

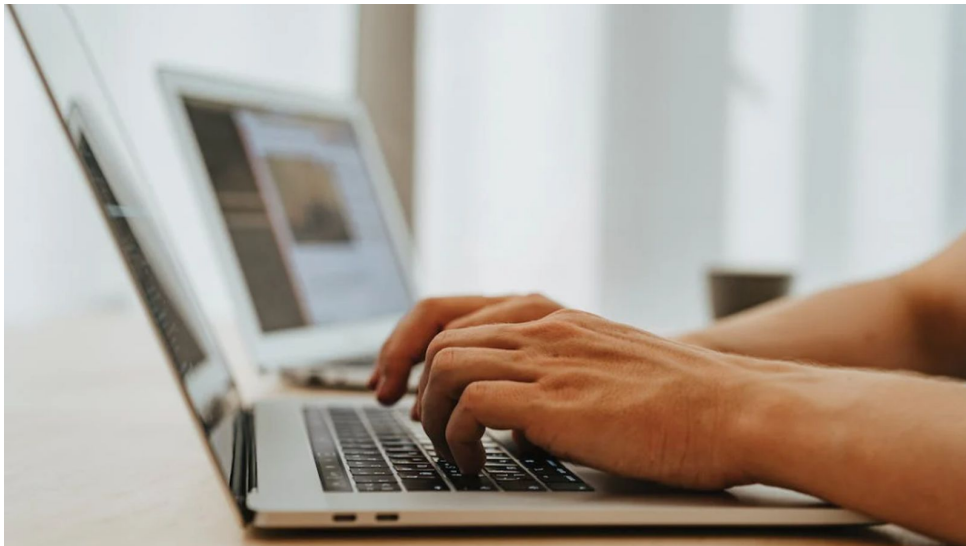


STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

System pro sdílení nabíječek elektroaut s využitím OCPP



Autor: Filip Jedlička
Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování
Třída: IT4
Školní rok: 2025/26

Poděkování

Prostor k poděkování (například vedoucímu práce).

Prohlášení

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým a prezentačním účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 1. 1. 2026

.....
Podpis autora

Abstrakt

Tato práce se zabývá návrhem a realizací systému pro sdílení domácích nabíječek elektromobilů, který uživatelům umožňuje zpřístupnit svou nabíječku ostatním podobně jako model Airbnb. V Česku sice není infrastruktura nabíjecích stanic kriticky nedostatečná, ale komunitní projekt tohoto typu může zvýšit jejich dostupnost, zlevnit nabíjení a urychlit návratnost domácích fotovoltaických elektráren. Existuje komunita uživatelů, kteří si nabíječky půjčují neformálně, avšak doposud se domlouvají složitě přes chatovací platformy.

Cílem projektu je vytvořit funkční prototyp platformy, která dokáže komunikovat s domácími nabíječkami přes protokol OCPP 1.6, zpracovat RFID autorizaci a umožnit automatické účtování plateb za nabíjení. Implementace zahrnuje databázový model v PostgreSQL a robustní API rozhraní pro správu nabíjecích stanic. Řešení je založeno na dvou backendových službách – OCPP serveru v Node.js a API serveru ve FastAPI. Systém je navržen jako backendové řešení (API-first), které připravuje veškerá data pro klientské aplikace.

Výsledkem je funkční prototyp systému umožňující registraci uživatelů i nabíječek, zpracování RFID přístupů a výpočet cen nabíjení. Projekt zároveň vytváří prostor pro budoucí rozšíření o klientskou webovou aplikaci, rezervace a platební brány.

Klíčová slova

OCPP 1.6J, nabíjecí stanice, elektromobilita, sdílení nabíječek, RFID autorizace, FastAPI, Node.js, WebSocket server, PostgreSQL.

Abstract

This thesis focuses on the design and implementation of a system for sharing home electric vehicle chargers, allowing users to make their chargers available to others in a manner similar to the Airbnb model. Although the charging infrastructure in the Czech Republic is not critically insufficient, a community-driven project of this type can increase charger availability, reduce charging costs, and accelerate the return on investment for home photovoltaic systems. A community of users who share chargers already exists, but they currently rely on informal and inefficient coordination through chat platforms.

The goal of the project is to create a functioning prototype of a platform capable of communicating with home chargers via the OCPP 1.6 protocol, processing RFID authorization, and automatically calculating charging payments. The implementation includes a PostgreSQL database model and a robust API interface for charging station management. The solution is built on two backend services — an OCPP server developed in Node.js and an API server using FastAPI. The system is designed as a backend-first solution, providing data for future client applications.

The result is a functional prototype enabling user and charger registration, RFID processing, and charging price calculation. The project additionally lays the groundwork for future extensions, including a client web application, reservations, and payment gateways.

Keywords

OCPP 1.6J, charging station, electromobility, charger sharing, RFID authorization, FastAPI, Node.js, WebSocket server, PostgreSQL.

Obsah

Úvod	3
1 Výběr postupů a technologií	5
1.1 OCPP protokol	5
1.2 OCPP Backend	5
1.3 API Backend	5
1.4 Databázové systémy	6
1.5 Analýza komunikace a testovací konzole	6
2 Tipy k psaní	7
2.1 Základy	7
2.2 Pokročilejší tipy	11
3 Když dokončuji práci	13
A Spot diagramy a další	19

ÚVOD

V posledních letech zaznamenává Česká republika, i díky dotačním titulům Evropské unie, výrazný nárůst počtu domácích fotovoltaických elektráren. Tento trend, společně s rostoucí dostupností elektromobilů a tlakem na snižování provozních nákladů, vedl k masivnímu rozšíření privátní nabíjecí infrastruktury. Tyto domácí nabíjecí body (wallboxy) však zůstávají po většinu času nevyužité, jelikož slouží primárně pouze pro potřeby majitele nemovitosti.

Sdílení těchto soukromých nabíječek by přineslo benefity všem stranám: majitelům by zkrátilo návratnost investice do fotovoltaiky a wallboxu, a zároveň by došlo k efektivnímu zahuštění sítě nabíjecích stanic, která je v tuzemsku stále nedostatečná.

V současnosti je trh se sdílením privátních nabíječek roztržštěný. Existují neformální komunity na sociálních sítích, kde je domluva zdoluhavá a manuální, nebo komerční projekty (např. evmapa.cz či goplugable.com), které jsou však často administrativně náročné, vyžadují specifický hardware nebo cílí primárně na firemní klientelu.

Cílem této práce je navrhnout a implementovat otevřený systém, který propojí existující domácí nabíječky a zajistí jejich automatizovanou správu. Systém bude zajišťovat komunikaci se stanicí, autorizaci uživatelů pomocí RFID karet, řízení nabíjecího procesu a následné vyúčtování spotřebované energie. Důraz je kladen na to, aby řešení nevyžadovalo od majitelů nákup proprietárního hardware, ale využívalo standardizovaný protokol.

1 VÝBĚR POSTUPŮ A TECHNOLOGIÍ

Při návrhu architektury systému byl kladen důraz na modularitu, škálovatelnost a použití moderních, ale ověřených technologií. Systém je navržen jako sada mikroslužeb běžících v kontejnerizovaném prostředí Docker.

1.1 OCPP PROTOKOL

Jako klíčový prvek komunikace mezi serverem a hardwarem nabíjecí stanice byl zvolen protokol **OCPP (Open Charge Point Protocol)** ve verzi **1.6 JSON (OCPP-J)**. Ačkoliv již existují novější specifikace (2.0.1, 2.1), verze 1.6 je v současnosti průmyslovým standardem s nejširší podporou výrobců domácích wallboxů. Použití formátu JSON namísto staršího SOAP (XML) výrazně zjednodušuje parsování zpráv a snižuje datovou náročnost, což je klíčové při komunikaci přes mobilní sítě. Pokud by měl projekt v budoucnu pokračovat, je architektura připravena na implementaci novějších verzí pro zajištění maximální kompatibility.

1.2 OCPP BACKEND

Pro službu zajišťující přímou komunikaci s nabíječkami (CSMS - Charging Station Management System) bylo zvoleno prostředí **Node.js**. Hlavním důvodem je jeho asynchronní architektura řízená událostmi (event-driven), která je ideální pro zpracování velkého množství WebSocket spojení v reálném čase. Node.js efektivně funguje jako router, který přijímá zprávy od nabíječek, validuje je a předává relevantní data do API backendu.

1.3 API BACKEND

Druhou částí systému je API backend, pro který byl zvolen jazyk **Python** s frameworkem **FastAPI**. Tato služba tvoří „mozek“ celého systému. Umožňuje OCPP backendu (a v budoucnu i dalším klientům, jako jsou webové či mobilní aplikace) komunikovat s databází a nabízí sadu nástrojů pro správu uživatelů a nabíječek. FastAPI bylo zvoleno pro svou rychlost, automatickou generaci dokumentace (Swagger UI) a striktní typovou kontrolu dat.

1.4 DATABÁZOVÉ SYSTÉMY

Architektura využívá kombinaci dvou databázových systémů pro různé typy dat:

- **PostgreSQL:** Slouží jako primární perzistentní úložiště pro trvalá data (uživatelé, transakce, definice nabíječek).
- **Redis:** Slouží jako in-memory úložiště pro dočasná a často měněná data. Ukládají se zde například aktuální stavy konektorů nebo poslední autorizované RFID tagy pro rychlý přístup.

1.5 ANALÝZA KOMUNIKACE A TESTOVACÍ KONZOLE

Před samotnou implementací serveru bylo nutné detailně pochopit tok zpráv v protokolu OCPP. Za tímto účelem byla v rané fázi projektu vytvořena **interaktivní testovací konzole** (CLI aplikace). Tento nástroj umožňoval simulovat serverovou stranu a manuálně odpovídat na zprávy přicházející z reálné nebo simulované nabíječky. Díky tomuto experimentálnímu přístupu bylo možné přesně zmapovat sekvence zpráv pro procesy jako `BootNotification` či `StartTransaction` a ověřit chování nabíječky při různých chybových stavech ještě před napsáním finálního kódu.

2 TIPY K PSANÍ

Jak už jsem psal výše \LaTeX je dosti komplexní systém, který umožňuje psát velmi rozsáhlé text. Jeho autor Donald Knuth ho stvořil, aby mohl vydat jeho učebnici *The Art of Computer Programming* a dodnes se je využíván pro sazbu skript, učebnic, článků či závěrečných prací. V této kapitole najdeš ukázky různých funkcí a balíčků \LaTeX u od těch nejzákladnějších až po složitější. Neznamená to nutně, že všechny musíš použít, ale když potřebuješ pomoci, tak je dobré mít oporu.

Pokud s \LaTeX em úplně začínáš tak ti můžu doporučit příručku *Ne příliš stručný úvod do systému $\text{\LaTeX}2\epsilon$* [2]. Případně spoustu užitečných informací najdeš na Wikibooks [3]. Pokud narazíš na nějaký problém googli. Na internetu je spousta fór, kde pravděpodobně už někdo podobný problém řešil. Asi nejvíce otho najdeš na stránce *TeX - LaTeX Stackexchange* [4].

2.1 ZÁKLADY: TEXT, OBRÁZKY, TABULKY A CITACE

Psaní v \LaTeX u není žádná věda, stačí psát normálně do zdrojového souboru. Pokud bys chtěl psát obrázky či číslovaný seznam, pak můžeš použít prostředí `itemize` či `enumerate`. Často je důležité používat nezlomitelnou mezeru. Tu uděláš pomocí `~` (tildy). Pokud budeš chtít psát uvozovky použij příkaz `uv`, pomocí něj se ti vytvoří uvozovky podle příslušného jazyka. V česku tedy ve formátu 99 66. Použití příkazu najdeš níže v textu.

Občas je zapotřebí \LaTeX u pomoci při rozdělování slov. To se udělá snadno vložením symbolů `\-` mezi jednotlivé slabiky.

2.1.1 Tabulky

U tabulek platí to stejné co u obrázků. Zarovnávají se na střed a nechávají se „plavat“ v textu. Tabulka narozdíl od textu, má popisek nahoře. U tabulky 2.1 je použit balíček `booktabs`, pomocí kterého je celá tabulka naformátovaná.

Seznam jak obrázků tak tabulek je pak vytvořen pomocí příkazů `listoftables` a `listoffigures` na konci práce před literaturou.

Tabulka 2.1: Tato tabulka slouží jako ukázka toho, jak mohou tabulky vypadat.

záhlaví	této	tabulky
obsah	tabulky	už
není	oddělený	čarami

2.1.2 Obrázky

U obrázků je dobré používat vektorové formáty, pokud to jde. \LaTeX se nejvíc kamarádí s formátem PDF. Do známého PDFka lze z jiných vektorových formátů (ať už SVG či EPS) obrázky přenést snadno pomocí grafických programů, jako je třeba Inkscape. \LaTeX si rozhodně poradí i s tradičními formáty PNG a JPG, avšak tyto obrázky mohou zabírat více prostoru a při tisku se může projevit nižší rozlišení obrázků. Pokud chceš používat tyto obrázky, rozhodně měj na paměti, aby měli rozlišení alespoň 250 indálně 330 ppi.

Obrázky se vkládají do prostředí `figure`, při úpravě šířky je možné krom tradičních jednotek jako cm nebo mm použít také jako jednotku šířku stránky `textwidth` to se hodí zejména když chceš mít více podobrázků.

U každého obrázku je důležité aby měl popisek, `caption`. Do popisku napiš, co na obrázku je, případně nějaký další popis, tak aby čtenář následně neměl sebemenší pochybnost. U obrázků co nejsou tvoje nezapomeň ani citaci. Jinak by to totiž znamenalo, že jsi obrázek dělal ty sám, což není etické přivlastňovat si cizí díla. Popisek obrázku je věta, proto musí vždy končit tečkou.

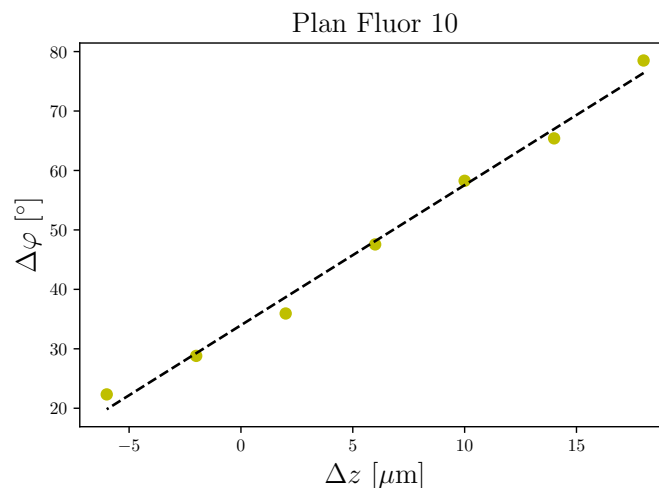


Obrázek 2.1: Logo SŠPU Opava [5].

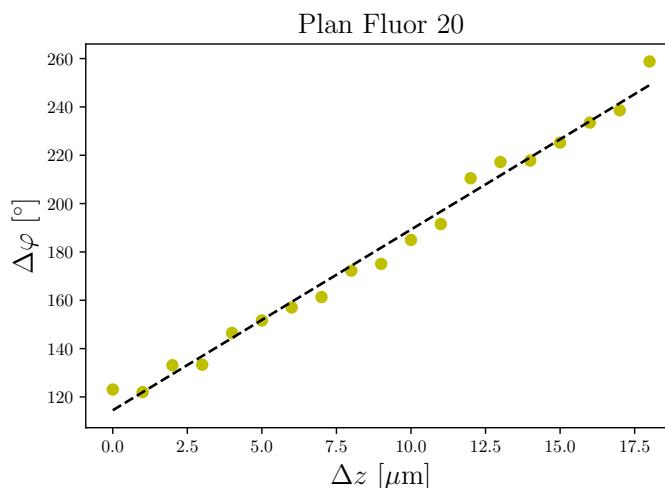
Když chceš odkazovat na obrázek, stačí pak už jen napsat příkaz `ref` a do závorek napsat označení obrázku. Třeba logo SOČky, můžeš vidět na obrázku 2.1 [?].

Pokud bys měl více podobrázků přichází do hry balíček `subcaption`. Pomocí něj lze vysázet i podobrázky. U podobrázků se popisek píše pouze jeden, dolů. Je v tomto případě vhodné použít navíc hranaté závorky, do nichž se napíše kratší popisek, který se následně ukáže v seznamu obrázků.

Všimni si, že obrázky jsou naschvál široké. Je to proto, aby byly dobře čitelné. Také si



(a)



(b)

Obrázek 2.2: Graf závislosti rotace DH PSF $\Delta\varphi$ na defokusaci objektivu Δz , (a) při použití objektivu Plan Fluor 10, (b) při použití objektivu Plan Fluor 20. Měřená data (žluté body) jsou lineárně proloženy (přerušovaná přímka).

všimni popisku grafů. Ačkoli nejspíš netušíš co je to DH PSF či defokusace objektivu mělo by ti být jasné, že je důležité přesně graf popsat. To znamená co je na vodorovné ose, co je na svislé ose. V jakých jednotkách veličiny jsou. Které body co znamenají, která křivka má jaký význam. Napsat samotné „ $\Delta\varphi$ “ je málo, vždy raději připomeň, co daná značka znamená.

2.1.3 Literatura

V \LaTeX u lze dělat seznam literatury dvěma způsoby. V této šabloně jsem použil ten, kdy se seznam literatury píše přímo do práce. Pro jeho vygenerování doporučuji použít některý z generátorů, jako jsou například Citace PRO [6]. Pomocí citací lze vygenerovat přímo dokument,

který se pak už jen překopíruje do textu a člověk nemusí nic zvýrazňovat. Dále lze využít Bibtex, který rozhodně do budoucna hodlám zaimplementovat do šablony, avšak jeho použití nemusí být tak přátelské k začátečníkům.

Pokud bys chtěl odkazovat na vícero zdrojů stačí je napsat vedle sebe oddělené čárkou [2, 6, 7]. Případně můžu odkaz na konkrétní stránku dát do hranatých závorek, viz [7, str. 1]

2.1.4 Programový kód

Pro vložení programového kódu do dokumentu LaTeX s možností zvýraznění syntaxe můžete použít balíček `listings`. Tento balíček nabízí široké možnosti pro formátování kódu, včetně zvýraznění syntaxe pro různé programovací jazyky.

Nejprve je třeba do preamble LaTeX dokumentu přidat `usepackage{listings}` a nastavit příslušné parametry. Příklad nastavení pro jazyk Python by mohl vypadat takto:

```
1 # Python code here
2 def hello_world():
3     print("Hello, world!")
```

Kód 2.1: Ukázka Python kódu

```
1 // JavaScript code here
2 function helloWorld() {
3     console.log("Hello, world!");
4 }
```

Kód 2.2: Ukázka JS kódu

```
1 /* eslint-env es6 */
2 /* eslint-disable no-unused-vars */
3
4 import Axios from 'axios'
5 import { BASE_URL } from './utils/api'
6 import { getAPIToken } from './utils/helpers'
7
8 export default class User {
9     constructor () {
10         this.id = null
11         this.username = null
12         this.email = ''
```

```

13     this.isActive = false
14     this.lastLogin = '' // ISO 8601 formatted timestamp.
15     this.lastPWChange = '' // ISO 8601 formatted timestamp.
16   }
17 }
18
19 const getUserProfile = async (id) => {
20   let user = new User()
21   await Axios.get(
22     `${BASE_URL}/users/${id}`,
23     {
24       headers: {
25         'Authorization': `Token ${getAPIToken()}`,
26       }
27     }
28   ).then(response => {
29     // ...
30   }).catch(error => {
31     // ...
32   })
33 }

```

Kód 2.3: ES6 (ECMAScript-2015) Listing

2.2 POKROČILEJŠÍ TIPY, KTERÉ SE MOHOU HODIT

2.2.1 Rovnice

Sazba matematiky je věda sama o sobě. Ačkoli Word prošel obrovskou změnou a je v tomto mnohem lepší, tak $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ je pro to přímo (ještě jsem neviděl matematika, co by používal Word). Spolu s balíčky `amsmath` a `amsfonts` snad neexistuje nic, co by se používalo a $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ by to nezvládl. Ať už jde o základní věci jako řecká písmenka – $\alpha, \beta, \gamma, \dots$ – integrály – $\int_{l_i}^{l_f} \tau dl$ – až třeba po speciální písmena – $\mathcal{F} : \mathbb{R}^n \rightarrow \mathbb{R}^m$. Pro případ, že bys potřeboval nějaké speciální integrály, je tu balíček `esint`, pomocí něj můžeš napsat třeba

$$\oint_{S(V)} \vec{E} \cdot d\vec{S} = \iiint_V (\vec{\nabla} \cdot \vec{E}) dV.$$

Jak můžeš vidět tak rovnice lze psát jednak do textu a nebo pokud se jedná o nějakou důležitou nebo rozsáhlejší rovnici tak na samostatný řádek. Pokud je rovnice opravdu důležitá,

tak je vhodné ji také číslovat. Pak se na ni můžeš dále odkazovat v textu.

$$\vec{F} = m\vec{a} \quad (2.1)$$

... Například podle druhého Newtonova zákona, rovnice (2.1) ... Zároveň je vždy nutné vysvětlit co která veličina znamená. V tomto případě bych napsal, že v druhém Newtonově zákoně vektor síly \vec{F} odpovídá součinu hmotnosti tělesa m a jeho zrychlení \vec{a} .

Věřím, že se sazbou matematiky ti pomůže tvůj školitel, případně mi můžeš napsat (mail je v úvodu). Jednotlivé funkcionality spolu se seznamem znaků nalezneš jednak v Ne příliš stručném úvodu [2] nebo na Wikibooks v sekcích *Mathematics* a *Advanced mathematics* [3].

3 KDYŽ DOKONČUJI PRÁCI

Každou práci je dobré zkontrolovat, aby v ní nebyly pravopisné chyby, nebyla těžkopádně napsaná – byla čtivá – a neobsahovala žádný typografický nedostatek. Proto, když práci sepíšeš, nech ji chvíli odležet, třeba týden. Pak si ji po sobě znovu přečti. Hned uvidíš, kolik věcí bys napsal jinak případně kde tě bije do očí jaká chyba. Dej práci přechíst také svému školiteli a případně češtináři. Zajistíš tak, že bude obsahovat méně chyb.

Pak můžeš práci vytisknout a hurá do soutěže.

ZÁVĚR

Věřím, že jsem ti spolu se šablonou poskytl několik tipů, jak napsat práci. Ať už jde o úplné začátky s \LaTeX em. Či ukázkou toho, co vše s ním zvládneš. Pokud bys měl k šabloně libovolné dotazy, rouhodně se na mě obrať. \LaTeX ové práci dodá určitou krásu, tak doufám, že ti dodá sebevědomí a uspěješ při soutěži. A i kdyby ne vzpomeň si, kolik ses toho musel naučit a hned uvidíš o jaký kus ses posunul.

LITERATURA

- [1] DOKULIL Jakub. *Šablona pro psaní SOČ v programu L^AT_EX* [Online]. Brno, 2020 [cit. 2020-08-24]. Dostupné z: https://github.com/Kubiczek36/SOC_sablona
- [2] OETIKER, Tobias, Hubert PARTL, Irene HYNA, Elisabeth SCHEGL, Michal KOČER a Pavel SÝKORA. *Ne příliš stručný úvod do systému L^AT_EX2_ε* [online]. 1998 [cit. 2020-08-24]. Dostupné z: <https://www.jaroska.cz/elearning/informatika/typografie/lshort2e-cz.pdf>
- [3] *Wikibooks: L^AT_EX* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2020-08-24]. Dostupné z: <https://en.wikibooks.org/wiki/LaTeX>
- [4] *TeX - L^AT_EX Stack Exchange* [online]. Stack Exchange, 2020 [cit. 2020-09-01]. Dostupné z: <https://tex.stackexchange.com>
- [5] *Střední škola průmyslová a umělecká Opava* [online]. [cit. 2023-11-11]. Dostupné z: <https://www.sspu-opava.cz>
- [6] *Citace PRO* [online]. Citace.com, 2020 [cit. 2020-08-31]. Dostupné z: <https://www.citacepro.com>
- [7] BORN, Max a Emil WOLF. *Principles of optics: electromagnetic theory of propagation, interference and diffraction of light*. 7th (expanded) edition. Reprinted with corrections 2002. 15th printing 2019. Cambridge: Cambridge University Press, 2019. ISBN 978-0-521-64222-4.

Seznam obrázků

2.1	Logo SŠPU Opava [5].	8
2.2	Graf závislosti rotace DH PSF $\Delta\phi$ na defokusaci objektivu Δz	9

Seznam tabulek

2.1	Tato tabulka slouží jako ukázka toho, jak mohou tabulky vypadat.	8
-----	--	---

PŘÍLOHA A SPOT DIAGRAMY A DALŠÍ