SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE
dokumentace

Eshop v Djangu

Petr Valášek

Obor: 18-20-M/01 INFORMAČNÍ TECHNOLOGIE
se zaměřením na počítačové sítě a programování

Třída: IT4

Školní rok: 2023/2024

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

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

V Opavě 22.04.2024

podpis autora práce

ABSTRAKT

Projekt je vytvořený v prostředí Django, představuje elektronický obchod - Eshop, který nabízí uživatelům širokou škálu funkcí a možností. Jednou z klíčových vlastností tohoto e-shopu je jeho responzivní design, který umožňuje pohodlné a přehledné procházení a nákupy na různých zařízeních, včetně mobilních telefonů, tabletů a počítačů.

Uživatelé mají možnost vytvořit si účet pomocí intuitivního a uživatelsky přívětivého rozhraní pro přihlášení a registraci. Eshop umožňuje přidávat recenze a sdílet tak zkušenosti s produktem s ostatními zákazníky.

Pro dosažení responzivního designu byly v projektu využity moderní technologie jako je framework Bootstrap.

Klíčová slova: Bootstrap, responsivní, Django, Eshop, recenze

ABSTRACT

This project, created within the Django environment, represents an electronic store - Eshop, offering users a wide range of features and options. One of the key features of this e-shop is its responsive design, allowing comfortable and clear browsing and shopping on various devices, including mobile phones, tablets, and computers.Users have the option to create an account using an intuitive and user-friendly interface for login and registration. The Eshop allows users to add reviews and share product experiences with other customers.To achieve a responsive design, modern technologies such as the Bootstrap framework were used in the project.

Keywords: Bootstrap, responsive, Django, Eshop, reviews

OBSAH

ÚVOD

V dnešní digitální éře se internet stal nevyhnutelnou součástí našich životů, poskytujícím nejen informace, ale i možnosti nákupů a obchodování. V rámci mé závěrečné studijní práce jsem se rozhodl prozkoumat svět webových aplikací a zaměřit se na vývoj Eshopu v Djangu. Toto rozhodnutí vyplývá z přesvědčení o síle a potenciálu moderních technologií, které nám umožňují vytvářet robustní a uživatelsky přívětivé platformy pro elektronický obchod.eshop v djangu

Důvodem mého zájmu o vývoj Eshopu v Djangu je především jeho vysoká úroveň flexibility a efektivity. Django, jako vysokoúrovňový webový framework napsaný v Pythonu, nabízí širokou škálu nástrojů a funkcí pro rychlý a efektivní vývoj webových aplikací jako vestavěné administrátorské rozhraní.

Dalším klíčovým faktorem, který mě motivoval k výběru této problematiky, je současný trend digitalizace obchodních procesů a posun k online prostředí. V dnešní době je důležité nejen mít přítomnost na internetu, ale také vytvořit uživatelsky přívětivé a funkční webové stránky, které dokážou oslovit a zaujmout zákazníky. Z tohoto důvodu jsem se rozhodl prozkoumat možnosti vývoje Eshopu v Djangu a zkoumat jeho potenciál v oblasti elektronického obchodu.

1.1 Django

Django je vysoko úrovňový webový framework napsaný v Pythonu, který podporuje rychlý vývoj webových aplikací. Je navržen tak, aby pomáhal vývojářům dodržovat princip

„Dont Repeat Yourself“ – DRY

To znamená, že s minimální duplikací kódu poskytuje robustní sadu nástrojů a funkcí, které zahrnují:

- ORM (Object-Relation Mapping) pro práci s databázemi
- Vestavěné administrační rozhraní pro správu dat
- Autentizaci a autorizaci uživatelů
- Formuláře a validaci dat
- Podporu pro vytváření API pomocí Django Rest Framework
- Šablonový systém pro generování HTML
- Middleware pro zpracování požadavků a odpovědí
- Podporu pro mezinárodní lokalizaci a další…

Django také podporuje principy běžného vývoje a obsahuje ochranu proti mnoha běžným útokům, jako je Cross Site Scripting – XXS, a Cross Site Request Forgery CSRF a SQL Injection. Je vhodný pro vývoj široké škály webových aplikací, od jednoduchých webových stránek po složité webové aplikace a API

1.2 Django - Architektura

Django architekturu Model-View-Template (MVT).

M – Model

V – View

T - Template

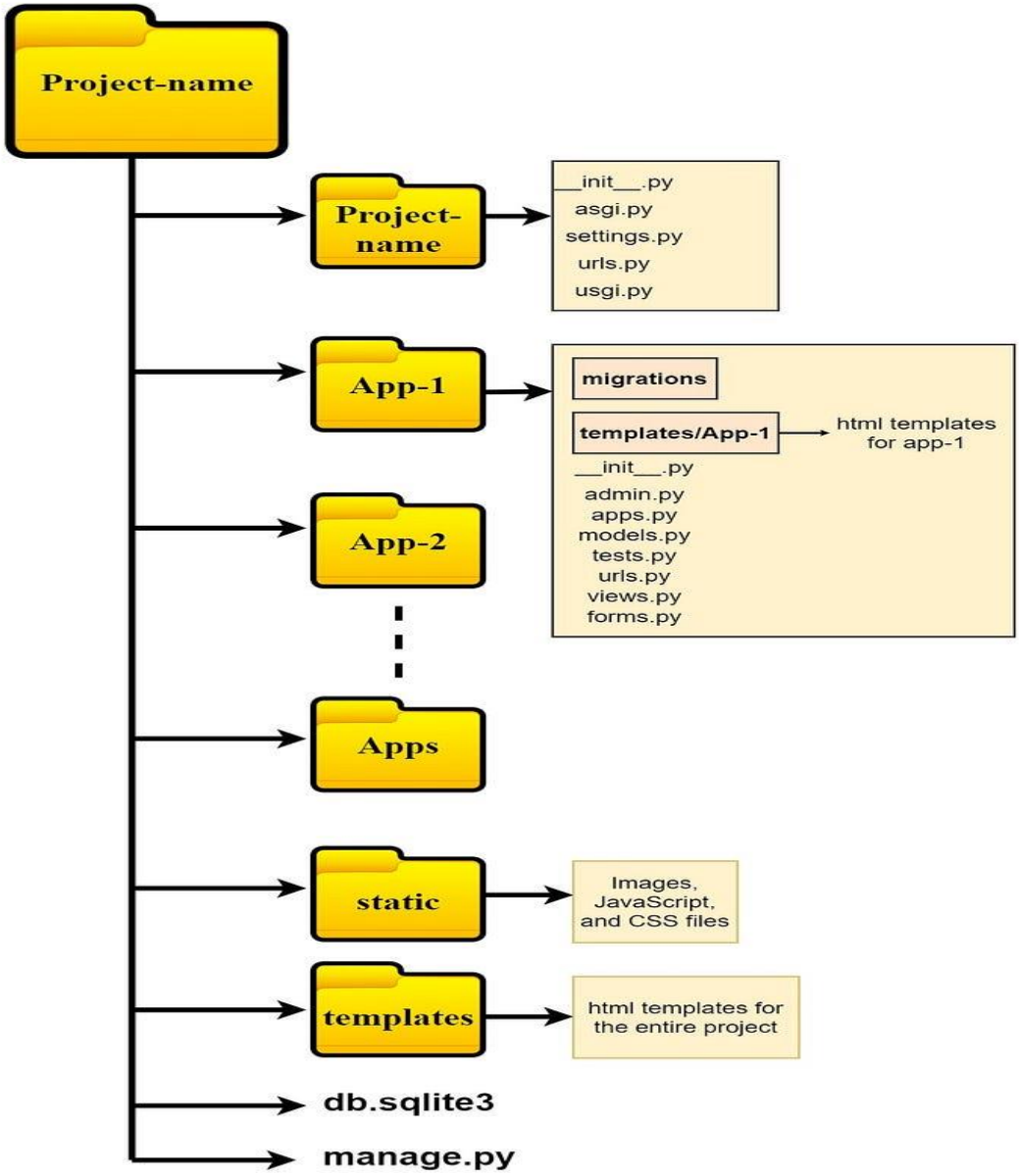
Zde je popis jak Django funguje:

- Model: Model je reprezentace databáze. Je to místo, kde definujete strukturu svých dat. Django používá ORM (Object-Relational Mapping) pro mapování modelů na databázové tabulky.

- View: View zpracovává požadavky a připravuje odpověď. View získává data z modelů, zpracovává je a předává je do šablony. View může být funkce nebo třída v Django.

- Template: Template je místo, kde je definováno, jak bude výstup prezentován uživateli. Django používá vlastní šablonovací jazyk pro generování HTML.

- URL Dispatcher: Django používá URL dispatcher, který směruje HTTP požadavky na odpovídající view na základě URL.



Když Django přijme HTTP požadavek, URL dispatcher rozhodne, který view má zpracovat požadavek na základě URL. View poté získá potřebná data z modelů, zpracuje je a předá je do šablony. Šablona poté generuje HTML, který je odeslán zpět klientovi jako odpověď. Django také poskytuje další funkce, jako je autentizace, formuláře, administrační rozhraní, CSRF ochrana a mnoho dalšího, které usnadňují vývoj webových aplikací.

1.2.1 CSRF Ochrana

CSRF (Cross-Site Request Forgery) je typ útoku, kde neoprávněný aktér zneužívá důvěru, kterou webový server má v prohlížeči uživatele.

Útočník nutí oběť k provedení nechtěných akcí na webové stránce, na které je oběť přihlášena. Django poskytuje zabudovanou ochranu proti CSRF útokům. Tato ochrana je implementována jako middleware a je ve výchozím nastavení aktivní.

Django CSRF ochrana funguje tak, že při každém POST požadavku očekává CSRF token. Tento token je generován Django a je jedinečný pro každou session uživatele. Token je vložen do formuláře jako skryté pole a je odeslán zpět serveru při odeslání formuláře.

Django poté ověří, zda je token platný a zda odpovídá tokenům uloženým na serveru. Pokud token chybí nebo se neshoduje, Django vrátí chybu a požadavek je odmítnut.

To znamená, že pokud útočník pokusí odeslat škodlivý požadavek na server, nebude mít přístup k CSRF tokenu a požadavek bude odmítnut. V Django šablonách můžete vložit CSRF token do formuláře pomocí tagu {% csrf_token %}.

Pokud vytváříte formuláře pomocí Django formulářových tříd, CSRF token je automaticky vložen do formuláře.

```
<form method="POST">
    {% csrf_token %}
    <!-- další pole formuláře -->
</form>
```

1.2.2 M - Model(y)

V Djangu jsou modely reprezentace databázových tabulek. Každý model je definován jako třída, která dědí od django.db.models.Model.

Atributy třídy reprezentují pole v databázové tabulce a každé pole je instance třídy Field (např. CharField pro textové řetězce, IntegerField pro celá čísla, DateField pro data atd.).

Django automaticky vytvoří databázovou tabulku pro každý model. Django také automaticky vytvoří ID pole pro každý model jako primární klíč, pokud není explicitně definován jiný.

Modely jsou obvykle definovány v souboru models.py v aplikaci Django. Po definování modelů musíte vytvořit migrace pomocí příkazu python manage.py makemigrations a poté aplikovat migrace pomocí python manage.py migrate. Tím se provedou potřebné změny v databázi.

```
from django.db import models

class Product(models.Model):
    name = models.CharField(max_length=200)
    price = models.DecimalField(max_digits=5, decimal_pla-
ces=2)
    description = models.TextField()
```

1.2.3 V – View(s)

View v Django je místo, kde se definuje logika, která zpracovává uživatelské požadavky a připravuje odpovědi. View může být funkce def nazev(request) nebo třída class.

Třídní view jsou založeny na třídách a umožňují definovat metody pro různé HTTP metody.

View jsou obvykle definovány v souboru views.py v aplikaci Django. Každý view musí být přiřazen k URL, což se obvykle děje v souboru urls.py.

```
from django.http import HttpResponse
from django.views import View

class HelloWorldView(View):
    def get(self, request):
        return HttpResponse("Hello, World!")
```

1.2.4 T – Template(s)

Šablony (Templates) v Django jsou zodpovědné za prezentaci dat uživateli. Jsou to textové soubory, které definují, jak se data zobrazí. Django používá svůj vlastní jazyk pro šablony, který umožňuje dynamicky generovat HTML.

Šablony mohou obsahovat proměnné, které jsou nahrazeny skutečnými hodnotami při vykreslování šablony, a značky, které mohou ovlivnit logiku generování HTML.

Šablony jsou obvykle uloženy v adresáři templates v aplikaci Django. Když chcete použít šablonu v view, můžete použít metodu render(), která vezme název šablony a slovník proměnných a vrátí vykreslený

```
<!DOCTYPE html>
<html>
<head>
    <title>{{ title }}</title>
</head>
<body>
    <h1>{{ heading }}</h1>
    <p>{{ content }}</p>
</body>
</html>
```

Šablony jsou obvykle uloženy v adresáři templates v aplikaci Django. Když chcete použít šablonu v view, můžete použít metodu render(), která vezme název šablony a slovník proměnných a vrátí vykreslený HTML:

```
from django.shortcuts import render

def my_view(request):
    return render(request, 'my_template.html', {'title': 'My
Title', 'heading': 'My Heading', 'content': 'My Content'})
```

1.3 Django databáze

Django podporuje mnoho různých databázových systémů.

Ve výchozím nastavení Django používá **SQLite** databázi, ale je také plně kompatibilní s dalšími populárními databázovými systémy, jako jsou PostgreSQL, MySQL a Oracle.

Konkrétní databáze, kterou Django projekt používá, je konfigurována v souboru settings.py v sekci DATABASES.

1.4 Static

Statické soubory v Django jsou soubory, které nejsou dynamicky generovány. Jsou to obvykle CSS, JavaScript a obrázkové soubory, které jsou potřebné pro správné zobrazení webových stránek.

2 VYUŽITÉ TECHNOLOGIE

2.1 Základní

2.1.1 Pycharm

Jako vývojové prostředí (IDE) jsem si vybral PyCharm, který je velmi oblíbený mezi vývojáři Pythonu. Jedním z hlavních důvodů, proč jsem si vybral PyCharm, je jeho přehledné propojení s verzovacími systémy jako Git a platformami pro sdílení kódu jako GitHub.

PyCharm poskytuje uživatelsky přívětivé rozhraní pro práci s Gitem a GitHubem, což zjednodušuje takové úkony jako commitování změn, pushování a pullení aktualizací, vytváření a sloučení větví a další.

Kromě toho PyCharm nabízí možnost resetování větve na konkrétní commit, což je užitečné, pokud potřebujete vrátit svůj kód do předchozího stavu. Tuto operaci můžete provést přímo z rozhraní PyCharmu bez nutnosti používat příkazovou řádku.

2.1.2 Git

Git je distribuovaný systém pro správu verzí, který umožňuje vývojářům sledovat a spravovat změny v kódu. Git byl původně vytvořen Linusem Torvaldsem pro vývoj jádra Linuxu, ale nyní je široce používán v mnoha různých projektech. Git umožňuje vývojářům vytvářet "větve" kódu, což umožňuje pracovat na různých funkcích nebo opravách chyb odděleně od hlavní větve kódu. Když je práce na větvi dokončena, může být sloučena zpět do hlavní větve. Git také umožňuje vývojářům "commitovat" změny, což je způsob, jak uložit stav kódu v daném okamžiku.

2.1.3 Github

GitHub je webová služba, která poskytuje hosting pro repozitáře Git. GitHub umožňuje vývojářům sdílet svůj kód s ostatními, spolupracovat na projektech a sledovat změny v kódu

v čase. GitHub také poskytuje řadu dalších funkcí, jako je sledování problémů, požadavků na tažení, wiki pro dokumentaci a další.

2.1.4 SQLITE

SQLite je knihovna v jazyce C, která poskytuje lehký diskový databázový systém, který nevyžaduje samostatný serverový proces a umožňuje přístup k databázi pomocí nezávislého souboru na disku. SQLite je velmi populární v mobilních aplikacích a je také často používán v webových aplikacích pro prototypování nebo pro jednoduché aplikace.

2.2 Widget_tweaks

Knihovna widget_tweaks je doplňková knihovna pro Django, která poskytuje řadu užitečných funkcí pro práci s formulářovými widgety. Tato knihovna umožňuje vývojářům snadno manipulovat a měnit vlastnosti formulářových widgetů přímo v šablonách Django.

```
{% load widget_tweaks %}
<!-- ... -->
<form method="post">
    {% csrf_token %}
    {{ form|crispy }}
    {% for field in form %}
        {% render_field field class="form-control" %}
    {% endfor %}
    <button type="submit">Odeslat</button>
</form>
<!-- ... -->
```

2.3 Django.contrib.auth

V kontextu Django, django.contrib.auth je modul, který poskytuje systém pro autentizaci a autorizaci. Tento modul je součástí Django "contrib" balíčku, což je sada doplňkových modulů dodávaných s Django, které poskytují běžně používané funkce.

Autentizace je proces ověřování identity uživatele. To znamená, že systém musí potvrdit, že je uživatel tím, kým tvrdí, že je. Django poskytuje několik nástrojů pro autentizaci, včetně

formulářů pro přihlášení a odhlášení, jakož i modelu User pro ukládání informací o uživatelích.

2.3.1 User

Model User je vestavěný model v Django, který je součástí django.contrib.auth.models. Tento model je navržen tak, aby pokryl většinu běžných případů použití pro autentizaci a autorizaci uživatelů. Zde jsou některé z klíčových polí, které model User obsahuje:

```
from django.contrib.auth.models import User
```

username: Toto pole je určeno pro uživatelské jméno. Je to povinné pole a musí být unikátní napříč všemi instancemi modelu User.

password: Toto pole je určeno pro heslo uživatele. Django automaticky hashuje hesla, takže v databázi nejsou uložena jako čistý text.

email: Toto pole je určeno pro e-mailovou adresu uživatele.

first_name a **last_name**: Tato pole jsou určena pro jméno a příjmení uživatele.

is_staff: Toto pole je boolean, který určuje, zda má uživatel přístup k administrativnímu rozhraní Django.

is_superuser: Toto pole je boolean, který určuje, zda má uživatel všechna oprávnění bez explicitního přiřazení.

Třída User je většinou importována, aby byla použita ve formulářích EditProfileForm a SignUpForm ve forms.py. Tyto formuláře umožňují uživatelům upravit svůj profil nebo se zaregistrovat do systému.

2.4 Pillow

Pillow je knihovna v Pythonu, která umožňuje otevírat, manipulovat a ukládat různé formáty obrázkových souborů. Je to nástupce původní knihovny PIL (Python Imaging Library), ale je kompatibilní s původní knihovnou a přidává některé užitečné funkce.

Pillow podporuje širokou škálu formátů obrázků, včetně populárních formátů jako JPEG, PNG, TIFF a GIF. Poskytuje také nástroje pro práci s barevnými prostory, filtry, histogramy a dalšími aspekty digitálního zpracování obrazu.

2.5 DBVisualizer

Pro generaci diagramu jsem zvolil DBVisualizer, což je nástroj pro správu a vizualizaci relačních databází. Poskytuje uživatelské rozhraní pro připojení k různým typům databázových systémů, jako jsou MySQL, PostgreSQL, Oracle, SQL Server a mnoho dalších. S DbVisualizerem můžete procházet strukturu databáze, provádět dotazy SQL, upravovat data a vytvářet vizualizace dat. Je oblíbeným nástrojem mezi vývojáři a databázovými administrátory pro jejich každodenní práci s databázemi.

3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

3.1 Začátek

1. Instalace Pythonu
2. Instalace Djanga
3. Vytvoření projektu
4. Přidání jména vytvořené aplikace do INSTALLED_APPS v settings.py
5. Nastavení urls.py
6. Definice modelu v models.py
7. Registrace modelů v admin sekci
8. Vytvoření pohledu ve views.py
9. Napojení na urls.py
10. Tvorba šablony a napojení na statické soubory
11. Spuštění serveru - test

1.

2. pip install Django

3. py manage.py startapp Eshop

4.

5. path('cs/', include('Eshop.urls')),

6. class Objednavka(models.Model):produkt = models.ForeignKey(Produkt, on_delete=models.CASCADE, verbose_name='Produkt', help_text='Vyberte produkt')…

7. from .models import Produkt
   admin.site.register(Produkt)

8. def home(request):produkty = Produkt.objects.all()return render(request, 'hlavni/home.html', {'produkty': produkty})

9. path('', views.home, name='hlavni_stranka'),

10. {% load static %}
    {block title}
    TEST
    {endblock}

11. py manage.py runserver

3.2 Pokračování

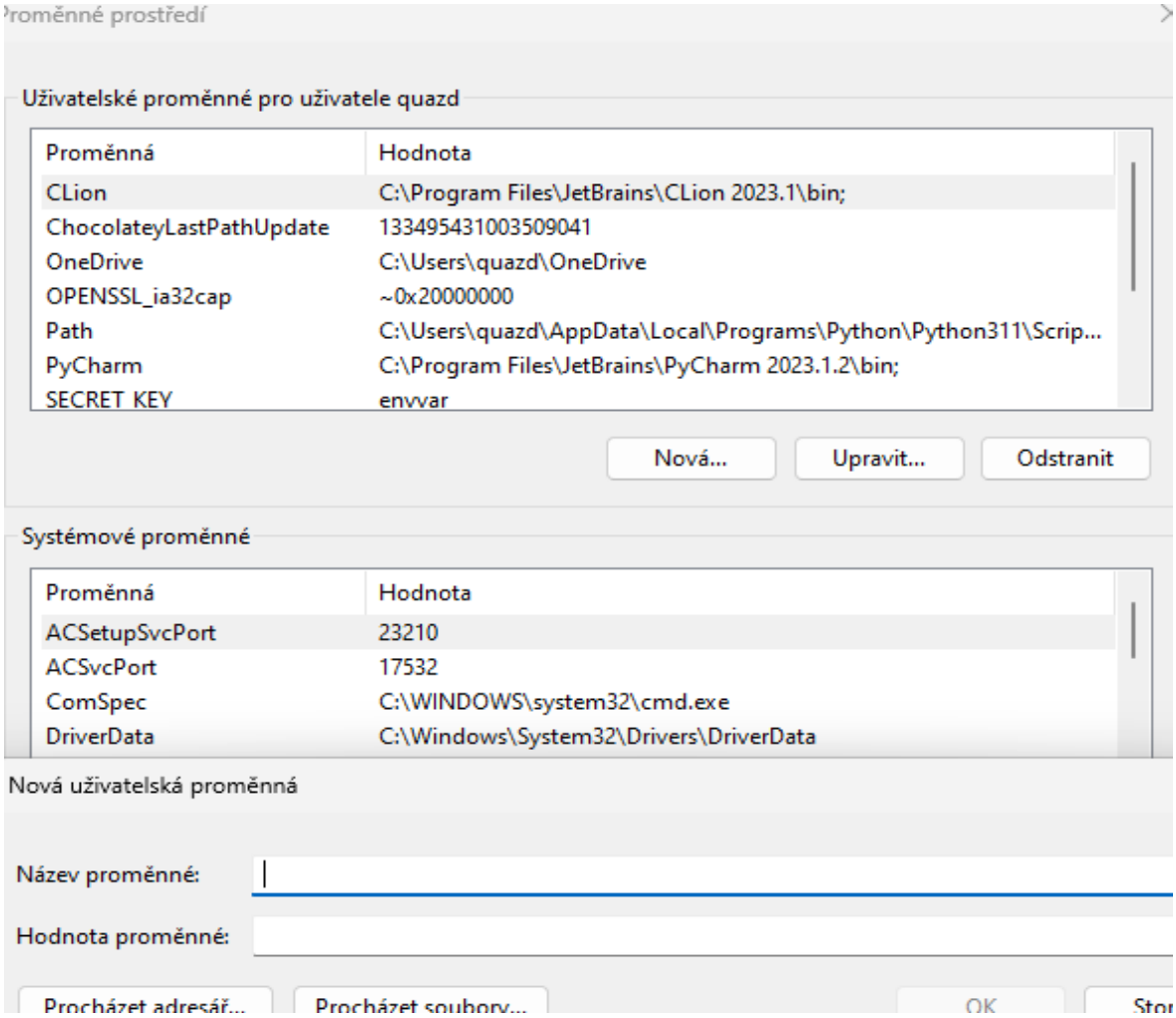
- Rozšířil jsem modely, které jsem „vyvolal“ v admin.py
- Vytvořil šablonu base.html na kterou napojil footer a navbar.
- hlavní stránka a detailní stránka produktu, poté jsem díky šablonám a celkové struktuře djanga jen vyvolával modely v šablonách.
- Přidal jsem login/registraci zapomocí User modelu z django.contrib.auth.

Ukázkova napojení:

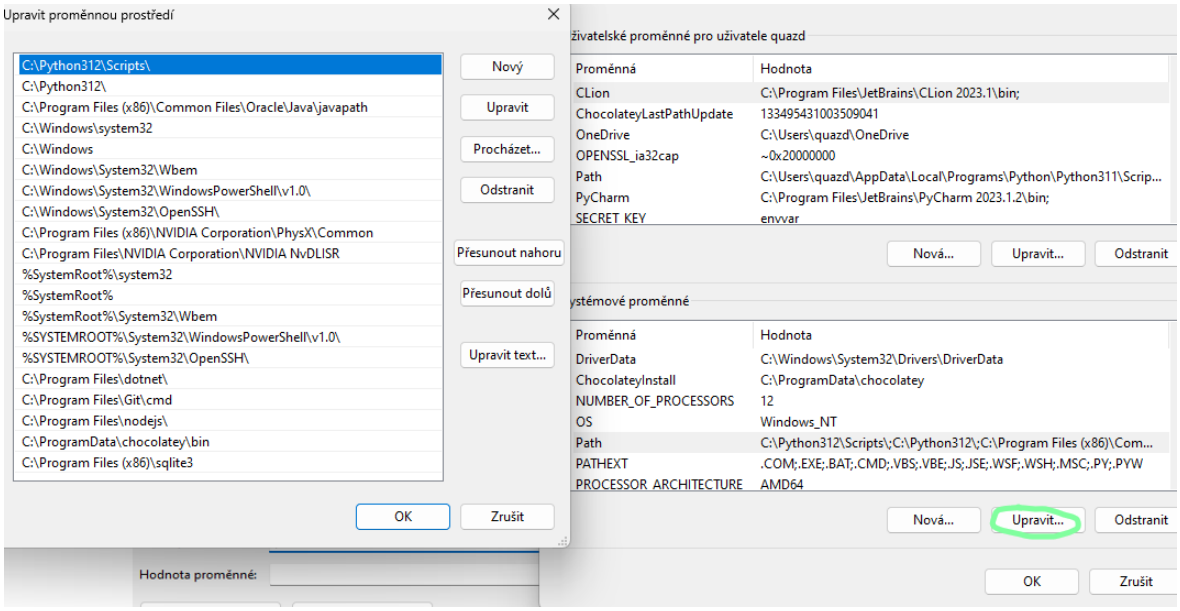
```
{% for brand in produkt.brand.all %}
    <div>
        <!-- Odkaz na stránku značky -->
        <a class="aer" href="{% url 'značka_podlahy'
brand.nazev %}">
            <!-- Název značky -->
            <h5 class="fw-bolder">{{ brand.nazev|capfirst
}}</h5>
            <!-- Podtržení -->
            <div class="underline-text2"></div>
        </a>
        <br>
        <!-- Název kategorie produktu -->
        <h5><b>{{ produkt.kategorie.nazev|capfirst
}}</b></h5>
    </div>
{% endfor %}
```

3.3 Vizualizace diagramu

Pro vizualizaci diagramu jsem použil DBVisualizer, jako první jsem musel stáhnout SQLite do počítače a vytvořit složku v Program Files (x86) a přidat ho do proměnného prostředí pro můj účet Microsoft



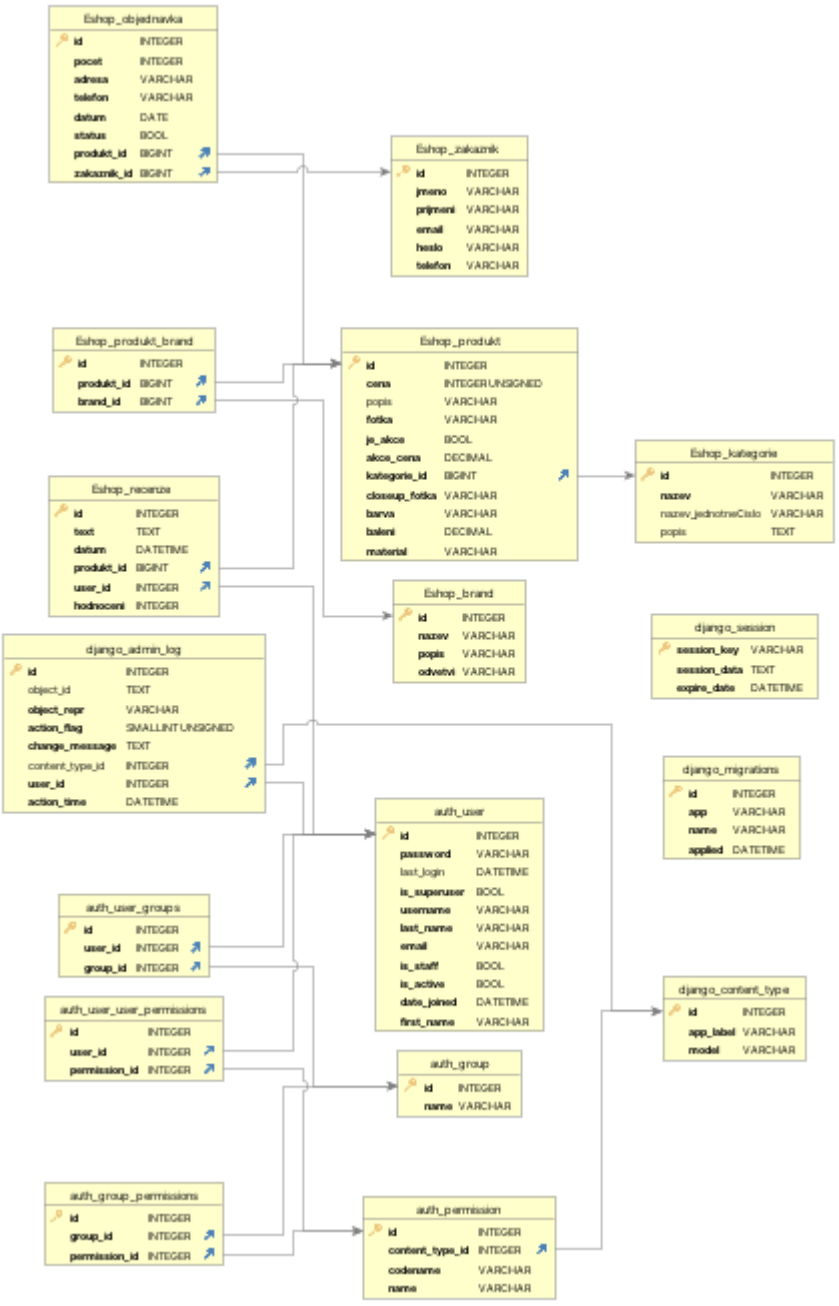
Poté už bylo jen třeba přidat cestu



Dále ověřit v CMD že je SQLite správně naistalován – pomocí jednoduchého příkazu sqlite3

Poté jen stačilo nainstalovat DBVisualizer a připojit databázi a najet do tables a zobrazit proper-

ties



4 VÝSLEDKY ŘEŠENÍ, VÝSTUPY, UŽIVATELSKÝ MANUÁL

4.1 Funkčnost Eshopu

Aplikace disponuje možností registrace, loginu, přidání recenzí, smazání recenze, změně hesla, změně nicknamu, zoomu fotky. Co bohužel aplikaci chybí je funkční nákupní košík s možností odeslat Objednávku.

4.2 Možná vylepšení

Určitě by se hodilo dodělat nákupní košík a popřípadě galerií fotek mezi kterými bude možnost přepínat. – Problém je že Eshop je dělán na podlahy a už tak bylo problém sehnat 2 stejné fotky podlahy a to přiblíženou fotku a fotku vzdálenou, když už je podlaha položená.

Také by se hodilo naimportovat kategorii „Ostatní“ a přidat do ní nějaké sady produktů.

Samozřejmě možnost přihlášení a registrace za pomocí Googlu atd… Problém je že Django nemá až tak velkou podporu v těchhle funkcích jako například Node – Next.js, který disponuje NextAuth.js, kdy naimportovat přihlášení pomocí Googlu zabere 5 minut. V djangu to může být na několik hodin.

Možná filtrace produktů dle ceny.

Search bar se mi zdá zbytečný, vzhledem k tomu, že podlahy nemají žádný specifický název.

Závěr

S Eshopem jsem až na nedodělaný košík spokojen. Cíl projektu byl proto splněn pouze z části. Jinak celý Eshop je responsivní jak na širokoúhlých obrazovkách tak i na zařízeních typu mobilu či tabletu.

https://github.com/Quazder/EshopDjango

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] Knihovna django.contrib.auth
https://docs.djangoproject.com/en/5.0/ref/contrib/auth/

[2] Database Vizualiser
https://www.dbvis.com

[3] Sqlite3
https://www.sqlite.org

[4] Widget Tweaks
https://pypi.org/project/django-widget-tweaks/

[5] CSS Generator
https://webcode.tools/css-generator

[6] Django
https://www.djangoproject.com

[7] django.contrib.auth
https://docs.djangoproject.com/en/5.0/ref/contrib/auth/

[8] django forms
https://docs.djangoproject.com/en/5.0/topics/forms/