End to end testing van android app

Verslag

VAK ITFactory

Academiejaar 2021-2022

Arnaud Provoost 3app/ai 02

Campus Geel, Kleinhoefstraat 4, BE-2440 Geel

LID VAN ASSOCIATIE KU LEUVEN

THOMAS MORE

INHOUD

INLEIDING

Het doel van deze stage was het schrijven automatische regressie testen voor een Android app.

Als start van de stage moest ik een proof of concept maken voor welke framework ik ging kiezen voor het schrijven van de testen. Bij de ontwikkeling hiervan heb ik samenspraak met de developer van QAdvice de programmeertaal gekozen. Nadat de POC goedgekeurd was hebben de code erachter verbeterd, zodat ze efficënter werkte.

Na deze POC ben ik dan begonnen met het schrijven van de eerste test case. Eerst als tekst en dan in code omzetten. Hierna werd mijn code verbeterd door mijn stagementor en een developer van het team.

Een nice to have was het kunnen draaien van deze testen op de jenkins pipeline. Doordat er geen hardware acceleratie op de pipeline kan gedraaid worden is dit niet gelukt.

DANKWOORD

Het was super mogelijkheid om voor QAdvice te mogen werken aan een opdracht voor De Lijn. Alle twee zijn het heel toffe bedrijven om voor te werken met goede groepssfeer en leuke teamactiviteiten. Daarvoor wil ik heel graag De Lijn en QAdvice bedanken. En in het bijzonder team TCE die mij direct thuis hebben laten voelen.

Ook wil ik graag nog is Gerry Meyen en Indy Naessens extra bedanken voor de hulp en grappige verhalen die zij aanboden tijdens de dagen op het kantoor.

Jelle Polders wil ik ook graag bedanken voor de opportuniteit om voor QAdvice te mogen werken.

Ook wil ik graag de docenten van Thomas More bedanken die mij het soms toch moeilijk maakten deze laatste drie jaar.

Als laatste wil ik ook alle andere collega's van De Lijn bedanken die mij geholpen hebben tijdens deze drie maanden in het Lijnhuis.

TERMINOLOGIE

- TCE: Team tevreden chauffeur en exploitant.
- MTC: Mobiele toepassing en controleurs.
- Test: De test omgeving die wordt gebruikt om nieuwe implementaties te testen.
- UAT: User acceptance testing. Omgeving waar business de implementaties test.
- Productie: De omgeving die open staat voor de buitenwereld/klanten.
- Indy: Mijn collega developer.
- Gerry: Mijn mede tester en stagementor.
- CCG: Controle gegevens die moeten ingevuld worden bij een controle.

INTRODUCTIE

Testing wordt nog door sommige IT'ers gezien als iets dat niet altijd hoeft en eigenlijk overbodig is. Maar als je kijkt naar de hoeveelheid projecten die naar buiten gaan met bugs zou je denken dat ze inzien dat testing heel belangrijk is.

Bij De Lijn wordt testing wel als heel belangrijk gezien. Er mag niets naar buiten zonder in test en UAT getest geweest te zijn. Doordat de nieuwe implementaties door deze twee omgevingen door moeten is de kans een stuk lager op bugs in productie.

Doordat je de bugs vroeger vindt in het project kosten ze minder om aan te passen. De manuele testen nemen echter enorm veel tijd in waardoor sommige implementaties lang in test/UAT blijven om pas later in productie te geraken.

De oplossing om deze testen sneller te laten lopen is het automatiseren van de regressie testen. Daarom was mijn stageopdracht het schrijven van de automatische testen voor de MTC-app.

De MTC-app wordt door een extern bedrijf geleverd. Daarom moeten we bij elke nieuwe update van de app deze regressietesten runnen zodat we zeker zijn dat de app nog altijd doet wat hij ervoor deed.

Deze testen moesten normaal gezien op Jenkins draaien zodat ze elke week gerund konden worden, maar doordat hardware acceleratie niet mogelijk is in Jenkins kan er geen emulator opgezet worden in Jenkins waardoor je ook de testen niet kunt runnen. Deze testen gaan hierdoor gedraaid worden lokale op een laptop van het team.

DE LIJN EN QADVICE

QADVICE

QAdvice is een bedrijf dat opgericht is geweest in 2012 door Jelle Polders. In de beginjaren lag de focus op de freelance werkzaamheden van Jelle Polders. Na het opbouwen van een stabiel klantenbestand en het leggen van sterke financiële basis, begon het bedrijf uit te groeien tot het team dat ze nu zijn.

QAdvice is gespecialiseerd in soft- en hardware testen met een sterke focus op testautomatisering. Ze hebben een hele waaier aan bedrijven en sectoren waar ze actief in zijn, zoals het bankwezen, de publieke sector en de elektronica sector.

DE LIJN

De Lijn is in 1991 ontstaan uit de opsplitsing van de Nationale Maatschappij van Buurtspoorwegen in een Vlaams en Waals deel. Het Vlaamse deel werd dan gefuseerd met de Maatschappij voor het intercommunaal Vervoer te Antwerpen en de Maatschappij voor Intercommunaal Vervoer te Gent.

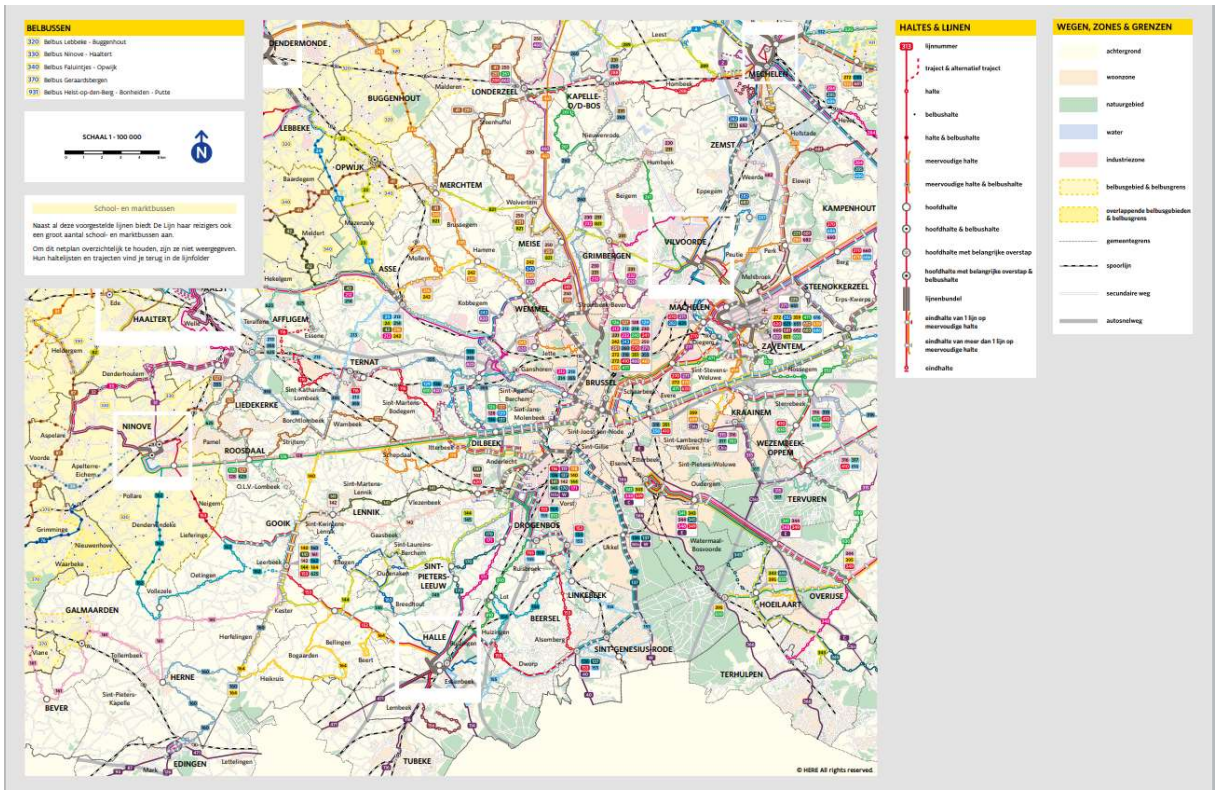
De zetel van het De Lijn bevindt zich in het Lijnhuis te Mechelen. Daar is mijn stage ook doorgegaan. Ook heeft De Lijn 5 exploitatie-entiteiten. In het lijnhuis bevinden zich ook het grootste deel van de Centrale Diensten.

HET LIJNHUIS IN MECHELEN

Het aanbod van De Lijn bestaat uit vaste lijnen met haltes en vertrek- en aankomsttijden. Niet alleen heeft De Lijn vaste lijnen, in sommige landelijke gebieden worden er belbussen ingezet. Deze bussen rijden alleen wanneer deze gereserveerd zijn. Ook heeft De Lijn in sommige steden in Vlaanderen tramnetten.

De Lijn heeft heel veel verschillende soorten lijnen. Zij hebben lijnen die door het centrum van steden gaan. Tussen deze steden rijden er streeklijnen. Ook rond de grote steden zoals Antwerpen, Brussel en Gent rijden er voorstedelijke ritten.

Voor bijna de helft van de ritten worden er exploitanten ingezet. Dit wil zeggen dat De Lijn bedrijven inzet om de ritten uit te rijden. Deze bedrijven zijn meestal kmo's die ook ander vervoer ernaast aanbieden. Deze exploitanten moeten alle regels van De Lijn volgen en zijn bijna onzichtbaar voor de klant.



LIJNEN ROND BRUSSEL

ROLLEND MATERIEEL

Trams: De Lijn heeft iets minder dan 400 trams in bezit. De meeste trams rijden op de premetro lijnen in Antwerpen. De oudste trams, de PCC's, worden geleidelijk uit dienst gezet.

LAGEVLOERTRAM IN GENT

Bussen: De Lijn heeft veel verschillende bussen en telt een totaal van 2240 bussen. Het grootste deel van deze bussen rijdt op diesel, maar De Lijn heeft besloten om vanaf 2019 alleen maar bussen te kopen die op hernieuwbare energie rijden.

Andere: Naast bussen en trams heeft De Lijn ook andere rollend materieel zoals de trambussen en trolleybussen.

MTC-APP

MTC-APP

De MTC-app is de app die zich bevindt op de tablet van alle controleurs van De Lijn. Deze app bevat alles wat de controleur tijdens zijn controles op de voertuigen van De Lijn nodig zou kunnen hebben.

Deze app wordt door een extern bedrijf geleverd aan De Lijn. Daardoor heeft men geen toegang tot de code base. Deze app is in react native geschreven dus werkt niet in met een browser.

VERSLAGEN

De app bevat veel verschillende verslagen. Op volgende afbeelding kan je alle verschillende verslagen zien die getest zijn. Al deze verslagen bevatten enorm veel invoervelden die moeten getest worden.

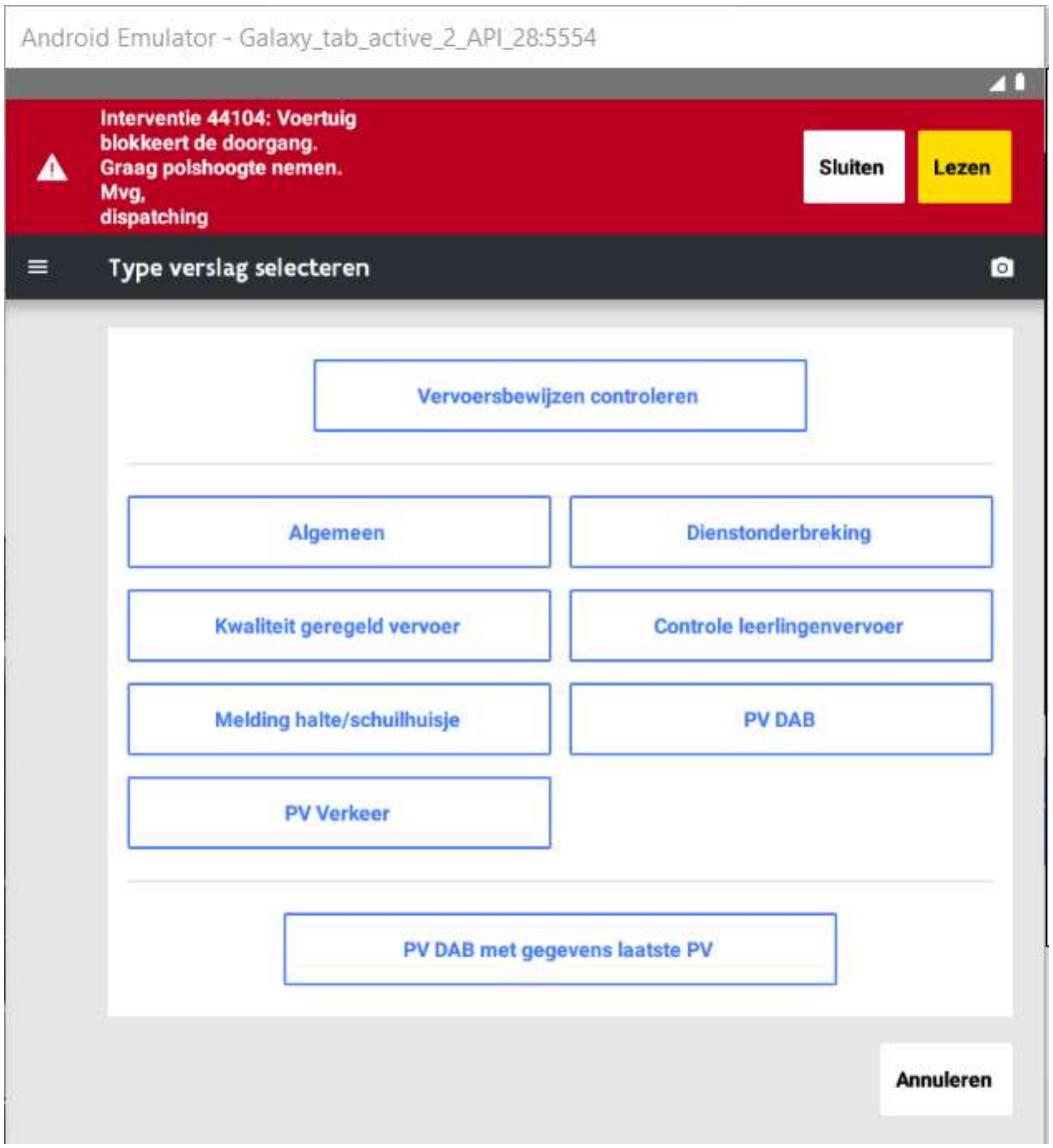


FOTO VAN MTC-APP MET ALLE VERSLAGEN OP

Ook worden de vervoersbewijzen gecontroleerd aan de hand van deze app. Hierdoor heb ik ook andere apps moeten gebruiken in de test omgeving. Een voorbeeld is de website om tickets aan te vragen en activeren.

PROJECT SCOPE

OMSCHRIJVING

Mijn project is 28 maart ‘22 begonnen en eindigt op 27 mei ’22. Het is verdeeld in 3 grote delen.

Het eerste deel is het opmaken van de POC voor de automatische testen. Aan de hand van die POC kon ik kiezen welk framework ik ging gebruiken voor de testen. Tijdens deze POC moest er ook een emulator opgezet worden op zowel mijn laptop als die van mijn begeleider Gerry. Daarom heb ik een gebruikershandleiding geschreven voor het lopen van mijn testen.

Het tweede deel is het schrijven van de test cases. De planning hiervoor werd gegeven door mijn stagementor. Hij koos eerst de verslagen die veel tijd innamen. En zo werd er dan een verslag afgewerkt. Nadat een verslag werd afgewerkt keken Gerry en Indy de automatisatie na.

In volgende tabel kan u vinden aan welk verslag ik heb gewerkt en in welke volgorde. Ook kan u zien hoe lang dit heeft geduurd. Als u meer uitleg wilt over wat ik elke week deed kan u op mijn website kijken naar mijn logboek.

	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13
POC met emulator													
01 inlog													
02 algmeen verslag													
03 pvdab													
04 pvverkeer													
05 dienstonderbreking													
06 kwaliteit geregeld vervoer													
07 controle leerlingen Vervoer													
08 melding schuilhuise en haltepaal													
09 controle vervoersbewijzen													
10 hamburgeritem													
Testen in Jenkins draaien met docker													

Als derde deel was het draaien van deze testen in een Docker container op Jenkins. Dit is uiteindelijk niet gelukt, maar we zoeken samen met het team nog naar een oplossing.

GELEVERDE ONDERDELEN

1. POC na 1 week
   Bevat:
   Gekozen framework
   Documentatie voor installeren emulator
2. Test Suites:
   Bevat:
   Alle test suites voor de verslagen geschreven en werkend gekregen.

En zo verder… Elk verslag was een ticket in het agile-board. Nadat ik mijn testen geschreven had maakten ik een pull request open. Deze werd dan verbeterd door Gerry en Indy.

3. Eind report
   Bevat:
   Documentatie op Confluence
   Testen op Bitbucket

Alles wat hierboven beschreven is, is opgeleverd. De testen worden later ook meer in diepte besproken.

POC

Emulator installeren op Windows

Introductie

Voor de POC te kunnen draaien heb ik een werkende emulator nodig waarop ik de app heb geïnstalleerd. Als eerste stap heb ik dan tutorials opgezocht over het opzetten van een Android emulator op Windows toestellen.

De eerste stap was het gebruiken van Android studio om alle nodige sdk's enz. te downloaden en te installeren. Deze emulator kon ik dan opstarten en mee verbinden van op mijn laptop.

Eens dat de emulator up and running was heb ik opgezocht hoe ik de app op de emulator kreeg. Dit bleek wat moeilijker te zijn dan gewoon een apk bestand te installeren op de emulator. Ik kreeg namenlijk geen apk bestand maar een aab file. Dit is een soort file die play store verwacht. Maar omdat dit een privé app is en dit zich alleen op de play store van De Lijn zelf bevindt moest ik daar zelf een apk bestand van maken.

Hiervoor heb ik een tool gevonden deze vormt de aab file om naar een apks bestand. Dit bestand bestaat uit verschillende apk's eentje voor elke mogelijkheid van hardware. Dit zorgt ervoor dat elk apk bestand alleen maar de packages bevat die nodig zijn voor je toestel.

Maar voor emulator gebruik ik dezelfde tool met gewoon andere opties. Zo vorm ik het aab bestand naar een apks bestand met een universal apk. Dit is een apk dat alle packages bevat, wat hem natuurlijk veel groter maakt. Maar dan maakt het niet uit welke emulator je gebruikt, met welke opties voor de hardware enz.

Nadat ik dan deze apk had gemaakt heb ik hem geïnstalleerd op de emulator. Nadien ben ik begonnen aan de 'echte' POC.

Testing Frameworks voor Android testing

Introductie

Er zijn 3 grote testing frameworks voor Android testing: Appium, Espresso en Detox. Alle drie hebben ze positieve en negatieve punten. Uiteindelijk heb ik geen keuze gehad over welk framework ik kon kiezen. Maar later hier meer over.

Omdat de app niet door De Lijn zelf is gemaakt is het moelijker om aanpassingen te vragen. De meeste van de invoervelden bevatten geen id. Waardoor ik naar de invoervelden moet zoeken op tekst. Wat voor onstabiliteit zorgt.

Mogelijke frameworks

- Appium: Is een open source test automation framework voor zowel native, hybrid en mobile web apps. Appium kan met heel veel verschillende talen werken waardoor je ook niet verbonden bent met één bepaalde taal. Appium werkt aan de hand van een server die dan met je emulator of echt toestel verbindt en aan de hand van deze server dan commands kan doorsturen naar de emulator. Met appium kan je een desktop app gebruiken om buttons enz op je app te vinden. Appium is ook een black box testing framework wat inhoud dat hij nooit rekening houd met de state van de app. Hij communiceert ook nooit direct met de app maar altijd via de emulator.

- Espresso: Is ook een black box testing framework maar je kan de framework alleen optimaal gebruiken wanneer je de codebase kent/hebt. Als je de codebase wel hebt zullen de testen

met espresso veel sneller runnen dan appium. Het is ook makkelijker om te leren dan appium. Espresso is ook onderhouden door Android/Google zelf wat ervoor zorgt dat het altijd compatibel zal blijven. In espresso kan je wel alleen uit java of kotlin kiezen als taal. Gerry had liever dat de framework in javascript/typescript kon gebruikt worden want dat is hij gewoon.

- Detox: Is een grey box testing framework je hebt dan sowieso de codebase nodig. Ook hier kan je alleen in javascript schrijven. Ook hier zijn de testen sneller dan in appium. De documentatie van Detox is ook niet uitgebreid genoeg. En er mist heel wat informatie of het is onduidelijk.

POC MET APPIUM

CONCLUSIE

Na het maken van een POC in appium en te hebben gekeken naar de 2 andere frameworks is er gekozen om verder te testen te schrijven in Appium. Dit komt omdat je voor de 2 andere frameworks de codebase nodig hebt om het framework 100% te kunnen gebruiken. Ook heb ik dat bekeken met Gerry en kon hij mijn code vrij makkelijk herkennen. Het lijkt ook op de testing frameworks die hij gebruikt voor zijn testen van de web apps.

De uiteindelijke POC waren de testcases voor het inloggen, namelijk het inloggen met geldige gegevens en het inloggen met foute gegevens. Deze worden later nog meer in detail besproken. Nadat deze POC goedgekeurd was mocht ik beginnen aan de echte test cases voor algemeen verslag.

Wat ik ook geleerd heb bij het schrijven van deze POC is informatie opzoeken over frameworks in documentatie en hoe belangrijk goede documentatie is. Sommige frameworks hebben maar heel weinig of zeer onduidelijke informatie.

REALISATIES

Ik heb alle verslagen in de MTC-app getest maar ook heb ik 2 extra test cases geschreven om de testen volledig te hebben geautomatiseerd. Zo heb ik het scannen van vervoersbewijzen getest. Ook het hamburger menu in de app heb ik getest. Maar hierover kan je later meer lezen.

Alle test cases voor de verslagen lijken op elkaar. Daarom heb ik gekozen om de meest interessante verslagen eruit te halen. Bij elke test case leg ik uit waarvoor deze dient. Ook kan je code voorbeelden zien van test cases.

In het totaal heb ik 10 test suites geschreven met elk ongeveer 5 tot 10 test cases. De test suite voor dienstonderbreking is een uitzondering, deze bevat 23 test cases maar hierover vindt u later meer informatie.

Verschillende test cases bevatten checkboxen. Bij deze checkboxen test ik niet alle mogelijkheden met elkaar. Maar elke keuze bij een checkbox wordt minstens één keer getest. Het is niet mogelijk om alle verschillende mogelijkheden te testen.

Ik heb niet alles kunnen testen. Zo zijn er sommige delen van de app die je nog altijd manueel zal moeten testen. Zoals de button voor het afsluiten van de app en het scannen van QR-codes voor een vervoerbewijs.

CONCLUSIE

Uit deze test suites heb ik heel veel geleerd van hoe het automatisch testen in elkaar zit. Ook gebruikte ik voorbeelden van Gerry. Deze waren dan in een andere framework geschreven, Cypress.

Het schrijven van automatische testen hangt ook heel hard samen met het schrijven van code. Dezelfde skills worden hierin gebruikt. Ook begrijpen de developers de code die de testers schrijven. Ik werd ook geholpen door Indy.

Na twee maanden heeft De Lijn een nieuwe versie gekregen van de MTC-app waarop ik mijn automatische testen heb kunnen tegen runnen. Zo hebben we opgemerkt dat het veel sneller werkt dan de manuele testen die vroeger gedaan werden.

Door deze testen hebben we ook een bug gevonden in de app in het hamburgermenu. Eén van de buttons bleek niet meer te werken na de update daarvoor is ook een ticket gemaakt in het jira bord van het developer team voor de MTC-app.

INLOGGEN

Al deze testen samen testen de werking van het inloggen op de app. Hieronder kan je zien wat er allemaal getest is geweest en op welke manier. Dit was de eerste testcases die ik heb geschreven voor dit project.

TEST CASES

1. Checken of alle velden en teksten beschikbaar zijn op de inlogpagina
   In deze test case check ik of de inlogpagina de juiste teksten bevat of de foto aanwezig is enz. Ook check ik of de juiste basiswaarden zich bevinden in het username- en password vakje. Hieronder kan u zien hoe ik check of dat tekstvakje bestaat en of de juiste tekst aanwezig is.

2. Login in een account met valide gegevens
   In deze test case check ik of je kan inloggen in een account als je de juiste waardes invult. Eens dat je bent ingelogd check ik of er op de pagina een titel is van 'Opdrachten voor xx/xx/xxxx' met xx/xx/xxxx als datum vandaag.

3. Login in een account met verkeerde gegevens
   In deze test case check ik wat er gebeurt als je verkeerd gegevens invult op de inlogpagina. En welke tekst er dan tevoorschijn komt.

CONCLUSIE

Uit deze test Cases heb ik de basisch geleerd van het schrijven van testen en met werken van de frameworks enz. Indy heeft me dan ook wat tips gegeven over het schrijven van code die lang leesbaar moet blijven maar ook onderhoudbaar door de mede testers.

ALGEMEEN VERSLAG

De volgende test cases draaien allemaal rond het algemeen verslag wat het kleinste verslag is in de app. Dit verslag wordt gebruikt bij een vaststelling die bij geen enkel ander verslag hoort. Ze is daardoor helemaal niet uitgebreid.

TEST CASES

1. Controle aanwezige items en standaardwaarden
   Deze test case checkt of alle velden bestaan en of ze de juiste waarden bevatten. Er zijn twee schermen. Op het eerste scherm bevinden zich gegevens rond het CCG en op de het volgend scherm bevinden zich gegevens rond de vaststelling die de controleur gemaakt heeft.

2. Enkel verplichte waarden
   Deze test case vult alleen maar de minimumwaarden nodig voor het doorsturen van het verslag. Nadat dit verslag doorgestuurd is checkt het of het verslag wel degelijk verstuurd is geweest.

3. Alle waarden
   In deze test case vult hij alle waarden in het verslag. Nadien stuurt hij het verslag door en checkt hij of dit wel degelijk gebeurd is.

4. Leeg verslag bewaren aanpassen en finaliseren
   Hierin bewaar ik een leeg verslag. Daarna open ik dit verslag opnieuw en pas ik een veld aan. Nadien stuur ik het verslag door en kijk of het verslag wel degelijk doorgestuurd is geweest.

5. Verslag verwijderen uit de opslag van de tablet
   Ik sla een verslag op in de opslag van de tablet. Nadien open ik dit verslag opnieuw en delete ik deze uit de tablet. Als laatste check ik dan of deze wel degelijk weg is uit de opslag.

6. Ongeldige waarden
   In deze test case vul ik te lange of ongeldige waarden in de invoervelden om te zien hoe de app hierop reageert. Nadien probeer ik dit verslag door te sturen. Wanneer dit faalt check ik de error die op het scherm wordt getoond of hij wel de juiste data bevat.

CONCLUSIE

Met dit verslag heb ik de basis geleerd van door de dom structuur te gaan. Dit komt omdat in de MTC-app er meestal geen id’s op de invoervelden staan waardoor je aan de hand van de dom structuur door de app moet gaan.

PVDAB

Pvdab is het eerste verslag dat direct als het wordt geopend naar de back-end gaat waardoor je deze niet zo makkelijk uit de opslag van de tablet kan halen. In dit verslag heb ik wel een deel niet kunnen automatiseren. Het scannen van een id is niet mogelijk in een emulator. Je kan wel foto's toevoegen aan de foto app waardoor je bijvoorbeeld wel een id zou kunnen scannen maar de plug-in die wordt gebruikt voor het scannen van de id. Het heeft een camera met auto focus nodig en die heeft een emulator niet.

TEST CASES

1. Controle aanwezige items en standaardwaarden
   Bij elk verslag moet de eerste test case zijn om te checken of alle velden nog bestaan. Dit zorgt ervoor dat de test zal falen wanneer er iets aan het uitzicht van de app zal veranderen. Ook worden de titels nagekeken.

2. Enkel verplichte waarden
   Zoals bij het algemeen verslag is hier de bedoeling dat ik de minimumwaarden invul en check of hij wel doorgestuurd is geweest. Het verschil hier is dat er veel meer minimumwaarden zijn dan in algemeen verslag.

3. Alle waarden
   Hier is het de bedoeling zoals bij algemeen verslag dat ik alle velden invul met geldige data. En daarna doorstuur naar de back-end en als laatste check ik of dit verslag wel degelijk naar de back-end vertrokken is.

4. Leeg verslag bewaren en aanpassen
   Zoals de titel het zegt ga ik het verslag leeg bewaren in de opslag van de tablet. En nadien hem terug opendoen en wat data aanpassen. Namelijk alle minimumwaarden invullen anders wordt het verslag ook niet doorgestuurd naar de back-end.

5. Verslag annuleren en verwijderen
   Dit is de test case waarbij ik een verslag aanmaak en annuleer. Annuleren betekent dat je uit het verslag gaat naar de home screen maar het verslag bestaat wel nog altijd in de opslag van de tablet en ook op de back-end.

6. Niet standaardwaarden
   Voor het testen van de checkboxen met verschillende waarden heb ik ook 2 test cases voor niet standaardwaarden zo kies ik een andere taal of andere geslacht en check ik of je nog altijd het verslag kunt doorsturen.

7. Verslag leeg finaliseren
   In deze test case probeer ik een leeg verslag te finaliseren. Dit zou niet moeten lukken. Dan check ik welke error je krijgt met welke errormessage.

8. Ongeldige waarden
   In deze test case vul ik te grote/lange waarden in waardoor dit verslag ook niet kan doorgestuurd worden naar de back-end. Ik check ook of alle foutief ingevulde velden terechtkomen in de errormessage.

Conclusie

In deze test case heb ik ermee leren omgaan dat het verslag direct naar de back-end wordt verstuurd. Ook heb ik geleerd om niet af te hangen van de grootte van het scherm want in deze test cases gebruik ik een scrol. In het begin stond die scrol nog op scrollen tot het einde van de pagina wat ervoor zorgt dat als de emulator een kleiner scherm heeft je misschien nooit aan bepaalde velden geraakt.

DIENSTONDERBREKING

Dienstonderbreking is ook een interessant verslag doordat dit het eerste verslag is waarbij er modals worden gebruikt. Dit zorgt wel voor wat uitdagingen. Deze modals moeten ook volledig getest worden zoals een apart verslag. Met voor elke modal moeten er 3 test cases geschreven worden. Daardoor is deze test suite de langste in het project.

TEST CASES

De test cases die hetzelfde zijn als de rest zal ik niet meer beschrijven.

1. Tegenpartij Leeg toevoegen en verwijderen
   Hierin test ik wat er gebeurt als ik een tegenpartij toevoeg aan dienstonderbreking of de juiste waarden worden doorgegeven. In deze test case wordt de tegenpartij leeg toegevoegd.

2. Tegenpartij vol toevoegen
   Hierin wordt weeral een tegenpartij toegevoegd maar deze keer zijn alle waarden ingevuld voor de tegenpartij. Zoals bij de vorige test case worden de waarde gecheckt wanneer hij doorgegeven wordt.

3. Tegenpartij ongeldige waarde
   In deze test case wordt ongeldige waarden doorgegeven naar de tegenpartij modal. Nadien is het de bedoeling om deze te annuleren. Want hij zal niet doorgestuurd worden door de te grote/lange waarden in de modal.

Zo zijn er drie modals. Voor al deze 3 modals zijn ongeveer dezelfde test cases daarom ga ik ze niet allemaal bespreken. Deze zijn wel beschikbaar als video op mijn portfoliowebsite.

CONTROLE VERVOERBEWIJZEN

Deze test suite is interessant omdat het scannen van vervoerbewijzen bijna volledig automatisch gebeurt. Het enige wat door de controleur wordt gedaan is bij het scannen van een vervoersbewijs op naam. Zo moet ik ook een vervoersbewijs aanvragen via de test-website van De Lijn.

TEST CASE

Zoals bij de vorige test suites check ik of alle velden bestaan met de juiste waarden. De volgende test case is het invullen van de minimumwaarden. Als derde zal ik alle waarden invullen en zoals altijd is er ook de test case waarbij ik ongeldige waarden invul. Al deze test cases testen niet het controleren van vervoersbewijzen.

Hier kan ik bijvoorbeeld het controleren van sms of m-tickets niet controleren omdat deze niet meer aangevraagd kunnen worden. Ook kan ik geen QR-codes scannen omdat deze dan bij elke test opnieuw moeten toegevoegd worden aan de emulator.

1. Sms of m-ticket controle geen geldig tickets
   Hierin check ik op een bestaand telefoonnummer. Omdat ik geen ticket kan kopen gelinkt aan een telefoonnummer kan ik alleen maar eentje scannen zonder geldig ticket.
2. Controle op naam geldig ticket
   Voor deze heb je een geldig ticket nodig die gelinkt is aan het ‘ArnaudTester’ account. Nadien vul ik de juist naam en familienaam in waardoor ik als antwoord krijg dat deze persoon een geldig account heeft.

TESTEN DRAAIEN IN CI PIPELINE

In het begin van het project was de bedoeling om de testen te draaien in een pipeline op jenkins. Hiervoor moest ik eerst de testen kunnen draaien in Docker. Daarmee ben ik begonnen het installeren van de nodige dependencies. Dit is gelukt ik kan de testen starten in de docker container. Maar natuurlijk falen ze want ze vinden geen emulator terug.

Dit kan dan ook gedraaid worden in Jenkins. Deze start automatisch wanneer er wordt gepushd naar de gekozen branch. Tijdens het gebruiken van mijn project zal deze op de development branch draaien.

De volgende stap was het installeren van een emulator in de docker container. Hiervoor heb je de java jdk en de android sdk nodig. Nadat dat allemaal gedaan was bleek het niet mogelijk om een emulator te draaien want je hebt hardware acceleratie nodig voor het draaien van de android emulator. Dat is jammer genoeg niet mogelijk in jenkins.

Hiervoor zijn we samen met het team naar oplossing gaan zoeken. Deze oplossingen heb ik wel niet kunnen implementeren doordat ik te weinig tijd had om dit nog te doen maar de informatie over de oplossingen heb ik doorgegeven aan het team. Het team zal dit dan bekijken met de technische architecten van De Lijn.

Er zijn verschillende oplossingen hiervoor zoals het draaien van de emulator in de cloud of een vm opstarten in azure met hardware acceleratie. Als je dit kan doen kan je ook de testen draaien vanop eender welke laptop en hoef je alleen maar npm te hebben geïnstalleerd.

DOCUMENTATIE

Confluence is de documentatie pagina die wordt geleverd door atlassian. Hierin heb ik een gebruikershandleiding geschreven voor het draaien van mijn testen. Ook heb ik hierin allemaal genoteerd wat ik niet heb kunnen testen. Maar in ditzelfde document staan ook de meest voorkomende fouten.

Aan de hand van dit document zou je de emulator moeten kunnen installeren en opzettenmaar ook er de testen runnen.

Wat ik ook heb gedaan tijdens dit project is het aanmaken van tickets in het atlassian platform. Dit niet alleen voor mijn eigen projecten. Maar wanneer de active directory van De Lijn down was heb ik een ticket geopend bij de technische architecten van De Lijn om het probleem te melden.

Ook het werken met branches enz. heb ik nog extra geleerd bij deze stage. Zo moest ik er zelf voor zorgen dat ik branches opendeed voor mijn jira tickets. Wanneer ik een volledige test suite had geschreven maakte ik ook een pull request. Zodat Gerry en Indy deze kon verbeteren. Nadien moest ik ook voor zorgen dat de branch werd gesloten.

CONCLUSIE

Dit project heeft bewezen dat het mogelijk is om testen te schrijven voor een Android app.

Door deze testen zal het team veel sneller bugs terugvinden die worden geïmplementeerd door een update van de app.

Wat ik ook geleerd heb uit dit project is communiceren met verschillende teams. Aan de hand van tickets open maken in jira enz. Ook mail versturen en tickets opvolgen en duidelijk maken wat je nodig hebt.

Het team TCE is een agile team waardoor ik ook de stand-up meetings mee heb gedaan, ook de sprint review meetings die om de 2 weken gebeurde op De Lijn. Na de sprint review meetings waren er ook de retrospective meetings waar ik kon zeggen wat ik liever anders zag maar ook waar de collega's hetzelfde konden doen.

Wat ook nieuw was voor mij was het aanmaken van de tickets en dat het niet even makkelijk is als op school als men een jira bord gebruikte. Zo zijn er een aantal regels niet altijd even goed gedocumenteerd.

Hopelijk zullen mijn testen ook later worden gebruikt om testen voor andere android native apps te automatiseren. Mijn testen waren ook de eerste testen voor een android native app binnen De Lijn. Tot nu toe werden deze apps manueel getest.

Ik heb ook enorm veel bijgeleerd over testen zelf. Dit was de eerste keer dat ik concreet testen schreef voor een bestaande applicatie. Ook omdat dit de eerste keer was dat dit binnen De Lijn gebeurde heb ik enorm veel informatie moeten opzoeken over de frameworks wat mij eigenlijk geleerd heeft om te werken met documentatie en niet alleen met een voorgemaakte samenvatting die men op school kreeg.

Ook al kan je wat problemen tegenkomen zoals de auto focus van de camera. Ook zijn de testen heel traag omdat je een emulator draait. De testen zijn ook minder stabiel dan het testen van web apps.

De emulator heeft ook nog altijd een kans om te stoppen waardoor de testen ook falen daarom is het ook beter om de testen één per één te runnen. Er zijn wel wat oplossingen die ik heb geïmplementeerd zoals het wachten na elke test.

Zoals hierboven ook is vermeld geweest is het nog niet mogelijk om deze testen in jenkins te draaien maar het team is bezig met deze implementatie.