



ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

ФАКУЛТЕТ „КОМПЮТЪРНИ СИСТЕМИ И ТЕХНОЛИГИИ”

ДИПЛОМНА РАБОТА

**Проектиране и разработване на web базирана система
за поръчки и плащания в заведения**

Научен ръководител:

гл. ас. д-р инж.

Камелия Райнова

Дипломант:

Борис Кирилов Костадинов,

КСИ, Ф.№ :121221157

София, 2025

Утвърдено дипломно задание

София, 2025

Съдържание

УВОД.....	5
ГЛАВА 1. ПОСТАНОВКА НА ДИПЛОМНАТА РАБОТА, ЦЕЛИ, ЗАДАЧИ И ФУНКЦИОНАЛНОСТИ	7
1.1 Постановка на задачата.....	7
1.2 Цели на дипломната работа.....	7
1.3 Основни функционалности на SmartBar.....	8
ГЛАВА 2: ПРОЕКТИРАНЕ НА СИСТЕМАТА SMARTBAR	11
2.1 Общ преглед на архитектурата	11
2.2 Избор на технологии.....	14
2.3 Структура на приложението.....	19
2.4 Проектиране на базата от данни	19
2.5 Поток на данни и логика.....	21
2.6 Предимства на архитектурата	22
2.7 Възможности за надграждане	23
ГЛАВА 3: РЕАЛИЗАЦИЯ НА УЕБ ПРИЛОЖЕНИЕТО	25
3.1 Общ преглед.....	25
3.2 Инициализиране и конфигурация.....	25
3.3 Регистрация и вход.....	25
3.4 Управление на продукти и категории.....	26
3.6 Създаване на поръчка	27
3.7 Онлайн плащане чрез PayPal	28
3.8 Генериране на фактура.....	29
3.9 Административен панел	29

3.10	Имейл известия	29
3.11	Визуален интерфейс.....	30
3.12	Тестване.....	30
ГЛАВА 4: РЪКОВОДСТВО ЗА РАБОТА С ПРИЛОЖЕНИЕТО		31
4.1	Начална страница (Home)	31
4.2	Регистрация и вход.....	32
4.3	Преглед на продукти	35
4.4	Количка	36
4.5	Поръчка	37
4.6	Плащане с PayPal	38
4.7	Преглед на завършена поръчка	40
4.8	История на поръчките.....	41
4.9	Административен панел (Django Admin).....	42
4.10	Работа с имейли.....	43
ГЛАВА 5: ЗАКЛЮЧЕНИЕ		45
ГЛАВА 6: ИЗПОЛЗВАНИ ИЗТОЧНИЦИ И ПРИЛОЖЕНИЯ		47
Приложение 1: Използвани фигури.....		47
Приложение 2: Списък на използвани литературни и онлайн източници.....		48

УВОД

В условията на съвременния дигитален свят, все повече заведения за хранене и развлечение – като ресторанти, барове и кафенета – се стремят да модернизират и автоматизират процесите на обслужване на клиентите. Това включва не само подобряване на вътрешната организация на работа, но и предоставяне на интуитивни, бързи и сигурни начини за поръчване и заплащане. В този контекст, уеб базираните системи за поръчки и онлайн плащания се превръщат в необходимост за всяко заведение, което цели да отговори на очакванията на модерните потребители.

Настоящата дипломна работа има за цел да проектира и реализира уеб приложение с наименование SmartBar, което предоставя цялостна система за дигитално обслужване в заведения – от избора и поръчката на продукти, до извършване на плащания през интернет. Системата е насочена към крайни потребители (клиенти на заведението), както и към администраторите и персонала, които управляват менюто, продуктите, поръчките и транзакциите.

По време на разработката на проекта бяха извършени редица инженерни дейности – анализ на изискванията, проектиране на архитектурата, изграждане на базата данни, създаване на интуитивен потребителски интерфейс, и интегриране на онлайн платежен модул (PayPal). Уеб приложението е разработено с помощта на съвременни технологии, сред които Python, Django, HTML, CSS, Bootstrap и JavaScript. За осигуряване на достъпност и надеждност, приложението е подготвено за хостване в облачна среда с използване на AWS и възможност за интеграция с имейл услуги и база данни.

Основната цел на проекта е да предложи решение, което:

- улеснява клиентите при заявяване на поръчки;
- автоматизира процеса на обработка и плащане;
- намалява натоварването на обслужващия персонал;
- подобрява цялостното потребителско изживяване в заведението.

В рамките на дипломната работа се представят всички основни фази от разработката на системата SmartBar – от формулирането на изискванията, през проектирането и реализирането ѝ, до извършване на тестове и анализ на резултатите. Освен това е включено ръководство за работа с приложението, както и препоръки за бъдещо разширяване на функционалността.

С настоящия проект се цели демонстрация на умения за проектиране, програмиране, интегриране на технологии и създаване на цялостно функциониращ софтуерен продукт, ориентиран към реален бизнес казус в сферата на ресторантьорството.

ГЛАВА 1. ПОСТАНОВКА НА ДИПЛОМНАТА РАБОТА, ЦЕЛИ, ЗАДАЧИ И ФУНКЦИОНАЛНОСТИ

1.1 Постановка на задачата

В последните години дигитализацията навлезе интензивно във всички сфери на обществения живот, включително и в ресторантьорството. Технологиите трансформират начина, по който клиентите взаимодействат със заведенията – от разглеждане на меню, през поръчване, до извършване на плащания. Стремешът към бързо, ефективно и безконтактно обслужване породил необходимостта от внедряване на цялостни уеб базирани решения, които улесняват както клиентите, така и служителите.

Проблемите, които традиционната организация на обслужване често създава – като забавяне при обслужване, липса на проследимост на поръчките, затруднения в комуникацията между сервитьори и кухня – налагат нуждата от софтуерни системи, които да автоматизират ключови процеси. Допълнително, в условията на пандемии, необходимостта от безконтактни методи на плащане и самообслужване се превърнала от удобство в изискване.

Заданието на дипломната работа е да се проектира, разработи и тества цялостно уеб базирано приложение за поръчки и онлайн разплащания в заведения от тип ресторант или бар. Разработената система трябва да предлага функционалност за клиентски поръчки, визуализация на менюто, добавяне в количка, плащане онлайн, както и административен панел за управление на продукти, категории и поръчки. Основната идея е чрез тази система да се подобри клиентското изживяване и да се намалят административните усилия от страна на персонала.

1.2 Цели на дипломната работа

Основната цел на дипломната работа е изграждането на напълно функционираща уеб базирана система за заведения, която да изпълнява следните функции:

- Предоставяне на дигитален интерфейс за преглед и избор на продукти от менюто на заведението.

- Възможност за регистриране и влизане на потребители.
- Обработка на поръчки с детайлна информация за избраните артикули и вариации (например размер, вкус, добавки).
- Онлайн плащане чрез интеграция с платформа за електронни разплащания (PayPal).
- Достъп до административен панел за управление на продуктите, категориите и постъпилите поръчки.
- Изпращане на автоматични имейли при регистрация, направена поръчка и плащане.
- Подобряване на процеса на обслужване чрез автоматизация и дигитализация.

Допълнителните цели включват:

- Сигурно съхранение на данни чрез използване на база данни.
- Добро потребителско изживяване (UX) чрез съвременен дизайн с помощта на Bootstrap.
- Модулна архитектура, позволяваща лесно разширение в бъдеще (напр. интеграция с мобилно приложение или QR меню).
- Хостване на приложението в облачна среда (напр. AWS или друг публичен хостинг).

1.3 Основни функционалности на SmartBar

SmartBar е уеб приложение, ориентирано към две основни групи потребители: клиенти на заведението и администратори/служители. По-долу са описани всички основни модули и функционалности, които системата предлага:

1. Регистрация и автентикация на потребители

- Възможност за създаване на потребителски акаунт чрез регистрационна форма.
- Вход и изход от системата с валидни данни.
- Потвърждение на регистрацията чрез имейл (активираща връзка).

- Възможност за възстановяване на забравена парола.

2. Преглед на менюто

- Менюто е разделено в категории (напр. Напитки, Храни, Десерти).
- Визуализация на продукти със снимка, цена и кратко описание.
- Поддържане на вариации на продукти (например: размер на напитка, добавки към храна).

3. Количка и преглед на поръчка

- Добавяне на продукти в количката с избрани вариации.
- Преглед на количката с възможност за корекция (увеличаване/намалване на количество, изтриване).
- Автоматично изчисляване на междинна сума и крайна цена.

4. Онлайн плащания

- Интеграция с PayPal за извършване на реални онлайн плащания.
- Преобразуване на валута (напр. EUR).
- Записване на статус на транзакцията (успешна, неуспешна, чакаща).
- Създаване на поръчка само след успешно плащане.

5. Административен панел

- Създаване, редакция и изтриване на продукти и категории.
- Преглед на всички поръчки с филтри по статус.
- Промяна на статус на поръчка (напр. „Приета“, „Завършена“, „Отказана“).
- Достъп до списъци с потребители и техните поръчки.

6. Имейл известия

- Изпращане на имейл при:

- Регистрация на нов потребител
- Потвърждение на поръчка
- Завършено плащане
- Имейл шаблони с автоматично попълнени данни (име, поръчка, дата и т.н.)

7. Фактуриране и преглед на завършена поръчка

- Потребителят може да прегледа направената поръчка.
- Създава се автоматично „фактура“ с всички детайли по транзакцията.
- Възможност за принтиране на документа директно от интерфейса.

8. Допълнителни възможности

- Отзивчив (responsive) дизайн — работи на компютър, таблет и телефон.
- Страници за „За нас“, „Контакт“ и друга информационна структура.
- Визуални съобщения и предупреждения при грешки, празна количка и др.

Тези функционалности превръщат SmartBag в завършена система, приложима в реална бизнес среда, и създават основа за по-нататъшно развитие – като мобилна версия, QR поръчки от маса, интеграция с POS терминали и др.

ГЛАВА 2: ПРОЕКТИРАНЕ НА СИСТЕМАТА SMARTBAR

2.1 Общ преглед на архитектурата

Уеб базираната система SmartBar е изградена върху многослойна архитектура, използваща парадигмата Model–View–Template (MVT), която е в основата на фреймуърка Django. MVT е концептуално близка до популярния Model–View–Controller (MVC) модел, като разликата е, че в Django View изпълнява и ролята на контролер, а шаблоните (Template) отговарят само за визуализацията на данните.

Тази архитектура позволява ясно разделение между бизнес логиката, обработката на заявки, визуалното представяне и съхранението на данни. Това води до по-добра модулност на кода, по-лесна поддръжка, по-бързо развитие на функционалности и по-висока устойчивост на грешки.

Компонентите на архитектурата:

Model (Модел – слой за достъп до данни)

Моделите в Django представляват Python класове, които дефинират структурата на данните и връзките между тях. Те се мапват директно към таблици в базата от данни чрез Object-Relational Mapping (ORM), което позволява работа с данните чрез Python обекти вместо чрез SQL заявки. Всеки модел в SmartBar описва конкретна логическа единица, например:

- Product – представя артикул от менюто;
- Order – съхранява информация за направена поръчка;
- CartItem – ред от количката;
- Payment – детайли за извършено плащане;
- Account – персонализиран потребителски модел.

ORM системата на Django управлява създаването на таблици, връзките между обектите, валидацията и миграциите на базата от данни.

View (Изглед – слой за бизнес логика и контрол на потока)

View компонентът обработва входящите HTTP заявки, извлича нужната информация от модела, обработва логиката и връща отговор под формата на HTML или JSON. Всеки view е Python функция или клас, която управлява дадена част от бизнес логиката – например:

- Зареждане на начална страница;
- Добавяне на продукт в количката;
- Създаване на поръчка;
- Потвърждение на плащане;
- Административни действия (редакция на продукти, статуси и др.).

Чрез view functions и URL routing заявките се насочват към подходящата логика. Така се гарантира, че за всеки потребителски или сървърен сценарий съществува централизирано място за обработка.

Template (Шаблон – слой за визуализация)

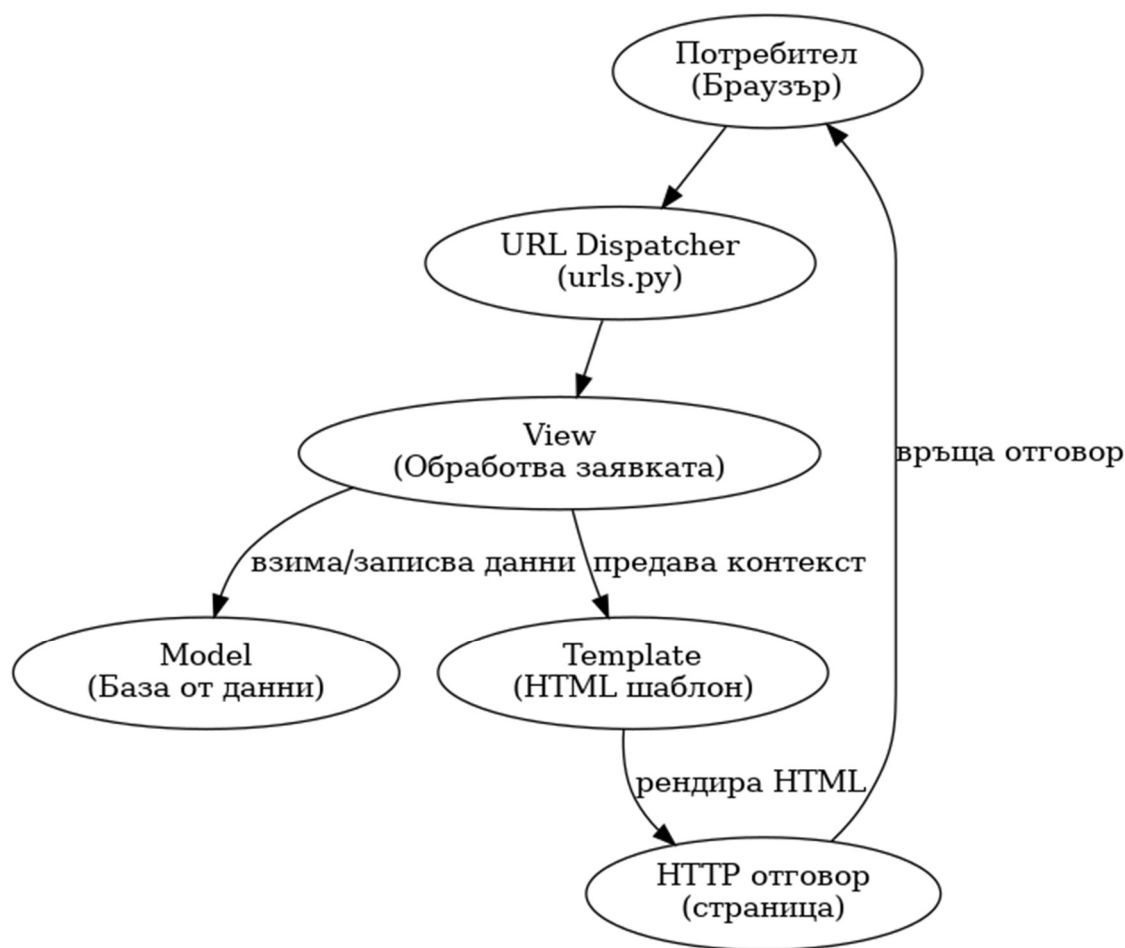
Шаблоните в Django използват HTML и вградения Django Template Language (DTL) – език, който позволява включване на динамични данни в шаблона чрез конструкции като `{% %}` (логика) и `{{ }}` (променливи). Те не съдържат бизнес логика, а само контрол на визуалното изобразяване на данни.

В SmartBar шаблоните дефинират как изглеждат страниците – начална страница, меню, количка, формуляри за вход и регистрация, детайли за поръчка и други. Използва се шаблонно наследяване (`{% extends %}`), което позволява многократна употреба на основна структура (напр. header, navbar, footer) и гарантира визуална консистентност.

Предимства на MVT архитектурата в SmartBar

1. **Разделение на отговорностите** – всеки слой е специализиран за конкретна задача: съхранение, логика или визуализация. Това улеснява отстраняването на грешки и разширяването на системата.
2. **Подобрена поддръжка** – различни разработчици могат паралелно да работят по шаблони, изгледи и модели без конфликти.

3. **Висока повторна употреба на компоненти** – един и същ модел може да се използва в различни изгледи, а един шаблон – в множество страници.
4. **Сигурност и мащабируемост** – благодарение на вградените в Django механизми (напр. CSRF защита, сесии, валидации) системата е устойчива още от първоначалния етап на разработка.
5. **Лесно внедряване и тестване** – наличието на структуриран код прави модулното тестване и бъдещия деплоймънт по-прозрачен и предвидим.



Архитектурната диаграма, визуализираща потока на заявките в системата SmartBar според MVT модела на Django - Фиг. 1

2.2 Избор на технологии

Изборът на технологии за изграждането на системата SmartBar е направен с оглед на надеждност, поддръжка от общността, сигурност и възможност за бъдещо разширение. Използвани са съвременни инструменти, които са доказани в реални уеб проекти и осигуряват стабилна основа за разработка и внедряване на уеб приложения.

2.2.1 Backend технологии:

Python



Python лого - Фиг.2

Python е интерпретиран, обектно-ориентиран език за програмиране, който се отличава с четимост и изчистен синтаксис. Благодарение на своята популярност и богата екосистема от библиотеки, Python е предпочитан избор при изграждане на уеб платформи, научни изчисления и автоматизация. В контекста на проекта SmartBar, Python служи като основен език за бизнес логиката на приложението, обработка на заявки и връзка с базата от данни.

Django



Django лого - Фиг.3

Django е високонивно уеб приложение фреймуърк, базиран на Python, който следва принципа "Don't Repeat Yourself" (DRY). Той предоставя вградена поддръжка за изграждане на RESTful API, ORM за работа с бази от данни, вграден административен интерфейс, формуляри, автентикация, управление на сесии, CSRF защита и други. Django е особено подходящ за проекти като SmartBar, тъй като ускорява разработката и намалява необходимостта от писане на повтарящ се код.

2.2.2 Frontend технологии:

HTML5



HTML5 лого - Фиг.4

HTML5 се използва за създаване на семантичната структура на уеб страниците. Той дефинира основните елементи от интерфейса – заглавия, параграфи, бутони, формуляри, изображения и други. HTML5 е напълно съвместим с всички съвременни браузъри и представлява гръбнака на клиентската част на SmartBar.

CSS3 и Bootstrap



CSS3 и Bootstrap лого - Фиг.5

CSS3 се използва за оформяне на визуалния вид на уеб страниците – цветове, шрифтове, разположения и анимации. В проекта е използвана и библиотеката Bootstrap, която предлага готови компоненти и класи, чрез които се изграждат адаптивни и мобилно-отзивчиви интерфейси. Това позволява интерфейсът на SmartBar да изглежда добре както на компютри, така и на таблети и мобилни устройства, без нужда от допълнителна стилизация.

JavaScript



JavaScript лого - Фиг.6

JavaScript е език за програмиране, който работи от страната на клиента (в браузъра) и осигурява интерактивност на страниците – динамично обновяване на съдържание, обработка на събития (клик, скрол, въвеждане на данни), както и интеграция с външни

скриптове. В системата SmartBar JavaScript се използва основно за визуализиране на PayPal бутона, както и за изпращане на AJAX заявки при финализиране на поръчка.

2.2.3 База от данни:

SQLite



SQLite лого - Фиг.7

SQLite е вградена релационна база от данни, която не изисква отделен сървър или конфигурация. Подходяща е за разработка и тестване, поради своята лекота, компактност и лесна настройка. Django работи с SQLite по подразбиране, което позволява бърз старт на проекта. В бъдеще, при мащабиране, базата лесно може да бъде заменена с PostgreSQL или друга по-мощна система.

2.2.4 Външни интеграции:

PayPal API



PayPal лого - Фиг.8

За реализацията на онлайн плащания в системата е използвана интеграция с PayPal API. Чрез официалния JavaScript SDK, приложението създава реална поръчка, която потребителят може да заплати чрез своя PayPal акаунт. По време на разработката е използвана тестовата Sandbox среда, а при финализиране – реална среда с поддръжка на плащания в евро. Интеграцията е реализирана с високо ниво на сигурност чрез токени и JSON заявка от клиента към сървъра.

SMTP (Gmail)

За изпращането на автоматични имейли от приложението (например при регистрация, потвърждение на поръчка или транзакция), е използван Gmail SMTP сървър. Django предоставя клас EmailMessage, чрез който лесно се изпращат текстови или HTML съобщения. SMTP връзката е защитена чрез TLS и удостоверяване чрез потребител и парола, съхранени безопасно чрез конфигурационен файл.

2.2.5 Хостинг и бъдеща поддръжка:

- Облачна инфраструктура и хостинг
По време на разработка приложението се хоства локално. То обаче е проектирано така, че може лесно да бъде пренесено към отдалечен сървър или облачна платформа. Подходящи варианти включват:
 - PythonAnywhere – платформа, специално създадена за Django приложения, подходяща за малки до средни проекти.
 - Heroku – удобен за автоматизирани деплой процеси, с безплатен план за малки приложения.
 - AWS EC2 + RDS – по-напреднала конфигурация с възможност за използване на PostgreSQL и вертикално мащабиране.
 - NGINX + Gunicorn/uWSGI – класическа конфигурация за продукционен сървър с висока производителност и контрол.

2.3 Структура на приложението

Приложението е разделено на следните модули (Django apps):

Модул	Описание
accounts	Управлява регистрация, вход и профили на потребителите
store	Съдържа всички продукти, категории и вариации
carts	Управлява количката и поръчките в нея
orders	Отговаря за създаване на поръчки, плащания и фактуриране
category	Обособява категории за продуктите
templates	Всички HTML файлове с визуализация на интерфейса

2.4 Проектиране на базата от данни

Базата от данни е проектирана така, че да осигури максимална нормализация, логическа последователност и възможност за разширение. Ето описанието на основните модели:

Account (User):

- Име, фамилия, имейл, телефон, парола (шифрована)
- Дата на създаване
- Активиращ флаг (is_active)
- Роля (по избор: клиент, служител, админ)

UserProfile:

- Връзка с Account (OneToOne)

- Снимка, адрес, град, държава, пощенски код

Category:

- Име на категория (напр. „Напитки“, „Храна“)
- Снимка и слаг за URL достъп

Product:

- Име, описание, цена, наличност, снимка
- Връзка към категория
- Поле „is_available“

Variation:

- Продукт
- Тип вариация (напр. „Размер“)
- Стойност (напр. „Малка“, „Голяма“)

Cart и CartItem:

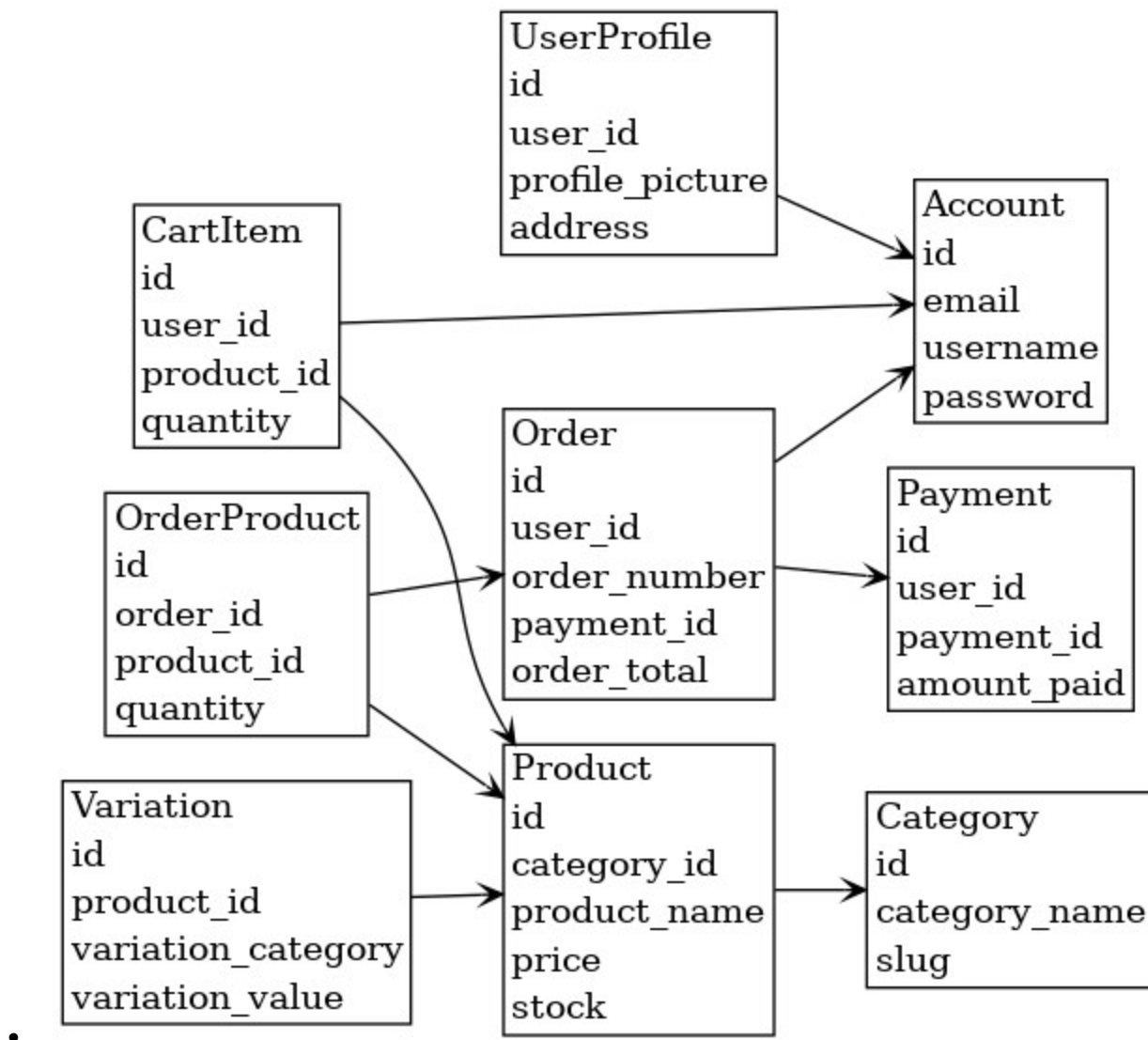
- Временна „кошница“, идентифицирана чрез сесия
- CartItem съдържа продукт, количество, вариации, сума

Order и OrderProduct:

- Информация за поръчката: потребител, адрес, сума, статус
- Свързани продукти, количество, цена, вариации
- Връзка с транзакцията (платежен запис)

Payment:

- ID на транзакция (от PayPal)
- Статус, метод, обща сума, дата



ER диаграма на базата данни - Фиг.9

2.5 Поток на данни и логика

1. Потребителят влиза в системата или се регистрира.
2. Разглежда продуктите и избира конкретни артикули с техните вариации.
3. Продуктите се добавят в количката, съхранявани във временно състояние.
4. При потвърждение, се генерира поръчка и се извиква PayPal API за заплащане.
5. След потвърдено плащане се записва транзакцията и поръчката става активна.

6. Администраторът вижда новата поръчка в панела и може да промени статуса.
7. Потребителят получава имейл с потвърждение и фактура.

2.6 Предимства на архитектурата

Избраната архитектура на системата SmartBar, базирана на модела Model–View–Template и модулната организация на Django, предоставя редица значими предимства както при разработката, така и при поддръжката и бъдещото развитие на системата:

- **Модулност и разделение на отговорностите**
Всеки компонент от функционалността е организиран в самостоятелно Django приложение (app), например accounts, store, orders, carts. Това позволява изолирана разработка и тестване, както и лесна повторна употреба на логика. При необходимост от добавяне на нова функционалност, тя може да бъде имплементирана като отделно приложение, без да се нарушава съществуващата структура.
- **Вградена сигурност на фреймуърка**
Django осигурява високо ниво на защита срещу често срещани уеб уязвимости, без нужда от допълнителна конфигурация. Сред тях са:
 - CSRF защита (Cross Site Request Forgery)
 - XSS защита (Cross Site Scripting) чрез автоматично ескейпване на съдържание в шаблоните
 - SQL инжекции – предотвратени чрез ORM
 - Управление на потребителски сесии, хеширане на пароли и контрол на достъпа
- **Гъвкавост и лесна поддръжка**
Системата е проектирана така, че да позволява добавяне на нови модели, изгледи, форми и маршрути без нужда от пренаписване на съществуващ код. Това прави поддръжката удобна и минимизира риска от грешки при бъдещи промени.

- **Скалируемост и бъдещо разширение**
Използваната архитектура и технологичен стек са подходящи както за малки, така и за по-мощни инсталации. Системата лесно може да бъде надградена с REST API, мобилни клиенти, допълнителни интеграции и механизми за анализ на данни. Това я прави подходяща не само за демонстрационна цел, но и за реално внедряване и развитие в търговска среда.

2.7 Възможности за надграждане

Благодарение на използваната архитектура, системата SmartBar е изградена с отворен подход, който позволява разширяване с минимални усилия. Сред възможните посоки за надграждане са:

- **Интеграция на REST API чрез Django REST Framework (DRF)**
Добавянето на RESTful интерфейс позволява външни системи (например мобилно приложение или POS терминали) да комуникират директно с уеб сървър, използвайки JSON. DRF предлага сериализация на данни, контрол на достъпа и лесно изграждане на API endpoints.
- **Разработка на мобилно приложение (Android/iOS)**
Системата може да бъде разширена с мобилна клиентска част, разработена с помощта на React Native, Flutter или Kotlin/Swift. Това ще позволи на клиентите да правят поръчки директно от своите устройства, а на персонала – да следи и обработва заявки в реално време.
- **QR код поръчки**
Генериране на индивидуален QR код за всяка маса в заведението, който при сканиране отвежда клиента към интерфейс за самостоятелна поръчка. Това позволява напълно безконтактно обслужване и елиминира необходимостта от сервитьор в началния етап на заявка.
- **Разширяване на поддържаните методи за плащане**
Освен PayPal, системата може да бъде интегрирана и с други платежни услуги като

Stripe, Revolut Pay, Apple Pay или банкови карти чрез платформите на iCard и BORICA. Това ще повиши удобството за клиентите и ще отвори допълнителни бизнес възможности.

- **Панел за персонала (бармани, готвачи)**
Специализиран административен панел за вътрешна употреба, чрез който персоналът може да вижда постъпващи поръчки в реално време, да маркира статуси (в процес, изпълнена, отказана) и да следи натовареността по сектори.
- **Аналитика и отчетност**
Добавяне на функционалности за статистика – най-поръчвани продукти, средна стойност на поръчка, оборот по периоди, активни потребители. Това би било особено полезно за собственици и мениджъри, които искат да вземат информирани бизнес решения.
- **Многоезична поддръжка и локализация**
Интерфейсът на системата може да бъде преведен на множество езици чрез механизма на Django i18n, което прави платформата подходяща за използване от международни заведения и клиенти.

Изграждането на SmartBar върху ясна, модулна и разширяема архитектура гарантира, че проектът не само отговаря на текущите цели и функционални изисквания, но е и устойчив във времето. Това осигурява стабилна основа за надграждане, внедряване в реална среда и адаптиране към бъдещи технологични и бизнес нужди.

ГЛАВА 3: РЕАЛИЗАЦИЯ НА УЕБ ПРИЛОЖЕНИЕТО

3.1 Общ преглед

Уеб приложението SmartBar е разработено с помощта на фреймуърка **Django** – мощен инструмент за изграждане на уеб системи чрез езика **Python**. Django е избран заради високата си надеждност, вградените функции за сигурност, лесната работа с бази от данни чрез ORM и възможността за бързо изграждане на административен интерфейс.

Приложението е организирано по модулен начин: всяка основна функционалност (потребители, продукти, количка, поръчки и категории) е разделена в отделно Django приложение, което позволява ясно структуриране на кода и лесна поддръжка.

3.2 Инициализиране и конфигурация

3.2.1 Създаване на проект

Проектът е създаден с командите:

```
django-admin startproject greatbar
cd greatbar
python -m venv env
source env/Scripts/activate # (Windows)
python manage.py runserver
```

3.2.2 Django приложения

За всяка логическа част на системата е създадено отделно приложение:

```
python manage.py startapp accounts
python manage.py startapp store
python manage.py startapp carts
python manage.py startapp orders
python manage.py startapp category
```

3.3 Регистрация и вход

Регистрацията на потребители е реализирана чрез персонализиран модел Account, който наследява Django класовете AbstractBaseUser и BaseUserManager. Потребителите се

регистрират с име, имейл, телефон, парола и след потвърждение по имейл — активират своя акаунт.

Функцията `register(request)` обработва формата за регистрация, като при успех се създава потребител и се изпраща активационен линк чрез SMTP.

```
def register(request):
    if request.method == 'POST':
        # проверка за съществуващ потребител
        # създаване на акаунт чрез Account.objects.create_user(...)
        # изпращане на активационен имейл
```

Изпращането на имейла се извършва чрез:

```
email = EmailMessage(subject, body, to=[user.email])
email.send()
```

3.4 Управление на продукти и категории

Продуктите са организирани в категории (напр. Храни, Напитки) чрез модел `Category`, който позволява лесна навигация и филтриране.

```
class Category(models.Model):
    category_name = models.CharField(max_length=50)
    slug = models.SlugField(unique=True)
```

Моделът `Product` съдържа информация за името, описанието, цената, количеството, категорията и изображението на продукта. Продуктите имат SEO-приятелски URL адреси чрез `slug` полета.

```
class Product(models.Model):
    product_name = models.CharField(max_length=100)
    category = models.ForeignKey(Category, on_delete=models.CASCADE)
    price = models.FloatField()
    stock = models.IntegerField()
    images = models.ImageField(upload_to='photos/products')
```

При нужда от вариации (например различен размер на кафе), се използва допълнителен модел `Variation`, който съхранява категорията на вариацията и конкретната стойност.

```
class Variation(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    variation_category = models.CharField(max_length=100)
    variation_value = models.CharField(max_length=100)
```

3.5 Количка и добавяне на продукти

Количката е съществена част от системата и позволява на потребителя да избира и съхранява продукти преди финализиране на поръчка.

Технически се използват два модела:

- `Cart`: свързва се със сесия или с потребителски профил.
- `CartItem`: съдържа конкретен продукт, количество и вариации.

```
class Cart(models.Model):
    cart_id = models.CharField(max_length=250)

class CartItem(models.Model):
    product = models.ForeignKey(Product, on_delete=models.CASCADE)
    quantity = models.IntegerField()
    cart = models.ForeignKey(Cart, on_delete=models.CASCADE)
```

Когато потребителят натисне „Добави в количка“, системата проверява дали този продукт вече съществува със същите вариации.

Ако да — количеството се увеличава.

Ако не — се създава нов запис.

```
def add_cart(request, product_id):
    product = Product.objects.get(id=product_id)
    # логика за проверка на съществуващ item
    # създаване или увеличаване на количество
```

3.6 Създаване на поръчка

При преминаване към плащане, потребителят попълва форма с адрес и информация за доставка. Данните се записват в модел Order, където се съхраняват също IP адрес, дата на създаване, сума и номер на поръчка.

Генерирането на уникален номер за поръчка става чрез комбинация от дата и ID:

```
data = Order()
data.first_name = form.cleaned_data['first_name']
data.order_total = grand_total
data.save()
order_number = current_date + str(data.id)
```

3.7 Онлайн плащане чрез PayPal

За реализиране на онлайн разплащанията е използван PayPal, като интеграцията първоначално е тествана чрез PayPal Sandbox — платформа за разработчици, позволяваща симулиране на плащания без реални средства.

Sandbox среда

- Създаден е разработвачески акаунт
- Генерирани са тестови акаунти: бизнес и клиентски
- Интеграция чрез JavaScript SDK:

```
<script src="https://www.paypal.com/sdk/js?client-id=...&currency=EUR"></script>
```

Преход към реална среда

След тестовите, sandbox идентификаторът се заменя с live client ID и приложението е готово за реални транзакции.

Обработка на плащане

Когато плащането е успешно, се извършва следната логика:

```
def payments(request):
    body = json.loads(request.body)
    payment = Payment.objects.create(...)
    order.payment = payment
    order.is_ordered = True
    order.save()
```

3.8 Генериране на фактура

След успешно плащане потребителят може да види детайли по поръчката във формат на фактура. Използва се шаблонът `order_detail.html`, като данните са показани в табличен вид и е наличен бутон за принтиране чрез JavaScript (`window.print()` върху конкретен `div`).

```
function printInvoice() {  
    window.print();  
}
```

3.9 Административен панел

Чрез Django Admin администраторите могат да управляват:

- Потребители
- Продукти и категории
- Поръчки и статуси

Интерфейсът е конфигуриран така, че да показва нужните полета, филтри и бързо търсене.

```
@admin.register(Product)  
class ProductAdmin(admin.ModelAdmin):  
    list_display = ('product_name', 'category', 'price', 'stock')
```

3.10 Имейл известия

SMTP имейл сървър на Gmail е използван за:

- Изпращане на активационни линкове при регистрация
- Потвърждения за направени поръчки
- Съобщения при промяна на статуси

```
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'  
EMAIL_HOST = 'smtp.gmail.com'  
EMAIL_PORT = 587  
EMAIL_HOST_USER = config('EMAIL_HOST_USER')  
EMAIL_HOST_PASSWORD = config('EMAIL_HOST_PASSWORD')  
EMAIL_USE_TLS = True
```

Изпращане на известие:

```
email = EmailMessage(subject, message, to=[user.email])
email.send()
```

3.11 Визуален интерфейс

Дизайнът е реализиран чрез Bootstrap и отговаря на принципите за адаптивен интерфейс (*responsive design*). Навигацията е интуитивна, страниците са олекотени, а действията са придружени от визуална обратна връзка (успешно добавяне, грешки и др.).

```
<div class="container">
  <div class="row">
    <div class="col-md-4">
      <!-- Карта с продукт -->
    </div>
  </div>
</div>
```

3.12 Тестване

Извършено е ръчно и функционално тестване на:

- Регистрация и вход с невалидни/валидни данни
- Добавяне и премахване от количка
- Финализиране на поръчка
- Проверка на имейл известия
- Проверка на PayPal транзакции чрез sandbox и live режим

ГЛАВА 4: РЪКОВОДСТВО ЗА РАБОТА С ПРИЛОЖЕНИЕТО

Тази глава описва подробно начина, по който се използва системата SmartBar както от обикновени потребители (клиенти), така и от администратори и персонал. Целта е да се даде ясно и систематизирано обяснение на всички интерфейсни елементи и логика на поведение в реална работна среда. Всички действия ще бъдат описани **стъпка по стъпка**, заедно с обяснение на функционалността, навигацията и резултатите.

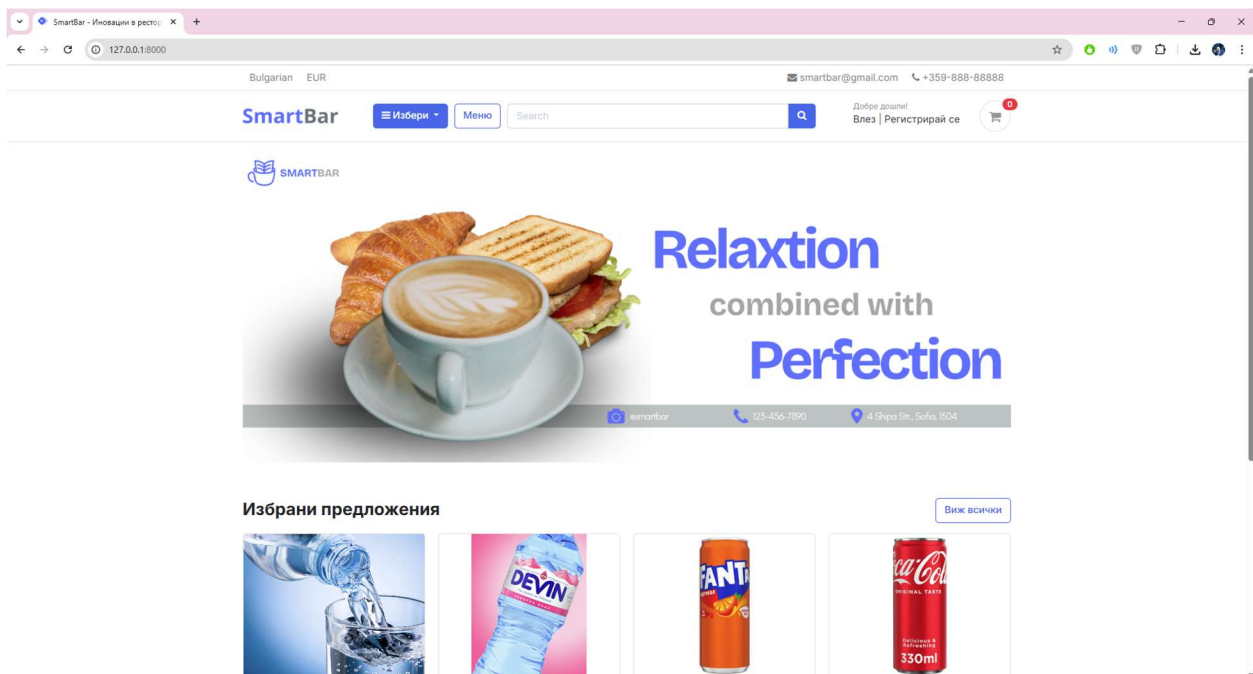
Тази глава ще даде пълно ръководство за работа със SmartBar – от първото посещение на сайта до успешно плащане и администриране. Системата е проектирана с интуитивен интерфейс, сигурна логика и пълна проследимост на действията. Всяка стъпка е съобразена както с очакванията на потребителите, така и с нуждите на служителите и управленския екип.

4.1 Начална страница (Home)

След отваряне на сайта (например <http://127.0.0.1:8000/>), потребителят попада на началната страница, която съдържа:

- Навигационна лента с:
 - Начало
 - Магазин (Меню)
 - Вход / Регистрация
 - Профил (ако е логнат)
 - Количка
- Банер изображение или слайдър с промоционално съдържание
- Препратки към различните категории

От тук потребителят може да се насочи към менюто и да започне да разглежда продукти, дори без да е влязъл в профил.



Начална старница (Home) - Фиг.10

4.2 Регистрация и вход

Регистрация:

1. Натиска се бутон „Регистрация“.
2. Зарежда се форма със следните полета:
 - Име
 - Фамилия
 - Имейл (използва се за вход)
 - Телефонен номер
 - Парола + потвърждение

Регистрирай се

Име	Фамилия
<input type="text" value="Борис"/>	<input type="text" value="Костадинов"/>
Емейл адрес	Телефонен номер
<input type="text" value="testmail@gmail.com"/>	<input type="text" value="0123456789"/>
Създай парола	Повтори паролата
<input type="password" value="....."/>	<input type="password" value="....."/>

Вече имаш профил? [Влез оттук](#)

Регистрирай се - Фиг.11

3. При изпращане на формата:

- В системата се създава нов потребител.
- Изпраща се имейл с активационен линк.
- До активиране на акаунта потребителят не може да влезе.

Изпратихме ви линк за потвърждение на вашия имейл адрес [testmail1001@gmail.com]

Вече потвърден? [Вход](#)

Изпратен емейл - Фиг.12

Вход:

Вход

[Забравена парола?](#)

Вход

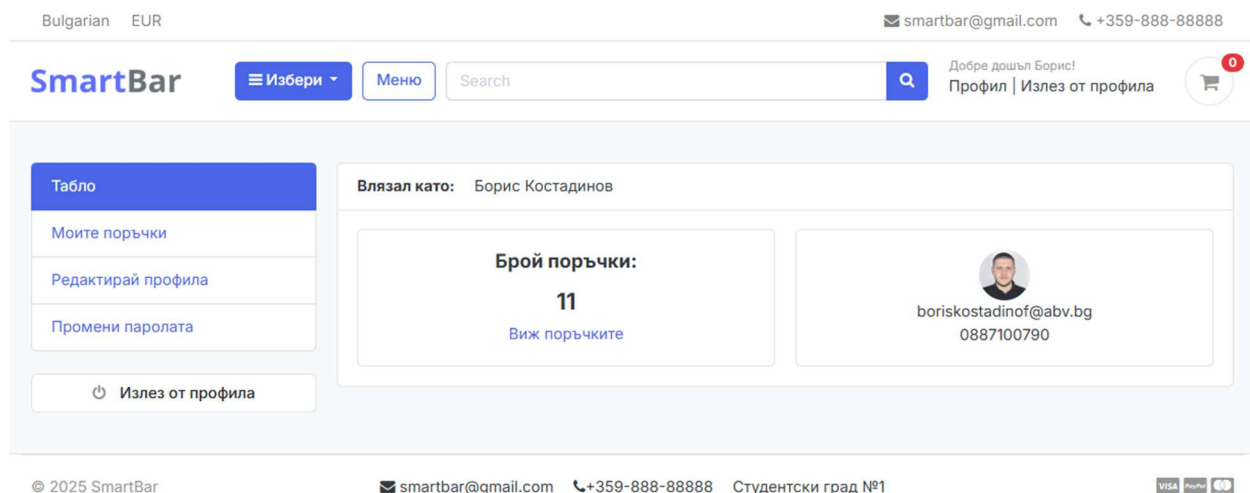
Нямаш профил? [Регистрирай се](#)

Вход - Фиг.13

След активиране:

1. Потребителят въвежда имейл и парола.
2. Системата го идентифицира и пренасочва към началната страница.

3. Навигацията се променя – появява се „Табло“, „Моите поръчки“, „Редактирай профила“, „Промени паролата“, „Излез от профила“



Моят профил - Фиг.14

4.3 Преглед на продукти

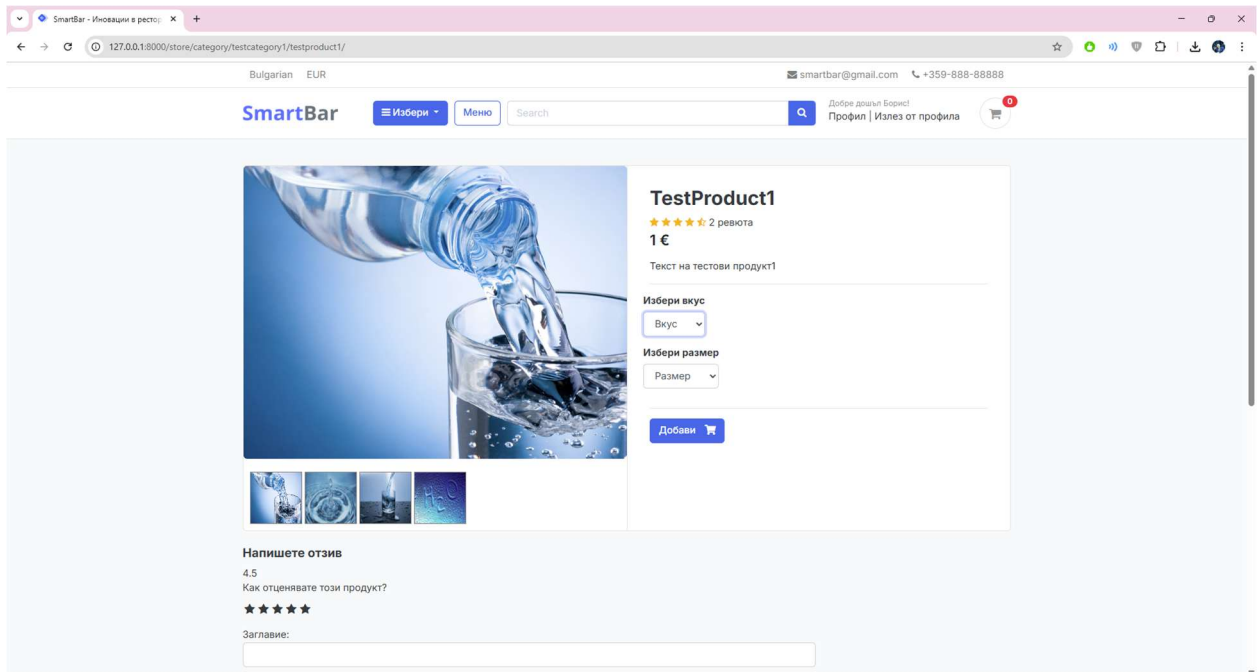
След кликване на „Меню“, се зарежда списък с продукти:

- Продуктите са подредени в мрежа (grid), по категории.
- Всеки продукт има:
 - Снимка
 - Име
 - Цена
 - Бутон „Виж продукта“

Детайли на продукта:

1. След натискане на „Виж продукта“, се зарежда подробна страница.
2. В нея има:
 - Подробно описание

- Възможност за избор на вариации (напр. размер: малък/голям)
- Бутон „Добави в количка“



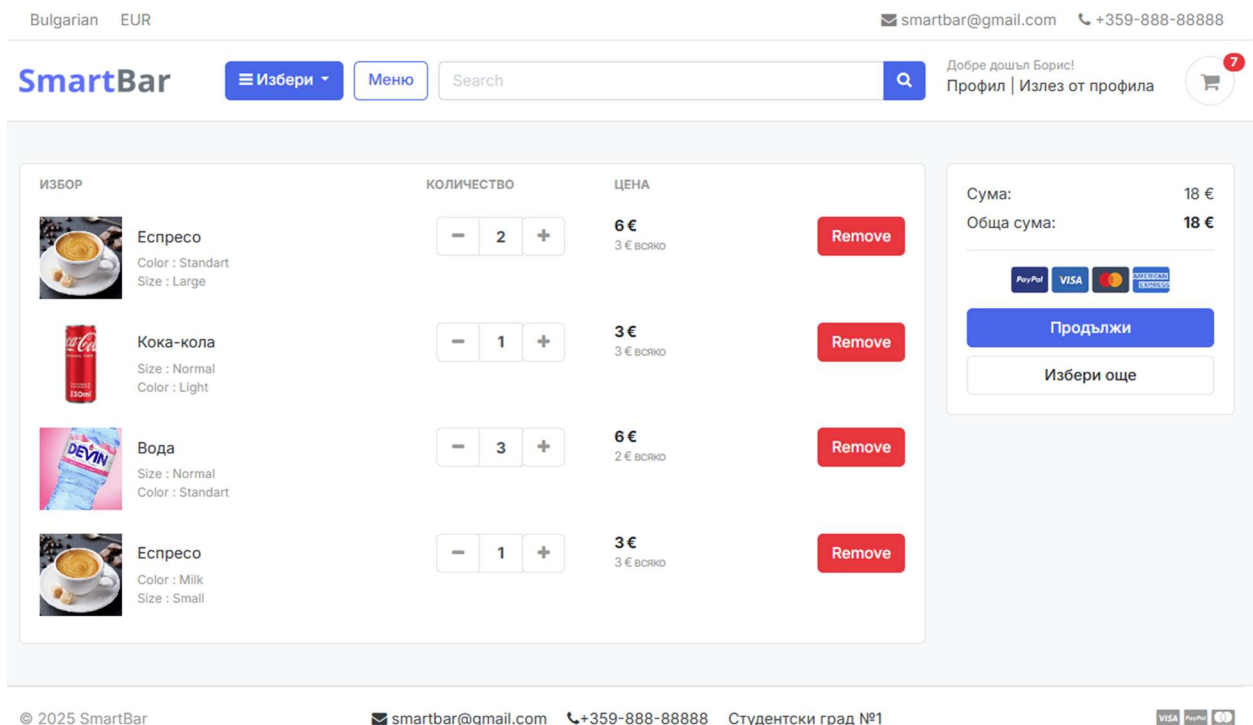
Продукт - Фиг.15

4.4 Количка

След добавяне, потребителят може да отиде в Количката, където:

- Вижда всички добавени продукти, със:
 - Избрани вариации
 - Количество
 - Единична цена
 - Обща цена за продукта
- Има възможности:
 - Увеличаване/намаляване на броя

- Премахване на продукт
- В долната част:
 - Обща сума
 - Крайна сума
 - Бутон „Продължи към плащане“



Количка - Фиг.16

4.5 Поръчка

След натискане на „Продължи към плащане“:

- Зарежда се страница с форма за адрес и доставка, включваща:
 - Име и фамилия
 - Телефон

- Имейл
- Държава, област, град
- Адрес
- Бележка към поръчката (по избор)

2. След потвърждение се създава поръчка в режим „в изчакване“ и потребителят се пренасочва към страницата за плащане.

Bulgarian EUR

smartbar@gmail.com +359-888-88888

SmartBar

Избери Меню

Search

Добре дошъл Борис!
Профил | Излез от профила

7

Данни за фактуриране

Име

Борис

Фамилия

Костадинов

Емейл

boriskostadinof@abv.bg

Телефонен номер

1234567888





Място

Маса 10

Допълнение

Коментар по поръчката

Лед и лимон

ПРОДУКТ	КОЛИЧЕСТВО	ЦЕНА
 <div>Еспресо Color : Standart Size : Large</div>	2	6 € 3 € всяка
 <div>Кока-кола Size : Normal Color : Light</div>	1	3 € 3 € всяка
 <div>Вода Size : Normal Color : Standart</div>	3	6 € 2 € всяка
 <div>Еспресо Color : Milk Size : Small</div>	1	3 € 3 € всяка

Направете поръчка

Добави продукт

© 2025 SmartBar

smartbar@gmail.com +359-888-88888 Студентски град №1

VISA

Master

Apple Pay

Данни за фактуриране - Фиг.17

4.6 Плащане с PayPal

На страницата за плащане:

- Зарежда се бутон на PayPal, чрез вграден SDK.

- Потребителят:
 1. Натиска бутона.
 2. Влиза в акаунта си в PayPal.
 3. Потвърждава плащането.
- След успешно плащане:
 - В базата се създава запис в таблицата Payment.
 - Поръчката се маркира като „Завършена“.
 - Изпраща се имейл с фактура и потвърждение.

Bulgarian
EUR
smartbar@gmail.com
+359-888-88888

SmartBar
Избери
Меню
Search
Добре дошъл Борис!
Профил | Излез от профила

Прегледайте поръчката си и извършете плащане





Данни за фактуриране

Борис Костадинов
Маса 10
boriskostadinof@abv.bg
1234567888
Коментар по поръчката: Лед и лимон





Начин на плащане

PayPal


Виж продуктите

ИМЕ	КОЛИЧЕСТВО	ЦЕНА
 Еспресо Color : Standart Size : Large	2	6 € 3 € each
 Кока-кола Size : Normal Color : Light	1	3 € 3 € each
 Вода Size : Normal Color : Standart	3	6 € 2 € each
 Еспресо Color : Milk Size : Small	1	3 € 3 € each

Цена: 18 €
Общо цена: 18 €









Pay with **PayPal**


Debit or Credit Card

Powered by **PayPal**

© 2025 SmartBar
smartbar@gmail.com
+359-888-88888
Студентски град №1


Плащане - Фиг.18

4.7 Преглед на завършена поръчка

След плащането, потребителят се пренасочва към страница order_complete/, където вижда:

- Уникален номер на поръчка
- Дата
- Статус на транзакцията (например „Завършена“)

- Таблица с продуктите
- Крайна цена
- Бутон за принтиране на фактурата – принтира се само избраната част от страницата.



Плащането е успешно

Направи нова поръчка

SmartBar

Поръчка #20250617107

Идентификационен номер на транзакцията #64A16594MM490805G

Дата на поръчката: June 16, 2025, 9:10 p.m.

СТАТУС: Завършен

Фактура

Борис Костадинов

Маса 10

Продукт	Количество	Цена
Еспресо <small>Color : Standart Size : Large</small>	2	3.0 €
Кока-кола <small>Size : Normal Color : Light</small>	1	3.0 €
Вода <small>Size : Normal Color : Standart</small>	3	2.0 €
Еспресо <small>Color : Milk Size : Small</small>	1	3.0 €
Сума:		18.0 €
Обща сума:		18.0 €

Приятно прекарване!

Печат

Успешно плащане и фактура с опция за печат - Фиг.19

4.8 История на поръчките

В потребителския профил има меню „Моите поръчки“, където клиентът:

- Вижда списък с предишни поръчки.
- Може да отвори всяка една.
- Преглед на продуктите, датата и статуса.

Bulgarian EUR smartbar@gmail.com +359-888-88888

SmartBar Избери Меню Search Добре дошъл Борис! Профил | Излез от профила

Табло

Моите поръчки

Редактирай профила

Промени паролата

Излез от профила

Твоята история на поръчки

Поръчка №	Име за фактуриране	Телефон	Обща цена	Дата
20250617107	Борис Костадинов	1234567888	18.0 €	June 16, 2025, 9:10 p.m.
20250614104	Борис Костадинов	0887100790	3.0 €	June 13, 2025, 10:11 p.m.
20250613103	Борис Костадинов	0887100790	17.0 €	June 13, 2025, 10:05 a.m.
20250613102	Борис Костадинов	0887100790	19.0 €	June 13, 2025, 7:10 a.m.
2025061294	Борис Костадинов	0887100790	110.0 €	June 12, 2025, 12:41 p.m.
2025061291	Борис Костадинов	0887100790	10.0 €	June 11, 2025, 10:42 p.m.
2025061287	Борис Костадинов	0887100790	20.0 €	June 11, 2025, 10:25 p.m.
2025061279	Борис Костадинов	0887100790	100.0 €	June 11, 2025, 10:06 p.m.
2025061277	Борис Костадинов	0887100790	40.8 €	June 11, 2025, 9:32 p.m.
2025061076	Boris Kostadinov	0887100790	40.8 €	June 10, 2025, 1:26 p.m.
2025061074	Boris Kostadinov	0887100790	10.2 €	June 9, 2025, 10:48 p.m.

© 2025 SmartBar smartbar@gmail.com +359-888-88888 Студентски град №1

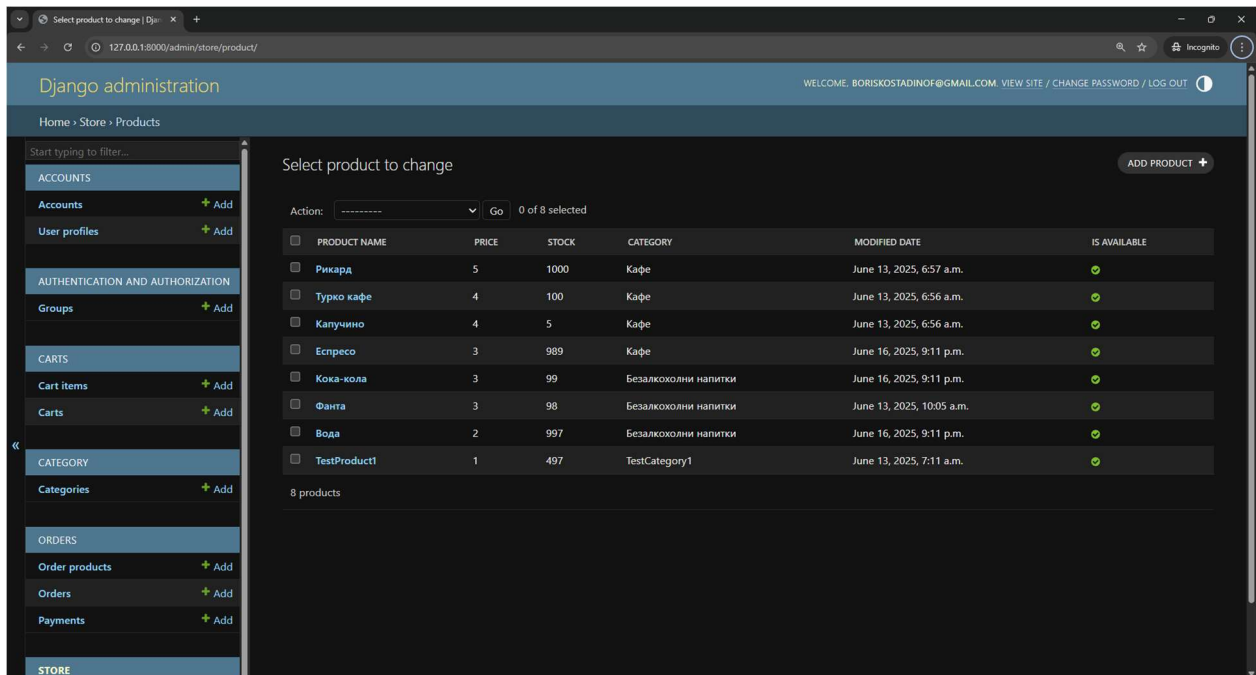
Моите поръчки - Фиг.20

4.9 Административен панел (Django Admin)

За служителите и администраторите:

- Достъпен е на /admin.
- Използват се Django ModelAdmin класове за управление.
- Възможности:

- Създаване и редакция на продукти, категории и вариации
- Преглед на поръчки
- Промяна на статуси
- Преглед на транзакции и профили




Административен панел - Фиг.21

4.10 Работа с имейли

- Всички важни действия (регистрация, нова поръчка, плащане) генерират автоматични имейли.
- Имейлите са HTML шаблони, изпратени чрез SMTP сървър.
- Работи се с EmailMessage от django.core.mail.

Please activate your account

от: greatbar200@gmail.com 
до: boriskostadinof@abv.bg

Hi b,

Please click on below link to confirm your registration.

<http://127.0.0.1:8000/accounts/activate/MTE/cqune7-a6af22b6ae4b87db9930aa99081ce40c/>

If you think it's not you, please ignore this email.

Изпратен емейл - Фиг.22

ГЛАВА 5: ЗАКЛЮЧЕНИЕ

В рамките на тази дипломна работа беше проектирана и разработена уеб базирана система за поръчки и плащания в заведения, наречена SmartBar. Основната цел на проекта беше да се създаде платформа, която да автоматизира и улесни процеса по приемане на поръчки и извършване на плащания в заведения като барове, ресторанти и кафенета. Реализираното решение цели както подобряване на потребителското изживяване, така и оптимизиране на работните процеси в обслужващия персонал.

Разработката премина през всички основни етапи на инженерния процес – анализ на нуждите, планиране, архитектурно проектиране, имплементация, тестове и валидиране на резултатите. Системата включва множество функционалности, сред които управление на продукти и категории, потребителска регистрация и вход, динамична количка, създаване и преглед на поръчки, както и интеграция с външна платежна услуга чрез PayPal. Специално внимание беше отделено на сигурността и гъвкавостта на системата, както и на възможността тя да се надгражда в бъдеще.

Системата SmartBar използва съвременни технологии, като Django за сървърна логика, Bootstrap за интерфейс и PayPal за онлайн разплащания. Всеки компонент беше реализиран така, че да бъде устойчив, лесен за поддръжка и разбираем от други разработчици. В допълнение към основната функционалност беше реализирана административна зона, чрез която могат да се управляват потребителите, поръчките и наличностите.

Един от ключовите аспекти в реализацията беше тестовата интеграция с PayPal чрез тяхната Sandbox среда. Това даде възможност да се симулират реални сценарии, без да се извършват реални транзакции. След успешното тестване, приложението беше конфигурирано за реална употреба с валидни идентификатори и валута, съответстваща на конкретните изисквания.

Сред основните предимства на SmartBar могат да се посочат: бързо обслужване на клиенти, безконтактно плащане, ясна структура на поръчките и възможност за последващо разширение – например добавяне на мобилно приложение, QR поръчки, статистика и отчетност.

Разработката на тази система позволи на автора да приложи на практика знания в областта на уеб програмирането, работата с бази от данни, потребителски интерфейси, сигурност на уеб приложения и интеграция с външни API услуги. Освен техническата реализация, важен акцент беше и върху потребителското изживяване, като приложението е проектирано така, че да бъде максимално интуитивно и достъпно.

В заключение може да се отбележи, че системата напълно отговаря на поставените изисквания и цели в дипломното задание. Тя е приложима в реални условия, като може да се внедри в малки и средни заведения, които търсят ефективен и достъпен начин да дигитализират своя процес по обслужване. Със сравнително малко усилия платформата може да бъде доразвита и персонализирана според нуждите на конкретен клиент или бизнес модел.

ГЛАВА 6: ИЗПОЛЗВАНИ ИЗТОЧНИЦИ И ПРИЛОЖЕНИЯ

Приложение 1: Използвани фигури

Архитектурната диаграма, визуализираща потока на заявките в системата SmartBar според MVT модела на Django - Фиг. 1	13
Python лого - Фиг.2	14
Django лого - Фиг.3.....	15
HTML5 лого - Фиг.4.....	15
CSS3 и Bootstrap лого - Фиг.5.....	16
JavaScript лого - Фиг.6.....	16
SQLite лого - Фиг.7	17
PayPal лого - Фиг.8	17
ER диаграма на базата данни - Фиг.9	21
Начална старница (Home) - Фиг.10.....	32
Регистрирай се - Фиг.11	33
Изпратен емейл - Фиг.12.....	34
Вход - Фиг.13.....	34
Моят профил - Фиг.14.....	35
Продукт - Фиг.15	36
Количка - Фиг.16.....	37
Данни за фактуриране - Фиг.17.....	38
Плащане - Фиг.18	40
Успешно плащане и фактура с опция за печат - Фиг.19	41
Моите поръчки - Фиг.20	42
Административен панел - Фиг.21	43
Изпратен емейл - Фиг.22.....	44

Приложение 2: Списък на използвани литературни и онлайн източници

1. Django Software Foundation. *Django Documentation (v5.2)*.
Достъпно на: <https://docs.djangoproject.com/en/5.2/>
2. Python Software Foundation. *Python 3.13 Official Documentation*.
Достъпно на: <https://docs.python.org/3/>
3. PayPal Developer Portal. *PayPal Checkout Integration Guide*.
Достъпно на: <https://developer.paypal.com/docs/checkout/>
4. Bootstrap. *Bootstrap v5.3 Documentation*.
Достъпно на: <https://getbootstrap.com/docs/5.3/getting-started/introduction/>
5. W3Schools. *HTML, CSS, JavaScript Tutorials*.
Достъпно на: <https://www.w3schools.com/>
6. Mozilla Developer Network (MDN). *JavaScript and Web APIs Documentation*.
Достъпно на: <https://developer.mozilla.org/en-US/docs/Web>
7. GitHub – проект на дипломанта: *SmartBar – Django-based Web Ordering System*.
Достъпно на: <https://github.com/boriskostadinov/greatbar-django>
8. Django Creating forms from models *ModelForm*
Достъпно на: <https://docs.djangoproject.com/en/5.2/topics/forms/modelforms/>
9. Django The messages framework
Достъпно на: <https://docs.djangoproject.com/en/5.2/ref/contrib/messages/>
10. Real Python. *How to Work with Django Models, Forms, and Views*.
Достъпно на: <https://realpython.com/>
11. Corey Schafer – YouTube канал: *Django Tutorials*.
Достъпно на: <https://www.youtube.com/user/schafer5>
12. Django REST Framework. *Official Documentation*.
Достъпно на: <https://www.django-rest-framework.org/>

13. Postman. *Testing APIs and integration debugging*.
Достъпно на: <https://www.postman.com/>
14. decouple · PyPI. *Python-Decouple – Settings Management*.
Достъпно на: <https://pypi.org/project/python-decouple/>
15. PythonAnywhere – хостинг платформа за Django приложения.
Достъпно на: <https://www.pythonanywhere.com/>
16. Gmail SMTP Setup – *Google Workspace Admin Help*.
Достъпно на: <https://support.google.com/a/answer/176600?hl=en>