***Disclaimer* | De lay-out van de PDF-versie voor dit document kan verschillen met de markdown versie, voor een accurate weergave zie markdown bestand in bitbucket.**

Testplan

- Inleiding
- Wanneer worden er testen geschreven?
- Waarvoor worden de testen geschreven
- Wat niet getest wordt
- Benodigdheden
- Hoe worden de testen geschreven.
- Test volledigheid

Inleiding

Om te controleren of de requirements uit het FO juist geïmplementeerd zijn moeten er tests geschreven worden.

In dit document wordt beschreven hoe, wat en waarover testen geschreven worden.

Wanneer worden er testen geschreven?

- Voor dit project worden unittests en E2E (End-to-End) testen geschreven.
- Unittests worden geschreven voor iedere stored procedure, trigger en check-constraint dat in dit project wordt gemaakt in MSSQL Server.
- Unittests worden gemaakt nadat de code zelf is geschreven om de code te testen.

Daarnaast worden er testen geschreven om te controleren of de bestaande gebruikers dezelfde rechten hebben zoals beschreven in het TO.

Waarvoor worden de testen geschreven

Alle code is getest op minimaal één success scenario en minimaal één edge-case.

Als voorbeeld met de check: leeftijd >= 18

- Het successcenario moet passen wanneer de leeftijd 25 is.
- De edgecase moet falen wanneer de leeftijd lager is dan 18.

Er wordt gecontroleerd of de afgesproken exceptions gegooid worden wanneer ze moeten, daarbij wordt ook gecontroleerd of er exceptions gegooid worden wanneer dat juist niet moet.

Wat niet getest wordt

- De insert scripts voor de mockdata worden niet getest.

Benodigdheden

Er is een specifieke dev-test database die de normale database nabootst.

Voor het schrijven van de testen wordt het tSQLt framework gebruikt.

No.	Resources	Descriptions
1.	Server	Een MSSQL database en een MongoDB staging area

No.	Resources	Descriptions
2.	Test tool	tSQLt.NewTestClass om een test tabel aan te maken en tSQLt.RunTestClass om de onderdelen te testen
3.	Netwerk	Alle testen worden lokaal getest

Hoe worden de testen geschreven.

Wanneer de tests worden aangemaakt moet je eerst vanuit tSQLt een testklasse aanmaken.

De testklasse heeft als naam de volledige naam van de functionaliteit + vooraan test_ (bijv test_TR_auto_update_manager_income of test_CH_player_older_than_18).

Als er meer dan twee tests zijn dan wordt een [SetUp] procedure toegevoegd aan de klasse waarin de fake tables worden opgezet.

De testen zelf worden geschreven als [testklasse].[test [naam van test]]

Testen zijn stored procedures.

De volgorde van de code in de test is:

- ARRANGE: voorbereiden test (verwachte tabel vullen)
- EXPECT: error (welke) of geen error
- ACT: de geteste actie
- ASSERT: wat zou het scenario moeten zijn.

De template van een losse test is:

```
CREATE OR ALTER PROCEDURE [testklasse].[test [naam test]]
AS
BEGIN
    --ARRANGE

    --EXPECT

    --ACT

    --ASSERT
END
```

Test volledigheid

Om testvolledigheid te controleren worden eerst alle unittests uitgevoerd om te kijken of het slagingspercentage 100% is. Zodra dit het geval is worden de resultaten van de testen in het testrapport uitgewerkt met bewijs van het slagen van de tests.