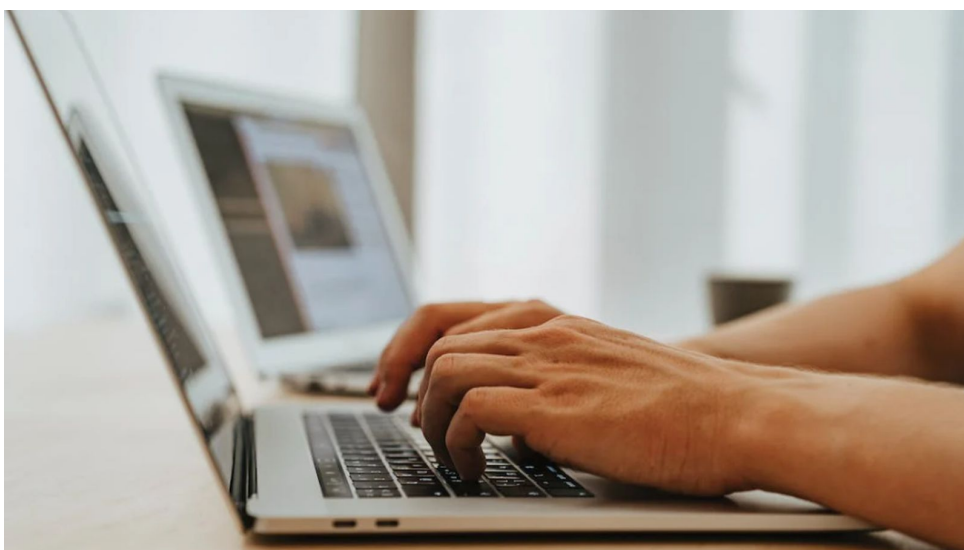


ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Django Social Auth



Autor: Petr Mičola
Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování
Třída: IT4
Školní rok: 2023/24

Poděkování

Na úvod bych chtěl poděkovat panu učiteli Mgr. Marku Lučnému za ochotu a podporu při tvorbě tohoto projektu.

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. 2024

.....
Podpis autora

Abstrakt

Tato práce se zaměřuje na vytvoření webové aplikace s autentizací uživatelů pomocí webového frameworku Django. Hlavním cílem projektu je implementace autentizace uživatelů prostřednictvím účtů GitHub a Microsoft, spolu s možností úpravy uživatelského profilu. Pro dosažení znovupoužitelnosti byl vyvinut backend, který může být integrován do jiných aplikací. Využití technologií jako Django, django-allauth, PostgreSQL a Docker zajišťuje robustnost a efektivitu aplikace. Gunicorn a Nginx byly zvoleny k obsluze webových požadavků, což přispívá k vysoké úrovni výkonu a škálovatelnosti. Soft UI Dashboard poskytuje moderní a přehledné prostředí administrace. Celková funkčnost aplikace je demonstrována na jednoduchém příkladu užití. Díky těmto implementacím má práce potenciál sloužit jako výchozí rámec pro vytváření dalších webových aplikací s podobnými požadavky na autentizaci uživatelů.

Klíčová slova

Django, webová aplikace, autentizace uživatelů, úprava profilu, znovupoužitelný backend

Abstract

This work focuses on creating a web application with user authentication using the Django web framework. The main project objective is to implement user authentication through GitHub and Microsoft accounts, along with the capability to edit user profiles. To achieve reusability, a backend has been developed, which can be integrated into other applications. Utilizing technologies such as Django, django-allauth, PostgreSQL, and Docker ensures the robustness and efficiency of the application. Gunicorn and Nginx have been chosen to handle web requests, contributing to a high level of performance and scalability. The Soft UI Dashboard provides a modern and clear administrative interface. The overall functionality of the application is demonstrated through a simple use case. Thanks to these implementations, the work has the potential to serve as a foundational framework for developing additional web applications with similar user authentication requirements.

Keywords

Django, web application, user authentication, profile editing, reusable backend

Obsah

Úvod	3
1 Webová aplikace	5
1.1 Úvod	5
1.2 Statické a dynamické webové stránky	5
1.3 Webový server	6
1.4 Aplikační server	6
1.5 Databáze	7
2 Framework	9
2.1 Úvod	9
2.2 Django	9
2.3 Model-Pohled-Šablona	10
2.4 Struktura projektu	12
3 Využité technologie	13
3.1 Django	13
3.2 Django-allauth	13
3.3 PostgreSQL	13
3.4 Docker	14
3.5 Gunicorn	14
3.6 Nginx	14
3.7 Soft UI Dashboard	15
4 Způsoby řešení, použité postupy	17
4.1 Vytvoření projektu	17
4.2 Autentizace Django-allauth	17
4.3 Vlastní uživatelský model	18
4.4 Citlivá data	18
4.5 Dockerizace aplikace	18
4.6 Administrace	19
4.7 Databáze PostgreSQL	20
5 Výsledky řešení, uživatelský manuál	21
6 Závěr	23
7 Seznam použitých informačních zdrojů	25
7.1 Základní struktura dokumentu	25

7.2	Práce s textem	26
7.3	Matematické vzorce a symboly	27
7.4	Práce s obrázky a tabulkami	28
7.5	Bibliografie a citace	30
8	Tipy k psaní	31
8.1	Základy	31
8.2	Pokročilejší tipy	35
9	Když dokončuji práci	37
A	Spot diagramy a další	43

ÚVOD

Mým cílem bylo vytvořit aplikační backend, který se bude starat o autentizaci uživatelů. Hlavním úkolem bylo umožnit uživatelům přihlašování skrze účty GitHub a Microsoft.

Proč jsem si vybral toto téma?

Každý student naší školy má účty GitHub a Microsoft. Proto jsem se rozhodl pro tento projekt – chtěl jsem vytvořit autentizační backend, jež můžou učitelé naší školy použít pro své vlastní aplikace.

Proč zrovna tyto technologie?

S webovým frameworkem Django jsme ve škole pracovali, což mi poskytlo solidní základy. Chtěl jsem využít svých znalostí ze studia a zároveň se něco naučit o ostatních technologiích, které řeší problematiku autentizace.

Struktura dokumentace

Tato dokumentace popisuje -

1 WEBOVÁ APLIKACE

1.1 ÚVOD

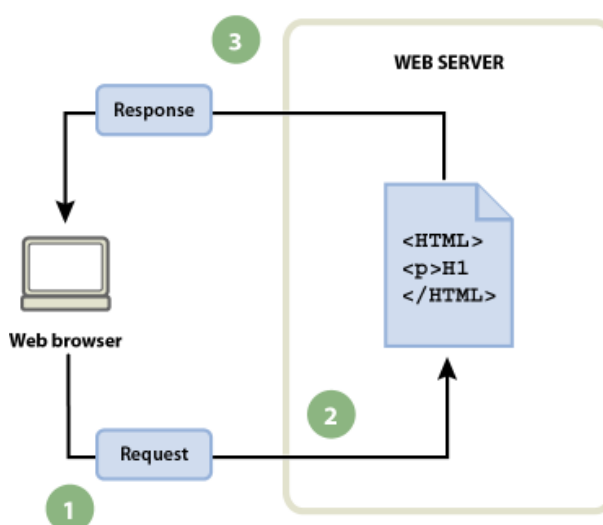
Webová aplikace je aplikace poskytovaná uživatelům z webového serveru přes počítačovou síť Internet. Uživatelé mohou přistupovat k webovým aplikacím prostřednictvím webového prohlížeče, takže nemusí instalovat žádný speciální software na svých zařízeních.

1.2 STATICKÉ A DYNAMICKÉ WEBOVÉ STRÁNKY

Webová aplikace obsahuje stránky s částečně nebo úplně neurčeným obsahem. Konečný obsah stránky se určí až tehdy, když návštěvník požádá o stránku z webového serveru. Webová aplikace je kolekcí statických a dynamických webových stránek.

1.2.1 Zpracování statických stránek

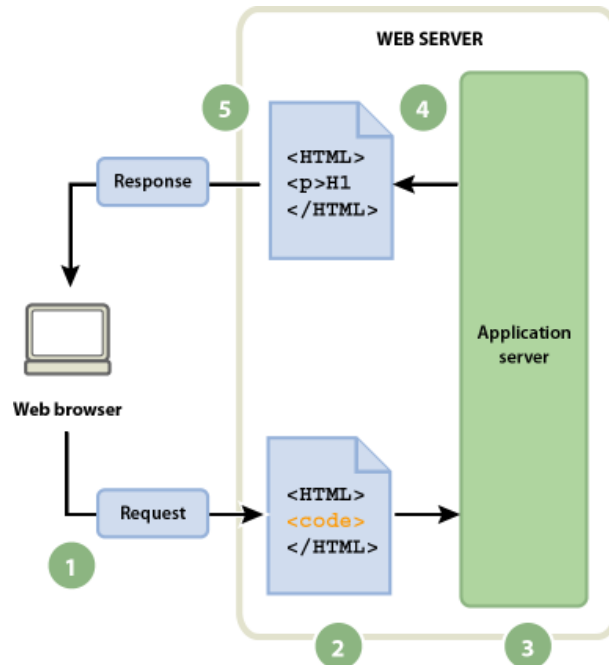
Když webový server přijme požadavek na statickou webovou stránku, pošle ji přímo prohlížeči, který o ni požádal. Statická webová stránka se nemění.



Obrázek 1.1: Zpracování statické webové stránky

1.2.2 Zpracování dynamických stránek

Naproti tomu, když webový server přijme požadavek na dynamickou stránku, předá stránku aplikačnímu serveru, který odpovídá za dokončení stránky. Aplikační server si přečte kód na stránce, dokončí stránku podle instrukcí v kódu a pak odstraní kód ze stránky.



Obrázek 1.2: Zpracování dynamické webové stránky

1.3 WEBOVÝ SERVER

Webový server je software, který posílá webové stránky na základě požadavků od webových prohlížečů. Požadavek na stránku se generuje, když návštěvník ve webovém prohlížeči klepne na odkaz na webové stránce, vybere záložku nebo zadá adresu URL do textového pole pro adresu.

1.4 APLIKAČNÍ SERVER

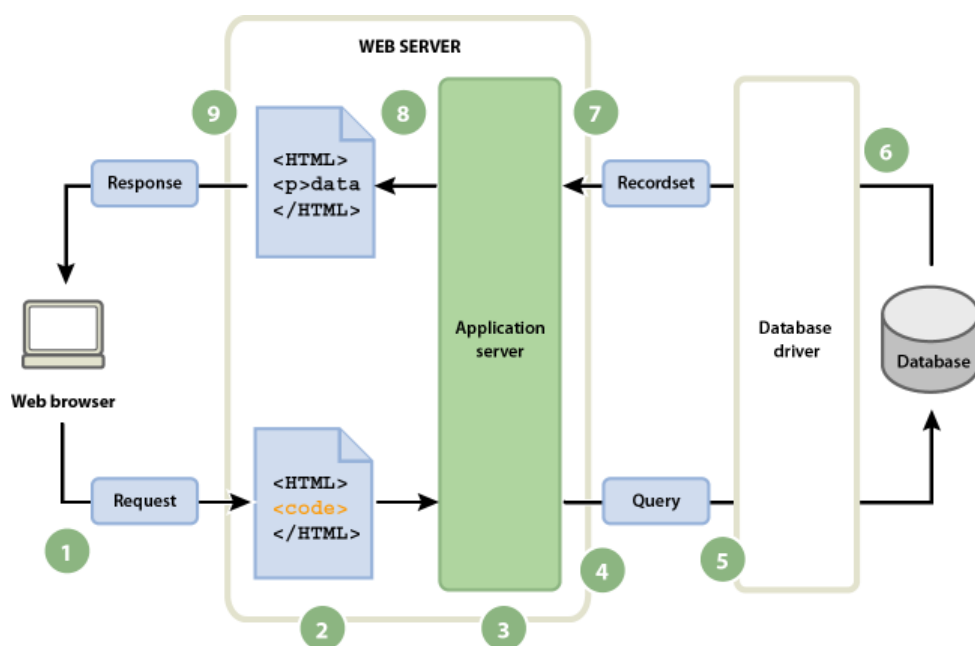
Aplikační server je software, který pomáhá webovému serveru zpracovat webové stránky obsahující skripty nebo tagy na straně serveru. Když webový server přijme požadavek na takovou stránku, předá stránku aplikačnímu serveru ke zpracování, než ji pošle prohlížeči.

1.5 DATABÁZE

Databáze je kolekce dat uložených v tabulkách. Každý řádek tabulky představuje jeden záznam a každý sloupec představuje pole v záznamu.

1.5.1 Přístupování k databázi

Aplikační server vám umožňuje pracovat s prostředky na straně serveru, jako jsou například databáze. Využitím databáze k uložení obsahu můžete návrh webu oddělit od datového obsahu, který chcete zobrazovat.



Obrázek 1.3: Přístupování k databázi

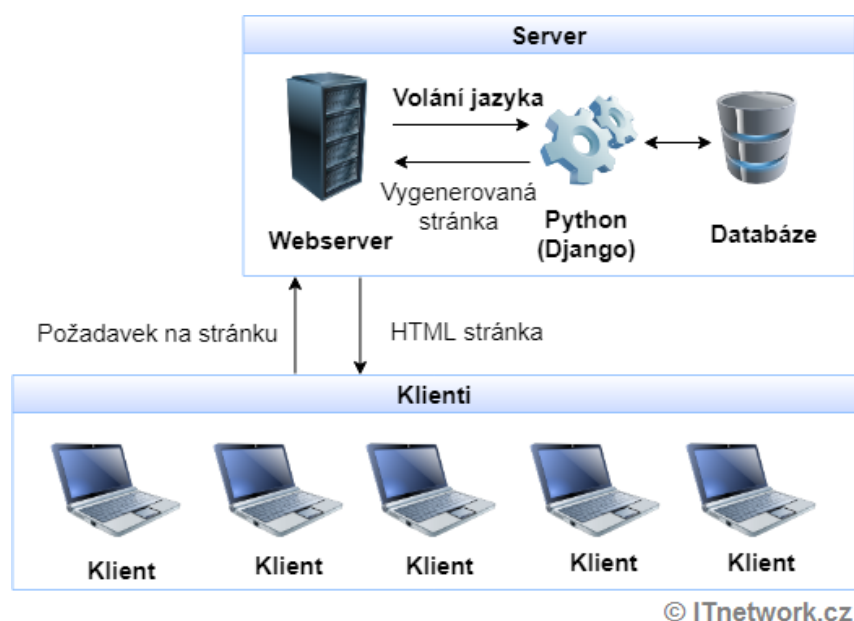
2 FRAMEWORK

2.1 ÚVOD

Framework je softwarová struktura pro podporu programování, vývoje a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovny API, podporu pro návrhové vzory nebo doporučené postupy při vývoji. Cílem frameworku je převzetí typických problémů dané oblasti, což umožní, aby se návrháři a vývojáři mohli soustředit pouze na své zadání.

2.2 DJANGO

Django IPA je open source webový aplikační framework napsaný v Pythonu, který se volně drží architektury Model-Pohled-Šablona. Hlavním úkolem Django je snadné vytvoření komplexních, databází řízených webových aplikací. Zaměřuje se na znovupoužitelnost a propojitelnost komponent, rychlý vývoj, v duchu „DRY“ (Don't Repeat Yourself) – neopakovat se.



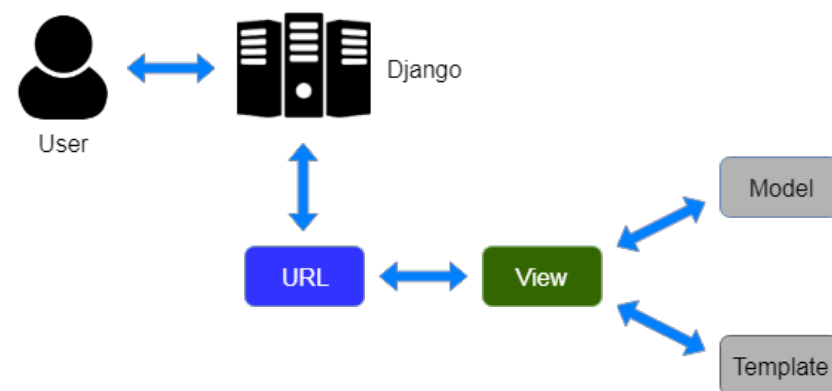
Obrázek 2.1: Diagram frameworku Django

2.2.1 Historie

Django bylo původně navrženo pro správu několika zpravodajsky orientovaných stránek společnosti The World Company v Lawrenci v Kansasu; později, v červnu 2005, bylo vydáno veřejně pod open-sourceovou licencí BSD. Framework byl pojmenován po jazzovém kytaristovi Django Reinhardtovi.

2.3 MODEL-POHLED-ŠABLONA

Model-View-Controller a je architektonický vzor, který je běžně používán při vývoji webových aplikací. V případě frameworku Django se tato architektura označuje jako Model-Pohled-Šablona (Model-View-Template).



Obrázek 2.2: Architektura Model-Pohled-Šablona

2.3.1 Model

Model reprezentuje datovou strukturu aplikace. Může to být například databázová tabulka, kde jsou ukládána data. Ve frameworku Django se modely definují jako třídy, které dědí od předdefinovaných modelových tříd poskytovaných frameworkem.

```
1 # models.py
2 from django.contrib.auth.models import AbstractUser
3 from django.db import models
4 class CustomUser(AbstractUser):
5     email = models.EmailField(unique=True)
6     username = models.CharField(unique=True, max_length=30)
7     profile_picture = models.ImageField(upload_to='profile_pictures/')
```

Kód 2.1: Příklad modelu Uživatel

2.3.2 Pohled

Pohled je část, která zpracovává požadavky od uživatelů a reaguje na ně. Obsahuje logiku pro získání dat z modelu a přípravu dat pro zobrazení. V Django se pohledy jsou implementovány jako funkce nebo třídy.

```
1  # views.py
2  from django.shortcuts import render
3  def quiz(request):
4      user = request.user
5      return render(request, 'quiz/quiz.html', {'user': user})
```

Kód 2.2: Příklad pohledu Kvíz

2.3.3 Šablona

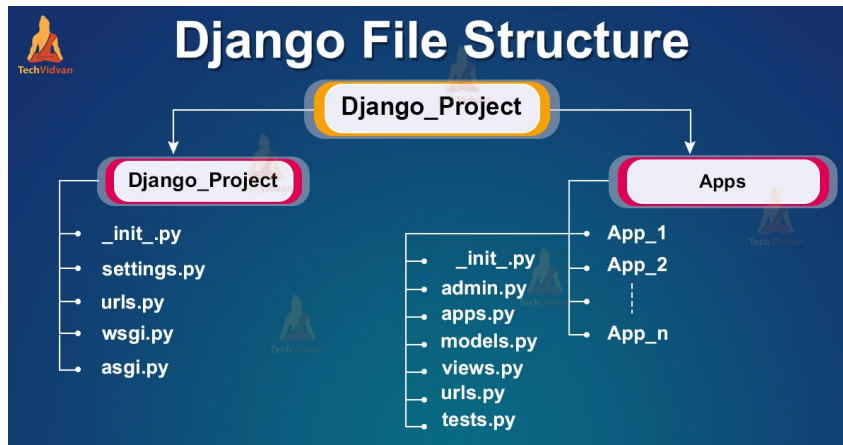
Šablona definuje, jak jsou data zobrazena uživateli. Jedná se o prezentaci, která může obsahovat HTML, CSS a speciální značky nebo proměnné, které jsou nahrazeny konkrétními daty během zpracování. Ve frameworku Django jsou šablony soubory s příponou .html, které oddělují prezentaci od logiky pohledu.

```
1  # base.html
2  <!DOCTYPE html>
3  <html>
4  <head>
5      <meta http-equiv="X-UA-Compatible" content="IE=edge">
6      <meta name="viewport" content="width=device-width, initial-scale=1.0">
7      <title>DSA - {% block head_title %}{% endblock head_title %}</title>
8      <link href='{% static "style.css" %}' rel='stylesheet'>
9  </head>
10  {% include 'components/navbar.html' %}
11  <body>
12  {% block content %}
13  {% endblock content %}
14  </body>
15  {% include 'components/footer.html' %}
16  </html>
```

Kód 2.3: Příklad základní šablony

2.4 STRUKTURA PROJEKTU

Projekty v Django mají specifickou strukturu, která pomáhá organizovat kód, šablony, statické soubory a další komponenty. Po instalaci Django se nový projekt vytvoří příkazem `django-admin startproject (název projektu)`.



Obrázek 2.3: Struktura projektu Django

2.4.1 Soubory projektu

Adresář projektu po první konfiguraci bude obsahovat tyto soubory:

- `manage.py` – Script, který má na starosti správu projektu.
- `settings.py` – Konfigurační script společný všem aplikacím v projektu.
- `urls.py` – Globální konfigurace URL.

2.4.2 Soubory aplikace

Vytvoření struktury nové aplikace proběhne po spuštění příkazu `python manage.py startapp (název aplikace)`. Po spuštění tohoto scriptu se vytvoří podadresář se strukturou:

- `views.py` – Obsahuje jednotlivé view funkce.
- `urls.py` – Obsahuje mapování URL na jednotlivá view.
- `models.py` – Obsahuje popis datového modelu aplikace.
- `tests.py` – Obsahuje jednotkové testy.

3 VYUŽITÉ TECHNOLOGIE

3.1 DJANGO

Django IPA je open source webový aplikační framework napsaný v Pythonu, který se volně drží architektury Model-Pohled-Šablona.



Obrázek 3.1: Django

3.2 DJANGO-ALLAUTH

Django-allauth je integrovaná sada aplikací Django řešící autentizaci, registraci a správu účtů třetích stran.

3.3 POSTGRESQL

PostgreSQL je objektově-relační databázový systém. Na jeho vývoji se podílí globální komunita vývojářů a firem.



Obrázek 3.2: PostgreSQL

3.4 DOCKER

Docker je open source software, jehož cílem je poskytnout jednotné rozhraní pro izolaci aplikací do kontejnerů.



Obrázek 3.3: Docker

3.5 GUNICORN

Gunicorn je open-source WSGI server napsaný v Pythonu, používaný pro spouštění webových aplikací. Jeho hlavním cílem je poskytovat efektivní a spolehlivé zpracování HTTP požadavků.



Obrázek 3.4: Gunicorn

3.6 NGINX

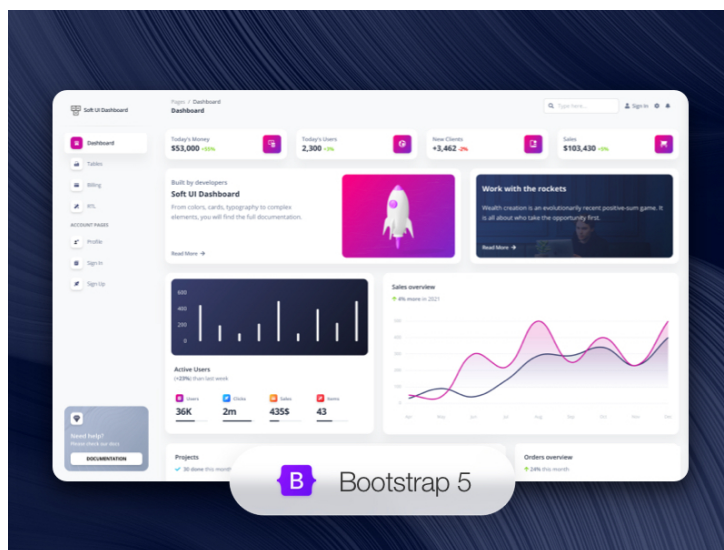
Nginx je softwarový open source webový server. Pracuje s protokoly HTTP (HTTPS), SMTP, POP3, IMAP a SSL. Zaměřuje se především na vysoký výkon a nízké nároky na paměť.



Obrázek 3.5: Nginx

3.7 SOFT UI DASHBOARD

Soft UI Dashboard je šablona administrace vytvořená pomocí Bootstrap 5.



Obrázek 3.6: Soft UI Dashboard

4 ZPŮSOBY ŘEŠENÍ, POUŽITÉ POSTUPY

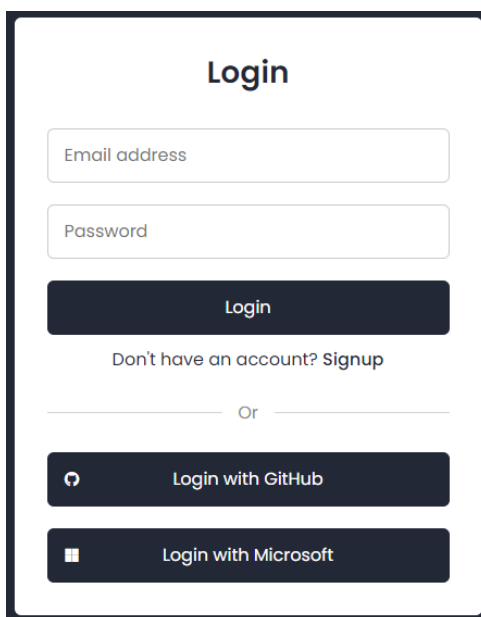
Při hledání vhodné technologie pro tuto aplikaci jsem narazil na Django. Tento webový aplikační framework je v problematice autentizace často využíván. S Djangem jsem navíc pracoval ve škole, tak jsem se rozhodl mé znalosti využít a Django použít.

4.1 VYTVOŘENÍ PROJEKTU

Prvním krokem bylo vytvoření projektu. Stačilo nainstalovat Python a framework Django. Příkazem se poté vytvořila základní struktura.

4.2 AUTENTIZACE DJANGO-ALLAUTH

Dále bylo mým úkolem umožnit uživatelům registraci a přihlášení pomocí aplikací třetích stran (GitHub a Microsoft). Pro tuto funkci jsem zvolil balíček django-allauth, který proces vytváření autentizace značně urychlil a autentizaci zabezpečil.



The image shows a login form with the following elements:

- Title:** Login
- Input Fields:** Email address, Password
- Buttons:** Login
- Text:** Don't have an account? Signup
- Separator:** Or
- Social Login Buttons:** Login with GitHub, Login with Microsoft

Obrázek 4.1: Formulář přihlášení

4.2.1 Aplikace třetích stran

Balíček django-allauth bylo potřeba nainstalovat. Poté jsem musel udělat pár změn v settings.py. Pro funkčnost přihlášení aplikací třetích stran jsem musel tyto aplikace přidat do struktury SOCIALACCOUNT_PROVIDERS. K tomu jsem potřeboval klíče těchto poskytovatelů, ty jsem získal zaregistrováním a nastavením aplikací OAuth.

4.2.2 Kontrola funkčnosti

Pro kontrolu funkčnosti jsem django-allauth musel přidat do cest urls.py a vytvořit jednoduchý pohled ve views.py, který vracel šablonu základní stránky.

4.3 VLASTNÍ UŽIVATELSKÝ MODEL

Pro využití vlastního modelu uživatele jsem v models.py vytvořil model User, který vycházel z Django AbstractUser. Tomuto modelu jsem přiřadil email, uživatelské jméno a profilové foto, vše s patřičnými parametry. Model jsem nakonec musel zaregistrovat v admin.py aplikace a nastavit v settings.py.

4.4 CITLIVÁ DATA

Dalším úkolem bylo aplikaci zdockerizovat. Před tím jsem se však musel postarat o citlivá data (klíče aplikací třetích stran a Django samotného). Tyto data jsem se ze souborů rozhodl skrýt a všechna umístit do jednoho souboru .env.

4.4.1 Python-decouple

Abych poté mohl k datům přistupovat nainstaloval jsem balíček python-decouple. Následně stačilo z balíčku importovat object config a citlivé proměnné nahradit config("název proměnné v souboru .env").

4.5 DOCKERIZACE APLIKACE




Dockerizace je proces přizpůsobení a zapouzdření aplikace do kontejneru pomocí technologie Docker. Kontejner je samostatný a izolovaný balíček, který obsahuje veškeré potřebné soubory aplikace.

4.5.1 Docker Desktop

Jako první jsem si nainstaloval program Docker Desktop, který umožňuje práci s kontejnery v grafickém prostředí.

4.5.2 Docker Compose

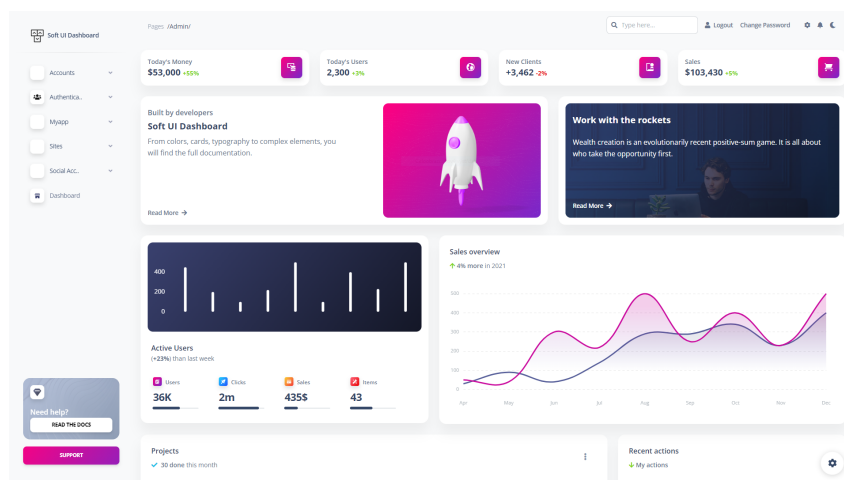
Dále jsem také využil nástroj Docker Compose, díky němuž jsem mohl definovat a spouštět více kontejnerů současně jako součást jedné aplikace. V mém případě to byl jeden kontejner s databází a jeden kontejner se samotnou aplikací.

Name	Image
 zaverecna-prace-main	
 web-1 6390180fef30	zaverecna-prace-main
 db-1 163b2e2a3373	postgres:15

Obrázek 4.2: Docker kontejnery

4.6 ADMINISTRACE

Pro administraci uživatelů jsem se rozhodl použít původní administraci Django a upravit si její vzhled podle mých preferencí.



Obrázek 4.3: Administrace na adrese /admin

4.6.1 Šablona Soft UI Dashboard

K úpravě administrace jsem vybral šablonu vytvořenou pomocí Bootstrap 5 Soft UI Dashboard. Tuto technologii lze získat dvěma způsoby - stáhnutím souborů z GitHubu nebo nainstalováním balíčku `django-admin-soft-dashboard`. Rozhodl jsem se využít druhou možnost a balíček nainstaloval.

4.7 DATABÁZE POSTGRESQL

Django v základním nastavení používá databázi SQLite, ta ale pro produkční účely není vhodná. Databázi SQLite jsem proto vyměnil za velice populární databázi PostgreSQL, která má dobrou podporu Djangem. Databázi jsem vytvořil a propojil s aplikací.

4.7.1 Docker entrypoint

Vytvoření souboru `entrypoint.sh`, který využívá Docker entrypoint přišlo vhod, protože tento soubor zkontroluje funkčnost databáze předtím, než se aplikace nainstaluje.

5 VÝSLEDKY ŘEŠENÍ, UŽIVATELSKÝ MANUÁL

6 ZÁVĚR

7 SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

TEMPLATE

- Vynikající kvalitu sazby, zvláště pro matematické vzorce.
- Automatizované generování obsahu, seznamů obrázků, tabulek a bibliografických odkazů.
- Možnost snadno pracovat s komplexními dokumenty jako jsou disertace nebo knihy.
- Rozsáhlé možnosti přizpůsobení a širokou škálu balíčků rozšiřujících jeho funkčnost.

V následujících sekcích se podrobněji podíváme na základní prvky \LaTeX a naučíme se, jak je používat k vytváření kvalitních dokumentů.

7.1 ZÁKLADNÍ STRUKTURA DOKUMENTU

V této kapitole se podrobněji podíváme na základní strukturu dokumentu v \LaTeX u. Po porozumění této struktuře budete schopni vytvářet vlastní dokumenty s přizpůsobeným formátováním a strukturou.

7.1.1 Preambule dokumentu

Preambule je první částí každého \LaTeX ového dokumentu. Zde definujeme typ dokumentu, který chceme vytvořit, a nastavíme různé parametry, které ovlivňují celkový vzhled dokumentu. Preambule také často obsahuje příkazy pro načítání různých balíčků, které rozšiřují základní funkčnost \LaTeX u.

```
\documentclass[options]{class}  
\usepackage[options]{package}
```

7.1.2 Hlavní tělo dokumentu

Hlavní tělo dokumentu začíná příkazem `\begin{document}` a končí `\end{document}`. Veškerý obsah, který chcete mít ve svém dokumentu, by měl být umístěn mezi tyto dva příkazy.

7.1.3 Sekce a podsekce

Pro organizaci obsahu se často používají sekce a podsekce. Tyto struktury pomáhají čtenáři lépe navigovat dokumentem a rozdělit text do logických bloků.

```
\section{Název sekce}  
\subsection{Název podsekce}  
\subsubsection{Název podpodsekce}
```

7.1.4 Odstavce a rozestupy

V \LaTeX u je nový odstavec vytvořen jednoduše vložením jedné nebo více prázdných řádek. Rozestupy mezi odstavci, stejně jako zarovnání textu, můžeme upravit podle potřeby.

7.2 PRÁCE S TEXTEM

Tato část se zaměřuje na základní techniky práce s textem v \LaTeX u, včetně formátování textu, vytváření seznamů a využití křížových odkazů a poznámek pod čarou.

7.2.1 Formátování textu

Formátování textu je klíčovým prvkem pro zvýraznění důležitých informací a zlepšení čitelnosti dokumentu.

Zvýraznění textu

V \LaTeX u existuje několik způsobů, jak zvýraznit text. Můžeme použít tučné písmo, kurzívu nebo podtržení.

```
\textbf{tučné písmo}, \textit{kurzíva}, \underline{podtržený text}
```

Seznamy a výčty

Seznamy jsou užitečné pro strukturování informací a jejich uspořádání do čitelné formy. \LaTeX podporuje nečíslované, číslované a popisné seznamy.


```

\begin{itemize}
\item Nečíslovaný seznam
\end{itemize}

\begin{enumerate}
\item Číslovaný seznam
\end{enumerate}

\begin{description}
\item[Popisek] Popisný seznam
\end{description}

```

7.2.2 Křížové odkazy a poznámky pod čarou

Křížové odkazy a poznámky pod čarou jsou důležité pro odkazování na jiné části dokumentu a poskytování dodatečných informací.

Křížové odkazy

Pomocí křížových odkazů můžeme odkazovat na jiné sekce, obrázky nebo tabulky v dokumentu.

```

\label{sec:nazev_sekce}
Odkaz na sekci \ref{sec:nazev_sekce}.

```

Poznámky pod čarou

Poznámky pod čarou poskytují dodatečné informace bez přerušení toku hlavního textu.

```
Text s poznámkou pod čarou.\footnote{Text poznámky pod čarou.}
```

7.3 MATEMATICKÉ VZORCE A SYMBOLY

Tato část poskytuje přehled o vkládání matematických vzorců a symbolů do dokumentů v \LaTeX u, což je nezbytné pro tvorbu akademických a vědeckých textů.

7.3.1 Základní matematické prostředí

L^AT_EX nabízí několik prostředí pro práci s matematikou, včetně "math" pro základní matematické výrazy a "displaymath" pro samostatné rovnice.

```
$z = x + y$ % Inline matematika  
\begin{displaymath}  
z = x + y  
\end{displaymath}
```

7.3.2 Rovnice a symboly

Matematické rovnice a symboly jsou základem mnoha vědeckých dokumentů, a L^AT_EX poskytuje širokou škálu nástrojů pro jejich efektivní použití.

Vložení jednoduché rovnice

Pro vložení jednoduché rovnice můžeme použít prostředí "equation" nebo "align" pro více rovnic s zarovnáním.

```
\begin{equation}  
E = mc^2  
\end{equation}
```

Pokročilé matematické výrazy

Pro složitější matematické výrazy, jako jsou integrály, sumy nebo frakce, L^AT_EX nabízí rozsáhlé možnosti.

```
\begin{equation}  
\int_0^{\infty} e^{-x} \, dx  
\end{equation}
```

7.4 PRÁCE S OBRÁZKY A TABULKAMI

Tato kapitola je zaměřena na vkládání a formátování obrázků a tabulek v L^AT_EXu, což jsou klíčové dovednosti pro vytváření vizuálně atraktivních a informativních dokumentů.

7.4.1 Vkládání obrázků

Vkládání obrázků do dokumentů \LaTeX umožňuje autorům přidávat vizuální prvky, které podporují a doplňují textový obsah.

Formáty obrázků

\LaTeX podporuje různé formáty obrázků, včetně populárních formátů jako JPEG, PNG a PDF. Výběr správného formátu je důležitý pro kvalitu a velikost souboru.

```
\includegraphics[width=0.5\textwidth]{obrazek.jpg}
```

Pozicování obrázků

Správné pozicování obrázků je klíčové pro zachování čitelnosti a estetiky dokumentu. \LaTeX nabízí několik možností, jak ovlivnit umístění obrázků v textu.

```
\begin{figure}[h]
\centering
\includegraphics[width=0.5\textwidth]{obrazek.jpg}
\caption{Popisek obrázku}
\label{fig:obrazek}
\end{figure}
```

7.4.2 Vytváření tabulek

Tabulky jsou nezbytné pro organizované a efektivní prezentování dat. \LaTeX umožňuje vytváření jak jednoduchých, tak složitých tabulek.

Základní tabulky

Pro vytváření základních tabulek lze využít prostředí "tabular". Jednoduchá tabulka může být vytvořena bez složitých formátovacích nástrojů.

```
\begin{tabular}{|c|c|c|}
\hline
A & B & C \\
\hline
1 & 2 & 3 \\
\hline
\end{tabular}
```

Pokročilé tabulky

Pro složitější tabulky, jako jsou tabulky s více řádky nebo sloupci, lze použít pokročilé formátovací možnosti, jako jsou sloučené buňky a speciální zarovnání.

```
\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|}
\hline
\multirrow{2}{*}{A} & B1 & C1 \\
\cline{2-3}
& B2 & C2 \\
\hline
\end{tabular}
\caption{Pokročilá tabulka}
\label{tab:pokrocila_tabulka}
\end{table}
```

7.5 BIBLIOGRAFIE A CITACE

Tato kapitola poskytuje podrobný návod na vytváření bibliografie a správné citování zdrojů v \LaTeX u, což jsou nezbytné dovednosti pro akademické psaní a publikování.

7.5.1 Vytváření bibliografie

\LaTeX umožňuje efektivní správu bibliografických záznamů a jejich automatické formátování. Tento proces zahrnuje několik kroků od definování zdrojů po jejich začlenění do dokumentu.

```
\begin{thebibliography}{99}
\bibitem{nazev}
Autor, \emph{Název knihy}, Nakladatelství, Rok.
\end{thebibliography}
```

7.5.2 Citování zdrojů

Správné citování zdrojů je klíčové pro akademickou integritu a umožňuje čtenářům dohledat zmiňované informace. V \LaTeX u je možné citovat zdroje jednoduše pomocí příkazu `\cite`.

Jak bylo zmíněno v `\cite{nazev}`, ...

8 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 nalezneš 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].

8.1 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.

8.1.1 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 8.1 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 8.1: Tato tabulka slouží jako ukázka toho, jak mohou tabulky vypadat.

záhlaví	této	tabulky
obsah	tabulky	už
není	oddělený	čarami

8.1.2 Obrázky

U obrázků je dobré používat vektorové formáty, pokud to jde. \LaTeX se nejvíce 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.

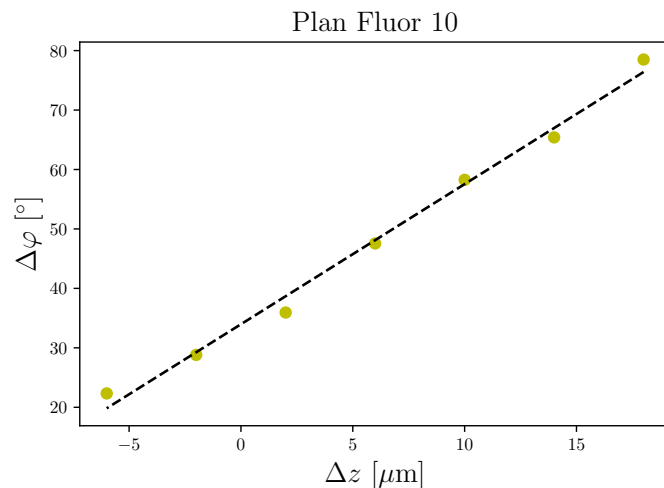


Obrázek 8.1: Logo SŠPU Opava [5].

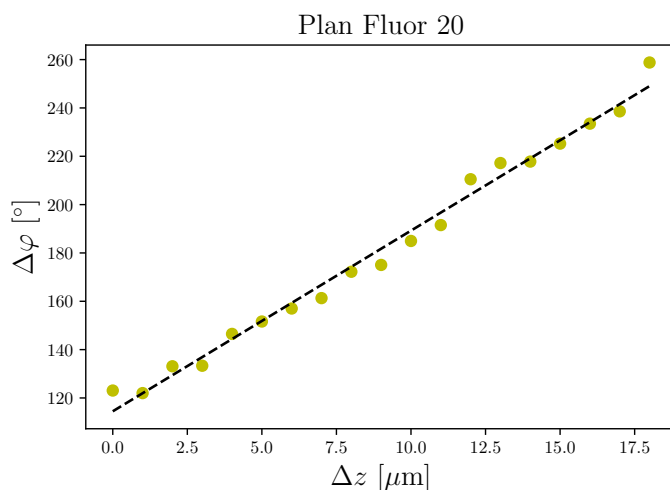
Když chceš odkazovat na obrázek, stačí pak už jen napsat příkaz `ref` a do závorek napsat označení obrázku. Třeba logo SOČky, můžeš vidět na obrázku 8.1 [?].

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 8.2: 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 popsát. 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.

8.1.3 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]

8.1.4 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 8.1: Ukázka Python kódu

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

Kód 8.2: 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 8.3: ES6 (ECMAScript-2015) Listing

8.2 POKROČILEJŠÍ TIPY, KTERÉ SE MOHOU HODIT

8.2.1 Rovnice

Sazba matematiky je věda sama o sobě. Ačkoli Word prošel obrovskou změnou a je v tomto mnohem lepší, tak \LaTeX 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 \LaTeX by to nezvládl. Ať už jde o základní věci jako řecká písmenka – $\alpha, \beta, \gamma, \dots$ – integrály – $\int_{l_i}^f \tau dl$ – 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.

$$\vec{F} = m\vec{a} \quad (8.1)$$

... Například podle druhého Newtonova zákona, rovnice (8.1) ... 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].

9 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 chvílku 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 tvé 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 LaTeX2e* [online]. 1998 [cit. 2020-08-24]. Dostupné z: <https://www.jaroska.cz/elearning/informatika/typografie/lshort2e-cz.pdf>
- [3] *Wikibooks: LaTeX* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2020-08-24]. Dostupné z: <https://en.wikibooks.org/wiki/LaTeX>
- [4] *TeX - LaTeX 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

1.1	Zpracování statické webové stránky	5
1.2	Zpracování dynamické webové stránky	6
1.3	Přístupování k databázi	7

2.1	Diagram frameworku Django	9
2.2	Architektura Model-Pohled-Šablona	10
2.3	Struktura projektu Django	12
3.1	Django	13
3.2	PostgreSQL	13
3.3	Docker	14
3.4	Gunicorn	14
3.5	Nginx	14
3.6	Soft UI Dashboard	15
4.1	Formulář přihlášení	17
4.2	Docker kontejnery	19
4.3	Administrace na adrese /admin	19
8.1	Logo SŠPU Opava [5].	32
8.2	Graf závislosti rotace DH PSF $\Delta\varphi$ na defokusaci objektivu Δz	33

Seznam tabulek

8.1	Tato tabulka slouží jako ukázka toho, jak mohou tabulky vypadat.	32
-----	--	----

PŘÍLOHA A SPOT DIAGRAMY A DALŠÍ