

Bakalářská práce



České  
vysoké  
učení technické  
v Praze

**F3**

Fakulta elektrotechnická  
Katedra počítačů

## Vytvoření REST služby pro konverzi LaTeX souborů do PDF

**TADEÁŠ KYRAL**

Vedoucí: Ing. LUKÁŠ ZOUBEK

Obor: Softwarové inženýrství a technologie

Zaměření: žádné

Listopad 2018



## Poděkování

V první řadě bych chtěl poděkovat svému vedoucímu JMENO za příjemnou spolupráci a cenné rady při našich konzultacích. Dále bych chtěl poděkovat mé rodině a kamarádům, kteří mě při psaní této práce podporovali.

## Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně, a že jsem uvedl veškerou použitou literaturu.

V Praze, 2. listopadu 2018

## Abstrakt

Práce pojednává o návrhu a vývoji webové služby pro konverzi LaTeXu do PDF, pro uživatele v Moodle a CourseWare. Doplněk má uživatelům usnadnit práci s LaTeXovými soubory přímo skrze portál Moodle, bez potřeby stahování a kompilace na svém stroji. Byla naimplementována služba na server a následně byla zapojena do provozu. Uživatelům portálů se usnadnila práce s LaTeX soubory.

**Klíčová slova:** PDF, LaTeX, PDF, Moodle, CourseWare, Web service, REST, Java EE

**Vedoucí:** Ing. LUKÁŠ ZOUBEK  
Katedra ekonomiky, manažerství a humanitních věd

## Abstract

Text follows...

**Keywords:** PDF, LaTeX, PDF, Moodle, CourseWare, Web service, REST, Java EE

**Title translation:** Create REST service for conversion from LaTeX to PDF

# Obsah

<b>1 Úvod</b>	<b>1</b>
1.1 Cíle práce .....	1
<b>2 Teorie a technologie</b>	<b>2</b>
2.1 REST .....	2
2.2 PDF .....	3
2.3 LaTeX .....	4
2.4 Moodle a CourseWare .....	5
2.5 FURPS+ .....	5
<b>3 Analýza</b>	<b>7</b>
3.1 Požadavky .....	7
3.2 Existující řešení .....	9
3.3 Kompilátory .....	10
<b>4 Vlastní řešení</b>	<b>12</b>
4.1 Návrh .....	12
<b>5 Implementace</b>	<b>15</b>
<b>6 Závěr</b>	<b>16</b>
<b>Literatura</b>	<b>17</b>
<b>A Seznam zdrojů</b>	<b>18</b>
<b>B Seznam zkratk</b>	<b>19</b>
<b>C Testování</b>	<b>20</b>

## Obrázky

2.1 Struktura objektů .....	4
4.1 Sekvenční diagram .....	13

## Tabulky

3.1 Funkční požadavky .....	7
3.2 Nefunkční požadavky .....	8

# Kapitola 1

## Úvod

$\text{\LaTeX}$  patří v akademickém prostředí mezi velmi oblíbené typografické systémy a je hojně využíván ke psaní skript a odborných prací. Mnoho akademiků využívá  $\text{\LaTeX}$  i k vytváření materiálů pro studenty, primárně v oblasti matematiky, jelikož nabízí jednoduché nástroje ke psaní vzorců. Hlavním zdrojem materiálů pro studenty jsou portály Moodle a CourseWare, kam vyučující dokumenty nahrávají a můžou je tam i upravovat. Nynější editor podporuje jenom řádkové příkazy a není schopen zpracovat celý  $\text{\LaTeX}$  dokument natož s více zdrojovými soubory. Tento stav mnoha učitelům nevyhovuje, protože by chtěli svoje dokumenty upravovat a přímo kompilovat v prohlížeči bez potřeby je stahovat a následně zase nahrávat.

Práce se dělí na čtyři hlavní části. Nejdříve proběhne seznámení s potřebnou teorií a technologiemi, dále budou specifikovány požadavky na službu z čehož se najdou již podobné existující řešení a porovnají se s požadavky. Následně se přistoupí k návrhu samotné aplikace.

### 1.1 Cíle práce

Hlavním cílem bakalářské práce je poskytnout konverzi  $\text{\LaTeX}$  souborů do PDF pro uživatele Moodle a CourseWare. Čehož bude dosaženo implementováním webové služby, která bude schopna komunikovat pomocí REST Api. K tomu vedoucí jednotlivé dílčí cíle jako analýza existujících řešení, návrh, implementace a v neposlední řadě testování.

## Kapitola 2

### Teorie a technologie

V této části si přiblížíme technologie potřebné k návrhu a vývoji webové služby specifikované v zadání práce. Postupně budou vysvětleny všechny zásadní pojmy, které pomůžou čtenáři doplnit znalosti v dané problematice.

#### 2.1 REST

Representational state transfer (dále jen REST) je architektura pro komunikaci mezi distribuovanými systémy. Termín zavedl R. T. Fielding ve své disertační práci[3], kde toto rozhraní bylo taktéž popsáno a vymezeno. Mimo jiné stanovil i těchto 5 základních pravidel, které by měly být dodrženy, aby se aplikace nazývala RESTful[4]:

- Klient-Server(Client-Server) - Toto omezení staví na principu oddělení zodpovědností (Separation of Concerns), nebo-li je klientská část starající se o uživatelské rozhraní a serverová část přistupující k databázi. Zlepšuje škálovatelnost systému a zjednodušuje použitelnost na různých platformách.
- Bezstavovost(Stateless) - Každý požadavek musí přenášet všechna související data, server totiž neuchovává žádné informace o nynějším spojení a každý požadavek bere jako nový.
- Keš(Cache) - Data přenášená v odpovědi mohou být označena jako kešovatelná, tudíž si je klient může uložit a kdykoliv použít znova.
- Jednotné rozhraní(Unified interface) - Základem tohoto omezení je princip HATEOAS (Hypermedia As The Engine Of Application State), které říká, že klient nepotřebuje znát pravidla komunikace dopředu a data musí obsahovat odkazy na další data v aplikaci. Měla by být jasně definovaná adresa zdroje (např. URI), reprezentace přenášených dat (např. HTML), typ média (např. JSON)
- Vrstvený systém(Layered system) - Přidáním vrstev se aplikace, kde každá vrstva je izolovaná a může komunikovat jenom se sousedícími vrstvami, zpřehlední a zlepší se její škálovatelnost.

Nejčastějším typem protokolu využívající tuto architekturu je Hypertext Transfer Protocol(HTTP). Pomocí čtyř hlavních metod GET, PUT, POST, DELETE v požadavku poslaného z klienta server buď odpovídající vrátí požadovaná data, přijme data poslané v těle a uloží do databáze, nebo data vymaže.



## ■ 2.2 PDF

Portable Document Format(dále jen PDF), jak už název napovídá, jedná se o formát dokumentů, jejichž hlavním cílem je poskytování nezávislosti na platformě. Velmi podobný programovacímu jazyku PostScript z kterého vzešel. Syntaxe souboru je nejlépe pochopitelná jako čtyři komponenty[1]:

- Objekty(Objects) - Základní jednotka celého souboru např. Pole, Čísla, Řetězce znaků. Jednotlivé objekty se popisují množinou znaků, která je definována lexikálními konvencemi.
- Struktura souboru(File Structure) - Samotný soubor se skládá ze čtyř částí, jak je i vidět na obrázku 2.1, ty si lehce popíšeme:
  - hlavička - identifikuje verzi PDF
  - tělo - obsahuje použité objekty, které reprezentují obsah dokumentu
  - tabulka odkazů - obsahuje odkazy na objekty v podobě počtu bytů od začátku souboru, kvůli náhodnému přístupu bez potřeby číst celý soubor
  - závěrečná sekce - udává pozici tabulky odkazů a speciálních objektů

Tato struktura napomáhá k náhodnému přístupu k jednotlivým částem a usnadňuje jejich aktualizaci.

- Struktura dokumentu(Document Structure) - Popisuje hierarchickou strukturu objektů v těle dokumentu.
- Content stream - Objekt v kterém se nacházejí instrukce k vykreslování grafických elementů. Každá stránka má minimálně jeden a na rozdíl od ostatních objektů je procházen sekvenčně.



## 2.3 LaTeX

Vychází z typografického sázecího systému T<sub>E</sub>X, který popisuje Pavel Satrapa ve své knize [5]: „*Patří do rodiny tak zvaných značkovacích jazyků (markup languages) a dal by se zjednodušeně charakterizovat jako programovací jazyk pro sazbu textů. Jeho základním vstupem je textový soubor, který obsahuje jak sázený dokument, tak příkazy ovlivňující sazbu. Určité znaky mají přiřazen speciální význam a jejich prostřednictvím jsou v textu odlišeny řídicí konstrukce. Typickým příkladem je zpětné lomítko, jímž začínají příkazy.*“

L<sup>A</sup>T<sub>E</sub>X je rozšíření T<sub>E</sub>Xu o balíček přednastavených řídicích konstrukcí. Hlavní cílem těchto systémů je jednoduchost pro psaní matematických a jiných vzorců. Ovšem je také velmi oblíben kvůli silné možnosti jednoduše upravovat dokumenty ke svému zalíbení, přestože prvotní seznámení je náročnější oproti jiným nástrojům ke psaní.

Soubor se skládá z preambule, která obsahuje nastavení pro celý dokument, toto nastavení může být v jiném souboru a případně používané balíčky. Druhá část je dokumentu v kterém se mimo jiné nachází kapitoly, sekce a podobně. Základní syntaxe může vypadat takto.

```
\documentclass{thesis}
\usepackage{csquotes}

\begin{document}
Hello world!
\end{document}
```

Základní soubory mají příponu `tex` kromě souboru s nastavením, ten je ve formátu `cls`, neboli `class file`. Tyto soubory mohou být uspořádány do stromové struktury, kde v kořenu stromu je jeden hlavní soubor, do kterého jsou vnořovány další, ale vše může

být i v jednom. To napomáhá přehlednosti velkých dokumentů a také používání již vytvořených.

Pro získání výsledného dokumentu, například ve formátu PDF, je potřeba vše zkompileovat. To provádí kompilátor.

## ■ 2.4 Moodle a CourseWare

Ačkoliv tyto portály se v mnoha aspektech liší v rámci této práce je můžeme považovat za ekvivalentní. Oba slouží pro podporu studia a to hlavně v podobě poskytování informací o předmětech studentům, které je mají zapsané. Vyučující takto může předat informace o harmonogramu kurzu, přednášky a jiné materiály. Moodle je komplexnější open-source systém s možností upravovat ho ke své potřebě, který navíc poskytuje prostředí pro vytváření a psaní testů, zapisování docházky, komunikaci se studenty a jiné.

## ■ 2.5 FURPS+

Vychází z klasifikace požadavků FURPS(**F**unctionality, **U**sability, **R**eliability, **P**erformance, **S**upportability) s kterou přišel Robert Grady v roce 1992. Roku 1999, Jacobson a el rozšířili specifikaci o znaménko "+", které přidává požadavky a omezení na návrh, implementaci, rozhraní a hardware[2].

### ■ Funkční požadavky

- Funkčnost(Functionality) - Požadavky popisující všechny hlavní prvky produktu i důležité aspekty z pohledu architektury např. lokalizovaný systém pro více jazyků.

### ■ Nefunkční požadavky

- Použitelnost(Usability) - Zaměřuje se na uživatelskou přívětivost nejenom samotné aplikace, ale i dokumentace, týkající se estetiky a konzistence.
- Spolehlivost(Reliability) - Spolehlivost systému v podobě doby běhu, správnosti fungování a četnosti výpadků.
- Výkon(Performance) - Vypovídá o výkonnosti systému, jak rychle dokáže zpracovávat požadavky, spustit se atd.
- Podporovatelnost(Supportability) - Popisuje testovatelnost, škálovatelnost, konfigurovatelnost...
- Návrh(Design) - Omezení na návrh systému např. požadavek na relační databázi
- Implementace(Implementation) - Specifikuje typ programovacího jazyku, platformu apod.
- Rozhraní(Interface) - Komunikace s externími systémy
- Fyzické(Physical) - Definuje požadavky na hardware, na kterém daný software poběží i co se týče fyzické velikosti



## Kapitola 3

### Analýza

#### 3.1 Požadavky

Nyní představíme požadavky kladené na aplikaci. Ty jsou buď požadované zadávajícím nebo odvozené z potřeb, ke kterým bude služba používána. Ale i z omezení plynoucích z prostředí v jakém bude provozována, v tomto případě pro fakultu státní vysoké školy, jmenovitě Fakulta elektrotechnická, ČVUT.

Za pomoci metody FURPS+ rozdělíme požadavky a omezení na funkční a nefunkční.

##### 3.1.1 Funkční požadavky

Typ	Požadavky
Funkčnost	<ul style="list-style-type: none"><li>• Webová služba s REST rozhraním pro komunikaci</li><li>• Umět přijmout požadavky a soubory</li><li>• Zkompilování <math>\text{\LaTeX}</math> souborů s požadovaným nastavením do PDF</li><li>• Uložení PDF a jeho poskytnutí</li><li>• Uživatel může zvolit tyto nastavení:<ul style="list-style-type: none"><li>• Přidat vodoznak na každou stranu PDF</li><li>• Nakonec souboru přidat stránku s předem stanoveným obsahem</li><li>• Výsledný soubor PDF bude chráněný</li><li>• Výsledný soubor PDF nebude kopírovatelný nebo tisknutelný</li></ul></li></ul>

**Tabulka 3.1:** Funkční požadavky

Služba má jediný a specifický účel, z toho plyne menší množství funkcionalit.

### 3.1.2 Nefunkční požadavky

Typ	Požadavky
Použitelnost	<ul style="list-style-type: none"> <li>Dokumentace k rozhraní na swagger</li> <li>Přístup skrz REST Api, pouze s tokenem</li> </ul>
Spolehlivost	<ul style="list-style-type: none"> <li>Služba není kritická</li> <li>Uptime 95% času</li> <li>Ochrana proti špatnému vstupu</li> <li>Autorizace pomocí tokenů</li> </ul>
Výkon	<ul style="list-style-type: none"> <li>Zvládnutí obsluhy desítky požadavků najednou</li> <li>Ukládání výsledných dokumentů po dobu jednoho měsíce</li> <li>Maximální doba tvorby dokumentu 5 minut</li> </ul>
Podporovatelnost	<ul style="list-style-type: none"> <li>Služba může být rozšířena o kompilování samotného TeXu a může podporovat vytváření i jiných formátů z L<sup>A</sup>T<sub>E</sub>Xu</li> </ul>
Implementace	<ul style="list-style-type: none"> <li>Platforma - Java EE</li> <li>Komunikace - REST Api</li> </ul>
Rozhraní	<ul style="list-style-type: none"> <li>Komunikuje s Moodle a CourseWare, tyto portály posílají data</li> </ul>
Fyzické	<ul style="list-style-type: none"> <li>Musí být provozováno na serverech ČVUT</li> </ul>

**Tabulka 3.2:** Nefunkční požadavky

Některé z požadavků zdůvodníme.

- Java EE - Je požadováno kvůli prostředí vývoje, kde to je hlavní platforma. Tudíž je zajištěna podpora.
- Služba není kritická - Poskytuje funkčnost, která nijak neovlivňuje základní funkcionality určených portálů.
- Musí být provozováno na serverech ČVUT - Jelikož aplikace bude pracovat s dokumenty a popřípadě i informacemi, spojenými s pracovníky školy, je potřeba tyto data chránit a uchovávat na vlastních serverech.

Z tabulky je vidět, že služba není nijak kritická a působí jenom jako doplněk do výše zmíněných portálů. Musí ovšem splňovat vyšší bezpečnostní nároky zapříčiněné prostředím v jakém se bude používat.

## 3.2 Existující řešení

Na základě stanovených požadavků v minulé kapitole přejdeme k analýze již existujících řešení. Momentálně uživatelé Moodle mohou vkládat do svých souborů řádkové příkazy, což není úplně dostačující a nesplňuje to požadavky. Samozřejmě se dají používat pro vytváření PDF z  $\text{\LaTeX}$  souborů kompilátory, které si můžete stáhnout a používat lokálně. To ovšem není předmětem této práce, a proto se podíváme na webové služby, které více odpovídají potřebám. Pár vybraných si představíme.

### 3.2.1 OverLeaf

Placená služba<sup>1</sup> pro tvorbu  $\text{\LaTeX}$  dokumentů, která umožňuje i používání s omezeními zadarmo. Je velmi oblíbená hlavně kvůli hezkému prostředí pro tvorbu dokumentů a jejich správu. Také nabízí výhody pro určité zájmové skupiny, nejzajímavější vzhledem k tématu této práce je předplatitelská služba OverLeaf Commons<sup>2</sup>. Ta poskytuje sdílené prostředí se všemi výhodami pro zaměstnance a studenty univerzity.

### 3.2.2 BlueLaTeX

Open-source služba<sup>3</sup> pro kompilaci  $\text{\LaTeX}$  souborů. Kompilovat můžete na jejich serveru nebo nabízejí kód pro spuštění na vlastním. Kromě samotné serverové implementace je k dispozici i webový klient. Vše je pouze zatím v Beta verzi a samotní tvůrci varují před možnými nedostatky a problémy. Hlavním lákadlem je souběžná spolupráce více lidí na jednom dokumentu a také možnost provozovat server lokálně.

### 3.2.3 ScienceSoft

Starší služba<sup>4</sup>, která mimo kompilace  $\text{\LaTeX}$  souborů vložených přímo na stránky poskytuje i jiné rozhraní pro posílání souborů a to skrz REST Api a SOAP. Jak bylo řečeno, tak tato služba je už starší tudíž pro kompilaci používá TexLive 2008.

### 3.2.4 ShareLaTeX

Velmi podobný BlueLaTeXu, tedy open-source<sup>5</sup> nabízející jimi hostovanou verzi nebo lokální verzi pro vlastní potřebu, obě verze obsahují mnoho funkcí včetně grafického prostředí pro uživatele. Pod jménem ShareLaTeX se vyskytuje jenom výše zmíněné, ovšem společně s OverLeaf také spravují službu Pro<sup>6</sup>, která je určená pro firmy, ale i univerzity. Ta se pyšní možností nasazení na vlastních serverech, správou uživatelů, podporou a zabezpečením.

<sup>1</sup><https://www.overleaf.com/for/universities>

<sup>2</sup><https://www.overleaf.com/for/universities>

<sup>3</sup><http://www.bluelatex.org/>

<sup>4</sup><http://sciencesoft.at/latex/index?lang=en>

<sup>5</sup><https://github.com/sharelatex/clsi-sharelatex>

<sup>6</sup><https://www.overleaf.com/for/enterprises>

### ■ 3.2.5 Porovnání

Každé z těchto řešení má své problémy, ať už že je zastaralé a neaktualizované nebo nesplňuje zanalyzované požadavky (kapitola 3.1). Do první skupiny patří ScienceSoft a BlueLaTeX, kde první z jmenovaných ani neposkytuje kód pro implementaci na lokálním serveru a druhý je v nedodělaném stavu, což může vést k nefunkčnosti a bezpečnostním rizikům. Služba Overleaf Commons nesplňuje stejný požadavek jako řešení od ScienceSoft, ale nabízí zajímavé prostředí pro vytváření a správu L<sup>A</sup>T<sub>E</sub>X dokumentů pro studenty i zaměstnance, o čemž by univerzita mohla popřemýšlet. Nejnadějnější možností se zdá být ShareLaTeX, který poskytuje k použití i jenom backendovou část pro kompilaci L<sup>A</sup>T<sub>E</sub>X a komunikaci, ale celý je implementovaný v CoffeeScriptu, což si rozporuje s požadavkem na implementaci, kde je Java EE.

## ■ 3.3 Kompilátory

Nejdůležitější částí celé aplikace bude kompilátor, který bude provádět kompilaci L<sup>A</sup>T<sub>E</sub>X souborů, ale může poskytnout i podporu pro operace s výsledným PDF. Tyto operace vycházejí ze stanovených funkčních požadavků (kapitola 3.1.1). Jelikož kompilace bude probíhat na serveru s Linuxovým operačním systémem.

### ■ 3.3.1 MikTeX

MikTeX<sup>7</sup> je velmi oblíbený hlavně na operačním systému Windows, díky příjemné instalaci a kvůli možnosti doinstalovávat balíčky "on-the-fly", neboli za běhu. Ale je poskytován i mimo jiné pro Linux a to například pro potřeby této práce v zajímavé verzi "Just enough TeX". Tato instalace obsahuje jen to nejnútnější, tedy bez zbytečných balíčků navíc. Je vyvíjen jediným programátorem, který se stará o celou distribuci, což je do budoucna lehce rizikové z pohledu udržitelnosti. Podporované Linuxové operační systémy jsou např. Ubuntu 16.04, Ubuntu 18.04 a Debian 9.

### ■ 3.3.2 TeXLive

Existují dvě různé cesty jak TeXLive<sup>8</sup> nainstalovat a od toho se odvíjí správa balíčků. Verze Native TeX Live a distribuce přímo pro daný operační systém. Liší se ve počtu balíčků po instalaci a způsobu jejich aktualizací. V Native verzi probíhají aktualizace pomocí TeX Live Manager nebo-li tlmgr, jinak se o to stará samotný operační systém. Pokud si vyberete Native je možno si i zvolit schéma, které určuje množství balíčků např. scheme-basic obsahuje jenom to nejnútnější.

### ■ 3.3.3 Porovnání

Oba kompilátory jsou si velmi podobné a liší se jenom v drobnostech, některé už byly nastíněny v jejich popisech. Jedním z důležitých aspektů je velikost instalace, tedy i s balíčky, v této kategorii nabízejí oba to samé. K tomuto se váže práce s balíčky,

<sup>7</sup><https://miktex.org/kb/just-enough-tex>

<sup>8</sup><https://www.tug.org/texlive/pkginstall.html>



kde už je jiná situace a to kvůli jasné výhodě MikTexu stahovat balíčky během běhu. Něco podobného jde dosáhnout i TexLive, ale je nutno použít externí skripty, což může mnohem více ovlivňovat rychlost kompilace.

## Kapitola 4

### Vlastní řešení

V této části navrhne podobu vlastního řešení problému specifikovaného v předchozích kapitolách.

#### 4.1 Návrh

Tato práce se zabývá webovou službou, která bude poskytovat svoje rozhraní a funkcionalitu portálům Moodle a CourseWare. Jedná se o jednoduchou aplikaci s jediným účelem, tedy kompilace  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  souborů do PDF. Tudíž není potřeba ani persistentní vrstvy, výsledný soubor PDF bude uložen do klasického souborového systému.

##### 4.1.1 Platforma

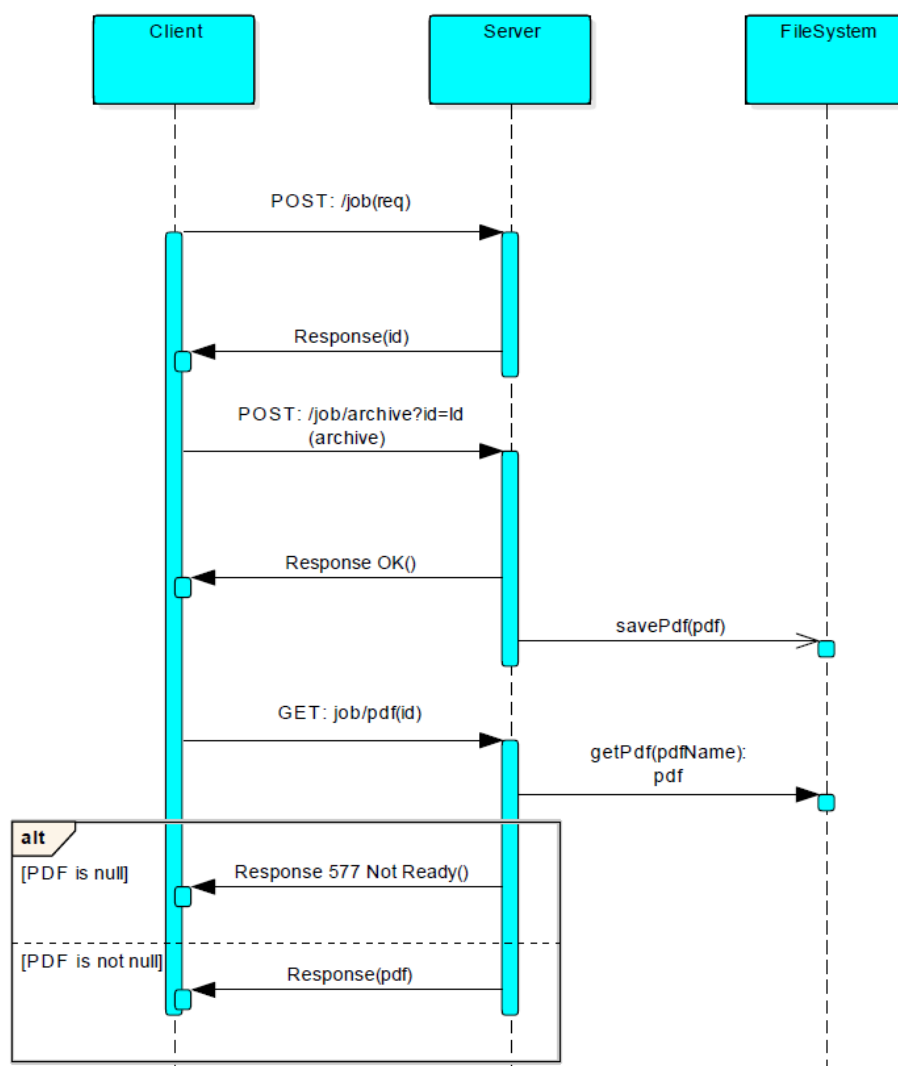
Na základě požadavků bude aplikace postavena na Java EE, což je platforma pro vývoj webových aplikací, rozšiřující standardní Javu SE. Oproti ní poskytuje některé zásadní techniky navíc, jednu se tedy představme.

**Vkládání závislostí** (dependency injection) umožňuje objektu používat jiné objekty bez potřeby ho zatěžovat jejich vytvářením. Objekty, které můžeme takto vkládat se nazývají beans a právě o jejich vytváření a zánik se stará Contexts and Dependency Injection(dále jen CDI) kontejner.

Dále je potřeba specifikovat aplikační server, ten poskytuje pro webové aplikace běhové prostředí, tedy zajišťuje správu databázových spojení apod. Na základě zkušeností je vybrán open-source Payara, který staví na GlassFish, oproti němu poskytuje častější aktualizace a opravy chyb.

##### 4.1.2 Komunikace

Pro komunikaci mezi klientem a serverem je vybráno REST Api. Poskytuje jednoduché rozhraní s kterým je schopen komunikovat jakýkoliv systém pouze za znalosti struktury požadavků a odpovědí. Posílání zpráv bude postaveno nad protokolem HTTP pomocí GET a POST metod. Na obrázku 4.1 je zobrazena komunikace mezi serverem a klientem.



Obrázek 4.1: Sekvenční diagram

Kompletní dokumentace k rozhraní se nachází na <https://app.swaggerhub.com/apis/Tadky/Thesis/1>. Na ukázkou vypíšeme alespoň používané zdroje:

- **POST** /job?token=<clientToken> - Slouží k předání požadavků na výsledné PDF.
- **POST** /job/archive?id=<jobId>&token=<clientToken> - Musí obsahovat zip archiv s potřebnými soubory pro kompilaci.
- **GET** /job/pdf?id=<jobId>&token=<clientToken> - Vrací výsledné PDF, pokud už je hotové.

Autentizace bude probíhat skrz tokeny, které budou mít dané portály vygenerované a budou se nimi ověřovat.





## **Kapitola 5**

### **Implementace**



## Kapitola 6

### Závěr

Text follows



## Literatura

- [1] Document management - portable document format - part 1: Pdf 1.7. [https://www.adobe.com/content/dam/acom/en/devnet/pdf/PDF32000\\_2008.pdf](https://www.adobe.com/content/dam/acom/en/devnet/pdf/PDF32000_2008.pdf), 2008.
- [2] DUTOIT, B. B. . A. H. *Object-Oriented Software Engineering Using UML, Patterns, and Java*. 2009.
- [3] FIELDING, R. T. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, PhD Dissertation, University of California, Irvine, 2000. Online, Přístup 17.11.2018.
- [4] HESAM, MASNE, S., HARRY, AND RAJAN. REST API Tutorial. <https://restfulapi.net/>. Online, Přístup 17.11.2018.
- [5] SATRAPA, P. *LaTeX pro pragmatiky*. 2011.

# Příloha A

## Seznam zdrojů

- Obrázek 3.1 – Vytvořeno autorem 15. 4. 2018.
- Obrázek 4.1 – Vytvořeno autorem 15. 4. 2018.
- Obrázek 7.1 – Vytvořeno autorem 15. 4. 2018.
- Obrázek 7.2 – Vytvořeno autorem 12. 3. 2018.
- Obrázek 8.1 – Vytvořeno autorem 12. 3. 2018.
- Obrázek 8.2 – Vytvořeno autorem 12. 3. 2018.
- Obrázek 10.1 – Vytvořeno autorem 17. 5. 2018.
- Obrázek 10.2 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 7.1 – Vytvořeno autorem 26. 2. 2018.
- Tabulka 7.2 – Vytvořeno autorem 26. 2. 2018.
- Tabulka 9.1 – Vytvořeno autorem 8. 5. 2018.
- Tabulka 9.2 – Vytvořeno autorem 8. 5. 2018.
- Tabulka 9.3 – Vytvořeno autorem 8. 5. 2018.
- Tabulka 10.1 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 10.2 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 10.3 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 10.4 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 10.5 – Vytvořeno autorem 17. 5. 2018.
- Tabulka 10.6 – Vytvořeno autorem 18. 5. 2018.
- Tabulka 10.7 – Vytvořeno autorem 18. 5. 2018.
- Tabulka 11.1 – Vytvořeno autorem 18. 5. 2018.





## **Příloha B**

### **Seznam zkratek**

**LMS** – Learning Management System

**SWOT** – Strengths, Weaknesses, Opportunities, Threats

**GDPR** – General Data Protection Regulation

**HTTP** – Hypertext Transfer Protocol

**API** – Aplikační rozhraní

**CW** – CourseWare

**JPA** – Java Persistence API

**JDBC** – Java Database Connectivity



## **Příloha C**

### **Testování**