Verantwoording - Evaluatie - Reflectie Beroepsproduct 1

Auteur: Adam Oubelkas

Studentnummer: 2150912

Aanmaakdatum: 12-01-2019

Versie: 1.0

# Inleiding

In dit verslag wordt de deelproduct en processen met betrekking tot de ontwikkeling van de applicatie, uit het project ‘Examendossier - CBRDocs’, verantwoord, geëvalueerd en op gereflecteerd.

Verantwoording Software 1

Alle technische ontwerpen uit het Conceptueel Enterprise Model zijn omgezet als entiteiten in de database en applicatie. Hierover werd een acceptatietest geformuleerd en uitgevoerd om de voorheen genoemde producten te toetsen op afgesproken Requirements van de klant. Hierin werden de entiteiten van verschillende kwalificaties en identificaties uit de technische ontwerpen meegenomen, evenals de door klant en stakeholders gewenste kwaliteitseigenschappen van het eindproduct.

Gedurende het ontwikkelproces van de database is ervoor gekozen om alle entiteiten en hun verwante associaties en relaties uit het ERD, die overeenkomstig is aan het Conceptueel Enterprise Model, na te bootsen. Door deze keuze is de visie van zowel de klant/stakeholders als de ontwikkelaar op één lijn blijven liggen. Hiernaast zijn er ook Views (tijdelijke tabellen) geïmplementeerd omdat de SQL-statements hiervan vaak uitgevoerd zullen worden binnen de applicatie. Voor het voorzien van gevulde gegevens in de database, heb ik zoveel mogelijk geprobeerd testgegevens uit, een door de klant aangeleverde, CSV-bestand te halen. Door deze ingevoerde gegevens zijn er meer variatie en mogelijkheden ontstaan voor het maken en bewerken van examens in het front-end van de applicatie.

Tijdens het ontwikkelproces van de applicatie werd bij een aantal klassen, methodes en statements commentaar toegevoegd ter verduidelijking van de bedoeling en functionaliteit(en) hiervan. Met de kennis die ik heb opgedaan uit het schoolvak ‘Programmeren 2’ heb ik overerving, exceptie-handeling toegepast in mijn broncode, zodat de gehele applicatie efficiënt en optimaal functioneert volgens de opgestelde Requirements. Voor het invoeren van examentijden in het front-end, heb ik een externe bibliotheek geïnstalleerd binnen het project om gebruik te maken van een time Picker. Dit component functioneert ongeveer als een combo box waarmee de gebruiker gemakkelijk een tijdstip voor een examen kan selecteren.

Binnen de loop van de uitvoering van, de op voorhand opgestelde, acceptatietest, werd er zoveel mogelijk ongewenst gedrag opgespoord en gecorrigeerd. Dit met als doel om de gehele applicatie aan alle door de klant vastgestelde Requirements te laten voldoen.

Ná het uitvoeren van de acceptatietest werden de testresultaten hieruit gebruikt om de database en applicatie zoveel mogelijk te optimaliseren.

Uiteindelijk resulteerde de ontwikkelprocessen van de database en applicatie in een volledig eindproduct dat getest en getoetst is op Requirements uit de acceptatietest.

Evaluatie Software 1

Tijdens het ontwikkelen van de database en applicatie kwam ik erachter dat een aantal (nieuwe) klassen en functies bijgewerkt of aangemaakt moesten worden om de applicatie aan alle Requirements te voldoen. Zodra alle broncode werd geschreven om de applicatie met alle toegepaste Requirements te starten, functioneerde de applicatie niet naar behoren tijdens het debuggen en testen. Hierdoor moesten een tiental klassen en functies geoptimaliseerd en tientallen keren opnieuw getest en getoetst worden. Dit verliep stug en langzaam omdat ik nog niet bekend was met alle mogelijke manieren om een Java-applicatie in de programmeeromgeving ‘Eclipse’ te debuggen.

Daarnaast moest de uitwisseling van gegevens tussen de MVVM-lagen overdacht en gelijktijdig aangepast worden. Evenals de structuur (attributen van en associaties tussen verschillende klassen) en opbouw (overerving) binnen alle broncode moesten constant veranderd worden om de door de klant gewenste kwaliteiten uit de Requirements te realiseren.

Gebruikersinvoer voor het bekijken/aanmaken/bewerken/verwijderen van examengegevens moesten ook gevalideerd worden op front-end niveau, zodat de gebruiker, indien nodig, hierover een melding krijgt. Binnen het front-end werd de gebruiker hiermee concreet geïnformeerd, zodat de gebruiker wist welke acties zijn uitgevoerd of uitgevoerd moesten worden door zichzelf.

Reflectie Software 1

In het volgende project zal ik proberen binding te implementeren tussen comboxen en het geselecteerde item uit de selectietabel, om ervoor te zorgen dat er maar één keer geklikt hoeft te worden om deze comboxen te wijzen op het geselecteerde item. Binnen het front-end van de applicatie uit het huidige project moet er twee keer geklikt worden op een examen om deze gegevens in comboboxen te tonen.

Tijdens het ontwikkelproces heb ik bij niet alle noodzakelijke klassen, methodes, attributen en statements commentaar toegepast waardoor andere programmeurs niet kunnen begrijpen wat mijn bedoeling was met al deze broncode. Dit zal ik in het volgend project anders aanpakken door tijdens het maken en bijwerken van klassen, methodes, attributen en statements niet alleen concreet commentaar te geven over de bedoeling en gebruik hiervan maar ook JavaDoc meteen toe te passen.

Betreft methodes zal ik, indien nodig, optionele parameters proberen daar te implementeren zodat een aantal parameters niet altijd gevuld hoeven te worden als deze toch niet in alle gevallen gebruikt worden binnen de methode.

Voor het opvangen van applicatiefouten middels try-catch-finally-statements zal ik in het volgende project ervoor zorgen dat naar de juiste soorten applicatiefouten (exceptions) wordt gekeken en dit op een correcte manier verwerkt worden binnen de applicatie. Dit deed ik gedurende dit project niet waardoor de applicatie onstabiel kon worden en niet meer kon functioneren, met als gevolg dat deze ongewenst gedrag kon vertonen.