

DEBRECENI SZC MECHWART ANDRÁS

GÉPIPARI ÉS INFORMATIKAI

TECHNIKUM

VIZSGAMUNKA DOKUMENTÁCIÓ

E-KERESKEDELMI OLDAL

Készítette:

Nagy Attila Sándor

Plangár Ábel József

Varjú Balázs

2024

Köszönetnyilvánítás

Ezúton szeretnénk megköszönni tanárainknak, osztályfőnökeinknek kitartó munkájukat, segítőkészségüket, mert nélkülük ez a záródolgozat nem készülhetett volna el.

Tartalom

Köszönetnyilvánítás	2
Vizsgaremek témája: Webshop készítése Django segítségével	5
Fejlesztői dokumentáció	5
Felhasznált technológiák	6
Python/Django	6
HTML/CSS	6
Bootstrap	6
Funkcionális követelmények	6
Felhasználói oldalak	6
Főoldal	6
Kosár oldal	7
Regisztráció és bejelentkezés	7
Rendelések oldal	8
Adminisztrációs felület	9
Kategóriák kezelése	9
Rendelések kezelése	10
A fejlesztés kezdő lépései	10
Virtuális környezet létrehozása	10
Virtuális környezet aktiválása	10
Django telepítése	11
Új Django projekt létrehozása	11
Alkalmazás hozzáadása	11
Az adatbázis kapcsolati ábrája	12
Felhasználói dokumentáció	13
A program bemutatása	13
Regisztráció/Bejelentkezés	13
Kijelentkezés	13
Szűrés	13
Néhány kihívás, érdekességek	14
1. Felhasználói interakció kezelése	15
2. Adatkezelés és adatbázis-műveletek	15
3. Routolás és átirányítások kezelése	15
4. Kosárkezelés	15
1. Dinamikus működés	16

2. Kosárkezelés	16
3. Form validáció	16
Teszt dokumentáció	16
Tapasztalat	16
Fejlesztési lehetőségek.....	18
1. Keresés megvalósítása:	18
2. Profil oldal:	18
3. Az oldal kinézetének javítása:	18
4. Bankkártyás fizetési lehetőség:	18
Irodalomjegyzék	18

Vizsgaremek témája: Webshop készítése Django segítségével

A modern kereskedelmi világban elengedhetetlenül fontos az online jelenlét és az internetes értékesítés lehetőségeinek kiaknázása. Kereskedőként ennek jegyében választottuk vizsgaremekünk témáját: egy webshop létrehozását Django keretrendszerrel.

A témaválasztásunkat több tényező indokolta. Először is, a Django egy erőteljes és kifejezetten alkalmas eszköz webalkalmazások fejlesztésére, amely gyors és hatékony munkavégzést tesz lehetővé. Másodszor, a webshopok népszerűsége és kereslete folyamatosan növekszik, és ezáltal a szoftverfejlesztés iránti igény is magas. Harmadszor, a webshopok fejlesztése számos érdekes és kihívást jelentő problémát kínál, amelyek megoldása során fejlődni tudunk.

A tervezett program egy modern, felhasználóbarát és rugalmas weboldal lesz, amely számos fontos funkciót kínál az online vásárlók számára. Az elképzelt webshop lehetővé fogja tenni a felhasználók számára a termékek könnyű böngészését, keresését és vásárlását, valamint a regisztrációt, bejelentkezést és a kosárkezelést. Emellett lehetőséget biztosítunk a termékek részletes leírására, képek feltöltésére és azok online megjelenítésére.

Célközönségünk elsősorban kis- és középvállalkozások, valamint egyéni vállalkozók, akik szeretnék online jelenlétet kialakítani, és értékesíteni termékeiket vagy szolgáltatásaikat az interneten keresztül. A programunk intuitív felhasználói felülettel és könnyen kezelhető adminisztrációs felülettel rendelkezik, így akár kezdő felhasználók is könnyen képesek lesznek üzemeltetni és karbantartani a webshopjukat.

Fejlesztői dokumentáció

A program elkészítése azért vált szükségessé, mert a vállalkozásunknak ki kell használnia az online piacok által kínált lehetőségeket, és fel kell készülnie a jövő kereskedelmi trendjeire. Az interneten történő jelenlét nemcsak a vállalkozásunk számára kínál új értékesítési csatornát, hanem lehetőséget teremt a célközönségünk számára, hogy kényelmesen és hatékonyan vásárolhasson tőlünk bárhol, bármikor.

Az egyik fő különbség más hasonló létező programokhoz képest az lesz, hogy az általunk készített webshop teljesen testreszabható és skálázható lesz. Ez azt jelenti, hogy a vállalkozásunknak lehetősége lesz saját igényeire és piaci helyzetére szabni a webshop funkcióit és megjelenését, így egyedi és konkurenciától megkülönböztethető online értékesítési felületet hozhatunk létre.

Felhasznált technológiák

Python/Django: A webalkalmazás alapját a Django keretrendszer képezi, amely hatékonyan kezeli a webshop adatbázisát (SQLite) és üzleti logikáját.

HTML/CSS: Az oldalak szerkezetének és stílusának megtervezése és megvalósítása HTML-ben és CSS-ben.

Bootstrap: A Bootstrap keretrendszer segítségével gyorsan és könnyen reszponzív és stílusos felhasználói interfészt hozunk létre.

A tanulmányaink során megismert Python és a Django keretrendszer lehetővé teszi számunkra, hogy hatékonyan és gyorsan fejlesszünk modern webalkalmazásokat, míg az SQLite, az HTML és a Bootstrap segítségével megvalósíthatjuk az alkalmazások tartalmi és vizuális megjelenítését.

A fejlesztés során használt szoftverek és eszközök: a Windows 10 operációs rendszer, amely stabil és széles körben elterjedt platformot biztosít a fejlesztéshez.

A Visual Studio Code egy könnyűsúlyú, de erőteljes kódszerkesztő, amely rendkívül népszerű a webfejlesztők körében. Az integrált fejlesztői környezet (IDE) számos kiterjesztést és funkciót kínál, amelyek segítségével hatékonyan tudunk kódolni, hibát keresni és projektjeinket kezelni.

A Mozilla Firefox böngésző pedig ideális választás a webalkalmazások teszteléséhez és ellenőrzéséhez. A böngésző fejlett fejlesztői eszközökkel rendelkezik, amelyek segítségével könnyedén ellenőrizhetjük a weboldalak megjelenését és működését, valamint hibakeresést végezhetünk.

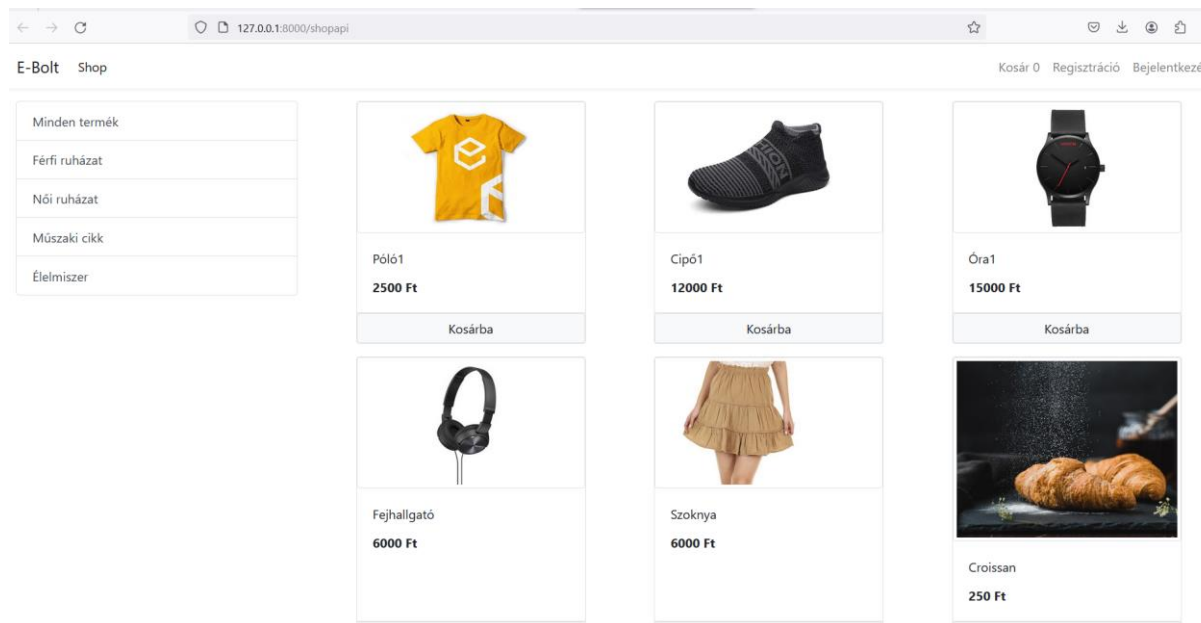
A fejlesztéshez használt hardverek: Intel i5 processzor, amely erőteljes és megbízható teljesítményt nyújt a fejlesztési feladatokhoz. A 8 GB RAM elegendő memóriát biztosít a programok futtatásához és a fejlesztési folyamat során keletkező adatok kezeléséhez.

Az integrált videokártya megfelelő grafikus teljesítményt nyújt az alapvető grafikai feladatokhoz és az IDE futtatásához, míg a 512 GB SSD (Solid State Drive) nagy tárolókapacitást és gyors adatelérést biztosít a projekt fájlok és adatbázisok tárolásához.

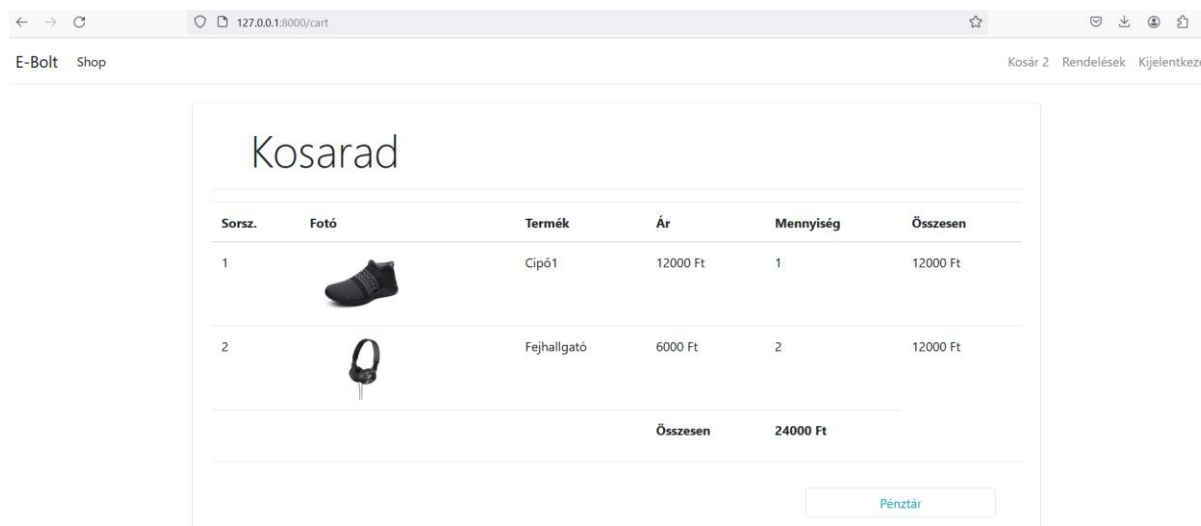
Funkcionális követelmények

Felhasználói oldalak:

Főoldal: Termékek megjelenítése, kategóriák böngészése, információk a termékekről, hozzáadás a kosárhoz.



Kosár oldal: Termékek listája a kosárban, árak, mennyiségek megjelenítése, rendelés leadása.



Regisztráció és bejelentkezés: Felhasználói fiókok kezelése.

← → ↻ 127.0.0.1:8000/signup ☆ 🛒 👤 📄

E-Bolt Shop Kosár 0 Regisztráció Bejelentkezés

Regisztráció

Keresztnév

Vezetéknév

Telefonszám

Email

Jelszó

Regisztráció

27.0.0.1:8000/signup

← → ↻ 127.0.0.1:8000/login ☆ 🛒 👤 📄

E-Bolt Shop Kosár 0 Regisztráció Bejelentkezés

Bejelentkezés

Email

Jelszó





Bejelentkezés

Rendelések oldal: A leadott rendelések megtekintése.

← → ↻ 127.0.0.1:8000/orders

E-Bolt Shop Kosár 0 Rendelések Kijelentkezés

Rendeléseid

Sorsz.	Fotó	Termék	Dátum	Ár	Mennyiség	Összesen	Status
1		Croissan	April 19, 2024	250 Ft	1	250 Ft	Függőben
2		Óra1	April 16, 2024	15000 Ft	2	30000 Ft	Függőben
3		Fejhallgató	April 16, 2024	6000 Ft	1	6000 Ft	Függőben
4		Póló1	April 15, 2024	2500 Ft	2	5000 Ft	Teljesítve

127.0.0.1:8000

Adminisztrációs felület:

Termékek kezelése: Új termék hozzáadása, meglévő szerkesztése vagy törlése.

← → ↻ 127.0.0.1:8000/admin/shopapi/product/

Django administration WELCOME, ATIS VIEW SITE / CHANGE PASSWORD / LOG OUT

Home > Shopapi > Products

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

- Groups + Add
- Users + Add

SHOPAPI

- Categories + Add
- Customers + Add
- Orders + Add
- Products + Add

Select product to change

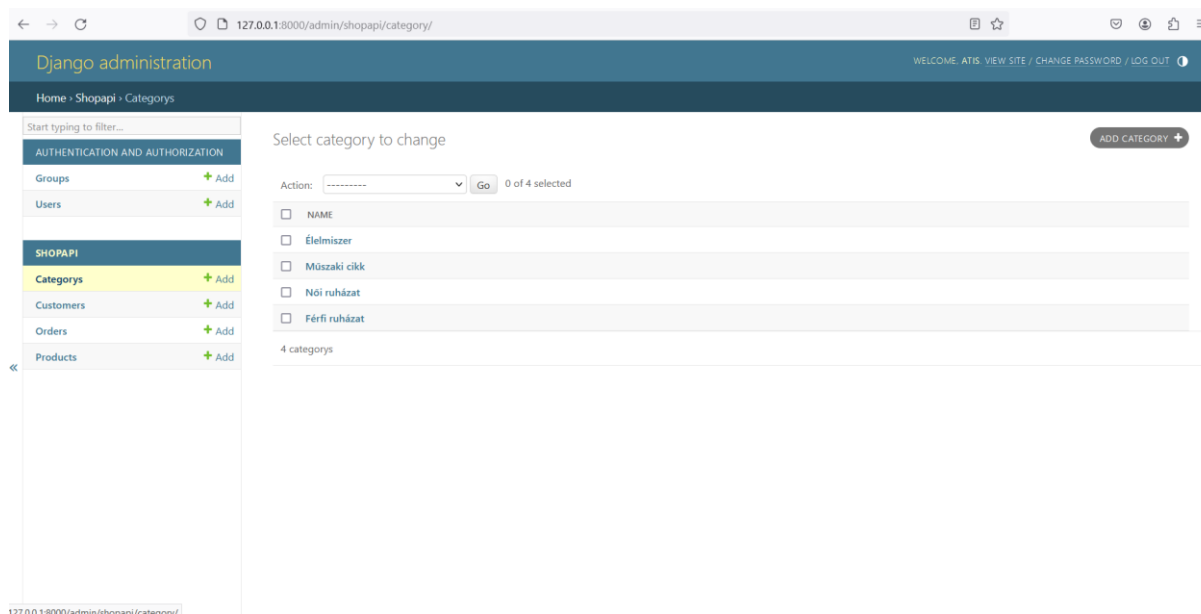
Action: Go 0 of 6 selected

<input type="checkbox"/>	NAME	PRICE	CATEGORY
<input type="checkbox"/>	Croissan	250	Élelmiszer
<input type="checkbox"/>	Szoknya	6000	Női ruházat
<input type="checkbox"/>	Fejhallgató	6000	Műszaki cikk
<input type="checkbox"/>	Óra1	15000	Műszaki cikk
<input type="checkbox"/>	Cipő1	12000	Férfi ruházat
<input type="checkbox"/>	Póló1	2500	Férfi ruházat

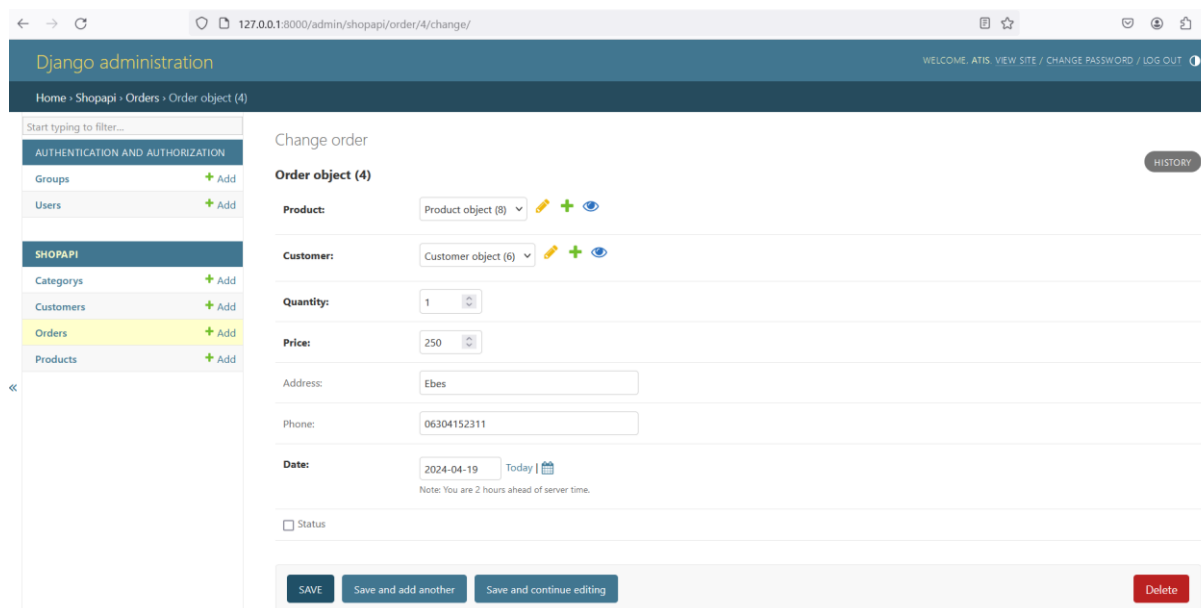
6 products

ADD PRODUCT +

Kategóriák kezelése: Kategóriák létrehozása, szerkesztése vagy törlése.



Rendelések kezelése: Beérkező rendelések listázása, státuszuk módosítása.



A fejlesztés kezdő lépései:

Virtuális környezet létrehozása: A projekt elkészítéséhez és a különböző Python csomagok telepítéséhez létrehozok egy virtuális környezetet. Ez egy izolált környezet, ahol a projekt függőségei és a Python verziója elkülönül a számítógépem többi részétől. Megnyitom a parancssort (terminált), a projekt mappában, ahol el szeretném tárolni a kódot. Ott verziókövetést állítok be git init paranccsal majd beírom be a következő parancsot: `python -m venv .` Így létrehozom a virtuális környezetet a Webbolt mappában.

Virtuális környezet aktiválása: Miután létrehoztam a virtuális környezetet, aktiválom azt. Ehhez egyszerűen beírom a parancssorba a következőt: Windows rendszeren: `.\Scripts\activate`

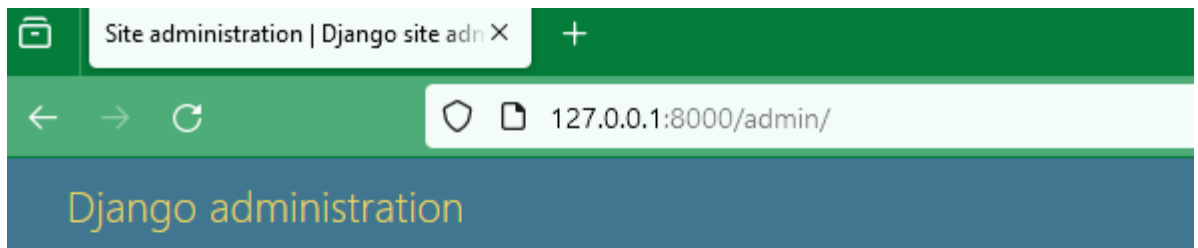
Django telepítése: Amikor a virtuális környezet aktív, telepítem a Django-t. Egyszerűen beírom a parancssorba a következőt: `pip install django`.

Új Django projekt létrehozása: Miután telepítettem a Django-t, létrehozok egy új Django projektet. Ehhez beírom a következőt: `django-admin startproject .`. A `python manage.py migrate` paranccsal pedig létrehozom a `db.sqlite3` adatbázist.

Alkalmazás hozzáadása: A Django projekt több alkalmazást is tartalmazhat, amelyek különböző részeket vagy funkciókat képviselnek. Így miután létrehoztam a projektet, létrehozok egy új alkalmazást, ami segít majd a kódom strukturáltabbá tételében és könnyebben kezelhetővé tételében. Ehhez a parancssorba írom: `python manage.py startapp shopapi`. Így létrehozom az új alkalmazást, aminek a neve `shopapi` lesz.

Futtatom az alapértelmezett Django webszerveret: a `python manage.py runserver` parancs megadásával. A Django belsőleg egy alapértelmezett webszerveret biztosít, ahol elindíthatjuk alkalmazásainkat.

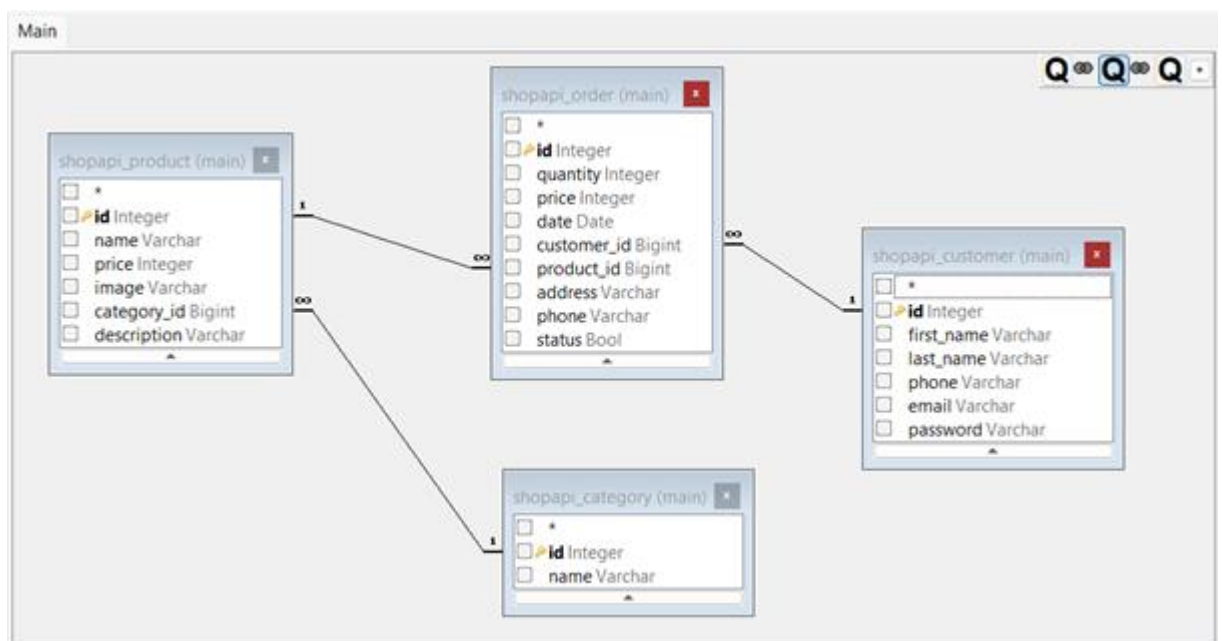
Az alábbi képernyőképen láthatók a szükséges modellek, amelyeket létre kell hoznunk. Ezek a modellek olyan táblák, amelyek az SQLite adatbázisban lesznek tárolva.



Site administration

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change
Users	+ Add	Change
SHOPAPI		
Categorys	+ Add	Change
Customers	+ Add	Change
Orders	+ Add	Change
Products	+ Add	Change

Az adatbázis kapcsolati ábrája:



Az adatbázis kapcsolati ábrája a következő táblákból áll: kategóriák, ügyfelek, rendelések és termékek. A kategóriák tábla tartalmazza az összes termékkategóriát, amelyeket a vásárlók böngészhetnek. Az ügyfelek tábla tartalmazza az összes regisztrált felhasználót, akik rendeléseket tehetnek a webáruházban. A rendelések tábla rögzíti az egyes ügyfelek által leadott rendeléseket, beleértve a termékek mennyiségét és az árakat. A termékek tábla tartalmazza az összes eladó terméket, amelyeket a vásárlók megvásárolhatnak. Ezek a táblák kapcsolatban állnak egymással elsődleges kulcsok és idegen kulcsok segítségével, hogy lehetővé tegyék az adatok egyszerű és hatékony lekérdezését és kezelését.

Felhasználói dokumentáció:

A webshop egy webböngészőben futtatható alkalmazás. Nem szükséges semmit sem telepíteni a futtatásához. A működéséhez csupán aktív internet kapcsolatra, illetve egy eszközre van szükség, amin van webböngésző.

A program bemutatása:

Regisztráció/Bejelentkezés: Ez a két gomb a felső sorban található jobb oldalt. Regisztráció során meg kell adni bizonyos adatainkat, fontos, hogy valódi, hivatalos adatokat legyenek megadva. Belépni egy fiókba csak sikeres regisztráció után lehet.

Kijelentkezés: Ez a funkció csak bejelentkezés után látható, ott, ahol előtte a bejelentkezés gomb volt látható. A kijelentkezés gombra kattintva a rendszer kilépteti a felhasználót.

Szűrés: Szűrni a bal oldalon lévő menüben lehet.

Néhány kihívás, érdekességek

```
api > views > home.py > Index > post
from django.shortcuts import render, redirect , HttpResponseRedirect
from shopapi.models.product import Product
from shopapi.models.category import Category
from django.views import View
# Create your views here.
class Index(View):

    def post(self , request):
        product = request.POST.get('product')
        remove = request.POST.get('remove')
        cart = request.session.get('cart')
        if cart:
            quantity = cart.get(product)
            if quantity:
                if remove:
                    if quantity<=1:
                        cart.pop(product)
                    else:
                        cart[product] = quantity-1
                else:
                    cart[product] = quantity+1
            else:
                cart[product] = 1
        else:
            cart = {}
            cart[product] = 1

        request.session['cart'] = cart
        print('cart' , request.session['cart'])
        return redirect('homepage')

    def get(self , request):
        return HttpResponseRedirect(f'/shopapi{request.get_full_path()[1:]}')

def shopapi(request):
    cart = request.session.get('cart')
    if not cart:
        request.session['cart'] = {}
    products = None
    categories = Category.get_all_categories()
    categoryID = request.GET.get('category')
    if categoryID:
        products = Product.get_all_products_by_categoryid(categoryID)
    else:
        products = Product.get_all_products();

    data = {}
    data['products'] = products
    data['categories'] = categories

    print('Te vagy:' , request.session.get('email'))

    return render(request , 'index.html' , data)
```

Ez a kódrészlet érdekes és kihívást jelentő volt több szempontból is:

1. **Felhasználói interakció kezelése:** A 'post' metódusban a felhasználói kattintásokra válaszolva kezeli a kosárba helyezést és eltávolítást, ami egy fontos felhasználói interakció. Ez a részletezett működés lehetővé teszi a vásárlók számára, hogy könnyen módosítsák a kosarukat.
2. **Adatkezelés és adatbázis-műveletek:** A kódban szereplő adatkezelés és adatbázis-műveletek (például termékek és kategóriák lekérdezése) fontos fejlesztői kihívást jelentenek. Ezek az operációk hatékonyan történnek a Django ORM használatával.
3. **Routolás és átirányítások kezelése:** A 'get' metódus a routolás és átirányítások kezeléséért felelős, amelyek biztosítják, hogy a felhasználó megfelelő oldalra irányítva legyen a különböző felhasználói műveletek során.
4. **Kosárkezelés:** Az adatoknak a felhasználói sessionben való tárolása és kezelése (például kosárba helyezett termékek) további kihívást jelentett. A megfelelő kezelés biztosítja, hogy a felhasználói élmény zökkenőmentes legyen, és a kosár tartalma ne vesszen el a munkamenetek között.

A következő frontend kódrészlet szintén érdekes volt számunkra:

```
<div class="card-footer p-0 no-gutters">
    {% if product|is_in_cart:request.session.cart%}
    <div class="row no-gutters">
        <form action="/#{{product.id}}" class="col-2" method="post">
            {% csrf_token %}
            <input hidden type="text" name="product" value="{{product.id}}">
            <input hidden type="text" name="remove" value='True'>
            <input type="submit" value=" - " class="btn btn-block btn-secondary">
        </form>
        <div class="text-center col">{{product|cart_quantity:request.session.cart}} db</div>
        <form action="/#{{product.id}}" class="col-2" method="post">
            {% csrf_token %}
            <input hidden type="text" name="product" value="{{product.id}}">
            <input type="submit" value=" + " class="btn btn-block btn-secondary">
        </form>
    </div>
    {% else %}
    <form action="/#{{product.id}}" method="POST" class="btn-block">
        {% csrf_token %}
        <input hidden type="text" name="product" value="{{product.id}}">
        <input type="submit" class="float-right btn btn-light border form-control" value="Kosárba">
    </form>
    {% endif %}
</div>
```

Ez a kódrészlet egy termék kártyájának láblécét határozza meg egy webshopban:

1. Dinamikus működés: A kód lehetővé teszi a felhasználók számára, hogy dinamikusan módosítsák a termék mennyiségét a kosárban, vagy akár hozzáadják azt a kosárhoz vagy eltávolítsák belőle, anélkül, hogy frissítenék az oldalt. Ez a felhasználói élményt javítja, és könnyebbé teszi a vásárlást.

2. Kosárkezelés: A kód figyelembe veszi a már kosárban lévő termékeket, és lehetővé teszi a felhasználók számára, hogy növeljék vagy csökkentsék a termék mennyiségét a kosárban. Ez segít a felhasználóknak abban, hogy könnyebben kezeljék a kosár tartalmát, anélkül, hogy elnavigálnának a kosár oldalra.

3. Form validáció: A kód használ CSRF védelmet a biztonság érdekében, amikor az űrlapokat elküldik. Ez segít megvédeni az alkalmazást az általános biztonsági fenyegetések ellen, mint például a Cross-Site Request Forgery (CSRF).

Teszt dokumentáció

Az oldal le lett tesztelve háromfajta böngészővel:

Firefox

Google Chrome

Microsoft Edge

Tapasztalat:

Számottevő különbség nem volt tapasztalható. A tesztelésre főként a környezetünkben élő potenciális felhasználókat kértük fel. A többség szerint az oldal átlátható és könnyen használható volt. A funkciók használata során egy fő problémába ütköztek, a navigációs sáv nem volt rögzítve az oldal tetején. Ezt a problémát a Bootstrap keretrendszer által biztosított 'sticky-top' osztály használatával oldottuk meg. Ez az osztály a Bootstrap CSS része, és automatikusan rögzíti a navigációs sávot az oldal tetejéhez, így az mindig látható marad, amikor az oldalt görgetik.

A formok ellenőrzése sokat segített a használatban, mert amikor megpróbáltak nem oda illő adatot felvinni, akkor a rendszer jelezte. A telefonszám bekérésénél például először elfogadta a betű karaktereket is.

A következő kódrészlet megoldotta a problémát:

```
elif not str(customer.phone).isdigit():
    error_message = 'A telefonszám csak számot tartalmazhat!'
```

A fejlesztéssel párhuzamosan történt a gépelési vagy írásjel hibák megtalálása, javítása.

Az autentikáció sikeres tesztelése után megbizonyosodtunk arról, hogy a felhasználók sikeresen be tudnak jelentkezni és kijelentkezni a webalkalmazásban, valamint a middleware megfelelően irányítja át őket a bejelentkezési oldalra hibás adatok esetén. A tesztek eredményei alapján megbízhatóan működik az autentikációs rendszer, és a felhasználók biztonságosan használhatják az alkalmazást.

Fejlesztési lehetőségek

Íme, néhány jövőbeli fejlesztési lehetőség, amelyekkel tovább fejlesztenénk az alkalmazást:

1. Keresés megvalósítása: Készítenénk egy keresőmezőt, amely lehetővé teszi a felhasználók számára, hogy gyorsan és könnyen megtalálják az általuk keresett termékeket. Ehhez használhatunk egy keresőmotort, amely lehetővé teszi az adatbázisban való keresést.

2. Profil oldal: Készítenénk egy profil oldalt, ahol a felhasználók szerkeszthetik az adatokat, például jelszómódosítás. Ez lehetővé teszi számukra, hogy könnyen kezeljék fiókjukat.

3. Az oldal kinézetének javítása: Frissítenénk az oldal designját és elrendezését, hogy vonzóvá és felhasználóbarátabbá tegye az élményt.

4. Bankkártyás fizetési lehetőség: Integrálni egy bankkártyás fizetési eljárást az oldaladra, amely lehetővé teszi a felhasználók számára, hogy közvetlenül az oldalon keresztül fizethessenek. Ehhez szükség lehet egy megbízható fizetési átjáróra és SSL tanúsítványra az adatbiztonság biztosítása érdekében.

Irodalomjegyzék

A vizsgaremekben használt források:

- <https://www.w3schools.com/>
- <https://infojegyzet.hu/webszerkesztes/dokumentacio/>
- <https://www.pendragon-fb.eu/>
- <https://docs.djangoproject.com/en/5.0/>
- <https://www.youtube.com/@nemestamas>