SKRIBL

Vrije Universiteit Brussel

WE-DINF-6537
Project Software Engineering
Academiejaar 2014-2015

Software Test Plan

Douglas Horemans <*dhoreman@vub.ac.be*>
Hannah Pinson <*hpinson@vub.ac.be*>
Ivo Vervlimmeren <*ivervlim@vub.ac.be*>
Noah Van Es <*noahves@vub.ac.be*>
Pieter Steyaert <*psteyaer@vub.ac.be*>

2 maart 2015

Versiegeschiedenis

Versie	Datum	Auteurs[1]	Beschrijving
1.0	19/11/2014	Douglas Horemans	Eerste versie
2.0	15/12/2014	Douglas Horemans	tweede versie

[1] Alle versies worden nagelezen door de Document Master (Ivo Vervlimmeren)

Inhoudsopgave

1 Introductie

SKRIBL is een sociaal netwerk voor proffen en PHD studenten.
Dit document is een test plan voor de SKRIBL web-application. Het test plan beschrijft hoe de code & features worden getest.
Vermits het SKRIBL-team agile werkt, gaat dit testplan nog sterk veranderen doorheen de iteraties. Elke test heeft een testversie omdat de tests met de iteraties mee moeten evolueren. Agile werken houdt ook in dat de vorige geslaagde test, opnieuw getest worden en indien nodig aangepast worden.

2 Referenties

- SDD versie 1.0
- Internal manual `http://wilma.vub.ac.be/~se4_1415/internalManuals/test-framework.pdf`
- Jasmine `http://jasmine.github.io/2.0/introduction.html`
- IEEE 829

3 Definities

HTML Hyper Text Markup Language. Is een taal dat de inhoud van een webpagina beschrijft.

DOM Document Object Model. Is een soort boom dat de content van een webpagine bevat.

GUI Graphical User Interface.

API Application Programming Interface.

MVC Model view controller. Is de link tussen de front-end van een web app en de server(in een Model view).

4 Domein

In het SDD, sectie 3.2, wordt de architectuur van de SKRIBL web-application beschreven. De sectie beschrijft de drie tiers dat de web-application gebruikt, namelijk de databasetier, servertier en clienttier. Deze drie tiers moeten tests implementeren, die in de sectie "Test items"worden beschreven.

5 Test items

Alle geïmplementeerde items & features moeten getest worden. De tests kunnen in verschillende groepen tests verdeeld worden. Deze groepen worden verder uitgelegd.

Item	Test versie
Databasetier tests	1.0
Servertier tests	1.0
Clienttier tests test	1.0

5.1 Databasetier tests

De database houdt alle gegevens bij die te maken hebben met users, publicaties,... In deze tier moet het toevoegen, uithalen en verwijderen van die gegevens getest worden. Het is ook belangrijk dat na het manipuleren van de database deze niet corrupt wordt.

5.1.1 Data-persistentie tests

Tests dat nagaan of data wel op de juiste manier wordt toegevoegd in de database, en daadwerkelijk bewaard blijven.

5.1.2 Data-opvraging tests

Tests die controleren of de gevraagde data teruggegeven wordt aan de server tier wanneer deze data opvraagt.

5.1.3 Data-verwijdering tests

Controleren of data op de juiste manier verwijderd wordt. Het verwijderen van bijvoorbeeld een user, moet meerdere elementen in de database verwijderen, dit word gecontroleerd.

5.2 Servertier tests

De servertier is een soort communicatie laag tussen de database en de user. De server moet HTTP requests behandelen en voor de validatie hiervan zorgen.

5.2.1 HTTP requests tests

Probeert juiste en foute HTTP requests te sturen naar de server en bekijkt het antwoord van de server.

5.2.2 Validatie tests

Deze tests proberen door middel van HTTP requests na te gaan of het valideren van data juist gebeurt.

5.3 Clienttier

De clienttier zorgt voor een user interface en het opsturen van HTTP requests naar de server.

5.3.1 Validation tests

Data die wordt opgestuurd bij een HTTP request moet aan sommige eisen voldoen.Bij elke request zijn die eisen anders.

Voorbeeld : login request verwacht:
Data = json dat username en password bevat

```
{
"username" : "skribl",
"password" : "Skribl123"
}
```

Header = moet als content type application/json hebben.

```
{
"Content-Type" : "application/json"
}
```

Validatie moet op dezelfde manier als op de server gebeuren. Het moet bij de twee tiers gebeuren voor security redenen.
Indien dit niet zou gebeuren zou injection in de database mogelijk zijn!

5.3.2 HTTP requests test

Probeert geldige HTTP requests te sturen naar de server en vergelijkt de verwachte response status met de verkregen status.

6 Software risico gevallen

De SKRIBL web-application bestaat uit drie tiers. Indien een van de tiers niet goed werkt (test falen) kunnen de andere tiers niet goed werken. Een perfecte werking van de drie tiers is nodig voor het opstarten van de web-applicatie.

7 Functies

De features die getest worden zijn:

- Login
- Logout
- Register
- Delete user
- (Publicatie toevoegen)

8 Strategie

Om onze code te testen gebruiken we Jasmine 2.0.3. Jasmine is een javascript test framework met heel veel kracht. Maar ook gebruiken we Postman, een chrome extension om HTTP requests te sturen.

8.1 Jasmine 2.0.3

Om code te testen in Jasmine moet een specification file worden aangemaakt. Deze specification file beschrijft de tests die dienen uitgevoerd te worden.
Een specification file bestaat uit een of meerdere oproepen van de describe function. Describe komt overheen met een groep tests. Deze functie verwacht als argumenten een string en een functie die effectief de tests gaat implementeren.
De tests in deze describe functie worden aangemaakt aan de hand van de it functie. De it functie verwacht ook een naam-string en een functie, maar in deze functie worden de tests beschreven.
Om tests te beschrijven gebruiken we matchers. Matchers kunnen elementen vergelijken, controleren of het element wel gedefinieerd is en nog veel anderen! Het is ook mogelijk om zelf matchers te schrijven. Voorbeelden van matchers :

- expect().toBe();
- expect().toEqual();
- expect().toMatch();
- ...

Omdat we in Jasmine matchers kunnen schrijven wordt het mogelijk om alles in javascript te controleren.

8.1.1 Tests uitvoeren

Omdat Jasmine een standalone software is , is het uitvoeren van tests heel gemakkelijk. Wanneer een specification bestand geschreven is moet het pad naar deze alleen maar toegevoegd worden in een html bestand. Wanneer dit gedaan is dient het html bestand alleen maar geopend te worden in een webbrowser. In de webbrowser geeft Jasmine de geslaagde en gefaalde tests weer. Indien een test faalt wordt zijn describe naam en it naam getoond.

8.1.2 Extra

Een manual over Jasmine werd opgesteld door de test manager voor het SKRIBL-team. Deze staat in de referenties en legt het gebruik van Jasmine in detail uit.

8.2 Postman

Postman laat toe om op een heel simpele manier HTTP requests samen te stellen en vervolgens op te sturen aan de hand van een GUI. Het is daarna het werk van de tester om te kijken of de teruggegeven status de verwachte is of niet. In principe zijn alle HTTP responses tussen 200 en 299 juist.

9 Stop criteria

Wanneer een test item voor al zijn tests slaagt, kan de code weer aangepast worden en nieuwe tests geschreven worden. Indien dit niet het geval is , worden er geen nieuwe tests geschreven. Dit voorkomt tijdsverlies.

10 Omgeving

Jasmine heeft geen bepaald environment nodig om tests te kunnen uitvoeren . Het enige dat gedaan moet worden is de library uitbreiden met de gewenste source en specification bestanden.
Daarentegen heeft Postman Google Chrome nodig, aangezien het een chrome extension is.

11 Verantwoordelijkheden

In het SKRIBL-team schrijft iedereen zijn eigen tests. Indien een lid denkt dat zijn code af is, omdat het aan al zijn tests voldoet, bekijkt de Test Manager de tests en voegt er eventueel enkelen bij. De tests worden dan opnieuw uitgevoerd en het bestand eventueel opnieuw aangepast tot het voldoet aan alle tests.

12 Goedkeuring

Wanneer de Test Manager alle tests heeft bekeken en geaccepteerd, en de code aan alle tests voldoet, moet de Quality Assurance Manager de code bekijken en deze wel of niet valideren. Indien de Quality Assurance Manager de code niet valideert moet de programmeur zijn code aanpassen tot de code aan de eisen van de Test Manager en Quality Assurance Manager voldoet.
Wanneer code wordt gevalideerd kan deze gepushed worden naar de master branch van de SKRIBL repository.