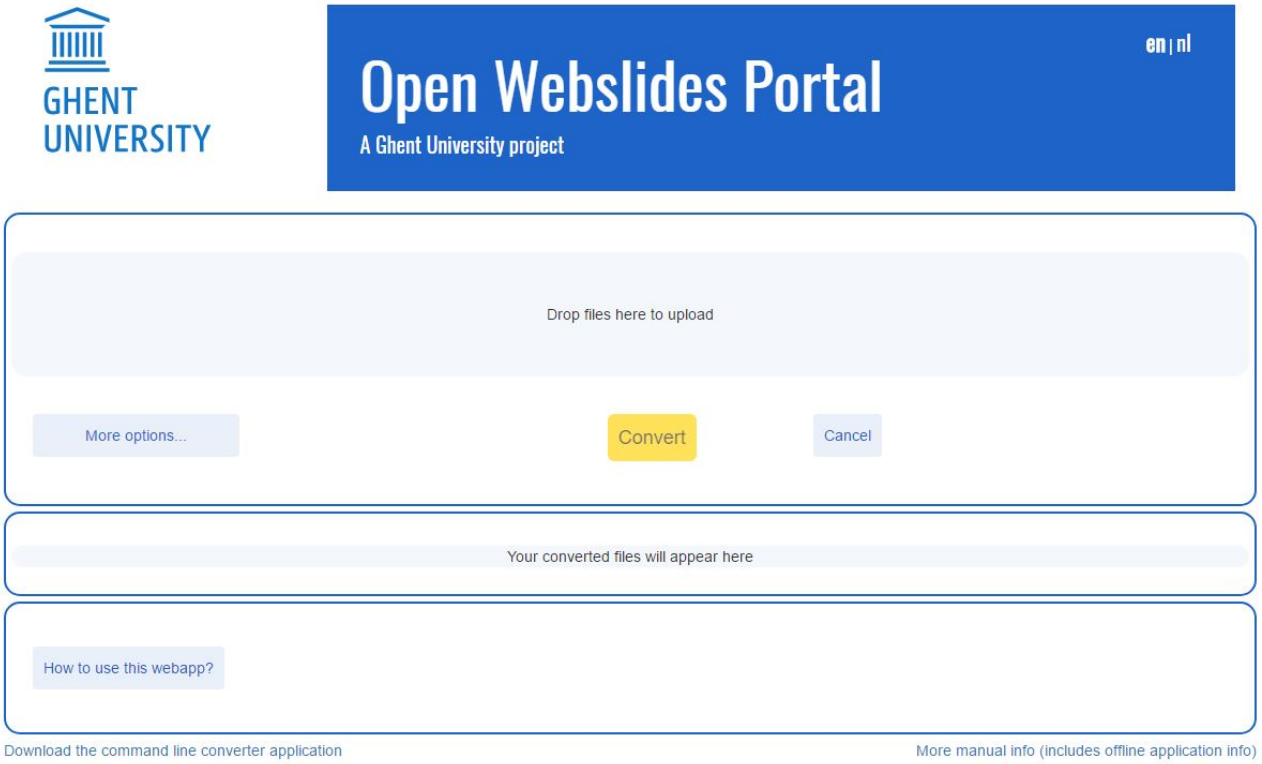
UNIVERSITEIT GENT

FACULTEIT INGENIEURSWETENSCHAPPEN EN ARCHITECTUUR

3e Bachelor Informatica
Academiejaar 2016-2017

Faculteit Ingenieurswetenschappen en Architectuur
Valentin Vaerwijckweg 1 - 9000 Gent

Open Webslides



Eindverslag bachelorproef
22/05/2017

Gertjan BOSTEELS
Joann CHAN
Laurens D'HOOGE
Jonas VANDAMME
Karel VANDEN HOUTE

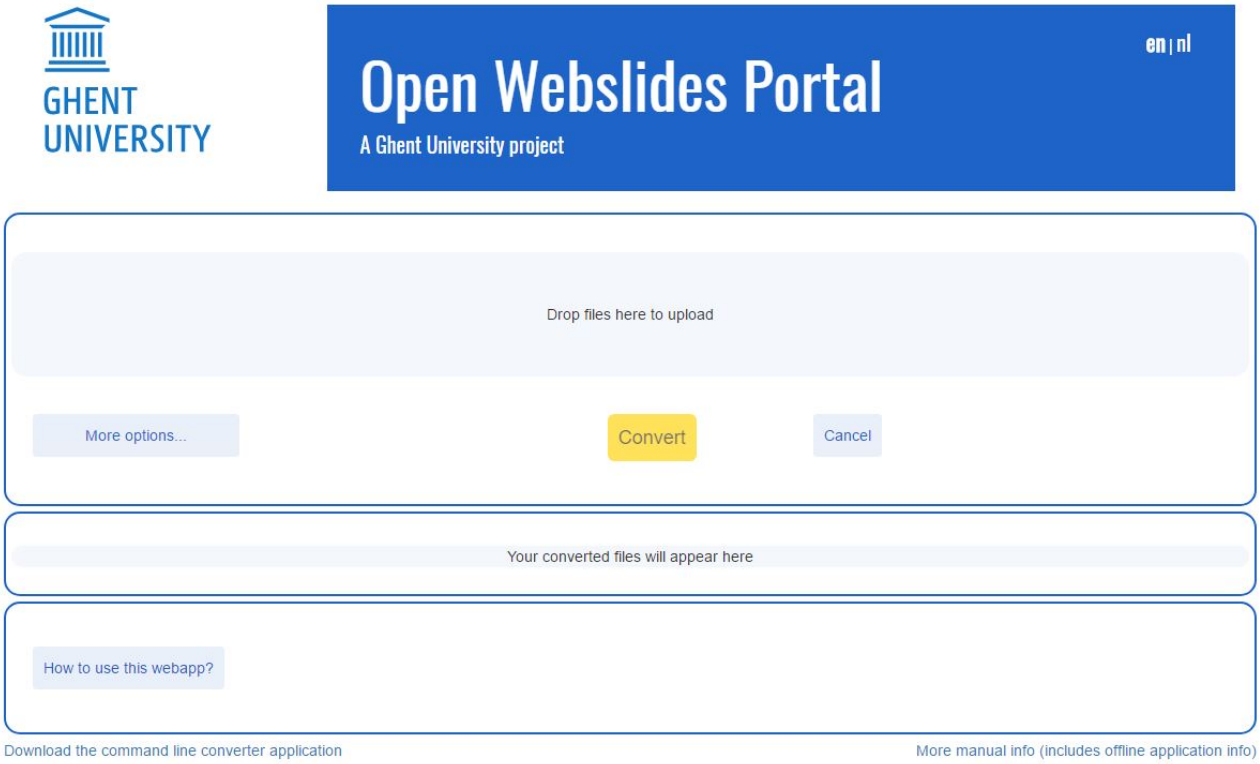
UNIVERSITEIT GENT

FACULTEIT INGENIEURSWETENSCHAPPEN EN ARCHITECTUUR

3e Bachelor Informatica
Academiejaar 2016-2017

Faculteit Ingenieurswetenschappen en Architectuur
Valentin Vaerwijckweg 1 - 9000 Gent

Open Webslides



Eindverslag bachelorproef
22/05/2017

Gertjan BOSTEELS
Joann CHAN
Laurens D'HOOGE
Jonas VANDAMME
Karel VANDEN HOUTE

Open Webslides

Open Webslides is een project van de Universiteit Gent. Het doel is om een toegankelijke en (inter)actieve leeromgeving aan te bieden die cocreatie bevordert. Dit wordt gedaan aan de hand van online cursusslides gemaakt in HTML. In dit project werken we aan een converter om bestaande slides uit pptx- en pdf-bestanden om te zetten naar Open Webslides. Dit doen we op zo'n manier zodat het relatief eenvoudig is om later uit te breiden naar andere formaten.
Momenteel kunnen pptx- en pdf-bestanden geconverteerd worden naar Open Webslides. Dit kan zowel via de website gebeuren als lokaal met een jar-bestand.

Voor dit project bedanken wij graag Ann Van Overberghe voor opvolging van het project en het helpen wegwijs maken in SCRUM. Daarnaast willen wij ook onze technische coaches bedanken, in het bijzonder dhr. Pieter-Jan Maenhaut, mvr. Veerle Ongenae en dhr. Martijn Saelens. Ten slotte zouden wij ook onze opdrachtgevers willen bedanken, dhr. Ruben Verborgh en mvr. Esther De Loof, voor hun feedback, enthousiasme en heldere communicatie.

Inhoudsopgave

Lijst van figuren

Hoofdstuk 1: Inleiding

1.1 Context

Open Webslides zijn online dia's, gemaakt met behulp van webtechnologieën, namelijk HyperText Markup Language (HTML), Cascading Style Sheets (CSS) en JavaScript. Deze slides zijn op dezelfde manier gemaakt als websites, waardoor alle mogelijkheden van het web geïntegreerd kunnen worden in deze dia's. YouTube-video's, interactieve webpagina's, ... kunnen eenvoudig toegevoegd worden aan de slides. Andere sites kunnen ook gemakkelijk bereikt worden door een eenvoudige link naar de website toe te voegen aan de dia.
Elke webslide heeft ook een eigen webadres, waardoor er overal verwezen kan worden naar een specifieke dia. Hierdoor kan het cursusmateriaal op elk platform toegankelijk gemaakt worden. Bovendien kunnen webslides op elk toestel bekeken worden. Het is niet nodig om hiervoor extra software te installeren, zoals wel het geval is bij PowerPoints.

1.2 Probleemstelling

Momenteel worden de studenten niet voldoende betrokken bij het lesmateriaal voor een vak. Het cursusmateriaal komt van de professor of lesgever en de studenten hebben weinig of geen zeggenschap in de inhoud. Met Open Webslides wil men daar nu verandering in brengen. Dit wordt onder andere gedaan door de studenten aanpassingen of aanvullingen te laten maken aan de webslides.
Open Webslides kunnen momenteel bewerkt en getoond worden, maar hiervoor is nog altijd technische kennis nodig. Men wil Open Webslides toegankelijk maken voor iedereen door de technische details van de slides weg te abstraheren. Om een vlotte overgang naar webslides te garanderen, is er nood aan een krachtige conversietool die zowel eenvoudig in gebruik is als accuraat.

1.3 Doelstelling

Met Open Webslides wil men het cursusmateriaal moderniseren. Er wordt een gebruiksvriendelijk en duidelijk platform gecreëerd om cocreatie van het leermateriaal te bevorderen. Hierdoor kunnen zowel docenten, studenten, onderzoekers, ... betrokken worden bij de creatie van het cursusmateriaal. Open Webslides moeten toegankelijk gemaakt worden voor iedereen.
Lesgevers en studenten krijgen de mogelijkheid om hun bestaande slides om te zetten naar het nieuwe webslides formaat of om enkel de inhoud van hun slides over te houden. Indien gekozen wordt om enkel de inhoud uit bestaande slides te extraheren, dan kan het resultaat daarvan rechtstreeks gebruikt worden als bouwblok in de webinterface om Open Webslides te maken.

1.4 Structuur van het verslag

In dit verslag wordt uitgelegd hoe pptx-bestanden en pdf-bestanden worden omgezet naar Open Webslides.

Het eerste deel beslaat de vorige pagina's en dient om zowel de context van dit project als de probleemstellingen en doelstellingen te schetsen.
Het tweede hoofdstuk bevat de gebruikersaspecten, waarin de high-level requirements besproken worden. Ook de use case diagrammen, de product backlog en feature list/sprint backlog zijn in dit hoofdstuk te vinden.
Het volgende hoofdstuk bevat de systeemarchitectuur. In dit hoofdstuk wordt het high-level systeem model besproken. De klassendiagrammen van de PPT-structuur en de converter worden hier uitgelegd, gevolgd door de werking van de PPTConverter, PDFConverter en de Writer. Vervolgens wordt de webapplicatie zelf besproken. Hoofdstuk 3 wordt afgesloten met de bijhorende activiteitendiagrammen en sequentiediagrammen.
In hoofdstuk 4 staat het testplan waarin uitleg wordt gegeven over de verschillende testen die uitgevoerd werden.
Het volgende hoofdstuk beslaat de evaluatie en discussies.
Hoofdstuk 6 bevat een handleiding voor de gebruikers. Deze handleiding beschrijft hoe gebruikers hun pptx-bestanden en pdf-bestanden kunnen omzetten naar Open Webslides, zowel via de site als lokaal.

Hoofdstuk 2: Gebruikersaspecten

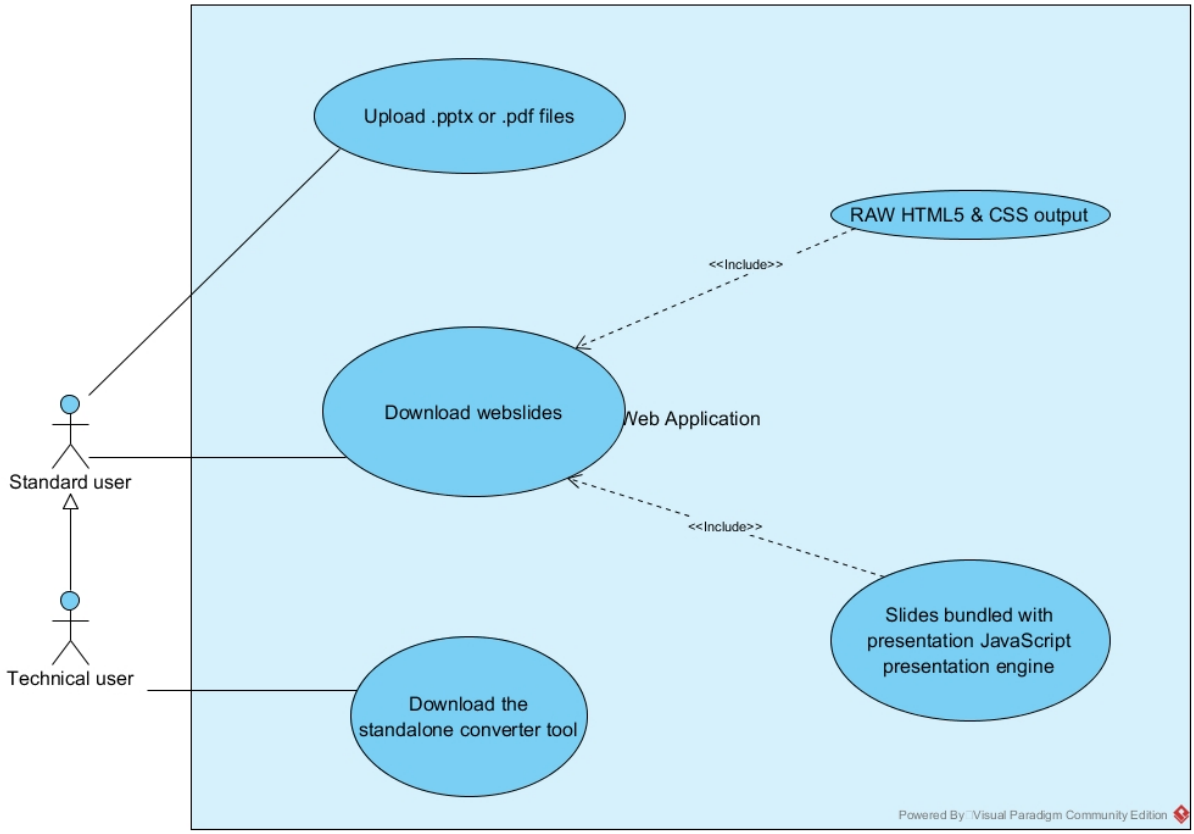
2.1 High-level requirements

De webapplicatie moet docenten en andere gebruikers de mogelijkheid geven hun huidig presentatiemateriaal (zowel MS PowerPoint als pdf-bestanden) om te zetten naar webslides. Daarvoor worden twee scenario's voorzien. Als eerste kunnen de huidige slides geüpload worden naar een webapplicatie om die onmiddellijk te converteren naar webslides. Ten tweede worden een aantal opties opengesteld voor de gebruiker. Op die manier kan de uitvoer aangepast worden naar de specifieke wensen van de lesgever.

Naast de webapplicatie moet de converter ook bruikbaar zijn als lokale applicatie. Op die manier krijgen gebruikers met programmeerervaring de mogelijkheid bestanden te converteren vanop de commandolijn. Ook via die weg moet met vlaggen en parameters de mogelijkheid bestaan een aantal opties aan te passen naar wens. Deze offline tool kan eenvoudig in een toekomstig platvorm ingebouwd worden.

2.2 Use case diagrammen

In onderstaand use case diagram 2.1 wordt beschreven wat gebruikers kunnen uitvoeren.



Figuur 2.1: Web application Use Cases

2.3 Use cases

Use Cases - Website
Succesvolle situatie

1. De gebruiker voegt een geldig pptx- of pdf-bestand toe op de server en klikt op convert.
2. De website geeft een link terug om het geconverteerde bestand te downloaden.
3. De gebruiker downloadt het bestand door op de link te klikken.

Niet succesvolle situatie 1

1. De gebruiker voegt bestand toe dat geen pptx- of pdf-bestand is.
2. De website geeft een foutmelding dat een fout bestand werd geüpload.

Niet succesvolle situatie 2

1. De gebruiker voegt een ongeldig pptx- of pdf-bestand toe op de server en klikt op convert.
2. De website aanvaardt het bestand, maar geeft niets terug.

Use Cases - Command Line Tool
Succesvolle situatie

1. De gebruiker voert een correct commando uit met een geldig pptx- of pdf-bestand.
2. De uitvoer wordt succesvol aangemaakt in een output map.

Niet succesvolle situatie 1

1. De gebruiker voegt een bestand toe dat geen pptx- of pdf-bestand is.
2. De console geeft een *file not supported error*.

Niet succesvolle situatie 2

1. De gebruiker voegt een ongeldig pptx- of pdf-bestand toe.
2. De console geeft een *not working powerpoint error*.

Niet succesvolle situatie 3

1. Gebruiker voert een niet-werkend commando in (bv. schrijffout in type, logging, ...) met een geldig pptx- of pdf-bestand.
2. De uitvoer wordt succesvol aangemaakt in een output map met standaardwaarden.

2.4 Product-backlog

De meest uitgebreide vorm van het project is driedelig: een conversietool voor .pptx bestanden, een abstractielaag boven Git voor opslag en vergelijking van webslides en een online webslides *editor*. Het communicatiemiddel tussen de klant en het team is de *product-backlog*, een niet-technische oplijsting van verwachte functionaliteit. Deze wordt door beide partijen opgesteld en mondeling besproken bij de samenkomst. Het resultaat is een startpunt voor het team dat goed afgesteld is op de verwachtingen van de klant en van het team.

Er was een voorkeur om te starten met de conversietool voor pptx-bestanden. De klant had liever één goede maar uitgebreide converter dan enkele kleinere en simpele tools of applicaties. Later is na overleg ook nog pdf-ondersteuning toegevoegd.

Voor sprints 1 en 2 waren volgende lijsten het vertrekpunt:

2.4.1 Product backlog 1

- 1 pagina file-upload met zeer simpele UI (drag en drop)
- Controle of juiste bestanden worden geüpload
- 1-click-conversie: Ondersteuning voor enkel pptx
- Omzetting van eenvoudige elementen (bullet points, figuren, headers, footers, lijsten. . .)
- Bijhouden van uploads en conversies
- Logging voorzien: belangrijkste gebeurtenissen uitschrijven met timestamp naar een bestand

Een uitgebreidere versie van deze lijst en de ideeën voor de andere delen zijn te lezen op dit online *whiteboard* `https://sketchboard.me/tAlAqCb7gbau#/`. Op de wiki sectie van de Github repository voor dit project staat een aparte pagina met verwijzingen naar de whiteboards samen met een korte beschrijving.

2.4.2 Product backlog 2

- Nieuw code design van uitvoer van de converter, een meer modulair ontwerp. Voorzien van logische eenheden waarin er wijzigingen kunnen aangebracht worden in de datastructuur zoals automatisch herkennen van titels en lijsten.
- Ondersteuning voor PDF Convertie
- Nieuwe webpagina voor file upload gebruik makend van andere technologie (zonder JSF)
- Fouten in de huidige HTML uitvoer oplossen (nesting van lijsten, speciale karakters)
- Nieuw code design datastructuur van powerpoints en pdf's. Een uniforme manier om de informatie die een ppt voorstelt op te slaan.
- Uitbreiding logging, meer gedetailleerd
- Koppeling tussen de verschillende modules van het project → website werkt samen met directoryguard en convertor

2.4.3 Product backlog 3

- Bugfix speciale karakters bij het converteren van powerpoints
- Logica na HTML, cleanup van de gegenereerde html code (Strong tags, p tag in li tag)
- User interface moet lineaire interface zijn van boven naar onder (bovenaan upload, daaronder convert, daaronder download)
- Webpagina moet default naar 'shower' formaat converten
- Uitleg van uploadpagina op webpagina zetten (Je kan pptx- en pdf-bestanden uploaden via drag en drop...)
- Download all knop (Mogelijkheid om meerdere conversies gelijktijdig te downloaden met één knop wanneer meerdere bestanden geüpload worden)
- Instabox → Bestand slepen in instabox, dit downloadt direct een geconverteerde versie
- PDF conversie uitbreiden (bv. tabellen met dezelfde functies als pptx)

2.5 Volledige feature list / sprint-backlog sprint 1

Feature	*Complexiteit*
JSF Front-end	
Initieel Project	1
Bestanden uploaden met Primefaces	1
Validatie bestandsextensie	1
CSS Layout	1
Sessie en herkenning inputbestand is gelijk aan outputbestand	1
Toevoegen icoon	1
Uploadpagina met verschillende opties (met/zonder opmaak)	1
Directory Guard Back-end	
Logging	2
Koppeling backend en converter	1
Zipper	1
Verwijderen van ENTRY_MODIFY	1
Openen van converters in threads i.p.v. processen	1
Gebruik entrypoint converter i.p.v. main	1
Blocking loop voor log van converter	1
Converter	
Initieel project	1
Zorgen voor output	1
Analyse Apache POI	1
Entrypoint voor ConversionThread	1
Dummyverwerker input	1
Output in output folder	1
Input via command line argument	1
Configuratiebestand	1
Template	2
HTML uitvoer	1
Diagrammen	
Klassendiagram converter	1
Sequentiediagram directory guard	1
Componentendiagram geheel	1
Documentatie	
Handleiding directory guard	1
Handleiding converter	1
Handleiding webapplicatie (JSF Front)	1

2.6 Onverwachte sprint-backlog items sprint 1

Feature	*Complexiteit*
JSF Front-end	
Bug: Verandering van Locale crasht JQuery	1
Directory Guard	
Bug: Blokkering uitschrijven van logfile converter	1
Logger	
Bug: Voorkom memory overflow en data loss	1

2.7 Volledige feature list / sprint-backlog sprint 2

Feature	*Complexiteit*
Webapplicatie full-stack	
Herschrijven webapplicatie (geen JSF meer)	3
Web front-end dropzone js	1
Front-end layout css	2
Back-end Servlet programming	3
Websockets 2-way communicatie front-end back-end	3
Session management	1
Sequentiediagrammen + handleiding webapp	2
Foutafhandeling + testing (bruikbaarheid, robuustheid, veiligheid)	2
Webapp updates + patches	2
PPTX converter	
Basisklassen HTML output writer	3
Detailuitwerking HTMLWriter voor alle PPTObjecten	2
Powerpointconversie diverse toevoegingen en optimalisaties	4
Datastructuur placeholder object	1
documentatie: UML-diagrammen	1
Converter lijst-nesting verbeteren	1
Testing JUnit HTMLWriter tests	1
Testing JUnit TemplateWriter tests	1
Bugfix: herhaling titel	1
Bugfix: herkenning speciale tekens	1
Bugfix: PPT objecten laten invullen met slides door PPTConverter	1
Aanpasbare titels opleidingsonderdeel en hoofdstuk in TemplateWriter	1
ConverterFactory: werp exceptie bij fout bestandstype	1
Converter zip output mogelijkheid + meer modulariteit inbouwen	3
PDF converter	
PDF inlezen in eigen datastructuur (zie pptx data-ontwerp)	4
Intelligente convertermodules	
Logica voor tekstherkenning	2
Ondersteunende deelprojecten	
Zipper: uitbreiding voor zippen van mappen	1
Gebruik output library in converter ipv klassen	1
Analyse BLOB database MySql	2

2.8 Volledige feature list / sprint-backlog sprint 3

Feature	*Complexiteit*
Webapplicatie full-stack	
Default naar shower veranderen	1
Logische flow UI	1
PPTX Converter	
Bugs PPTX converter oplossen	1
Toevoegen headers en footers	1
Bugfix: speciale karakters	1
Intelligente convertermodules	
Groeperen Textparts met dezelfde FontDecoration	1
PDF Converter	
Bugs PDF converter oplossen	1
Toevoegen headers en footers	1
HTMLWriter	
Verwijderen <p>-tags in -tags	1

Hoofdstuk 3: Systeemarchitectuur

3.1 High-level systeem model

Diagram 3.1 is een *big-picture-view* van het geheel. Om het project zo modulair mogelijk te houden, is geopteerd om de deelprojecten zo veel mogelijk van elkaar te scheiden.
Het belangrijkste deel voor de klant is de converter. Daarom is besloten om enkel de essentiële functionaliteit van de converter daar te implementeren. *Cross-cutting concerns*, gedeelde functionaliteit die vereist is in verschillende delen van professionele applicaties, werden apart geschreven (bijvoorbeeld de logger). De webapplicatie is het bindmiddel tussen de gebruiker en de convertor. De webapplicatie staat in voor alles wat niet rechtstreeks met het conversieproces te maken heeft.

Het geheel draait op een Ubuntu server *virtual machine* (VM), waarbij de webapplicatie beheerd wordt door een Glassfish webserver.
Het diagram is ook online beschikbaar in een hogere resolutie op:
`https://www.dropbox.com/s/xmfntu6sfxdcmjk/Architecture_overview.png?dl=0`

3.2 Klassendiagrammen

3.2.1 PowerPointstructuur

Op figuur 3.2 bevindt zich het klassendiagram van de PowerPointstructuur.
Het doel van de gebruikte datastructuur is om een PowerPoint zo goed mogelijk na te bootsen.
De inhoud van een PowerPoint wordt bijgehouden in een PPT-object. Een PPT-object bevat vervolgens verschillende slides in volgorde. Een Slide-object kan net als in een PowerPoint verschillende elementen bevatten, PPTObject's genaamd. Zo een element kan een Title-, Text-, Textpart-, PPTList-, Table-, Video-, Image- of Chart-object zijn. Elk van deze elementen implementeert de PPTObject-interface. Title en Textpart zijn de eenvoudigste elementen, zij bevatten enkel tekst met enkele layoutvariabelen zoals size, font, fontdecoration (bijvoorbeeld bold en italic) ... In het Text-object worden verschillende Textpart-objecten bijgehouden die tot eenzelfde paragraaf behoren.

Ter verduidelijking: wanneer een zin in het midden een vetgedrukt woord bevat, zal het opgeslagen worden als verschillende Textpart-objecten. Er is een linker Textpart (met daarin alle tekst links van het vetgedrukte woord), een midden Textpart (het vetgedrukte woord) en een rechter Textpart (met alle tekst rechts van het vetgedrukte woord). Deze Textpart-objecten worden opgeslagen in een List.

Het PPTList-object is iets specialer, het kan een lijst van PPTObject-objecten bevatten en zijn datastructuur lijkt dus op die van een Slide-object. Een Image/Video heeft een bestandsnaam die bijhoudt waar het bestand lokaal is opgeslagen, en twee variabelen voor de dimensie, één voor zijn grootte en één voor zijn locatie. Andere objecten zijn een video, tabellen en hyperlinks.

Verder bevat een PPT-object een PPTInsight. Dit object houdt 'Insights' over de PowerPoint bij, bijvoorbeeld de nodige tijd om de conversie uit te voeren, het aantal verschillende en gelijke woorden, ...

3.2.2 Werking Converters

Om bestaande slides om te zetten naar Webslides, zijn er converters nodig. Natuurlijk kan deze conversie niet altijd op dezelfde manier werken. Zo zal bijvoorbeeld een PowerPoint anders worden omgezet dan een PDF. Om de conversie eenvoudig en soepel te laten verlopen, moeten alle converters de interface IConverter implementeren. Deze vereist dat er een parse(PPT,String [,ZipOutputStream]) en een setOutput(Output) methode is. De parse-methode zal een bestand, dat wordt meegegeven met de constructor, omzetten naar de PPT-datastructuur. Met setOutput(Output) kan aan de converter meegegeven worden waar hij eventuele info- en/of foutberichten kwijt kan. Dit klassendiagram wordt getoond in figuur 3.3.

Het kiezen van de juiste converter gebeurt via de ConverterFactory. Wanneer een nieuwe converter wordt gevraagd met de getConverter(File) methode, zal deze op basis van de extensie van het bestand een gepaste converter teruggeven (en het bestand meegeven aan de converter via zijn constructor). Vervolgens kan er -indien gewenst- een Output worden ingesteld en daarna geparsed worden naar een PPT-object.

PPTConverter

De PPTConverter maakt gebruik van Apache XSLF (een onderdeel van Apache POI, `https://poi.apache.org/slideshow/xslf-cookbook.html`), een library die het mogelijk maakt PowerPoints uit te lezen. Dit werd in combinatie met SAX gebruikt. Via de library van Apache is het mogelijk de nuttige XML uit te lezen met SAX en de overige data via methodes op te vragen. Zo wordt alle tekst uitgelezen via SAX, en worden van images, charts, tables, videos reeds het ID bewaard. Met dit ID kan later de juiste data opgevraagd worden met de MediaHandler klasse. De MediaHandler zal gebruik maken van enkele *native* Apache XSLF methodes om de data uit de PowerPoint te halen. Ook zal hij eventuele Hyperlinks toevoegen aan de tekstobjecten. Door het ID al op voorhand uit te lezen, garanderen we een juiste volgorde van objecten. Voorafgaand aan de MediaHandler, zal de GarbageHandler de PPT-structuur filteren op eventuele null-objecten die erin geslopen zijn. Dit kan gebeuren wanneer de originele PowerPoint veel is aangepast en elementen verwijderd zijn. Dit wordt niet altijd correct opgeslagen in de XML van de PowerPoint (er blijven lege elementen achter). Verder maakt de PPTConverter ook gebruik van een TextHandler en een InsightHandler. De TextHandler zal zorgen voor spaties tussen woorden (deze worden in de XML van de PowerPoint niet bewaard). De InsightHandler zal het PPTInsight object van het huidige PPT-object updaten. De conversie zal altijd volledig doorlopen worden. Alle mogelijke fouten worden opgevangen en uitgeschreven naar de logs, maar nadien zal de conversie verder gezet worden.

PDFConverter

De PDFConverter maakt gebruik van Apache PDFbox (`https://pdfbox.apache.org/`), een library waarmee PDF kan worden uitgelezen. Het doel van deze library is echter niet PowerPoints die als PDF zijn opgeslagen uit te lezen, wat het gebruik toch wat moeilijker maakt. Om tabellen te herkennen wordt gebruik gemaakt van tabula, een project gebaseerd op PDFBox dat specifiek ontworpen is om tabellen te herkennen uit een PDF. (`http://tabula.technology/`). De conversie begint door een file om te zetten naar PDDocument, dat is het object waaruit streams gelezen kunnen worden. Dit document wordt volledig overlopen door de library. In dit framework wordt van een aantal klassen overgeërfd om de verwerking van de stream te manipuleren. Onder andere van de klasse PDFTextStripper en PDFStreamEngine worden methoden overschreven.

Formaat
Er zijn grote verschillen tussen pdf's onderling. Wanneer een pdf werd gemaakt met behulp van 'print to pdf' zal er bijvoorbeeld minder informatie in het document zitten dan wanneer de pdf gemaakt werd met 'save as pdf'. Zo gaan hyperlinks verloren in een geprinte versie en is de manier waarop afbeeldingen bijgehouden worden fundamenteel anders.
Bij de geprinte versie is elke afbeelding terug te vinden in het document als een fysieke afbeelding. Bij een opgeslagen versie wordt echter gewerkt met verwijzingen. Dit houdt in dat indien een afbeelding meerdere malen voorkomt er slechts 1 fysieke versie herkend zal worden met een aantal verwijzingen.

De convertor zal de verwijzingen vinden maar de afbeelding niet. Deze mismatch wordt achteraf opgevangen.

Conversie
De conversie is opgesplitst in vijf delen:

1. Openen en decoderen van het pdf-bestand
2. Afbeeldingen uit het bestand halen en wegschrijven naar de opgegeven map of ZipOutputStream
3. Informatie over afbeeldingen en tekst uit file halen en in PPT-objecten stoppen
4. Tabellen herkennen met Tabula
5. Naverwerking

Het openen en decoderen van het pdf-bestand gebeurt reeds bij constructie, ernstige fouten worden hier al opgevangen of gemeld. De naverwerking bestaat eruit oneffenheden uit te conversie weg te werken. Onder andere:

- Dubbele tekst uit PPT-model verwijderen. Omdat tekst en tabellen afzonderlijk ingelezen worden, staat de tekst die in tabellen gevonden wordt eveneens als gewone tekst in het PPT-model. Dit wordt opgelost door de herkende tabellen te inspecteren en hun bijhorende tekst te verwijderen.
- Foute tabellen reduceren. De tabelextractie is niet foutloos, dikwijls worden tabs als tabellen geïnterpreteerd, dit wordt opgelost door tabellen die slechts 1 kolom hebben te elimineren. Ook tabellen die beginnen met typische lijstitems worden verwijderd.
- Placeholders voor onvindbare afbeeldingen. Door de mismatch tussen afbeeldingen en afbeeldingsinfo zijn er sommige afbeeldingen die niet gevonden kunnen worden, hier worden placeholders voorzien.

3.2.3 Logica

In dit project wordt ook een soort intelligentie gebruikt om bepaalde componenten te herkennen, zoals titels en (on)geordende lijsten. Deze componenten komen veel voor in Powerpoints en kunnen makkelijk voorgesteld worden in HTML met behulp van tags. Deze logica wordt uitgevoerd nadat het PPT-object wordt ingelezen door de converter, maar voordat dit object wordt uitgeschreven door de Writer. In de logica worden ook Textpart-objecten gegroepeerd met dezelfde FontDecoration, zodat het voor de HTMLWriter eenvoudiger wordt om de accentuering uit te schrijven. Titels worden herkend voor zowel pptx als pdf aan de hand van de minimum en maximum fontgrootte doorheen het hele PPT-object.
Het herkennen van geordende en ongeordende lijsten gebeurt enkel voor pdf-bestanden, omdat de gebruikte library voor de PDFConverter enkel de tekst en positie uit de documenten kan halen. Ongeordende en geordende lijsten worden op een andere manier herkend. Ongeordende lijsten maken gebruik van dezelfde symbolen voor dezelfde sublijst, terwijl er bij geordende lijsten wordt gewerkt met opeenvolgende cijfers of letters. Om lijsten te zoeken wordt er eerst gezocht naar de Text-objecten die een bullet bevatten. Aan de hand van de tekstpositie en het opsommingsteken wordt ook het niveau bepaald. Wat hierna volgt, werkt voor beide lijsten op dezelfde manier. Eerst wordt de bullet verwijderd aan het begin van de tekst. Als tweede zullen opeenvolgende lijnen die bij dezelfde opsomming horen aan hetzelfde Text-object worden toegevoegd. Ten slotte worden deze nieuwe Text-objecten toegevoegd aan een bestaande lijst of een nieuwe lijst, aan de hand van het niveau van de tekst. Voor het herkennen van lijsten wordt gewerkt met het *template method pattern*. Het grootste deel van de uitgevoerde code is hier gelijk. Alleen de implementatie voor het zoeken van de bullets met de bijhorende niveaus is verschillend voor geordende en ongeordende lijsten.
Het klassendiagram van de logica is te zien in diagram 3.4.

3.2.4 Writer

Eenmaal het bestand geconverteerd is naar een PPT-object, moet het nog uitgeschreven worden. De Writer is een abstracte klasse die verantwoordelijk is voor het wegschrijven van de inhoud van een PPT-object. In dit project zijn er twee concrete implementaties. De eerste is een HTMLWriter, die schrijft de inhoud van de slides weg als HTML5-code. Objecten die niet herkend worden of niet ondersteund zijn, worden vervangen door een Placeholder. In de output is het zo duidelijk dat er een element niet omgezet kon worden.
De tweede implementatie is een TemplateWriter. Die bevat zelf ook een HTMLWriter om de inhoud van de slides zelf uit te schrijven. Die output van de HTMLWriter wordt door de TemplateWriter omringd met een *header* en *footer* om zo de template van de pagina toe te voegen. De template bevat onder andere scripts om de slides onmiddellijk te kunnen afspelen als diavoorstelling.

3.3 Webapplicatie

3.3.1 Technologieën

- Front-end
  - HTML5
  - CSS (Bootstrap + eigen css)
  - JavaScript (Dropzone.js + JQuery)
  - AJAX
  - Websockets
- Back-end
  - Java Servlets
  - Pure Java
  - Glassfish 4.1.1 webserver

3.3.2 Gebruik en mogelijkheden

De webapplicatie is bereikbaar voor elke UGent gebruiker, rechtstreeks vanaf toestellen aangesloten op het UGent netwerk of met de UGent *VPN* (*Virtual Private Network*) indien men aangesloten is op een ander netwerk. Hyperlink: webslide.ugent.be
Het voorblad van dit verslag toont de hoofdpagina van de webapplicatie. Het design is bewust minimalistisch om het gebruiksgemak te verhogen.
Een gebruiker hoeft niet in te loggen en kan op het uploadvak klikken of er rechtstreeks bestanden naartoe slepen. Er kunnen maximaal acht bestanden gelijktijdig opgeladen worden.
Na een klik op de converteer knop verschijnen progressiebalkjes in elk van de bestandsiconen. Als het bestand geüpload werd met succes, zal de gebruiker een groen vinkje zien, zoniet zal er een rood kruis te zien zijn.
Na een succesvolle upload verschijnt onder de dropzone een roterend icoon om te signaleren dat het opgeladen bestand verwerkt wordt. Dat icoon verandert in een knop met de naam van het opgeladen bestand, waarop geklikt kan worden indien de server het bestand volledig kon verwerken. Als de converter het bestand niet of onvolledig kon verwerken zal de gebruiker gewaarschuwd worden.
In het normale geval krijgt de gebruiker een zip-bestand terug met daarin zijn/haar slides.

- raw: pure HTML5 uitvoer voor gebruikers die dat bestand willen gebruiken als basis in de online webslides editor die momenteel in ontwikkeling is.
- shower: HTML5 + Shower presentation engine is een Javascript/CSS combinatie specifiek om HTML te presenteren als slides. Deze optie is voor gebruikers die hun slides willen converteren om onmiddellijk te kunnen gebruiken.

3.3.3 Technische kenmerken

Websockets

De hoeksteen van de communicatie tussen client en server in deze applicatie is het gebruik van websockets. Een websocket biedt een oplossing voor een tekortkoming van het traditionele client-server model, namelijk communicatie van de server naar de client op initiatief van de server.
In deze applicatie worden websockets gebruikt om statusinformatie door te sturen naar elke gebruiker over de verwerking van zijn bestanden. Enkel essentiële informatie wordt doorgestuurd en er wordt ook alleen gereageerd als een bericht toekomt. Geen van beide partijen maakt gebruik van bronnen (CPU, netwerk I/O) om tussendoor informatie te vragen over het verwerkingsproces.
Deze berichtgebaseerde architectuur komt ook naar voor in de applicatielogica, om zo efficiënt mogelijk om te springen met de beschikbare bronnen. De implementatie van de websockets is voornamelijk te vinden in websocket.js (client-side) en in ServerEndpoint (server-side).

Applicatielogica

De applicatielogica is volledig geschreven in Java en maakt intensief gebruik van multithreading. De ConverterManager klasse bevat het gros van de server-side verwerking dat niets te maken heeft met communicatie naar het client toestel. De ConverterManager vervult volgende functies:

- Bijhouden welke gebruikers welke bestanden hebben geüpload om het resultaat aan de juiste gebruiker terug te kunnen geven.
- De verdeling van het converteerwerk voor alle bestanden over een beperkte set threads en de administratie van die threads.
- Het loggen van de eigen status en het beheer van de LogThread die de voortgang van het conversieproces voor de verschillende threads registreert en bewaart.
- Fungeren als callback voor de ConversionCallables die actief zijn in de threadpool, om nadien het ServerEndpoint op de hoogte te brengen dat een conversie voltooid is.
  Een volledig overzicht van de structuur en communicatie van de webapplicatie is te zien in figuur 3.5.

  Er moest verzekerd worden dat slechts één instantie van de ConverterManager bestaat voor de webapplicatie. Daarom is gekozen om een vorm van het *singleton pattern* te implementeren. Het initialization-on-demand holder idiom is de officiële naam voor dit pattern (een lazy-loaded versie van het singleton pattern). De uitwerking hiervan steunt op een private constructor en een publieke statische methode in de klasse waarvoor een unieke instantie vereist is, gecombineerd met een geneste, private, statische klasse die de houder is van de unieke instantie. In de implementatie in ConverterManager is een bijkomende statische methode voorzien in de geneste klasse om het mogelijk te maken de callbackparameter voor ConverterManager in te vullen.

  Andere kenmerken

  - Servlets: bestandsup- en download gebeuren door gebruik te maken van Servlets.
  - i18n *internationalization*: rekening houdend met het internationale karakter van de universiteit, is de volledige webapplicatie beschikbaar in het Nederlands en het Engels. Bijkomende talen kunnen met minimale inspanning toegevoegd worden.
  - Foutafhandeling: opvang van HTTP statuscodes 4xx en 5xx, gebeurt met speciale pagina's ter vervanging van de automatisch gegenereerde foutpagina's die Glassfish zou tonen. Het doel hiervan is om gebruikers beleefd te informeren dat er iets foutgelopen is met hun bestand en wat ze dan kunnen doen. Deze pagina's zijn ook in de beide talen beschikbaar.
  - Documentatie: om het gebruiksgemak te verhogen voor gebruikers is een korte handleiding in drie stappen voorzien op de hoofdpagina van de webapplicatie. Een uitgebreidere handleiding die ook meer technische informatie bevat, is te vinden via een link op de hoofdpagina. Elke methode is geannoteerd volgens de javadoc notatie en een versie van de javadoc is gebundeld in de release van het project.

3.4 Libraries

De gebruikte externe libraries zijn:

- Apache POI: inlezen PowerPoint
- Apache PDFBox: inlezen PDF (inclusief FontBox)
- Tabula: herkennen tabellen in PDF
- Apache Commons CLI: parsen van de argumenten op de commandolijn
- Apache Commons IO: makkelijke manipulatie van het bestandssysteem

3.4.1 Output

Er is ook een library ontwikkeld voor het loggen van gebeurtenissen. De Ouput library bestaat uit twee delen, een Output-deel en een Logger-deel. Het Output-deel zal zorgen voor het omleiden van de uitvoer van het programma. Er zijn verschillende soorten van Output; een StdOutput die voor uitvoer naar de console zorgt, een Logoutput die voor uitvoer naar een logbestand zorgt (Via de Logger) en een StdLogOutput die allebei doet.
Een speciale Output is de GuardOutput. Die vult een wachtrij van strings met de berichten, vergezeld van het ingestelde ID van de converter. Een ander proces of een andere thread kan die wachtrij dan asynchroon bericht per bericht uitlezen en verwerken.
Het Logger-deel maakt een logbestand aan en schrijft de uitvoer van een LogOutput weg naar dat bestand.

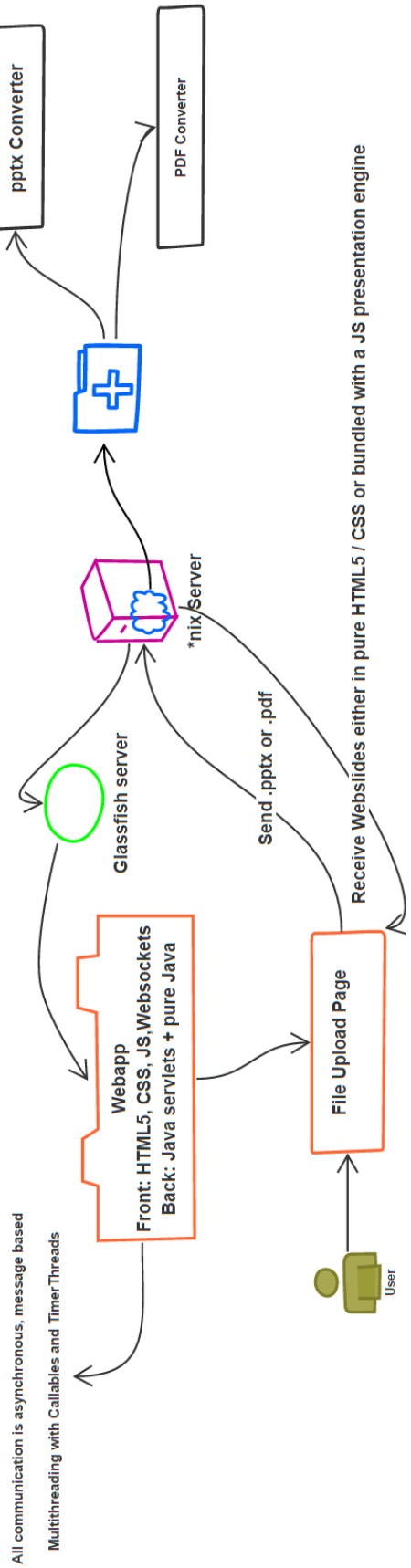
3.5 Activiteiten- en Sequentiediagrammen

3.5.1 Webapplicatie

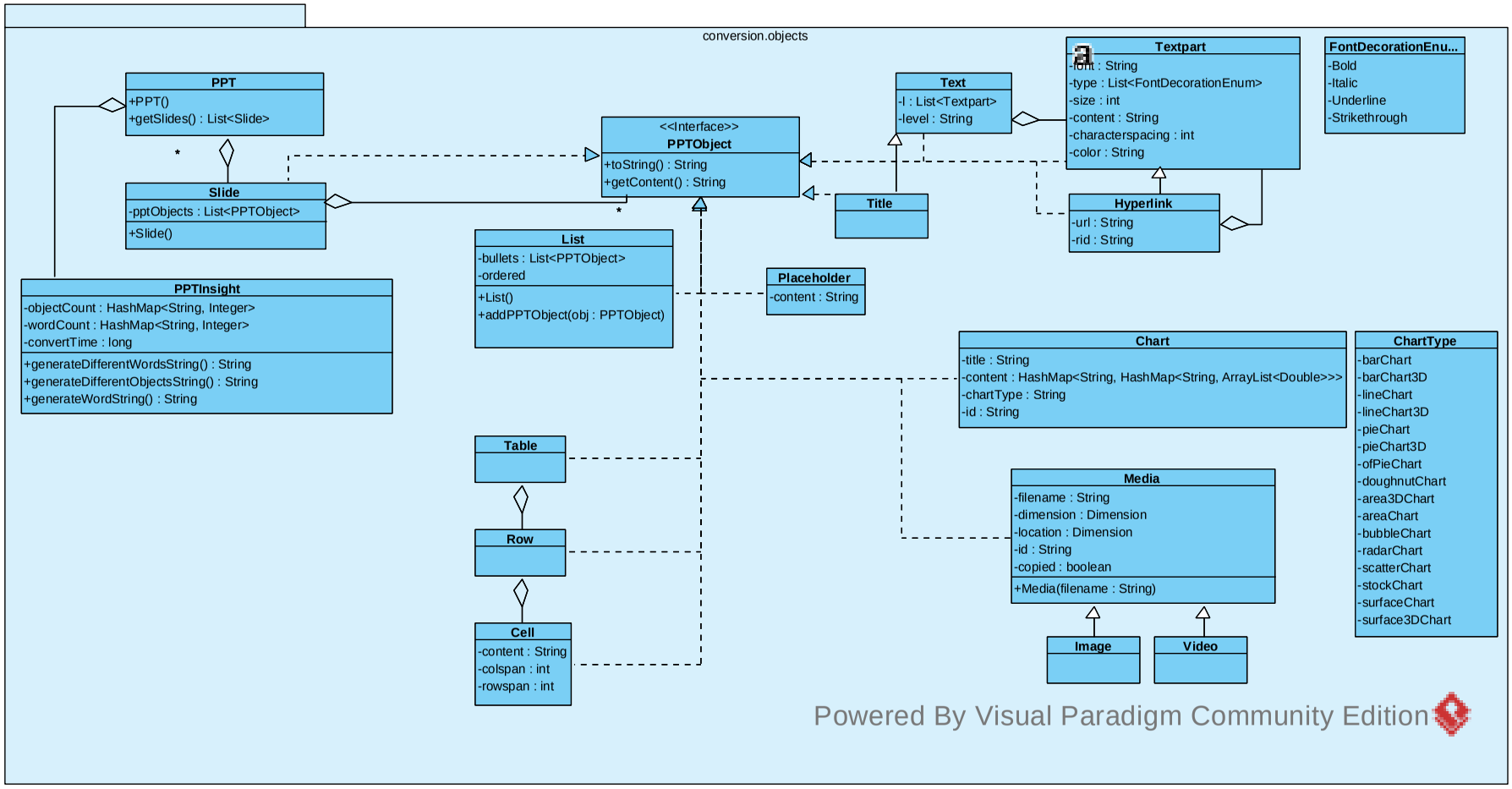
Diagram 3.5 toont de hele weg die een bestand doorloopt van upload tot download. Een A3-versie van dit diagram vindt u achteraan. Een downloadbare versie in hoge resolutie is te vinden op `https://www.dropbox.com/s/ltaui2q0a4wcsg7/Web%20application%20overview%20no-DB.pdf?dl=0`.

3.5.2 Converter

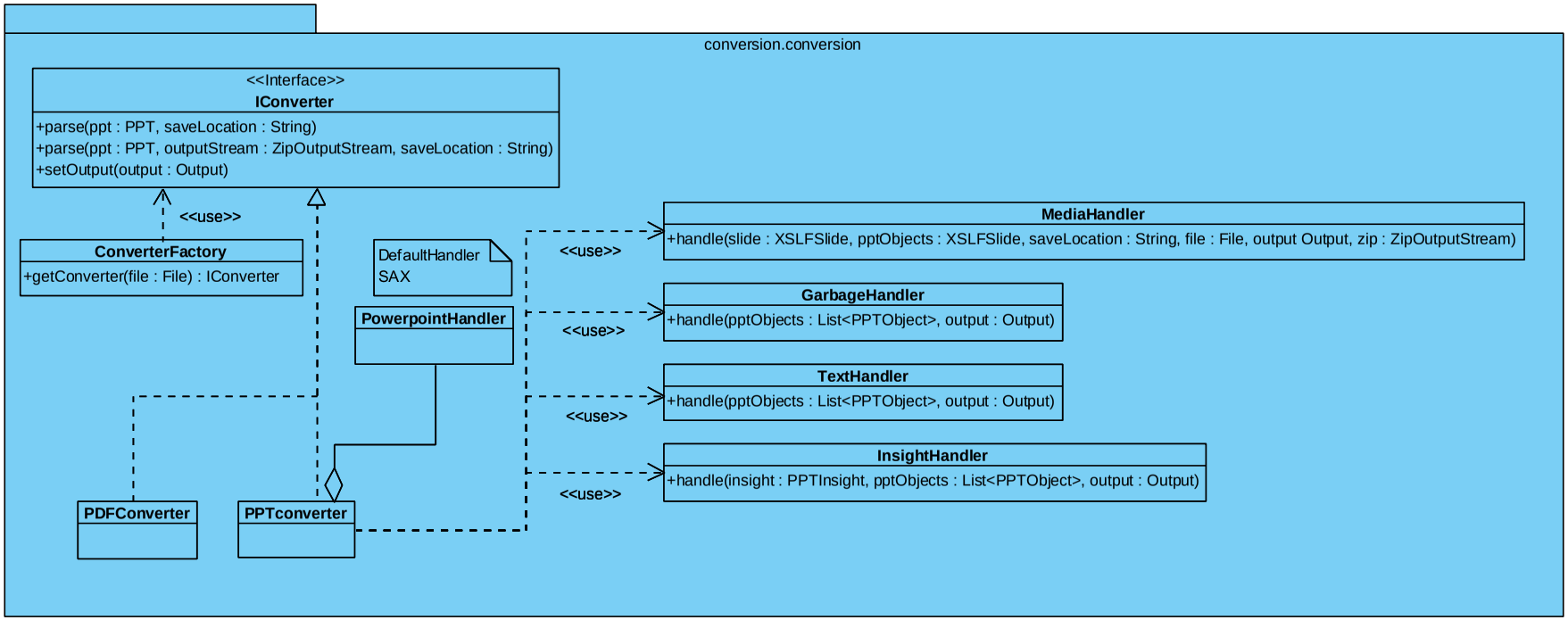
De OpenWebslidesConverter kan pptx-bestanden en pdf-bestanden converteren naar Open Webslides. Bij de uitvoer van het programma moet een bestandsnaam meegegeven worden. Dit bestand wordt aan de hand van de overige argumenten naar het gewenste formaat geconverteerd en eventueel in de juiste map geplaatst. In sequentiediagram 3.6 wordt getoond hoe dit gebeurt.



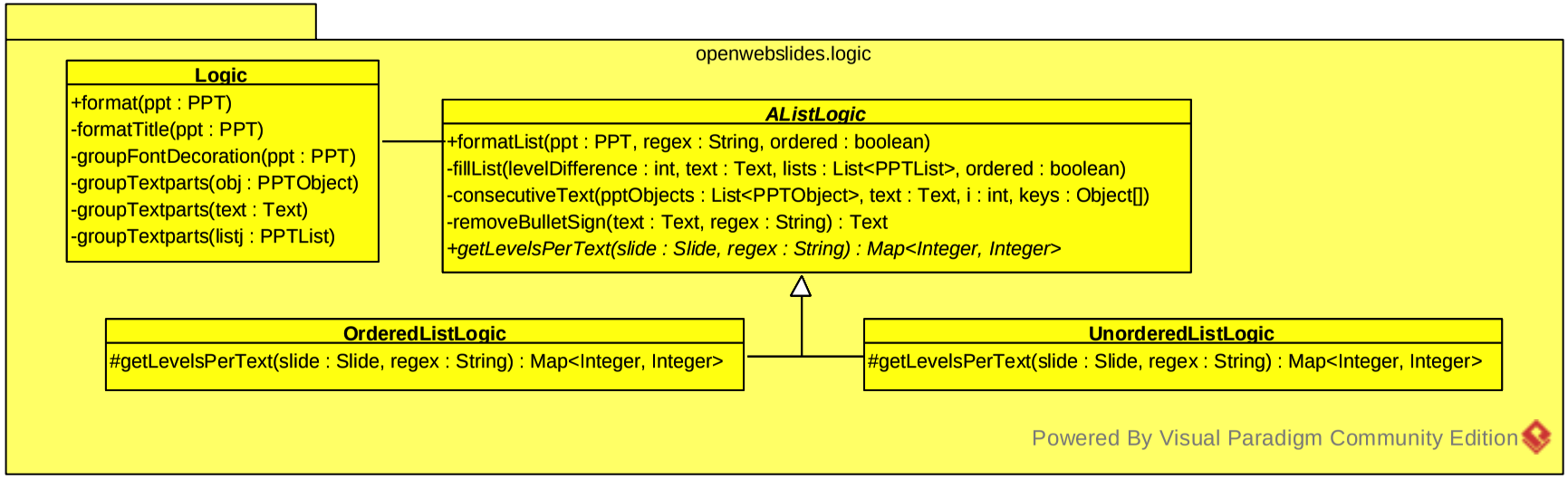
Figuur 3.1: Architectuur overzicht



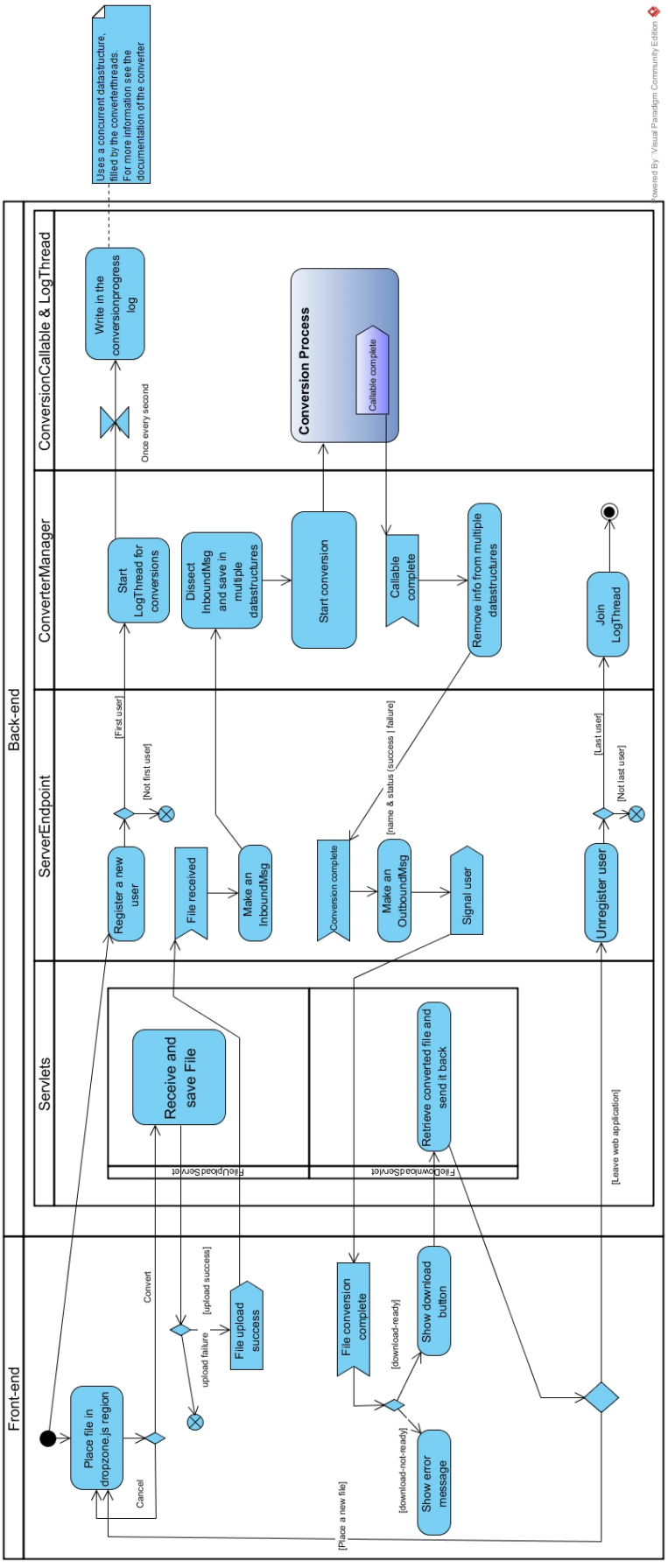
Figuur 3.2: Klassendiagram PPT



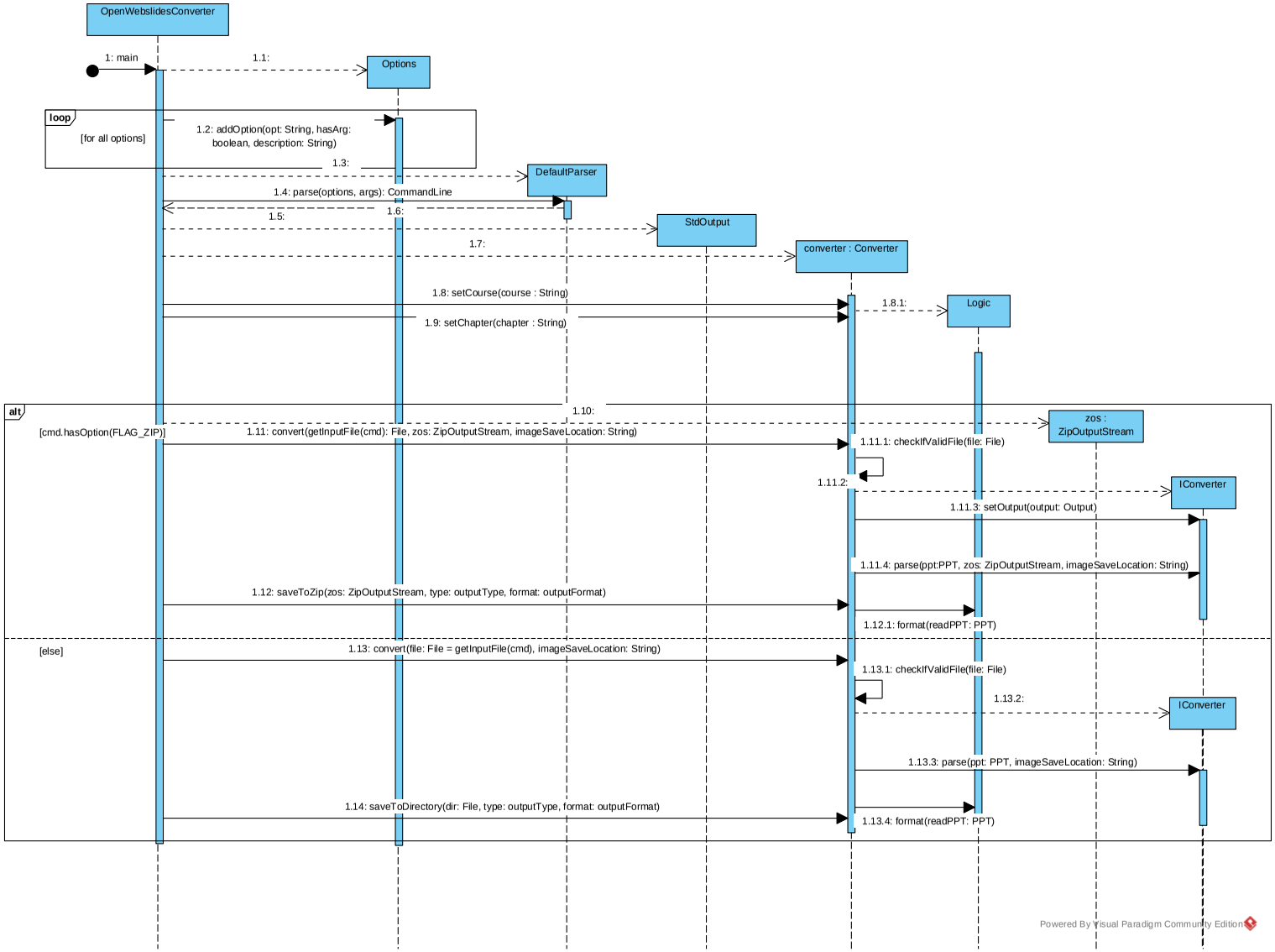
Figuur 3.3: Klassendiagram Factory



Figuur 3.4: Klassendiagram Logica



Figuur 3.5: Activiteitsdiagram webapplicatie



Figuur 3.6: Sequentiediagram OpenWebslidesConverter

Hoofdstuk 4: Testplan

Elke ontwikkelaar voert tijdens of na het ontwikkelen en/of uitbreiden van een module manuele testen uit. Dat wordt gedaan voor verschillende (soorten) invoerbestanden en waarden. Een voorbeeld daarvan is de reactie op randgevallen en corrupte bestanden. Daarnaast worden ook systematisch unit- en integratietesten uitgevoerd.

4.1 JUnit-tests

Er zijn geautomatiseerde unit-testen voor de HTMLWriter, TemplateWriter en het parsen van de argumenten op de commandolijn (converter). Er is een extra testklasse voor het uitschrijven van afbeeldingen naar HTML, waar alle mogelijke groottes getest worden (kleiner dan, binnen of groter dan een bepaald interval). Die testen zo goed als alle mogelijke invoer en controleren of de output gelijk is aan de verwachte waarde(n). Deze testen worden uitgevoerd bij elke wijziging aan de code van de overeenkomstige module.

4.2 Integratietesten

Op het niveau van de converter zijn er integratietesten om het correct samenwerken van de PDFConverter en PPTConverter, logger, outputklassen en logica te testen. Die testen kunnen allemaal tegelijk uitgevoerd worden, maar het resultaat moet handmatig gecontroleerd worden indien er geen excepties opgetreden zijn. Er wordt getest voor verschillende combinaties van argumenten (raw of shower, wel of niet uitschrijven naar een zipbestand, naar een andere *output directory*, wel of geen *course* en/of *chapter* ingesteld ... Deze testen worden uitgevoerd telkens een module een update heeft gekregen en toegevoegd is aan de converter.

4.3 Webapplicatie testing

Om de webapplicatie te testen zijn twee strategieën gebruikt. Tijdens de startfase van de ontwikkeling is vooral gebruik gemaakt van *Grey-box testing*. Hierbij wordt de hele keten geïnspecteerd van het uploaden tot het downloaden, zonder de *internals* van het conversieproces te onderzoeken.
Eens de webapplicatie een stabiele vorm had, is ook gebruik gemaakt van *Black-box testing*, waarbij er enkel gewerkt wordt vanuit het standpunt van een gewone gebruiker.
Specifieke testonderwerpen:

- Gebruiksgemak: Buitenstaanders gebruik laten maken van de applicatie en na observatie ook naar hun ervaring en feedback vragen. Het gebruik van deze methode was vooral bevorderlijk om de layout van de applicatie te verbeteren.
- Robuustheid: Proberen de limieten op te zoeken van de applicatie door o.a. zeer grote bestanden van verschillende gebruikers gelijktijdig aan de applicatie te overhandigen. Foutafhandeling proberen uit te lokken om te zien hoe de applicatie reageert in onverwachte situaties.
- Gebruik van systeembronnen: Monitoring van het gebruik van bronnen (CPU, geheugen, netwerk I/O, schijf I/O) voornamelijk op de server en in mindere mate bij de clients.

Het resultaat van deze testen is dat er door de volledig berichtgebaseerde architectuur amper verbruik is van systeembronnen op momenten waarop de applicatie *idle* is. Het geheugengebruik blijft ook beperkt door correct beheer van datastructuren in de applicatie. Het langdurig actief zijn van de applicatie op de server vertoont geen kruipende groei in het geheugengebruik.

- Beveiliging: tijdens de ontwikkeling is extensief gebruik gemaakt van de BurpSuite Proxy om netwerkverkeer te inspecteren. Hoewel het vroegere gebruik voornamelijk was om op een vlotte manier de communicatie te inspecteren, is de tool later ook gebruikt om eventuele zwakten op te sporen in de applicatie.
  Er is voornamelijk gelet op XSS, user impersonation, file-inclusion, malicious file-upload en directory-traversal. Deze testen zijn voornamelijk manueel uitgevoerd. Een andere test is geautomatiseerd uitgevoerd door OWASPs Zed Attack Proxy.
  Het resultaat van die laatste test is een security Filter die bijkomende headers toevoegt aan berichten om bv. gebruik te maken van de anti-XSS mogelijkheden die ingebouwd zijn in Google Chrome en Internet Explorer.

Hoofdstuk 5: Evaluatie en discussies

5.1 Performantie

Er zijn drie onderdelen die de performantie van de applicatie bepalen. Deze zijn de conversie van PowerPoint naar onze datastructuur in het geheugen, de logica die een verbetering van de elementen in de datastructuur doet en de conversie van de datastructuur naar uitvoer.

Uit de uitgevoerde testen blijkt de grootste geheugenbelasting te liggen bij het inlezen en verwerken van de oorspronkelijke bestanden. Het uitschrijven van de webslides als HTML heeft een verwaarloosbare invloed op het geheugen voor de meest voorkomende bestanden.

5.1.1 Conversie van PowerPoint naar datastructuur

De conversie van PowerPoint naar datastructuur is het onderdeel dat het meeste impact heeft op de performantie. Hij moet de grootste hoeveelheid data overlopen en testen, en mediabestanden kopieëren. Tussen PPTX en PDF conversie is een verschil in performantie. Ze werken op een compleet verschillende manier en hebben bijgevolg een verschillende uitvoeringssnelheid.

PPTX conversie

De basis van de PPTX conversie is het uitlezen van de xml van iedere slide met SAX. SAX is een van de meer performante manieren om xml uit te lezen (uitgebreide testing te vinden op `https://www.javacodegeeks.com/2011/12/jaxb-sax-dom-performance.html`). Ook is de code zo goed mogelijk geoptimaliseerd. Bij het uitlezen van xml met behulp van SAX gaat iedere keer de naam van het element uitgelezen worden en vervolgens getest worden of die gelijk is aan één van de constanten die in de code staat. De optimalisatie die hier gebeurt, is het logisch testen van deze naam. Zo zou het niet performant zijn om de naam te gaan vergelijken met constanten die te maken hebben met tekst wanneer er op dat moment geweten is dat er een foto wordt uitgelezen. Na het uitlezen is het PPT-object opgevuld. De informatie die deze bevat is echter nog niet compleet en moet dus door logica worden aangevuld. Op deze plek werd gekozen voor overzichtelijke, modulaire code boven performantie. Dit is echter een minimaal verschil. Op dit moment zit enkel de nuttige informatie van de slide in het geheugen, wat zich vertaalt in gemiddeld minder dan tien objecten (vaak zelfs enkel titel, lijst en foto). Daardoor is het verschil tussen deze objecten een of twee keer te overlopen verwaarloosbaar. De logica zal naast bewerkingen op de objecten ook zorgen dat de nodige mediabestanden uit de PowerPoint worden gekopieerd en opgeslagen op een gewenste locatie.
Tijd is normaalgezien niet relevant bij performantie, aangezien iedere computer waar het programma op uitgevoerd wordt andere hardware heeft. Om toch een indicatie te geven van grootte-orde zijn enkele testen uitgevoerd op een laptop met 8gb ram en een intel i5 5th-gen processor, met andere woorden een computer die de meerderheid van computergebruikers kan vertegenwoordigen. Op een dergelijke computer duurt het voor een pptx PowerPoint die 200 foto's, vijf video's en 200 slides bevat, tien seconden om ingelezen te worden. Hierbij neemt het kopiëren van de mediabestanden de meeste tijd in.

PDF conversie

Voor PDF conversie wordt er gesteund op PDFBox. Er wordt slechts 1 PDDocument aangemaakt op basis van het meegegeven bestand. Dit PDDocument wordt verschillende keren overlopen. Bij het inlezen zijn tekstextractie, hyperlinkherkenning en afbeeldingsinformatie gebundeld. Een tweede ronde is nodig om de fysieke afbeeldingen uit het bestand te halen. Een derde en laatste ronde herkent tabellen. Na het inlezen is het PPT-object ingevuld. Er volgt echter nog heel wat naverwerking waarbij het PPT-object volledig overlopen wordt. Bij de extractie blijkt vooral het ophalen en opslaan van de afbeeldingen vertragend te werken.

5.1.2 Verbetering van datastructuur

Nadat het pptx- of pdf-bestand wordt ingelezen door de bijhorende converter, worden bepaalde componenten herkend met behulp van logica. Het PPT-object wordt hierbij meerdere keren overlopen. Het omzetten van bv. gewone tekst naar lijsten gebeurt echter zeer snel en heeft weinig invloed op de performantie.

5.1.3 Conversie van datastructuur naar uitvoer

In vergelijking met het inlezen en verwerken van de oorspronkelijke bestanden gebruikt het uitschrijven naar HTML zeer weinig geheugen. Voor de meest voorkomende bestanden is er geen echte invloed op het geheugengebruik te merken. Alle PPT-objecten worden daarbij éénmaal overlopen. HTML code is in feite gewoon tekst en die tekst wordt per slide onmiddellijk weggeschreven naar een buffer. Op die manier is er nooit te veel tekst in het geheugen en kan het systeem zelf beslissen wanneer I/O naar het bestandssysteem nodig is.

5.2 Security

Het gebruik van een webapplicatie betekent dat gebruikers met de applicatie interageren. Er moet echter vanuit gegaan worden dat enkele actoren acties zullen uitvoeren, per toeval of moedwillig, die niet toegestaan of verwacht zijn. Daarom is in eerste instantie gezorgd voor nette foutafhandeling in de applicatie om robuuster te zijn. Specifiek voor de webapplicatie is gezorgd dat de server geen gebruik meer maakt van de standaardfoutberichten die in veel gevallen te veel informatie onthullen over de aard van de fout. Andere testen in verband met beveiliging zijn ook uitgevoerd, hiervoor wordt verwezen naar paragraaf 4.3 (Webapplicatie testing subsectie beveiliging).

5.3 Schaalbaarheid

Voor dit project wordt de conversie uitgevoerd op één enkele server. Alle bestanden worden dan ook geüpload naar hetzelfde bestandssysteem.
De convertor kan evengoed parallel uitgevoerd worden op meerdere servers. Dat kan zowel vanuit een bestandssysteem als een databank waar de bestanden in opgeslagen worden. Daarbij kan zowel gekozen worden voor één enkele databank of een gedistribueerd databanksysteem. Ook is de keuze voor een bestandssysteem of databank voor de geüploade bestanden onafhankelijk van diezelfde keuze voor de geconverteerde bestanden.

5.4 Problemen en geleerde lessen

Gedurende het project zijn een aantal lessen geleerd waarvan we hier een kort overzicht geven.

Als eerste blijkt het veel moeilijker om een pdf-bestand correct in te lezen en te verwerken. Hoewel pdf gezien wordt als een standaard blijkt er toch een verschil te zitten in de structuur. Zo zijn een geëxporteerde pdf en een geprinte pdf met een pdf-printer niet hetzelfde, wat je wel zou verwachten. Een gescheiden bestandssysteem voor elke gebruiker kan daarbij een oplossing zijn.

Ten tweede is het meermaals gebeurd dat een bepaalde functionaliteit getest is en perfect werkt lokaal, maar bij het uploaden naar de server toch niet blijkt te werken. De fout blijkt dan meestal te liggen bij andere omgevingsvariabelen, verkeerde paden of andere instellingen. Dit zou deels te vermijden zijn door de testen ook uit te voeren op andere omgevingen en besturingssystemen.

Als laatste is een goed beheer van bestanden noodzakelijk op de server. Een voorbeeld daarvan is wanneer twee gebruikers kort na elkaar een bestand met dezelfde bestandsnaam uploaden. Er moet voorkomen worden dat terwijl de convertor bezig is met het verwerken van het eerste bestand, het bestand overschreven wordt door de tweede gebruiker. Dat leidt tot een corrupt bestand en een mislukte conversie voor beide.

Hoofdstuk 6: Handleidingen

Er zijn twee manieren om het pptx- of pdf-bestand te converteren naar Open Webslides.

De eerste manier is het eenvoudigst. Via de website `http://webslide.ugent.be/` kunnen pptx- en pdf-bestanden geüpload worden. Met behulp van de bijkomende opties kan er gekozen worden voor Shower (standaard template) of voor raw (enkel de HTML5-code van de slides). De website kan gemakkelijk gebruikt worden door gebruikers zonder technische kennis.

Als tweede manier kan er ook lokaal gewerkt worden. Daarvoor moet het jar-bestand van het programma lokaal beschikbaar zijn. Op de website is een downloadlink te vinden. Het commando voor de conversie is:

java -jar OpenWebslidesConverter.jar -i <inputbestand> [opties]

met als inputbestand een pptx- of PDF-bestand. De mogelijke opties zijn:

vlag	**omschrijving**	**mogelijke waarden**	**standaardwaarde**
-o	*output directory*	*	output
-t	*output*type	raw | shower	raw
-zip	zipbestand als output		
-co	*course* (vak)**	*	
-ch	*chapter* (hoofdstuk)**	*	

** kunnen enkel ingesteld worden als het type (-t) *shower* is

Enkele parameters kunnen ook ingesteld worden met een *properties*-bestand. Meer info is te vinden in de uitgebreide handleiding op
`https://github.ugent.be/iii-vop2017/webslides-01/wiki/Converter-manual`.

Hoofdstuk 7: Besluit

Met Open Webslides wordt een toegankelijke en (inter)actieve leeromgeving aangeboden die cocreatie bevordert aan de hand van online cursusslides gemaakt in HTML. De bedoeling van dit project was om een converter te maken om bestaande slides uit pptx- en pdf-bestanden om te zetten naar Open Webslides.

Er is ondersteuning voor twee soorten bestanden, namelijk pptx-bestanden en pdf-bestanden. De structuur van de pptx-bestanden wordt overgenomen. Titels, lijsten, tabellen ... worden herkend. Deze objecten worden ook in de pdf herkend met behulp van logica.
Bestanden (pptx en pdf) kunnnen zowel via de website als lokaal omgezet worden naar Open Webslides. Er is een webapplicatie waar bestanden geüpload kunnen worden. Bij geldige pptx- en pdf-bestanden wordt een zip-folder teruggegeven met de inhoud van de geüploade bestanden omgezet naar Open Webslides. Er is ook keuze tussen meerdere opties, namelijk raw (pure HTML5) en Shower (standaard template). Op de webapplicatie is er ondersteuning voor meerdere talen.
Er is ook voor gezorgd dat meer technische gebruikers hun pptx-bestanden en pdf-bestanden lokaal kunnen converteren naar Open Webslides. Hiervoor werden zoals besproken meerdere opties gebruikt.