Universiteit Gent

Vakoverschrijdend project

Project Mobiliteitsbedrijf Gent

Testhandleiding

Groep 7:
Trésor Akimana
Nick De Smedt
Tom Hoet
Dean Parmentier
Simon Scheerlynck
Xavier Seyssens
Jonas Van Wilder
Lennart Vermeir

Lesverantwoordelijken:
Jan Goedgebeur
Annick Van Daele
Prof. Dr. Marko Van Dooren

2015-2016

Backend

Uitleg

We hebben zowel unit testen als integratietesten geschreven. Voor de laatsgenoemde hebben we gebruik gemaakt van DbUnit (een JUnit extensie) en een in-memory databank (nl. HSQLDB) voor de klassen die de Postgres databank aanspreken. Voor de klassen die MongoDB nodig hebben, gebruiken we Fongo (een in-memory java implementatie van MongoDB). Voor de unit testen maken we gebruik van JUnit, in combinatie met Mockito om te mocken.

Voor de namen van de testklassen volgen we de standaard, namelijk de naam van de klasse die getest wordt gevolgd door 'Test' indien het een unit test betreft (bvb. UserContollerTest), of gevolgd door 'IT' indien het een integratietest is (bvb. UserDAOIT). Bij de namen van de testmethoden hebben we geopteerd voor een vast formaat: de naam van de methode die getest wordt, gevolgd door een underscore met daarachter hetgeen we verwachten dat de test doet. (bvb. createUser_shouldCreateCorrect, getRoute_shouldReturnNull, ...).

Deze volledige test suite wordt bij ons automatisch uitgevoerd bij het builden van de applicatie op de server.

Handleiding

- Java 8 en Maven zijn vereist om de testen uit te voeren.
- Navigeer naar de directory backend/VOP7BackEnd.
- Voer **'mvn clean compile test integration-test install'** uit.
  - De code wordt nu gecompileerd en getest.
- De jar staat nu in de target directory.

Frontend

Uitleg

Om op een goede manier de frontend te kunnen testen hebben we zowel unittesten als integratietesten geschreven. Voor de unittesten hebben we de Karma-Jasmine framework gebruikt. De integratietesten hebben we met behulp van protractor geschreven.
Door het gebruik van Google Maps waren wel niet alle code testbaar. Om dit grotendeels op te lossen en ten minste onze eigen AngularJS-code te kunnen testen hebben we alle code die gebruik maakt van Google Maps in apparte entiteiten (nl. directives) geplaatst.

Handleiding

Vooraleer de frontend testen lokaal uitgevoerd kunnen worden moeten eerst Node.js en dan het testframework (Karma) genstalleerd worden. Om Node.js te installeren kan deze link worden gebruikt: .

Wanneer Node.js genstalleerd is, navigeer in de console naar de map /frontend/app/karma-jasmine-tests en voer het commando "npm installüit om de benodigde packages te installeren.

Om dan een Karma-server op te starten, voer "npm testüit. Deze testserver zal onmiddelijk de testen draaien en automatisch opnieuw telkens wanneer de bestanden gencludeerd in conf.js (in de array "files") aangepast worden. De testen genereren output in de console waar de de testserver opgestart is en coverage in de map /frontend/app/karma-jasmine-tests/coverage.
Merk op dat een groot deel (bijna alle) controllers geen unit tests hebben. Dit is omdat de google map-gerelateerde code ze ontestbaar maakt. Daarom voeren we in de plaats daarvan protractor-testen uit.
Om de Protractor-testen te starten, voer "npm protractorüit.