FoodFinder

Grupp 14

Verifiering och valideringsdokument

V. 1.0

20170317

Dokumenthistorik

Datum	Version	Beskrivning	Författare
170317	1.0	Stycke dokumentgranskning påbörjad	John Eklöf
170317	1.0	Riktlinjer för kod påbörjad	Filip Heidfors
170403	1.0	Testfall kravbaserad systemtestning påbörjad.	Elias Moltedo
170403	1.0	Påbörjat stycket med White-box Test	Alexander J. Drottsgård
170403	1.0	Fortsatt skriva på riktlinjer för kod, skrivit checklista för kodgranskning och skrivit beskrivande text för process kodgranskning	Filip Heidfors
170404	1.0	Redigerat checklista för kodgranskning och riktlinjer för kod samt skrivit syfte för dokumentet	Filip Heidfors
170405	1.0	Fortsatt på stycket med White-box Test (Unit Testing)	Alexander J. Drottsgård
170406	1.0	Lagt till ytterligare testfall	Alexander J. Drottsgård Filip Heidfors Elias Moltedo John Eklöf
170406	1.0	Skrivit stycke Testprocess	Filip Heidfors
170406	1.0	Fortsättning på dokumentgranskning	John Eklöf
170410	1.0	Färdigställt dokumentgranskning	John Eklöf
170410	1.0	Påbörjat spårningsmatris	John Eklöf

[Fler rader läggs till efterhand som det behövs. Utifrån beskrivningen ska det gå att förstå vad personen gjorde för typ av ändring. Så bara en text som säger ”ändring” räcker inte. Texten behöver exempelvis säga ”Lagt till testfall TF23-28” eller ”Infogat testrapport TR3”

Det kan finnas flera författare. Endast de som faktiskt är aktiva med att skriva texten listas som författare. Om man är två som sitter och jobbar tillsammans och en skriver men man hela tiden aktivt diskuterar det som skrivs så kan bägge personerna stå som författare. Om man har ett möte i hela gruppen där man diskuterar saker, en person tar anteckningar och skriver sedan rent detta i dokumentet så står endast denna person som författare.]

Innehåll

Verifiering och valideringsdokument

Syfte

<Text som beskriver syftet med dokumentet. Vad det innehåller och hur det relaterar till projektet. Det kan också beskrivas vad som inte står här om det förtydligar vad syftet är eller inte är.>
Det här dokumentet innehåller dokumentation för testfall, testrapporter, kod- och dokumentgranskning, kravbaserad systemtestning samt white-box testning och hur detta ska utföras. Syftet med detta är att verifiera och validera projektets olika artefakter för att öka kvaliteten på slutprodukten och minimera risken för fel under projektets gång.

Ordlista

<ord> <förklaring>
<ord> <förklaring>

[Ta upp förkortningar och uttryck som används i dokumentet och som kanske inte är självklara för en läsare som inte är insatt i projektet. Tänk också på att ta upp begrepp som skulle vara svåra att förstå för en läsare som skulle kunna vara en presumtiv kund eller referensperson i forma v exempelvis slutanvändare. Ordlistan ska ordnas i bokstavsordning.]

Referenser

[1] Tsui F., Karam O., Bernal B. (2013/2014). Essentials of Software Engineering 3rd Edition, Jones & Bartlett Learning

[2]
http://istqbexamcertification.com/what-is-statement-coverage-advantages-and-disadvantages/

<referens>

[Använd ett referenssystem och utforma referenser konsekvent enligt detta. Exempel på referenssystem enligt IEEE som är vanligt i tekniska dokument hittas på:
http://libguides.murdoch.edu.au/c.php?g=246207&p=1640218
]

Testprocess

<Text som beskriver den testprocess som används. Tänk på att denna ska matcha utvecklingsprocessen i projektplanen. Som minst ska kravbaserad systemtestning utföras med utgångspunkt i kravdokumentet. Beskrivningen ska ange hur testning genomförs, när testning sker i förhållande till utvecklingsprocessen och hur denna dokumenteras. Det ska också beskrivas hur eventuella problem som testningen identifierar ska hanteras.>

Tester som kommer utföras för det här projektet är kravbaserad systemtestning samt white-box testning i form av Unit Testning och Statement Coverage. För alla former av testning kommer flera testfall skrivas. Testerna kommer att utföras när de olika testfallen/kraven är implementerade i kod och dokumenteras i testrapporter. Eventuella testfall där resultatet inte stämmer överens med det förväntade resultatet för testfallet kommer att korrigeras och rättas till så fort som möjligt.

Kravbaserad systemtestning kommer att utföras genom att först skriva testfall för de olika kraven. När kraven är implementerade i kod utförs testerna enligt testfallen och dokumenteras i en testrapport. Eventuella testfall vars resultat inte stämmer med det förväntade resultatet ska fixas så fort tid finns tillgänglig. De olika white-box testerna kommer att utföras på samma vis som kravbaserad systemtestning när testfallen för white-box är implementerade.

[Generellt om referenser till beskrivningar av metoder i detta dokument: Detta kan vara (kurs)litteratur eller någon annan relevant källa. Om en referens som utgörs av en webbsida används så ska denna vara att betrakta som information med bekräftad auktoritet på området. Det vill säga Wikipedia eller liknande fungerar inte. Webbreferenser ska vara etablerade expertsidor så som officiella webbsidan för XP, Agile Alliance eller webbsidor underhållna av erkända branschorganisationer så som IEEE eller ACM.]

Granskning

Kodgranskning

Kodgranskning för det här projektet kommer att utföras i form av en anpassad variant av inspektion [1]. Gruppens ansvarig för kodgranskning kommer att kalla till ett möte och skicka ut en checklista för kodgranskning till övriga medlemmar. Inför mötet ska alla medlemmar granska vald kod med hjälp av checklistan och riktlinjerna för kod. Under mötet ska gruppen gå igenom och diskutera allas kodgranskningar och hur eventuella fel i koden kan lösas på bästa sätt. Under mötet kommer ansvarig för kodgranskning föra ett granskningsprotokoll och sedan utföra ändringarna som tagits upp på granskningsmötet. Kod som väljs ut för granskning kommer att vara klasser och metoder som hanterar implementering av krav med prioritering “Must” och “Should” samt långa och komplexa klasser och metoder. Valet av kod prioriteras på detta sätt då det är mest väsentligt att koden som hanterar “Must” och “Should” krav fungerar felfritt samt att ju mer komplex och lång koden är desto större risk är det att koden innehåller fel.

Riktlinjer för kod

Klasser:

- Klasser ska ha relevanta namn så det är tydligt för resterande gruppmedlemmar vad klasserna utför.
- Klasserna ska inte vara för långa.
- Inre klasser ska vara kortare än sin huvudklass. Om en inre klass är för lång så gör en egen klass av det.
- Klasserna ska vara av en viss typ (controller, boundary, entity) hantera olika saker, till exempel logik (controller) eller GUI (boundary). En klass som till exempel hanterar GUI ska innehålla minimal eller ingen logik utan hålla sig till sitt syfte.
- Alla klasser ska vara javadokumenterade

Variabler:

- Alla instansvariabler ska vara privata
- Alla variabler ska ha relevanta namn så det är tydligt vad variabeln representerar. Förkortningar ok så länge det fortfarande är tydligt.
- I variabler som innehåller flera ord ska orden separeras med med en stor bokstav för nytt ord. Till exempel “buttonSearch”.
- Alla variabler ska börja på liten bokstav.

Metoder:

- Alla metoder ska ha relevanta och tydliga namn både på själva metoden men även på eventuella parametrar för metoden.
- Metoder ska inte ha för mycket olika funktioner. Varje metod ska ha ett huvudfokus och hålla sig till det
- Alla metoder ska vara javadokumenterade
- Alla metoders kropp ska börja på samma rad som metodhuvudet efter ett

mellanslag.
- Alla metoders kropp ska avslutas en rad under sista raden kod för metoden
- Kod i en metod ska alltid börja på raden under metodhuvudet

Läsbarhet för koden:
- All kod ska vara korrekt indenterad och enkel att läsa för alla gruppmedlemmar. White space ska användas för att på ett tydligt sätt separera kodstycken.

Checklista för granskningsmöte
<Här visas den checklista som ska användas för granskningsmötet. Om man vill så kan denna lyftas ut som ett separat dokument men det ska då ges en referens till detta dokument här.>
- Har alla variabler, klasser (inklusive inre klasser) och metoder korrekt synlighet?
- Har alla instansvariabler, klasser och metoder namn så det på ett tydligt sätt framstår vad de representerar?
- Är alla variabelnamn, klassnamn och metodnamn skriva enligt riktlinjerna?
- Finns det någon variabel eller metod som inte används?
- Har alla variabler, klasser och metoder lämpliga namn som tydligt beskriver deras syfte?
- Är alla klasser och metoder dokumenterade med kommentarer som beskriver deras syfte?
- Finns det någon klass eller metod som är väldigt lång som kan och borde delas upp i flera klasser/metoder? - Redogör för hur det kan delas upp
- Finns det någon klass eller metod som är onödig och kan tas bort?
- Finns det någon metod eller kodstycke som inte hanterar möjliga fel på ett korrekt sätt?
- Finns det några varningar för koden som borde fixas?
- Har koden korrekt indentering?
- Är koden lättläst med white space på lämpliga platser?
- Finns det metoder som hade kunnat kortas ner och göras mer effektiva och lättlästa?
- Är kodens struktur konsistent? Skrivs metodhuvud, metodkropp och white space på ett konsistent sätt enligt riktlinjerna?

Dokumentgranskning
Dokumentgranskningen ska följa de principer som gäller för den mest formella granskningen, det vill säga inspektion. Den som är ansvarig för dokumentgranskningen har till uppgift att planera och strukturera hela granskningsarbetet. Samtliga gruppmedlemmar får sedan i uppgift att sätta sig in i materialet [1]. Sedan ska varje deltagare inkomma med synpunkter vid angivet granskningsmöte. Kallelse skickas ut av den gruppmedlem som är ansvarig för dokumentgranskningen. Varje kallelse ska ske med minst två dagars framförhållning så att alla gruppmedlemmar har tid och möjlighet att sätta sig in i materialet. Målsättningen är att två träffar ska arrangeras under projektets gång. De dokument som ska granskas är kravdokument, designdokument och verifierings och valideringsdokument. Dessa dokument

ligger till grund för hela arbetet då de är under ständig förändring. Ambitionen är att projektplanen ska följas oavsett vilka förändringar som sker i de övriga. Vid varje granskningsmöte ska det upprättas ett protokoll som redogör för samtliga synpunkter, vilka förändringar som bör göras, vem som ansvarar för ändringen samt när detta ska ske. Vidare ska gruppmedlemmarnas deltagande dokumenteras för själva granskningen. [Del av individuell fördjupning. Någon i gruppen måste ta ansvar för denna.]
< Text som beskriver en process för en formell inspektion för dokumentgranskning och hur denna genomförs i det här projektet. Texten ska innehålla referenser till minst en beskrivning av den metod som man utgår ifrån. Den beskrivning av metoden man utgår ifrån behöver sannolikt anpassas till ert projekts förutsättningar. Texten ska beskriva vilka anpassningar ni gör och hur dokumentgranskning genomförs i ert projekt. Texten ska också ange hur man väljer ut vilka dokument som ska granskas och motivera varför detta prioriteringssätt är lämpligt. Checklistor och mängd dokument ska vara så omfattande att ett granskningsmöte på 2 timmar fylls ut. Se även information under granskningsprotokoll mot slutet av dokumentet.>

Riktlinjer för dokument
<Här listas de riktlinjer som ska användas av gruppen för de dokument man skriver. Dessa ska omfatta hur text struktureras, hur man namnger filer, vilka typsnitt och layout, här ska definieras vilka ord som ska användas för vissa saker så att man är konsekvent i sin text. Det krävs även vissa mer dokumentspecifika riktlinjer som anger hur exempelvis krav ska utformas, hur användningsfallsbeskrivningar ska skrivas i designdokumentet eller hur testfall för kravbaserad testning ska skrivas. Om man vill så kan dessa lyftas ut som ett separat dokument men det ska då ges en referens till detta dokument här.>

Det här stycket kommer att lyfta fram de viktiga aspekter som krävs för kunna ha en fungerande dokumenthantering. För att öka tydligheten kring de filer som gruppen sedermera kommer att använda sig av behöver riktlinjer att tas fram. Ambitionen är att minska eventuell förvirring både inom gruppen men även för de som utifrån kommer att läsa våra dokument. Syftet med dokumentgranskningen är att säkerställa att de dokument som tas fram är tillräckligt bra. Alltså att de kan tjäna som underlag för framtida arbete.

Dokumentstruktur
Samtliga dokument ska följa den mall som anges nedan. Samtliga filer ska ha samma struktur och utformning för att skapa öka tydlighet.
Filnamn: Samtliga filnamn ska namnges utifrån följande: da33a_VT17_FoodFinder_v1.0
Typsnitt: Samtliga dokument ska skrivas i typsnittet Cambria.
Teckenfärg: All brödtext ska ha svart som teckenfärg
Layout: Försättsbladet ska innehålla information om gruppnamn, gruppnummer, dokumentnamn, versionsnummer, samt datum. Alla sidor ska innehålla sidhuvud med samma information som försättsbladet. Alla dokument ska innehålla dokumenthistorik, innehållsförteckning, syfte, ordlista och referenser. Alla sidor ska vara numrerade. Alla

huvudrubriker ska ha teckenstorlek 14 i fet stil. Alla underrubriker ska ha teckenstorlek 12 i fet stil. All brödtext ska skrivas i teckenstorlek 12.

Checklista för granskningsmöte

<Här visas den checklista som ska användas för granskningsmötet. Om man vill så kan denna lyftas ut som ett separat dokument men det ska då ges en referens till detta dokument här.>

Övergripande dokumentstruktur

- Är den övergripande dokumentstrukturen korrekt? Identifiera problem och brister med dokumenten.
- Följer dokumenten kraven som anges i dokumentstrukturen?
- Finns det förkortningar, ord eller liknande som borde förklaras i en ordlista?
- Är språket välformulerat? Finns det stavfel, dålig grammatik, eller meningsuppbyggnader som är bristfälliga?
- Är språket neutralt och objektivt?
- Är innehållsförteckningen uppdaterad?
- Kan man se en "röd tråd" genom alla texter?
- Är omfattningen av samtliga dokument tillräcklig? Motivera!
- Är samtliga referenser och dess fakta tillförlitlig?
- Finns det stegvisa instruktioner i dokumenten?
- Är dokumenten anpassad till rätt målgrupp?
- Finns det rätt informationsmängd baserat på vad uppgiften kräver?
- Tar dokumenten hänsyn till givna standarder(externa lagar), förordningar och lagar?

Kravhantering

- Granskning av respektive krav. Identifiera brister och styrkor med respektive krav. Bör något krav läggas till eller tas bort?
- Är kravet rätt formulerat?
- Kan kravet missförstås?
- Står olika krav i konflikt med varandra (**consistency**)?
- *Om ja, vilket annat krav står kravet i konflikt med?*
- Kan kravet spåras till produktbeskrivningen (**traceability**)?
- Kommer det slutliga produkten kunna testa eller på annat sätt verifiera att kravet är uppfyllt (**verifiability**)?
- Anges den den funktion eller egenskap som kravet beskriver någon annanstans i dokumenten (**adaptability, traceability**)?
- *Om ja, finns det referenser till kravet i dokumenten?*

Övergripande frågeställningar

- Övergripande frågeställningar, identifiera vad som är bra och vad som eventuellt saknas.
- Täcker kraven samtliga önskemål i produktbeskrivningen (**completeness**)?

- *Om nej, motivera?*

● Täcker kraven samtliga tillägg som har gjorts i produktbeskrivningen (**completeness**)?

Testning

Kravbaserad systemtestning

<Text som beskriver kravbaserad systemtestning och hur denna genomförs i det här projektet. Texten ska innehålla referenser till minst en beskrivning av den metod som man utgår ifrån. Den beskrivning av metoden man utgår ifrån behöver sannolikt anpassas till ert projekts förutsättningar. Texten ska beskriva vilka anpassningar ni gör och hur kravbaserad testning genomförs i ert projekt.>
Kravbaserad systemtestning [1] kommer att genomföras genom att först skriva testfall för de olika kraven. När kraven är implementerade i kod utförs testerna enligt testfallen och dokumenteras i en testrapport. Eventuella fel som identifieras vid testerna kommer att korrigeras och rättas till så fort tillfälle ges.

Prioritering

<Text som beskriver hur krav prioriteras för testning med denna metod.>

White box-testning: Unit Testing

[Två personer kan som sin individuella fördjupning utföra någon form av white-box-testning. Exakt vilken white-box-metod som används lämnas till personen/personerna och gruppen att bestämma – exempelvis enhetstest, statement coverage eller något annat. Om två personer arbetar med white-box-testning så måste dessa två personer arbeta med olika metoder för detta. Två avsnitt med varsin rubrik enligt ovan, en för varje metod, behövs då.]
<Text som beskriver den white-box-metod som används. Texten ska innehålla referenser till minst en beskrivning av denna metod som man utgår ifrån. Den beskrivning av metoden man utgår ifrån behöver sannolikt anpassas till ert projekts förutsättningar. Texten ska beskriva vilka anpassningar ni gör och hur testmetoden tillämpas i ert projekt.

Texten ska referera till de kodfiler med testkod/testfall som sannolikt behövs och tala om var dessa hittas och vad de heter. Filer med testkod ska lämnas in tillsammans med dokumentation när denna frågas efter (exempelvis vid RS-deadlines eller slutinlämning).

Det ska som ett absolut minimum finnas 20 relevanta testfall. Om en första prioriterad kod inte genererar så många testfall för metoden så ska ytterligare kod testas.>

Testning i form av Unit Testing[1] kommer att utföras för systemet. Systemets klasser kommer att testas isolerat. Genom integrerade tester så kommer man kunna

uppmärksamma problem för varje klass och det ger i längden en lägre risk för fel i systemet. Systemet innehåller ett antal klasser som är beroende av varandra, därför kommer stubs att skapas för att varje klass ska kunna testas isolerat. För att testa de delarna av systemet som är beroende av databas/API (i detta fall Foursquare API) kommer ett abstrakt interface att skapas. Då fler tester skrivs efterhand så kommer det att finnas testfall för systemet att köra under fortsättningen av systemets utveckling.

Prioritering

<Text som beskriver hur kod-moduler prioriteras för testning med denna metod. Prioriteringen ska motiveras till varför det är lämpligt att prioritera på detta vis.>
Ordningen i testfallen kommer att prioriteras utfrån hur våra krav ser ut.

White box-testning: Statement coverage

[Två personer kan som sin individuella fördjupning utföra någon form av white-box-testning. Exakt vilken white-box-metod som används lämnas till personen/personerna och gruppen att bestämma – exempelvis enhetstest, statement coverage eller något annat. Om två personer arbetar med white-box-testning så måste dessa två personer arbeta med olika metoder för detta. Två avsnitt med varsin rubrik enligt ovan, en för varje metod, behövs då.]
<Text som beskriver den white-box-metod som används. Texten ska innehålla referenser till minst en beskrivning av denna metod som man utgår ifrån. Den beskrivning av metoden man utgår ifrån behöver sannolikt anpassas till ert projekts förutsättningar. Texten ska beskriva vilka anpassningar ni gör och hur testmetoden tillämpas i ert projekt.

Texten ska referera till de kodfiler med testkod/testfall som sannolikt behövs och tala om var dessa hittas och vad de heter. Filer med testkod ska lämnas in tillsammans med dokumentation när denna frågas efter (exempelvis vid RS-deadlines eller slutinlämning).

Det ska som ett absolut minimum finnas 20 relevanta testfall. Om en första prioriterad kod inte genererar så många testfall för metoden så ska ytterligare kod testas.>

Som en del av white box testningen kommer statement coverage att utföras. Det innebär att alla villkor i koden kommer att testas [2]. Om en metod består av flera villkor kommer alla vägar som koden har att genomgås. Fördelen med statement coverage är att koden verifieras utifrån vad den ska och inte ska utföra. Testmetoden mäter kvaliteten på koden och undersöker samtidigt olika vägar som är möjliga i koden. Vilket innebär att alla möjliga utfall blir testade.

För att utföra statement coverage i projektet kommer olika villkorssatser från olika metoder i koden väljas ut. Tanken är att olika möjliga utfall ska prövas för att se hur koden beter sig och ifall kraven uppfylls med hjälp av villkoren.

Prioritering

<Text som beskriver hur kod-moduler prioriteras för testning med denna metod. Prioriteringen ska motiveras till varför det är lämpligt att prioritera på detta vis.>

Testfall kravbaserad systemtestning

Sökning

T = Test
S = Sökning

BEHÖVER FLYTTAS TS1 Systemets användning av gps

Förberedelser: Testet ska se om applikationen hanterar enhetens gps på rätt sätt. Ifall den hämtar användarens aktuella position (latitud och longitud) och utgår från den vid sökning. I testfallet behöver applikationen vara installerad på en mobil enhet.
Teststeg:

1. Genomföra en sökning från en mobil enhet med valfritt avstånd.
2. Stänga av applikationen och rensa RAM-minnet
3. Gå till en annan position cirka 100 meter bort. Sätta på applikationen igen på samma enhet
4. Göra en ny sökning med samma avstånd som i första steget.

Förväntat resultat: Applikationen ska vid varje ny sökning hämta användarens aktuella position.

TS1 Sökning filtrerat på olika prisklasser

Förberedelser: Testet ska se om applikationen hämtar och filtrerar resultat baserat på prisklass.
Teststeg:

1. En sökning genomförs av prisklass 4 (Högsta prisklass) med valfritt avstånd.
2. Undersök resultat från teststeg 1 och genomför en ny sökning med prisklass 3.
3. Undersök resultat från teststeg 2 och genomför en ny sökning med prisklass 2.
4. Undersök resultat från teststeg 3 och genomför en ny sökning med prisklass 1
5. Undersök resultat från teststeg 4.

Förväntat resultat: Applikationen ska vid teststeg 1 visa alla möjliga och tillgängliga resultat. Vid sänkning av prisklass ska resultatet filtrera bort alternativ som tillhör den/de högre klasserna.

TS2 Sökning filtrerat på avstånd

Förberedelser: Ha telefonens platstjänster igång
Testet ska se om applikationen hämtar resultat inom angiven radie. Radiens utgångspunkt hämtas via enhetens position.
Teststeg:

1. En sökning genomförs där valfri prisklass och valfritt avstånd väljs.
2. Undersök resultat från teststeg 1 och se ifall resultatet stämmer med angivet avstånd.
3. Applikationen stängs av rensa RAM-minne.

4. Applikationen sätts på och en ny sökning genomförs med en ny utgångspunkt, minst 100 meter från första.
5. Undersök resultat från teststeg 4 och se ifall resultatet inom stämmer med angivet avstånd.

Förväntat resultat: Resultatet är endast restauranger inom användarens valda radie.

/*
*__TS# Sökning av Restaurangtyp, kategorier.__
*Förberedelser: Testet …
*/

TS3 Slumpad sökning
Förberedelser: Ha telefonens platstjänster igång
Teststeg:

1. Välj prisklass (1-4) från drop-down menyn.
2. Ange avstånd i fältet “avstånd”.
3. Klicka på knappen “slumpa”

Förväntat resultat: Information om en slumpad restaurang vars prisklass och avstånd stämmer med användarens filtreringsval visas.

Sökresultat

T = Test
S = Sök
R = Resultat

TSR1 Restaurangers betyg visas i listan av sökresultatet
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning

Förväntat resultat: Sökresultatet som listar restauranger visar betyg enligt den datan som hämtats från API.

TSR2 Restaurangers prisklass visas i listan av sökresultatet
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning

Förväntat resultat: Sökresultatet som listar restauranger visar prisklass enligt den datan som hämtats från API.

TSR3 Restaurangers avstånd från positionen användaren gjorde en sökning visas i listan av sökresultatet
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning

Förväntat resultat: Sökresultatet som listar restauranger visar avstånd till restaurangen från positionen användaren gjorde sin sökning

Detaljerad Information

T = Test
D = Detaljerad
I = Information

TDI1 Möjlighet till direkt uppringning av restaurang
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning
2. Välj restaurang i resultatlistan
3. Klicka på ikon för uppringning

Förväntat resultat: Den detaljerade vyn som visar information om vald restaurang visar en ikon för enkel uppringning av restaurang från telefonnummer som hämtats från API.

TDI2 Restaurangers adresser visas i det detaljerade sökresultatet
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning
2. Välj restaurang i resultatlistan

Förväntat resultat: Den detaljerade vyn visar specifik information om adress till vald restaurang enligt den datan som hämtats från API.

TDI3 Användaren får vägbeskrivning till vald restaurang
Förberedelser: Ha mobilens platstjänster igång och tillgång till internet
Teststeg:

1. Gör en sökning
2. Väl restaurang i resultatlistan
3. Klicka på knapp för att få vägbeskrivning

Förväntat resultat: Mobilens valda kartapp öppnas och visar vägbeskrivning till vald restaurang enligt koordinaterna som hämtats från API.

Spårningsmatris

[Annan variant på diagram kan användas eller bild infogas. Matrisen blir betydligt större och kan delas upp i flera matriser utifrån olika kategorier av krav eller testfall. Ett X nedan i matrisen indikerar att kravet testas av testfallet.]

Sökning

	FS1	FS2	FS3	FS4	<id krav>
TS1		X			
TS2	X				
TS3				X	
<id testfall>					
<id testfall>					

Direkt information

	TSR1	TSR2	TSR3		
FDiI1	X				
FDiI2		X			
FDiI3			X		
<id testfall>					
<id testfall>					

Detaljerad information

	TDI1	TDI2	TDI3		
FDeI1	X				
FDeI2		X			
FDeI3			X		
<id testfall>					
<id testfall>					

<Testfall/materiel för annan metod>

[Dokumentera andra testfall som används med andra metoder eller annan lämplig dokumentation så som exempelvis checklistor eller sammanställningar för annan typ av verifiering och validering.]

Granskningsprotokoll

[Här infogas rapporter från granskningsmöten som granskar kod eller dokument. För varje rapport ska det anges vilken artefakt som granskades, tid och datum för granskningen, vilka som närvarade vid granskningen och i vilka roller. Det behöver tas fram en mall för hur granskningsprotokoll utformas. Detta ska dock ange vilken fil och rad/dokument och stycke ett problem noterats på, typ av problem, hur alvarligt problemet bedöms vara och vems ansvar det är att åtgärda problemet om det ska åtgärdas. Där ska även finnas plats för rekommendationer som ges fr hur ett problem åtgärdas eller om någon mer generell åtgärd bör ske av exempelvis process eller liknande.]

Granskningsprotokoll <granskningstyp> <Id för protokollet>

<Här infogas protokollet.>

[Observera att det blir flera olika protokoll – minst två från kodgranskning och två från dokumentgranskning.]

Testrapporter

[Testrapporter kan dokumenteras i egna dokument om gruppen önskar.]

Testrapport Sökning TR

Id testfall	datum	Kod/dokument version	Utfört av	Resultat
TS1				
TS2	170410	1.0	Alexander Drottsgård	Förväntning för testfallet är att restauranger inom vald radie ska visas och dessa förväntningar uppfylls.
TS3				

[Det kommer sannolikt att behövas fler rader och flera olika testrapporter. Vissa justeringar i utseende av rapporten kan behöva göras beroende på testtyp.]

Testrapport Sökresultat TSR

Id testfall	datum	Kod/dokument version	Utfört av	Resultat
TSR1	170410	1.0	John Eklöf	Sökresultatet som listar restauranger visar betyg enligt den datan som hämtats från API. Dessa förväntningar uppfylls inte.
TSR2	170410	1.0	John Eklöf	
TSR3				

Testrapport Detaljerad Information DIR

Id testfall	datum	Kod/dokument version	Utfört av	Resultat
TDI1				
TDI2				
TDI3				