammanfattning av krav)
Användaren ska ha möjlighet att ange en titel för packlistan.

F.A.W.1.1. - Ange kön
Användaren ska inte kunna ange kön.
Möjlighet att ange kön har valts bort för att undvika fastna i stereotyper. I och med att användaren ska kunna redigera packlistan (se 1.4) finns det möjlighet att personalisera listan och lägga till "könsspecifika" saker om användaren önskar så.

1.2 Generera packlistan

F.S.M.1.2.1 Ta reda på destinationens längd- och breddgrad
Baserad på användarens input ska systemet ta reda på destinationens koordinater (längd- och breddgrad).

F.S.M.1.2.2 Hämta aktuell väderinformation
Baserad på destinationens geografiska koordinater ska systemet ta reda på aktuell väderinformation (min och max temperatur samt regnrisk) om sådan finns tillgänglig.

F.S.M.1.2.3 Hämta historisk väderinformation
Om resan inte påbörjas inom den kommande veckan eller aktuell väderinformation inte finns tillgänglig ska systemet hämta historisk väderinformation baserad på användarens input.

F.S.M.1.2.4 Hämta antal föremål
Baserat på resans längd ska vissa föremål förekomma i större antal. Detta gäller till exempel vissa basplagg och underkläder, vars antal ska justeras beroende på hur många dagar resan varar.

F.S.M.1.2.5 Hämta föremål som är relaterade till planerade aktiviteter
Systemet ska kunna hämta föremål ur databasen som tillhör de valda aktiviteterna.

F.S.C.1.2.6 Justera föremål som är relaterade till resans etikett
(ej implementerat, se sammanfattning av krav)
Systemet ska kunna justera vilka föremål som hämtas från databasen baserat på den valda etiketten.

F.S.M.1.2.7 Hämta föremål som är relaterade till användarens boende på resan
Systemet ska kunna lägga till föremål i packlistan baserat på användarens boendeform.

F.S.M.1.2.8 Hämta föremål som är relaterade till användarens färdmedel
Systemet ska kunna hämta föremål ur databasen som är kategoriserade efter valt färdmedel.

F.S.C.1.2.9 Hämta föremål som är relaterade till medresenärer och deras ålder (ej implementerat, se sammanfattning av krav)
Systemet ska kunna hämta föremål ur databasen som är knutna till eventuella medresenärer och deras ålder. Till exempel ska det finnas specifika kategorier för barn i olika åldrar.

F.S.C.1.2.10 Hämta föremål som är relaterade till lokal kultur (ej implementerat, se sammanfattning av krav)
Systemet ska, beroende på destination, filtrera bort föremål som inte är lämpliga att ha med sig utifrån kulturella sedvänjor. På samma sätt ska systemet lägga till föremål som är viktiga ur ett kulturellt perspektiv.

1.3 Presentation av packlistan

F.S.M.1.3.1 - Kategorier
Packlistan som systemet har genererat utifrån användarens input ska presenteras överskådligt med hjälp av olika kategorier (t.ex. kläder, dokument, hygienartiklar).
Behövs för att ge användaren en bättre överblick.

F.S.M.1.3.2 - Antal
För saker vars mängd påverkas av resans längd ska antal visas upp.

F.S.C.1.3.3 - Vikt (till viss del implementerat, se sammanfattning av krav)
En uppskattning av packningens totalvikt ska visas upp.

F.S.C.1.3.4 - Information om väderdata som ligger till grund för urvalet
Användaren ska ha möjlighet att visa upp väderdatan som urvalet är baserad på.

1.4 Skapa konto (ej implementerat, se sammanfattning av krav)

F.S.C.1.4.1 - Skapa konto
Användaren ska kunna skapa ett konto med användarnamn och lösenord. När användaren registrerar sig ska hen uppge lösenordet två gånger och dessa ska matcha. Användarnamnet ska vara unikt.
Vid inloggning får användaren tillgång till fler funktioner: t.ex. spara packlistan, dela packlistan med andra.

1.5 Logga in (ej implementerat, se sammanfattning av krav)

F.S.C.1.5.1 - Logga in (ej implementerat, se sammanfattning av krav)
Användaren ska kunna logga in.

F.S.C.1.5.2 - Spara packlista
Användaren kan trycka på "spara-knappen" och får då en kod som kan användas för att generera listan igen på androidapplikationen och webbsidan.

F.S.C.1.5.3 - Visa upp sparad packlista (ej implementerat, se sammanfattning av krav)
En startsida som visar upp sparade listor i en lista.

2. Androidapplikation

2.1 Användarinput

F.A.M.2.1.1 - Ange resmål
Användaren ska ha möjlighet att ange resmål.
Det behövs för att applikationen ska kunna ta reda på vädret på resmålet.

F.A.M.2.1.2 - Ange resans datum
Användaren ska kunna ange avrese- och returresedatum
Datum behövs för att kunna beräkna resans längd samt hämta aktuell eller historisk väderinformation.

F.A.M.2.1.3 - Ange planerade aktiviteter
Användaren ska kunna ange planerade aktiviteter under resan.

F.A.M.2.1.4 - Aktiviteter
Användaren kommer att kunna välja mellan 8 aktiviteter (friluftsliv, sightseeing, vintersport, kultur, träning/motion, nattliv, sol och bad och annat).

F.A.C.2.1.5 - Ange etikett (ej implementerat, se sammanfattning av krav)
Användaren ska ha möjlighet välja bland olika "tags" (t.ex. miljövänlig, lyxig, romantisk etc.) för att bestämma typ av resa.

F.A.M.2.1.6 - Ange boende
Användaren ska kunna ange typ av boende.

F.A.M.2.1.7 - Boende
Användaren ska ha möjlighet att välja mellan 8 olika boende (hotell, tält, lägenhet, husbil-vagn, hos kompisar, stuga, hostel och annat).

F.A.M.2.1.8 - Ange färdmedel
Användaren kunna ange färdmedel.

F.A.M.2.1.9 - Färdmedel
Användaren ska ha möjlighet att välja mellan 8 olika färdmedel (flyg, färja, bil, cykel, buss, motorcykel, tåg och annat).

F.A.S.2.1.10 - Ange medresenärer (och ålder)
(ej implementerat, se sammanfattning av krav)
ännu ej specificerat

F.A.M.2.1.11 - Ange packlistans titel
Användaren ska ha möjlighet att ange en titel för packlistan.
Detta krävs för att identifiera sparade listor.

F.A.M.2.1.12 - Hämta lista från webb-app
Användaren ska kunna hämta en packlista från databasen via webbservern.
Krävs för integrationen mellan webb och android. Listan ska kunna skapas på webben och hämtas via servern.

F.A.M.2.1.13 - Hantera lista
Användaren ska kunna lägga till, ta bort, ändra antal och markera föremål i listan.
Flexibiliteten i att kunna manipulera listan direkt i appen krävs för att uppnå användbarhetskraven.

F.A.M.2.1.14 - Spara/ladda lista
Användaren ska kunna spara listan, med eventuella ändringar, i en lokal fil. Vid programstart ska den sparade listan kunna läsas in.
Markerade (packade) föremål ska också sparas så att det lätt går att återuppta påbörjad packning efter avbrott.

2.2 Generera packlistan

F.S.M.2.2.1 Hämta föremål som är relaterade till användarens färdmedel
Systemet ska kunna hämta föremål ur databasen som är kategoriserade efter valt färdmedel.

F.S.M.2.2.2 Hämta föremål som är relaterade till planerade aktiviteter
Systemet ska kunna hämta föremål ur databasen som tillhör de valda aktiviteterna.

F.S.M.2.2.3 Hämta föremål som är relaterade till användarens boende på resan

Systemet ska kunna lägga till föremål i packlistan baserat på användarens boendeform.

2.3 Presentation av packlistan

F.S.S.2.3.1 - Kategorier

Packlistan ska visas som en scrollningsbar lista där olika kategorier av föremål är sorterade under respektive rubrik.
Behövs för att ge användaren en bättre överblick.

F.S.M.2.3.2 - Antal

För saker vars mängd påverkas av resans längd ska antal visas upp.
En väsentlig funktion för listans struktur och presentation. Varje föremål ska endast uppta en rad i listan, därför behöver antalet synas på samma rad.

F.S.C.2.3.3 - Vikt

En uppskattning av packningens totalvikt ska visas upp.
Ett möjligt tillägg som syftar till att förbättra användbarheten .

2.4 Skapa konto

F.S.M.2.4.1 - Skapa konto

Användaren ska kunna skapa ett konto.
Vid inloggningen kan användaren få möjlighet till fler funktioner: t.ex. spara packlistan, dela packlistan med andra.

2.5 Logga in

F.S.M.2.5.1 - Logga in

Användaren ska kunna logga in.

Kvalitativa krav

Delas upp i två kategorier:

- Produktkrav
- Organisatoriska krav

Organisatoriska krav

IF.S.1 - Programmeringsspråk

Webb-applikationen ska använda python som programmeringsspråk.
Webb-hemsidan ska använda HTML, CSS och JavaScript.
Android-applikationen ska använda Java som programmeringsspråk.

Produktkrav

IF.A.2 - Lätt användbart UI
UI ska inte vara komplicerat och innehålla många knappar. Det ska finnas bara de viktigaste interaktioner.

IF.S.3 - Minnesutrymme
Android-applikationen ska inte ta upp mer än 50 MB på telefonen.

IF.S.4 - Skärmsstorleksstöd
Webbapplikationen ska anpassas till olika skärmstorlekar

Sammanfattning av krav

Under denna sammanfattning kommer det visas vilka krav som inte blivit implementerade och vilka krav som är implementerade till en viss del. Det kommer förklaras varför dessa krav inte blev implementerade eller varför de helt enkelt inte finns med.

Kraven som inte är implementerade:
Webb:
F.A.C.1.1.7 - Ange etikett
F.A.C.1.1.8 - Etiketter
F.A.C.1.1.13 - Ange medresenärer och ålder
F.A.M.1.1.14 - Ange packlistans titel
F.S.S.1.2.6 Justera föremål som är relaterade till resans etikett
F.S.S.1.2.9 Hämta föremål som är relaterade till medresenärer och deras ålder
F.S.C.1.2.10 Hämta föremål som är relaterade till lokal kultur
F.S.C.1.5.1 - Skapa konto
F.S.C.1.6.1 - Logga in
F.S.C.1.6.3 - Visa upp sparad packlista
Android:
F.A.C.2.1.5 - Ange etikett

De krav som är implementerade till en viss del:
F.S.C.1.3.3 - Vikt

F.A.C.1.1.7 - Ange etikett var en del av projektet en tid där projektgruppen ville att användaren skulle kunna välja ett alternativ till sin resa. Efter det började koda och bygga upp de olika produkterna ansåg gruppen att dessa etiketter var svåra att definiera och därför svåra att implementera. Eftersom gruppen inte använder sig av etiketter så finns det krav som inte blev implementerade som är kopplade till etiketter (**F.A.C.1.1.8 - etiketter**, **F.S.S.1.2.6 justera föremål som är relaterade till resans etikett och F.A.C.2.1.5 - Ange etikett**).

F.A.C.1.1.13 - Ange medresenärer och ålder var ett annat alternativ som funnits med i projektet sedan början. Det fanns inte mer tid att implementera detta. Det hade gått med lite kod och lite mer tid. Eftersom detta krav inte är implementerat är även kravet **F.S.S.1.2.9 Hämta föremål som är relaterade till medresenärer och deras ålder** inte implementerat.

F.A.M.1.1.14 - Ange packlistans titel blev ej implementerat i webbgränssnittet pga tidsbrist och för att andra funktioner hade prioritet.

F.S.C.1.2.10 Hämta föremål som är relaterade till lokal kultur var ett alternativ projektgruppen kom på som var relaterat till att man kanske reser till ett land där en viss klädnad bör bäras av de som är i detta land. Detta hade tagit väldigt mycket research och databasen hade blivit större och mer omfattande än vad den är nu. Allt detta hade tagit mer tid än vad som var budgeterat. Projektgruppen hade redan andra alternativ som man jobbade på och såg att detta kravet kanske kunde implementeras längre fram om produkten fortsattes att utvecklas.

F.S.C.1.5.1 - Skapa konto. Alla de krav som rör att man ska skapa ett konto eller logga in var en del av projektet då gruppen vill spara listor (detta är nu löst på ett annat sätt). Detta krav blev ej implementerat för att det micro ramverk (bottle) som användes till webbservern hade ett annat tillhörande ramverk(cork) som skulle hjälpa till att bygga upp skapa konto funktionen. Det som gjorde att detta inte implementerades var att cork går ej att installeras på windows datorer då det bibliotek som hjälper till att installera detta ramverk inte längre finns på windows. Efter en tid lagts ner på att försöka få detta att fungera var gruppen tvungen att gå vidare och den andra lösningen hittades. Detta påverkade även andra krav under skapa konto och logga in (**F.S.C.1.6.1 - Logga in).**

F.S.C.1.3.3 - Vikt är det kravet som är till största del helt implementerad. I både webbgränssnitt och android gränssnitt visas den totala vikten. Det som ej är implementerat är att man ej visar vad som är i handbagage och att totalvikten justeras om det görs ändringar i listan, till exempel om föremål läggs till eller tas bort.