# Testing strategy

Swop groep 12:
Jeroen De Coninck,
Tom De Bie,
Jeroen Van Gool,
Ruben Lapauw

## 1 Overzicht

Als teststrategy voor onze software maken we vooral gebruik van white-box testing met een een stukje black-box testing. White-box testing gebeurt met de interne kennis van het systeem, en test specifieke functies van de interne software. Terwijl black-box testing het systeem van buitenaf benaderd en aanvragen op het systeem uitvoert. Als black-box testing doen we enkele manuele tests via de UI, en testen we de belangrijke API-calls automatisch. Onze white-box testing strategie bestaat uit het testen van de belangrijkste methodes in het systeem. En deze onder zo veel mogelijk druk te zetten zodat eventuele randcondities en uitzonderingsgevallen duidelijk worden.

Alle usecases worden ook specifiek getest op een volledig verloop, deze wordt gedaan met black-box testing. Alle stappen worden "uitgevoerd" en er wordt gecontroleerd of deze een juiste uitvoer hebben. Dit zal ook manueel in de UI gedaan worden indien de API nog niet beschikbaar is. Er moet voor iedere preconditie getest worden of deze gecontroleerd wordt en of deze correct wordt afgehandeld. Dit gebeurt door juiste, verkeerde en geen informatie in te voeren.

## 2 Use cases

### 2.1 Prescribe treatment

De eerste preconditie, "er moet een doctor aangemeld zijn", kan niet getest worden vanwege het design. Deze wordt altijd automatisch afgedwongen. Aangezien een DoctorController de toegang tot deze usecase beheert, deze kan enkel verkregen worden door zich als doctor aan te melden. Er moet wel een controle uitgevoerd worden dat deze doctor niet al uitgelogd is.

De drie andere precondities moeten getest worden door een doctor zonder patientfile, de patientfile van een ontslagen patient of een patientfile zonder diagnose te gebruiken.

Na het testen van de precondities overlopen we de stappen van de usecase. Iedere stap wordt getest of dat alle uitvoerdata juist de verwachte data bevat.

#### 2.1.1 Flow

1. Het aanroepen van de usecase kan niet getest worden, dit stukje behoort tot de UI.

2. List of available treatments: Er wordt getest of er een aantal treatments bestaan. In de unit-testen van de verschillende treatments moet getest worden of deze beschikbaar zijn voor de UI.

3. Het selecteren van de treatment behoort opnieuw toe aan de UI.

4. Request treatment-input: Hier moet getest worden wat er gebeurt met foutieve invoer. Er kan niet getest worden of de invoer juist is aangezien men niet weet welke treatments beschikbaar zijn. Dit behoort toe aan de unit-testen van de treatments zelf, de uitvoer moet ook getest worden door de unit-testen. En de abstractie-laag kan getest worden met een vaste treatment te gebruiken, maar deze fouten zullen snel aan het licht komen in de UI.

5. Invoer van data behoort toe aan de UI. Hier worden Arguments gebruikt die moeten getest worden in hun eigen unit-testen.

6. De creatie van de treatment moet getest worden. Aangezien er geen enkele manier is om via de API te controleren of deze aangemaakt is, moet dit met een unit-test gebeuren. Er moet wel getest worden of de invoer kan gebroken worden. Er moet met interne unit-testen gecontroleerd worden of deze appointments juist gemaakt worden.

7. Het weergeven van de treatment behoort toe aan UI. Deze kan getest worden door een willekeurige treatment te nemen, in combinatie met het unit-testen van alle treatments.

#### 2.1.2 Alternate flow #1 en #2

6. (a) De diagnose is niet goedgekeurd: Er moet voor deze randconditie met white-box unit-testen gecontroleerd worden of deze niet geplanned is.

6. (b) Niet genoeg voorraad: Er moet voor deze randconditie met white-box unit-testen gecontroleerd worden of deze niet geplanned is.

### 2.2 Approve Diagnosis

Zoals bij de Prescribe treatment use case wordt de eerste preconditie, het ingelogd zijn van een dokter, impliciet afgehandeld door het feit dat de use case enkel gestart kan worden vanuit een DocterController. De tweede preconditie, "de patient aan wiens dossier de dokter aan het werken is is nog niet ontslagen", wordt ook op dezelfde manier afgehandeld als bij Prescribe treatment.

Daarna overlopen we weer de verscheidene stappen die de use case overloopt:

#### 2.2.1 Basic Flow

1. Het aanroepen van de usecase wordt weer afgehandeld in de UI.

2. Lijst van diagnoses die een second opinion nodig hebben: Voor het starten van de use-case worden er enkele van deze diagnoses aangemaakt voor deze en andere dokters, men kan dan in deze stap controleren of de juiste diagnoses getoond worden.

3. Selecteren diagnose behoort opnieuw toe aan de UI.

4. Goedkeuring diagnose, opnieuw in de UI.

5. Opslaan beslissing en planning behandeling: We controleren of het diagnose-object gemarkeerd is als goedgekeurd en indien er een behandeling aan vasthangt, of deze behandeling correct gepland is (correct zijnde op een moment dat de patient en benodigdheden vrij zijn op dat tijdstip en niet vroeger dan het huidige tijdstip).

6. Weergave behandeling, opnieuw in de UI.

#### 2.2.2 Alternate Flow

5. (a) Afkeuring diagnose

    1. Afkeuring diagnose, opnieuw in de UI.
    2. Opslaan beslissing: Controleren of het diagnose-object gemarkeerd is als ongeldig.
    3. Nieuwe diagnose: We kijken na of de nieuwe diagnose een second opinion vereist van de dokter die de eerste diagnose gesteld heeft, de rest van deze stap wordt afgehandeld in de tests van de use case Enter diagnosis.