

13-1-2017

Gebruikershandleiding

IPSENH



RZN

REPOSITORY ZONDER NAAM

Versiebeheer

Versie	Datum	Omschrijving	Wie
0.1	10-01-2017	Opzet document	Zairon
0.2	10-01-2017	Ontwikkelstraat	Zairon
0.3	11-01-2017	Code coverage	Zairon
0.4	13-1-2017	Back-end deel	Sander

Inhoud

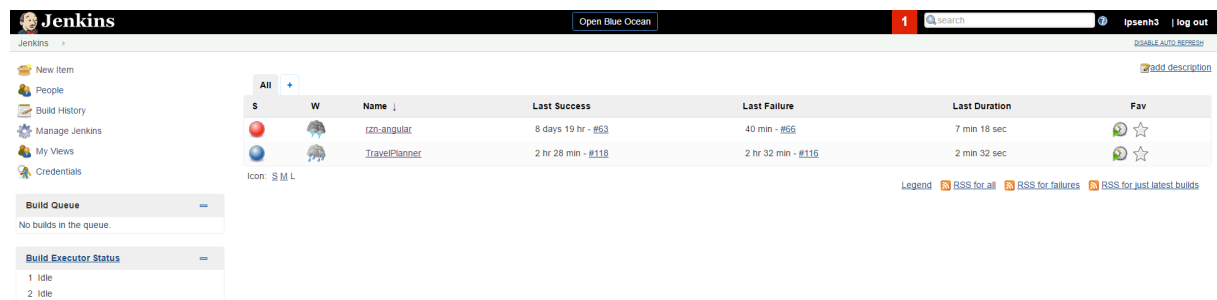
Ontwikkelstraat.....	3
Jenkins.....	3
Front-end	5
IDE	5
GIT	5
Resultaten Unit Tests Bekijken.....	5
Resultaten e2e Tests Bekijken.....	6
Code Coverage Bekijken	6
Jenkins Log	6
Back-end.....	7
IDE	7
GIT	7
SonarQube	7
IIS.....	10
Sails API	10

Ontwikkelstraat

In dit document wordt beschreven hoe een developer onze ontwikkelstraat kan gebruiken.

Jenkins

Jenkins is geïnstalleerd op de path: "C:\Users\Administrator\.jenkins". Met de server opgestart openen we de webbrowser, we gebruiken Google Chrome hiervoor. We navigeren naar localhost:8080 en inloggen met de correcte gebruikersnaam en wachtwoord. We krijgen dan het dashboard van Jenkins tevoorschijn. Jenkins draait op port 8080. Op de homepagina van Jenkins krijgen we de jobs te zien, voor de back-end(TravelPlanner) en de front-end(rzn-angular).



Hier kunnen we ook de status bekijken van de jobs, aan de linkerkant zien we "status of the build" en "weather report showing aggregated status of recent builds". Bij de eerste krijgen we een bal te zien, als deze blauw is betekent het dat het laatste build succesvol was, en als deze rood is betekent het dat het laatste build niet succesvol was. Bij de tweede status krijgen we een weer te zien, als de laatste builds gefaald zijn dan zien we regen of bliksem, als de laatste builds succesvol waren dan zien we een zonnetje.

Om alle builds te bekijken van een job kunnen we op de job klikken, vervolgens krijgen we een scherm met enkele opties en de "build history". Hier zien we alle builds met hun status erbij. Voor iedere build kunnen we de betreffende log bekijken door op de bal aan de linkerkant te klikken. Als we een build error hebben kunnen we in de log zien wat fout ging, dit zien we helemaal beneden in de log aangezien de build stopt bij het treffen van een fout.

Bij de opties kunnen we de "Changes" bekijken, dit zijn de commits die gedaan zijn en die vervolgens een build getriggert geeft. Bij "Workspace" kunnen we het project folder bekijken. Als we op "Build now" klikken dan kunnen we meteen een build starten als we dat willen. We kunnen de job verwijderen door op "Delete Project" te klikken. En om de job te configureren klikken we op "Configure", het configureren van een job is beschreven in het document Implementatie Ontwikkelstraat.

The screenshot shows the Jenkins web interface for a project named 'TravelPlanner'. The top navigation bar includes the Jenkins logo and the project name. On the left, a sidebar contains links to various dashboard features: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', 'Favorite', 'Git Polling Log', and 'Move'. The main content area is titled 'Project TravelPlanner' and includes links to 'Workspace' and 'Recent Changes'. Below these is a 'Permalinks' section with a list of links to specific build events. At the bottom left, a 'Build History' table displays the last four builds, including their IDs, status icons, and timestamps.

Project TravelPlanner

- [Back to Dashboard](#)
- [Status](#)
- [Changes](#)
- [Workspace](#)
- [Build Now](#)
- [Delete Project](#)
- [Configure](#)
- [Favorite](#)
- [Git Polling Log](#)
- [Move](#)

Permalinks

- [Last build \(#118\). 1 day 0 hr ago](#)
- [Last stable build \(#118\). 1 day 0 hr ago](#)
- [Last successful build \(#118\). 1 day 0 hr ago](#)
- [Last failed build \(#116\). 1 day 0 hr ago](#)
- [Last unsuccessful build \(#116\). 1 day 0 hr ago](#)
- [Last completed build \(#118\). 1 day 0 hr ago](#)

Build History

#118	Jan 9, 2017 10:57 AM
#117	Jan 9, 2017 10:55 AM
#116	Jan 9, 2017 10:53 AM
#115	Jan 9, 2017 10:46 AM

Als we terug naar het dashboard navigeren kunnen we op “Manage Jenkins” klikken om verschillende instellingen aan te passen. De nodige instellingen voor dit project zijn beschreven in het document Implementatie Ontwikkelsstraat.

Bij “Credentials” kunnen we credentials toevoegen of verwijderen. We gebruiken deze om te kunnen identificeren met de git repository en pullen. Bij de configuratie creëren we credentials tijdens het maken van een nieuwe job.

Front-end

Hier wordt beschreven hoe je moet omgaan met de front-end gedeelte.

IDE

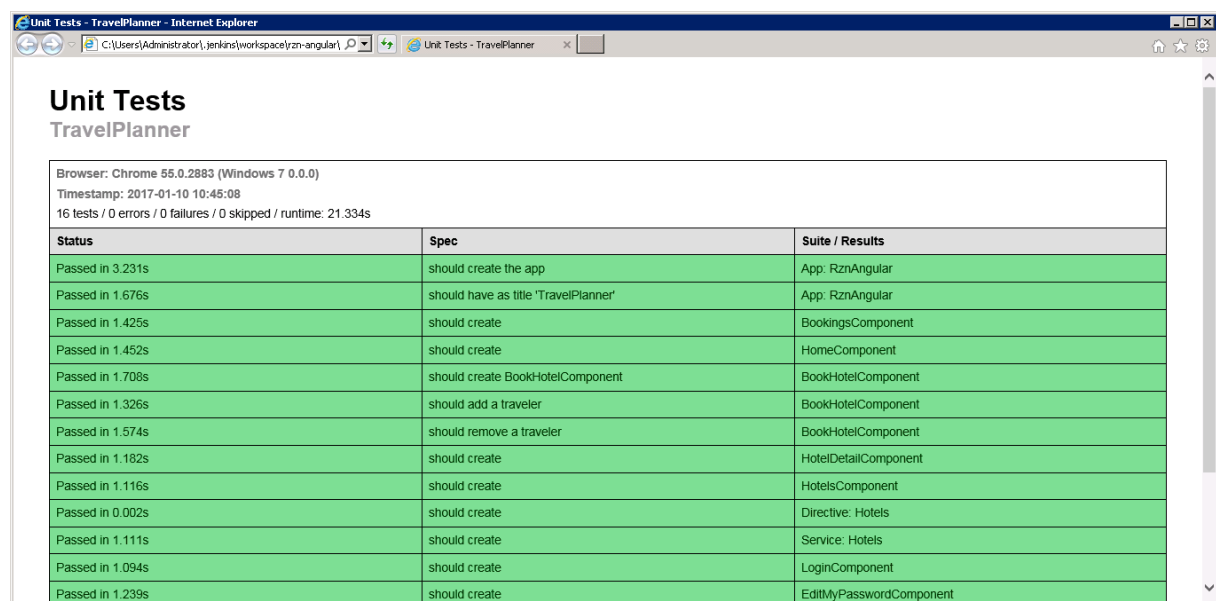
Als IDE voor de front-end gebruiken we JetBrains WebStorm. Deze installeren we op onze lokale machine. We halen het project met GIT van de repository en openen deze met WebStorm. Na het maken van changes in de IDE kunnen we in WebStorm met de ingebouwde GIT functionaliteit deze commiten en een push request doen naar de gewenste branch op de repository.

GIT

Voor de front-end hebben we een repository op GitHub. Deze is te vinden op <https://github.com/antonsteenvoorden/rzn-angular>. De flow om met GIT te werken gaat als volgt, changes worden gedaan en gepushed naar een nieuwe branch, vervolgens wordt er een pull request gedaan naar andere ontwikkelaars die deze kunnen bekijken en controleren (code review). Als de changes geaccepteerd worden kan de branch gemerged worden in de branch development. Development is onze standaard branch. Jenkins gaat vervolgens het build proces starten.

Resultaten Unit Tests Bekijken

Om de resultaten van de unit tests te bekijken kunnen we in de project folder met de commando `"npm run unittest-report"` de resultaten in de webbrowser bekijken, deze gaat een httpserver draaien met een html-file die als inhoud de resultaten van de tests bevat. We kunnen hier zien welke tests succesvol waren of niet, en hoe lang de test genomen heeft. We stoppen de server met `"ctrl c"`.



Status	Spec	Suite / Results
Passed in 3.231s	should create the app	App: RznAngular
Passed in 1.676s	should have as title 'TravelPlanner'	App: RznAngular
Passed in 1.425s	should create	BookingsComponent
Passed in 1.452s	should create	HomeComponent
Passed in 1.708s	should create BookHotelComponent	BookHotelComponent
Passed in 1.326s	should add a traveler	BookHotelComponent
Passed in 1.574s	should remove a traveler	BookHotelComponent
Passed in 1.182s	should create	HotelDetailComponent
Passed in 1.116s	should create	HotelsComponent
Passed in 0.002s	should create	Directive: Hotels
Passed in 1.111s	should create	Service: Hotels
Passed in 1.094s	should create	LoginComponent
Passed in 1.239s	should create	EditMyPasswordComponent

Resultaten e2e Tests Bekijken

Om de resultaten van de e2e tests te bekijken kunnen we in de project folder met de commando “npm run e2e-report” de resultaten in de webbrowser bekijken, deze gaat een httpserver draaien met een html-file die als inhoud de resultaten van de tests bevat. We kunnen hier zien welke tests succesvol waren of niet, en hoe lang de test genomen heeft. We krijgen ook een screenshot van de webbrowser te zien bij de test. We stoppen de server met “ctrl c”.

Code Coverage Bekijken

Om de code coverage te bekijken kunnen we in de project folder met de commando “npm run coverage-report” de code coverage bekijken, deze gaat ook een httpserver draaien met een html-file die als inhoud de code coverage bevat. We stoppen de server met “ctrl c”.

79.77% Statements 462/581 40.66% Branches 37/92 61.59% Functions 85/138 77.94% Lines 324/554

File	Statements	Branches	Functions	Lines
src/	100%	29/29	100%	4/4
src/app/	100%	94/94	50%	9/9
src/app/bookings/	75%	27/36	92.31%	4/8
src/app/guards/	93.33%	14/15	50%	3/3
src/app/home/	90.91%	10/11	100%	3/4
src/app/hotels/	72.64%	77/106	7.32%	14/23
src/app/hotels/hotel-detail/	81.25%	26/32	16.67%	5/7
src/app/hotels/hotel-detail/book-hotel/	81.67%	49/60	50%	10/14
src/app/login/	92.31%	12/13	100%	3/4
src/app/models/	54.29%	19/35	100%	4/8
src/app/my-profile/	83.33%	15/18	50%	2/4
src/app/my-profile/edit-my-password/	83.33%	10/12	100%	3/4
src/app/my-profile/edit-my-profile/	100%	7/7	100%	3/3
src/app/page-not-found/	100%	7/7	100%	3/3
src/app/register/	80.77%	21/26	100%	5/9
src/app/services/	60.66%	37/61	50%	6/16
src/app/weather/	65%	26/40	100%	4/15
src/environments/	100%	1/1	100%	0/0

Jenkins Log

Wanneer een stap van de build proces niet succesvol was, stopt de hele build proces voor de front-end. Alleen voor het linten van de source code geldt dit niet. Bij een niet succesvol build willen we de log ervan bekijken om te zien wat fout ging. Zoals eerder beschreven hoe je een log kan bekijken, gaan we nu naar de desbetreffende log en helemaal beneden in de log kunnen we zien welke stap van de build proces niet succesvol was, en wat precies fout ging. Omdat de build meteen stopt kan je altijd helemaal beneden in log de fout vinden.

Back-end

Hier wordt beschreven hoe je moet omgaan met de back-end gedeelte.

IDE

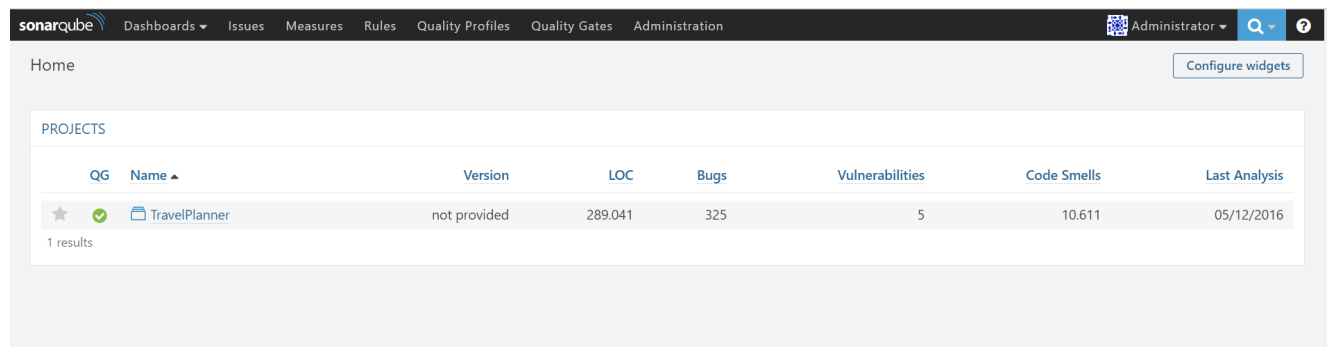
Voor de back-end gebruiken we Visual Studio. Visual Studio heeft een ingebouwde Team Explorer waar alle git commando's uitgevoerd kunnen worden. Via een knop op de website waar de repository staat, kan gemakkelijk de repository lokaal gekloond worden. Via de Team Explorer kunnen dan changes worden gemaakt.

GIT

De repository van de back-end is te vinden op <http://fortytwowindmills.visualstudio.com>. Hier dient ingelogd te worden, waarna er naar de repository van onze groep gegaan kan worden. Changes worden gedaan en gepushed naar een nieuwe branch, vervolgens wordt er een pull request gedaan naar andere ontwikkelaars die deze kunnen bekijken en controleren (code review). Als de changes geaccepteerd worden kan de branch gemerged worden in de branch development. Development is onze standaard branch. Jenkins gaat vervolgens het build proces starten.

SonarQube

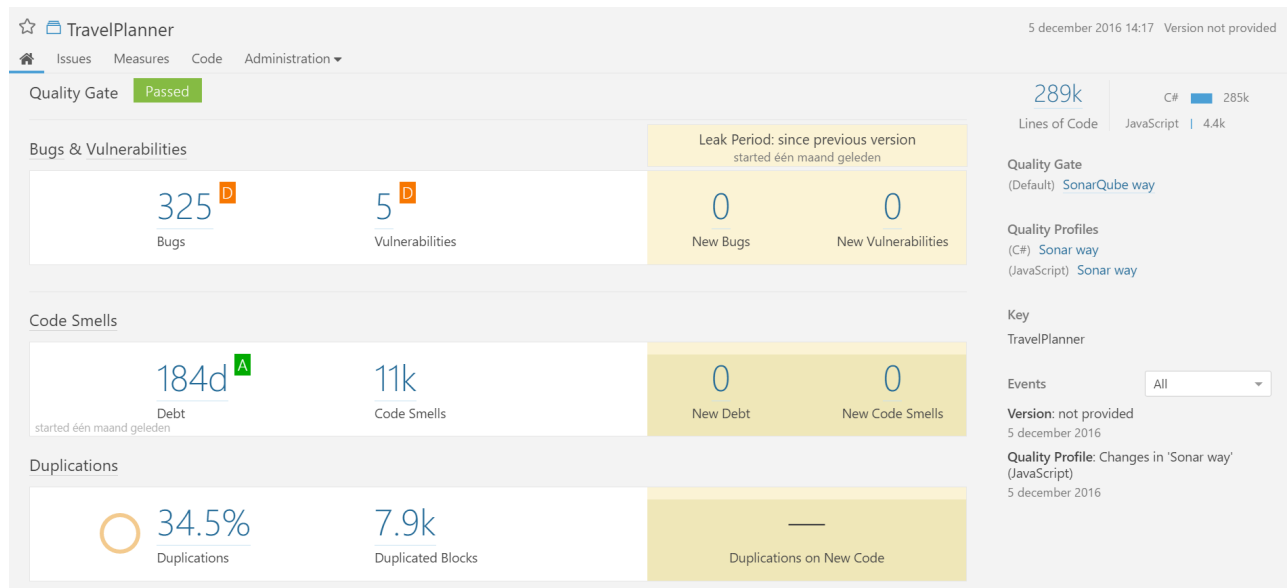
SonarQube is een tool waarmee de code kwaliteit gemeten kan worden van verschillende projecten. De gegevens van de metingen worden opgeslagen in een PostgreSQL database. De gebruikersinterface van SonarQube is te vinden op poort 9000. Door te surfen naar localhost:9000 kom je in de gebruikersinterface.



The screenshot shows the SonarQube web interface. At the top, there is a navigation bar with links for Dashboards, Issues, Measures, Rules, Quality Profiles, Quality Gates, and Administration. The user is logged in as Administrator. The main content area is titled 'Home' and features a 'Configure widgets' button. Below this, there is a section titled 'PROJECTS' which contains a table with the following columns: QG, Name, Version, LOC, Bugs, Vulnerabilities, Code Smells, and Last Analysis. A single project, 'TravelPlanner', is listed with a status of 'not provided', 289,041 LOC, 325 bugs, 5 vulnerabilities, 10,611 code smells, and a last analysis date of 05/12/2016. A '1 results' indicator is shown at the bottom left of the table.

QG	Name	Version	LOC	Bugs	Vulnerabilities	Code Smells	Last Analysis
★	TravelPlanner	not provided	289,041	325	5	10,611	05/12/2016

Hier is een overzicht te zien van alle projecten die gescand zijn met SonarQube. Er kan meer informatie over een project ingezien worden door op de naam van het project te klikken. Er opent zich een nieuw venster met een overzicht van de belangrijkste gegevens.



Zoals in dit voorbeeld te zien is, is de kwaliteitstest geslaagd. Ookwel de 'Quality Gate' genoemd. Deze is zelf naar wens in te stellen onder het administratiemenu. SonarQube heeft standaard een quality gate opgesteld. Deze moet per project aangepast worden naar de eisen van dat project. Naast de quality gates heeft SonarQube standaardregels opgesteld met betrekking tot specifieke programmeertalen. Deze regels zijn ook naar wens aan te passen in het administratiemenu.

De server staat zo ingesteld dat er elke week op dinsdag om 14:00 een script start die van de laatste Jenkins build een volledige scan maakt. De gegevens van deze scan worden opgeslagen in de PostgreSQL database en kunnen worden ingezien en geback-upt waar nodig.

Het administratiemenu kan worden benaderd nadat er is ingelogd met het administrator account. De inloggegevens van dit account staan standaard op id: Admin wachtwoord: admin.

Administration

Configuration ▾ Security ▾ Projects ▾ System ▾

General Settings

Edit global settings for this SonarQube instance.

3D Code Metrics

Analysis Scope

C#

General

Java

JavaScript

Scanner for MSBuild

SCM

Security

Technical Debt

Database Cleaner

Keep only one snapshot a day after

After this number of hours, if there are several snapshots during the same day, the DbCleaner keeps the most recent one and fully deletes the other ones.

24

(default)

Key: sonar.dbcleaner.hoursBeforeKeepingOnlyOneSna...

Clean directory/package history

If set to true, no history is kept at directory/package level. Setting this to false can cause database bloat.

☒

(default)

Key: sonar.dbcleaner.cleanDirectory

Keep only one snapshot a week after

After this number of weeks, if there are several snapshots during the same week, the DbCleaner keeps the most recent one and fully deletes the other ones

4

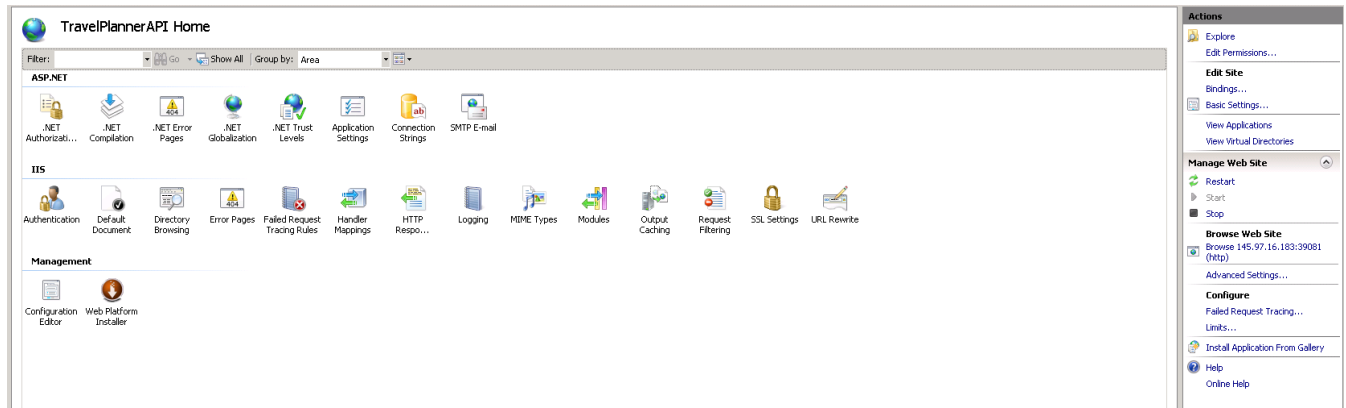
(default)

Key: sonar.dbcleaner.weeksBeforeKeepingOnlyOneSna...

In principe hoeft hier niets te veranderd worden, maar indien nodig kunnen alle SonarQube instellingen vanuit hier beheerd worden.

IIS

Om de backend te laten draaien maken we gebruik van IIS 7.5 met extra geïnstalleerde functies. Er is een snelkoppeling op het bureaublad van de server gemaakt naar de gebruikersinterface van IIS. Aan de linkerkant zijn de connection lijst. Wanneer je deze openklapt worden twee sites getoond. De Travelplanner API wordt op poort 5555 gedraaid en de back-end API draait op poort 39081. Vanaf hier kunnen verschillende instellingen voor beide websites aangepast worden. Tevens worden logbestanden bijgehouden. Deze zijn terug te vinden in het logs mapje op het bureaublad.



Sails API

In de tussentijd hebben we een Sails API gebruikt bij de front-end, omdat de back-end niet volledig klaar was. Deze is geïnstalleerd als een service op de server en is te benaderen op port 1337. Om de api te gebruiken kunnen we naar services gaan en zoeken voor "TravelPlanner API" en we starten deze service. We kunnen het ook opstarten door te gaan naar "C:\Users\Administrator\travelplanner-api" en hierin voeren we de commando "node app.js".