GYMNASIUM JANA KEPLERA

Parléřova 2/118, 169 00 Praha 6

Webová stránka pro hodnocení obědů ze školní jídelny

Maturitní práce

Autor: Adam Dočkálek

Třída: 4.C

Školní rok: 2021/2022

Předmět: Informatika

Vedoucí práce: Šimon Schierreich

Praha, 2022

GYMNASIUM JANA KEPLERA
Kabinet informatiky

ZADÁNÍ MATURITNÍ PRÁCE

Student:	**Adam Dočkálek**
Třída:	**4.C**
Školní rok:	**2021/2022**
Platnost zadání:	**30. 9. 2022**
Vedoucí práce:	**Šimon Schierreich**
Název práce:	**Webová aplikace pro hodnocení jídel ze školní jídelny**

Pokyny pro vypracování:

Cílem práce je vytvořit webovou stránku, která slouží k hodnocení jídel ze školní jídelny. Na stránku by se měla automaticky posílat konkrétní jídla z jídelníčku, který se nachází na stránkách školy. Stránka by také měla umět třídit určitá jídla do různých kategorií (např. zda-li jsou vegetariánská) a to buď automaticky, či manuálně. Každé jídlo by mělo mít stránku, kde je možné psát komentáře, fotografie a hodnocení.

Doporučená literatura:

[1] MARTIN, Robert C. *Design Principles and Design Patterns*. **www.objectmen- tor.com, 2000. Dostupné z: https://fi.ort.edu.uy/innovaportal/file/2032/1/design_principles.pdf.**

[2] FOWLER, Martin. *Patterns of Enterprise Application Architecture*. **Boston, Massachusetts, USA: Addison-Wesley Professional, 2003. The Addison-Wesley Signature Series. ISBN 978-0-321-12742-6.**

[3] EVANS, Eric. *Domain-Driven Design: Tackling Complexity in the Heart of Software*. **Boston, Massachusetts, USA: Addison-Wesley Professional, 2003. ISBN 978-0-32-112521-7.**

[4] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. **2., aktualiz. a dopl. vyd. Brno: Computer Press, 2007. ISBN 978-80-251-1503-9.**

URL repozitáře:

https://github.com/Docksein/Maturitni-prace

student *vedoucí práce*

V Praze dne 22. 10. 2021

Prohlášení

Prohlašuji, že jsem svou práci vypracoval samostatně a použil jsem pouze prameny a literaturu uvedené v seznamu bibliografických záznamů. Nemám žádné námitky proti zpřístupňování této práce v souladu se zákonem č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších předpisů.

V Praze dne 21. srpna 2022 Adam Dočkálek

Poděkování

Chtěl bych poděkovat vedoucímu práce Šimonu Schierreichovi za rady, které pomohly, aby práce byla funkční.

Abstrakt

Práce má za účel vytvořit webovou aplikaci, na které je možné hodnotit obědy ze školní jídelny Gymnázia Jana Keplera. K vytvoření aplikace byl použit programovací jazyk Python a webový framework Django, který umožňuje relativně snadnou tvorbu webových aplikací s velkou podporou. Na automatické posílání dat z webové stráknky jídelny byla použita knihovna pandas, která umožňuje brát data z google tabulek. Výsledkem práce je webová aplikace, která umožňuje přihlášeným uživatelům psát hodnocení a komentáře ke školním obědům, které jsou automaticky poslány do databáze každý pracovní den. Uživatel si také může prohlédnout vlastní hodnocení, která již napsal.

Klíčová slova

webová aplikace, hodnocení, obědy

Abstract

This work is aimed to create a web application, on which a user can review lunch foods from the school canteen of Grammar School of Johannes Kepler. To create this application the programming language that was used is Python and the web framework Django, which allows relatively easy development of web application with a large amount of support. The automatic pulling of data from the website of the school canteen uses the library pandas that can get data from google sheets. The result of this work is a web application that allows rating and commenting of lunch foods for logged-in users . The foods are automatically saved to database every workday. The user can also view all of his past reviews.

Keywords

web application, reviews, lunches

Obsah

1. Teoretická část

Cílem práce je vytvoření webové aplikace, na které je možné hodnotit jídla ze školní jídelny. Měla by mít několik funkcí ke kterým samozřejmě patří hodnocení jídel, psaní komentářu a přihlášení uživatelů, kteří mohou hodnocení psát. Samotná jídla by se měla automaticky přidávat do databáze.

Co se týče již existujících hodnotících stránek, je můžeme jich na internetu najít velké množství. Ze stránek pro hodnocení filmů je to například české csfd.cz, imbd.com nebo rottentomatoes.com. Z obecných stránek na hodnocení je to například metacritic.com. Jako stránky s hodnocením můžeme vzít také on-line obchody, které běžně mají pod produktem část s uživatelskými hodnoceními.

1.1 Hodnocení

Stránka by měla mít seznam jídel a každé jídlo by mělo mít možnost zapsat komentář a hodnocení. Jídla mají název, hodnocení, fotku a tagy. Na stránce by se měl zobrazovat průměr hodnocení od 1 do 5 a z této škály si vybírá uživatel a volitelně může napsat komentář k jeho hodnocení. Pokud byl komentář uložen, uživatel bude přesměrován a komentář se zobrazí pod popisem daného jídla, ke kterému komentář patří.

Uživatel by měl možnost napsat komentář k danému jídlu pouze jednou za den. Poté, jestli svůj názor změní, má možnost opět další hodnocení přidat.

1.2 Přihlašování uživatelů

Anonymní uživatel by měl možnost zobrazit si všechna jídla i jejich detaily(komentáře, hodnocení. apod.), ale neměl by mít možnost zapisovat komentáře, aby nebylo komentáře možné spamovat. Samotná registrace by měla jít přes google účty. Uživatel se přihlásí přes gmail a již by měl mít možnost psát komentáře.

1.3 Automatické přidávání jídel

Vzhledem k tomu, že jsou jídla přidávána každý týden na stránce školy, tak by mělo být možné data tahat a přidávat je do databáze pomocí programu automaticky.

2. Implementace

Pro vytvoření webové aplikace jsem použil knihovnu Django pro jazyk Python. S jazykem Python jsem měl již před tímto projektem zkušenosti, takže se mohu soustředit na vytváření samotné webové aplikace a nemusím se znovu učit syntaxi a logiku jiného jazyku. Django má rozsáhlou podporu, při instalaci obsahuje i zabudované předinstalované aplikace a funkce, které lze využití bez vytváření vlastních. Má také velkou on-line komunitu, takže je možné najít odpověď na otázky ohledně případných problémů při vývoji aplikace, které mohou nastat. Vzhledem k tomu, že jsem dělal webovou aplikaci poprvé v životě, tak bylo samozřejmostí, že se dopustím spousty chyb a v tomto případě je relativně jednoduché najít odpověď. Pro stylizování front-endu jsem použil framework Bootstrap, který mi byl doporučen vedoucím práce.

2.1 Databáze

Pro webovou aplikaci je potřeba databázi s několika tabuklami: tabulka Jídel, která má název, datum, kdy byla položka přidána, fotka jídla a klíč k tabulce s tagy. Na podporu image souborů je potřeba knihovnu Pillows.

Druhá je tabulka komentářů, která skladuje data jmen uživatelů, jednotlivá hodnocení(v tomto případě od jedné do pěti), datum přidání komentáře a klíč k jídlu, ke kterému patří daný komentář. Tyto dvě tabulky jsou mezi sebou ve vztahu One-to-many (jedno jídlo, více komentářů).

Třetí tabulkou jsou tagy, které mají pouze jméno. S tabulkou s jídly jsou ve vztahu Many-to-many.

Co se týče samotné databáze, ve vývoji používám databázi SQLite3. Je již zabudovaná při instalaci a nevyžaduje další nainstalované knihovny, či změny v kódu. Pokud by šla aplikace do fáze nasazení, tak by ovšem tato databáze nebyla ideální, protože postrádá určité funkce jako například více zapisování do databáze najednou. Ideální by byla databáze PostgreSQL, která se doporučuje jak oficiální django dokumentací, tak i komunitními uživateli. Pro změnění databáze by bylo potřeba nainstalovat knihovnu psycopg2 a změnit nastavení databáze v souboru settings.py.

Nejdříve bylo nutné vytvořit samotnou aplikaci na komentáře, která obsahuje předem vytvořené soubory, do kterých se píšou třídy a funkce, které potom komunikují s databází a front-end html soubory.

Architektura Djanga je založená na MVC, takže přístup k databázi zajišťují modely, které jsou v django aplikaci pod souborem models.py. V praxi pak implementace vypadá následovně:

```
class Tag (models.Model):

    name = models.CharField(max_length=100)

    def get_absolute_url(self):
        return reverse("tag_list", kwargs={"pk": self.id})

    def __str__(self):
```

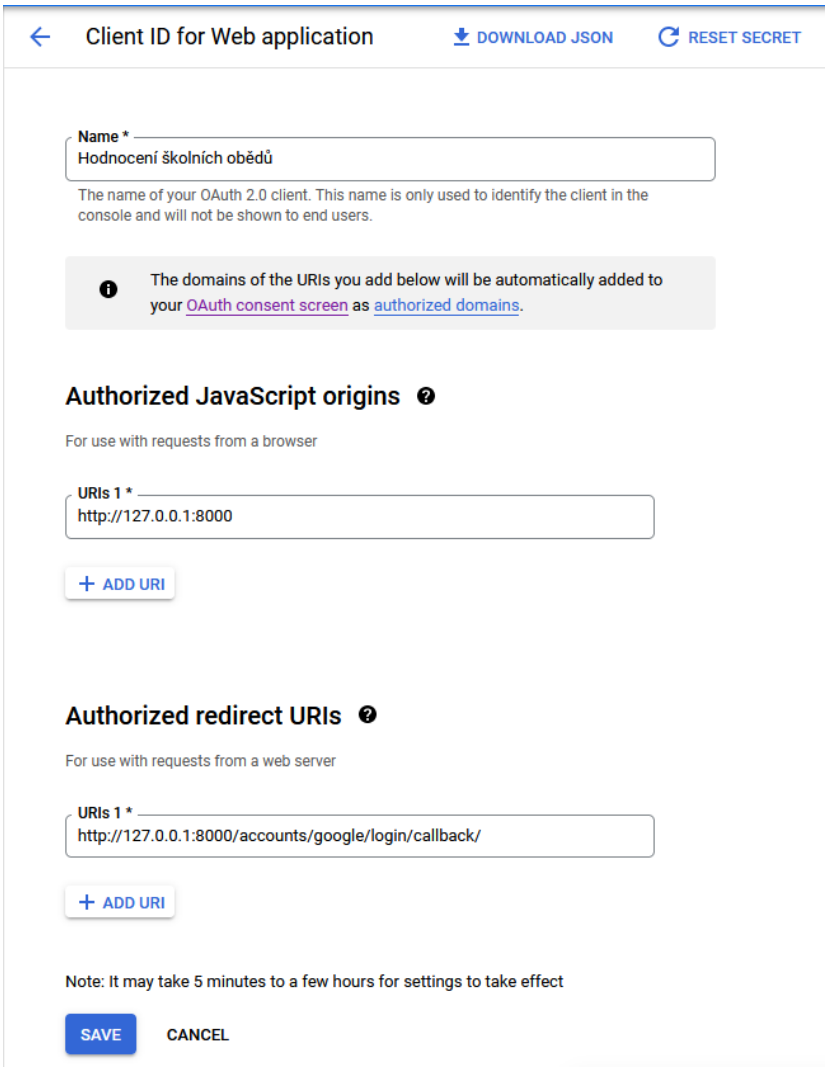
```
        return self.name


class Food (models.Model):

    title = models.CharField(max_length=200)
    upload_date = models.DateTimeField(default=timezone.now)
    picture = models.ImageField(upload_to="images/", blank = True, null=True)
    tags = models.ManyToManyField(Tag, blank=True)


    def average_rating(self):

        all_ratings = map(lambda x: x.ratings, self.review_set.all())
        return str(numpy.mean(list(all_ratings)))[:3]

    def average_rating_home(self):

        all_ratings = str(numpy.mean(list(map(lambda x: x.ratings, self.review_set.all()))))[:3]
        if all_ratings == "nan":
            return "0"
        else:
            return all_ratings


    def get_absolute_url(self):
        return reverse("food_reviews", kwargs={"food_id": self.id})

    def __str__(self):
        return self.title


class Review (models.Model):

    rating_choices = (
        (1, 1),
        (2, 2),
        (3, 3),
        (4, 4),
        (5, 5)
    )

    published_date = models.DateTimeField(default=timezone.now)
    food_key = models.ForeignKey(Food, on_delete=models.CASCADE)
    author_name = models.ForeignKey(User, on_delete=models.CASCADE)
    comment = models.CharField(max_length=1000)
    ratings = models.IntegerField(choices=rating_choices)
```

Dále v kódu můžeme vidět i některé funkce, funkce average_rating() slouží k vypsání průměru hodnocení, která je použita v html souborech, který používá django formy. Také například i funkce vypisující název jídla nebo funkce, která vypíše abolutní url link k danému jídlu.

2.2 Přihlašování uživatelů

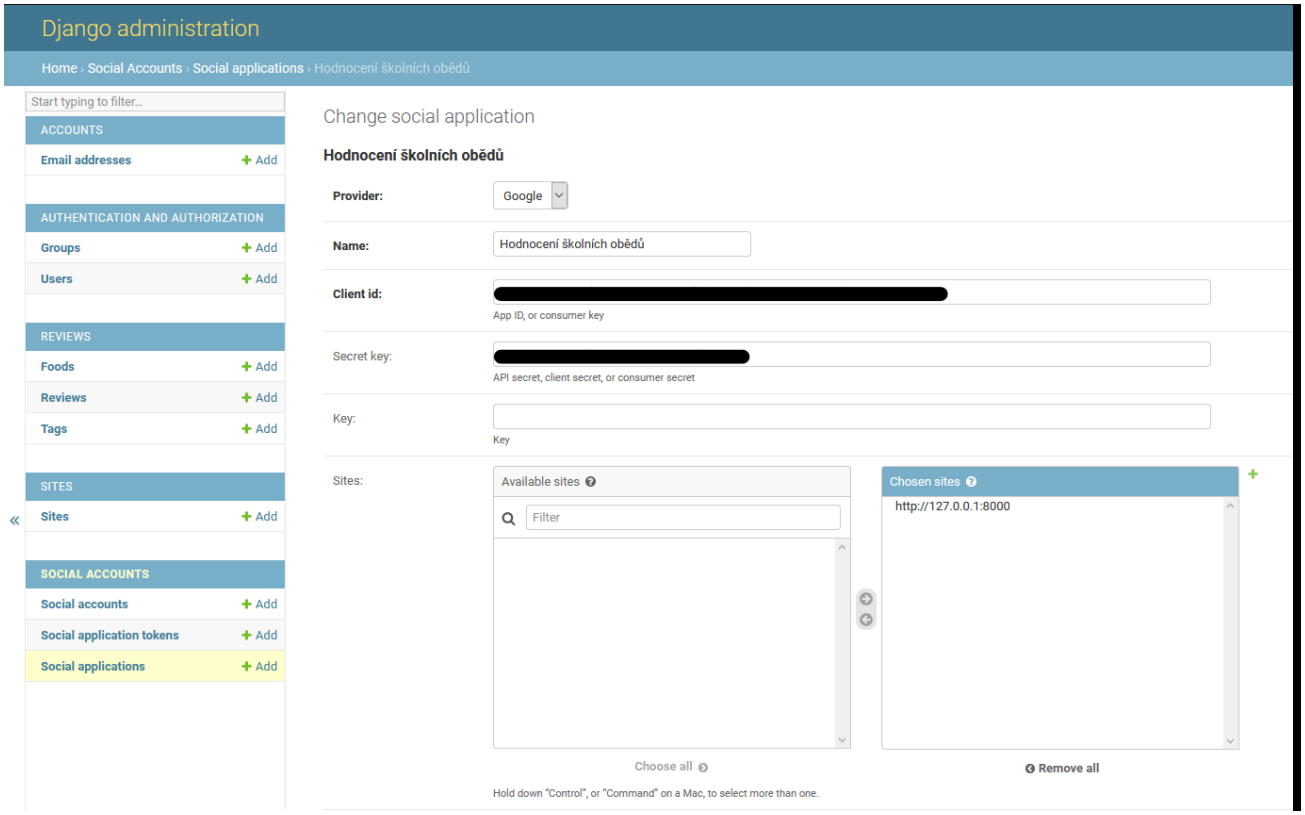
Pro registrování je použita aplikace 'django-allauth'. Samotná aplikace má zabudované přihlášení a registraci lokálně, ovšem má také možnost přihlášení přes účet z jiné sociální sítě. Pro tento projekt jsem vybral pouze přihlašování skrz google účty, které má přiřazený každý žák, takže se stačí přihlásit tímto účtem.

Abychom mohli využít google přihlašování, je nutné mít přístup k Google cloud platform. Zde je potřeba vytvořit nový projekt, nainstalovat Gmail API a vytvořit OAuth 2.0 credentials. Registrace credentials by měla vypadat zhruba takto:



Pro projekt by měl být k dispozici id (Client ID) a klíč (Client secret), který se přidá do databáze na admin stránce samotného webu. V části "Sites"se přidá jméno domény (v případě lokálního hostování http://127.0.0.1:8000 nebo localhost:8000) a v části "Social applications"se přidá provider

(Google), jméno webové aplikace, id s klíčem a stránka, kterou jsme přidali v části "Sites".



Bohužel účty gjk nemají k Google cloud platform povolený přístup, takže jsem musel využít svůj osobní účet.

2.3 Hodnocení a komentáře

Aby bylo formulář možné nechat renderovat na prohlížeči, je nejdříve nutné si vytvořit formu s poli v souboru forms.py, které chceme, aby vyplnil uživatel, kterými jsou hodnocení, což je povinná část a komentář, který je nepovinný. Tato forma je poté vyrenderována a uživatel vidí formulář se dvěmi poli, které by měl vyplnit. Aby nemohl nepřihlášený návštěvník stránky také psát do databáze, je tento formulář vyrenderovaný jen pro přihlášené uživatele díky zabudované funkci user.is_authenticated, pro nepřihlášené uživatele je pouze vidět text „Pro hodnocení se musíte přihlásit". Pokud uživatel již dnes dané jídlo hodnotil, tak jej také hodnotit nemůže.

Po úspěšném přidání hodnocení je uživateli zobrazena stránka s textem „Váš komentář byl úspěšně zapsán". Veškerá hodnocení pro dané jídlo jsou zobrazeny pod jídlem a formulářem pro poslání hodnocení. Je zobrazeno jméno autora, jeho hodnocení, datum napsání komentáře a případný komentář autora.

Při implementaci jsem narazil na problém, že při snaze uložit komentář stránka uživatele přesměrovala na seznam jídel a přesto že request na POST přišel zpět na back-end, hodnocení se v databázi neuložilo. Důvodem bylo, že request směřoval do jiné view funkce, která renderovala stránku se seznamem jídla. Ovšem tato funkce již nezvládala POST requesty z front-endu a tím pádem se již neukládaly do databáze. Důvodem byla akce form v html, která automaticky přesunula POST právě

funkci renderovací seznam jídel. Proto bylo nutné POST přesměrovat zpět do funkce, která zvládala zapsat komentář do databáze.

2.4 Automatické posílání jídel

Pro taháné jídel jsem využil knihovenu pandas. Knihovna pandas umožňuje práci z google tabulkami bez toho, aniž bych se potřeboval přihlašovat do google API, což by bylo v tomto případě zbytečné, protože tabulku potřebuji pouze číst. Podobný projekt na tahání jídel byl již vytvořen, proto jsem si část kódu půjčil z github repozitáře.

Pro automatické ukládání jsem využil knihovnu apscheduler. Jedna z jeho funkcích BackgroundScheduler umožňuje, aby scheduler běžel v pozadí aplikace na jiném vlákně a nebyla to jediná věc, která běží. Funkce se zapne každý pracovní den v 11 hodin. Nová jídla se přidají do databáze a jídlům, které v databázi již jsou, se přepíše čas, kdy byly naposledy v jídelníčku. Tagy a fotku k danému jídlu lze poté přidat manuálně.

3. Technická dokumentace

Pro stažení repozitáře do vlastního zařízení lze stáhnout repozitář přímo z Github.com nebo použít tento příkaz:

```
git clone https://github.com/Docksein/Maturitni-prace
```

Pro spuštění souboru je potřeba knihovny, které stáhnete následovně:

```
pip install django numpy django-allauth pandas apscheduler Pillow
```

Inicializování databáze:

```
python manage.py migrate
```

Spuštění:

```
python manage.py runserver
```

Pokud bychom chtěli uvést projekt do produkce, tak musíme nastavené do určité míry změnit. V souboru settings.py bychom měli přepsat tyto proměnné:

```
DEBUG proměnnou na DEBUG = False
```

Secret Key, můžeme přečíst např. z jiného souboru:

```
with open('/etc/secret_key.txt') as f:
    SECRET_KEY = f.read().strip()
```

A nastavení ALLOWED_HOSTS

Produkční nastavení lze také zkontrolovat pomocí příkazu:

```
python3 manage.py check --deploy
```

Závěr

Cíle, které byly u tohoto projektu určeny, tak byly splněny. Ukládání jídel je plně automatizované a je možné přidávat k němu tagy, fotky a psát k němu komentáře. Osobně jsem se naučil práci s Djangem a tvořením webových aplikací, protože jsem na žádném takovém projektu nikdy předtím nepracoval. Naučil jsem se jak fungují databáze v praxi a propojení databází, jak ukládat data z front-endu na back-end a jak potom data z databází zobrazit zpátky na webovém prohlížeči a aby byla data z jedné databáze propojené s daty v druhé na určité stránce. Osobně jsem se snažil, aby práce úspěšně naplnila zadání a abych se naučil nové věci, kterých bych do budoucna mohl využít.

Seznam použité literatury

[] *Bootstrap documentation v5.2.* URL: https://getbootstrap.com/docs/5.2/getting-started/introduction/.

[22a] *Django documentation*. 2022. URL: https://docs.djangoproject.com/en/4.1/.

[22b] *Questions tagged [django]*. 2022. URL: https://stackoverflow.com/questions/tagged/django.

[con22] MDN contributors. *Deploying Django to production*. 2022. URL: https://developer.mozilla.org/en-US/docs/Learn/Server-side/Django/Deployment.

[GR01] David Goodger a Guido van Rossum. *PEP 257 – Docstring Conventions*. 2001. URL: https://peps.python.org/pep-0257/.

[Grö22] Alex Grönholm. *Advanced Python Scheduler*. 2022. URL: https://apscheduler.readthedocs.io/en/3.x/index.html.

[Pen22] Raymond Penners. *django-allauth documentation*. 2022. URL: https://django-allauth.readthedocs.io/en/latest/.

[W3S22] W3Schools. *Django Tutorial*. 2022. URL: https://www.w3schools.com/django/.

Seznam obrázků

Seznam tabulek