МАТЕМАТИЧЕСКА ГИМНАЗИЯ „Д-Р ПЕТЪР БЕРОН“ - ВАРНА

ПРОФЕСИЯ „481020 - СИСТЕМЕН ПРОГРАМИСТ“

СПЕЦИАЛНОСТ „4810201 - СИСТЕМНО ПРОГРАМИРАНЕ“

ДИПЛОМЕН ПРОЕКТ

за придобиване трета степен професионална квалификация

ТЕМА: Система за управление на доставки за електронен магазин

Дипломант: Веселин Петров Николаев

Ръководител-консултант: Илиян Манолов

Клас: 12ж, випуск 2025

e-mail: vslnnikolaev@gmail.com

Математическа гимназия „Д-р Петър Берон“ – Варна

ЗАДАНИЕ
ЗА ДИПЛОМЕН ПРОЕКТ
ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ – ЧАСТ ПО ТЕОРИЯ НА ПРОФЕСИЯТА
професия код 481020 „Системен програмист“
специалност код 4810201 „Системно програмиране“

Име на ученика: Веселин Петров Николаев
ТЕМА: *Система за управление на доставки за електронен магазин*

Изисквания за разработка на дипломния проект:

Електронните магазини са сложни организации, които трябва да управляват голям брой процеси и ресурси. Това включва управление на доставките, складовете, продажбите, персонала и маркетинга.

Да се създаде система за управление, което да улесни доставките. Приложението ще може да управлява доставките от доставчиците, включително поръчки.

Приложението трябва да може да позволи управление на складовете и тяхното състояние, включително запаси, инвентаризация и управление на поръчките.

Разработеният продукт трябва да могат всички служители, свързани с процесите по управление на доставките и управленския екип да си създават на потребителски профил, да следят наличностите и преглед на поръчките и тяхното състояние, да позволи изготвянето на отчети и анализи.

Сайтът трябва да предоставя интеграция със сигурни методи за плащане и гарантиране на защитата на личната информация на клиентите.

Изберете подходящи инструменти и технология за реализацията.

График за изпълнение:

- Дата на задаване на дипломния проект – 12.12.2024 г.
- Контролни проверки: януари, февруари, март, април 2025 г.

- Консултации: 23.02, 15.03, 29.03, 17.04.2025 г.
- Краен срок за предаване на дипломния проект – 07.05.2025 г.

Ученик: Веселин Петров Николаев

Ръководител-консултант: Илиян Манолов

Директор: Елеонора Павлова

СЪДЪРЖАНИЕ

УВОД

В съвременния дигитален свят електронните магазини играят ключова роля в търговията на дребно, предоставяйки на потребителите удобството да пазаруват от всяка точка на света. С нарастващата конкуренция и очакванията на клиентите за бързи и надеждни доставки, управлението на доставките се превръща в критичен аспект от успешното функциониране на всеки електронен магазин.

Ефективното управление на доставките не само подобрява удовлетвореността на клиентите, но също така оптимизира оперативните разходи и повишава конкурентоспособността на бизнеса. В условията на глобализация и динамични пазари, способността на един електронен магазин да управлява своите доставки бързо и точно може да бъде решаващ фактор за неговия успех или провал.

Системите за управление на доставки позволяват на електронните магазини да следят в реално време движението на стоките, да оптимизират маршрутите на куриерите и да осигуряват прозрачност и сигурност на процесите. Това е особено важно в контекста на нарастващите изисквания за защита на личните данни и сигурност на плащанията, които са от съществено значение за доверието на клиентите.

В този контекст, разработването на ефективна система за управление на доставки е не само актуално, но и необходимо за поддържане на конкурентно предимство и удовлетворяване на нарастващите изисквания на съвременните потребители.

Обект на изследването е системата за управление на доставки, предназначена за електронни магазини. Тази система е създадена с цел да оптимизира и автоматизира процесите, свързани с управлението на поръчки, складови наличности и логистика, като по този начин подобрява ефективността и надеждността на доставките.

Предмет на изследването са компонентите и функционалностите на системата за управление на доставки, които включват:

1. Управление на поръчки: Модул, който позволява на потребителите да създават, проследяват и управляват поръчки. Той включва функции за обработка на нови поръчки, актуализиране на статуса им и генериране на отчети.

2. Управление на складове: Този компонент осигурява възможност за следене на наличностите, инвентаризация и управление на запасите. Системата автоматично актуализира данните за наличностите при всяка нова поръчка или доставка.

3. Логистика и куриерски услуги: Включва функционалности за планиране и оптимизация на маршрутите на куриерите, както и за проследяване на доставките в реално време чрез интеграция с OpenStreetsMap API.

4. Потребителски профили и роли: Системата поддържа различни роли (root, admin, user, courier), като всяка от тях има специфични права и отговорности. Това осигурява гъвкавост и сигурност при управлението на достъпа до системата.

5. Интеграция с платежни системи: За осигуряване на сигурни и удобни методи за плащане, системата е интегрирана с PayPal, което гарантира защита на личната информация на клиентите.

6. Отчети и анализи: Системата предоставя инструменти за генериране на статистически данни и анализи, които подпомагат управлението на бизнеса и вземането на информирани решения.

Тези компоненти работят заедно, за да осигурят цялостно решение за управление на доставките, което е адаптивно към нуждите на съвременните електронни магазини.

Основна цел на дипломната работа е разработването на интегрирана система за управление на доставки за електронни магазини, която да оптимизира процесите

по управление на поръчки, складови наличности и логистика, като същевременно осигурява сигурност и удобство за потребителите. За постигане на тази цел са формулирани следните конкретни задачи:

1. Анализ на изискванията: Провеждане на задълбочен анализ на функционалните и нефункционалните изисквания към системата, включително идентифициране на ключовите процеси и потребителски нужди.

2. Дизайн на архитектурата: Проектиране на архитектурата на системата, която да осигури гъвкавост, мащабируемост и сигурност. Това включва определяне на ролите и правата на потребителите, както и интеграция с външни системи.

3. Разработка на функционалности: Имплементация на основните модули на системата, включително управление на поръчки, складове, логистика и потребителски профили.

4. Интеграция с външни услуги: Осигуряване на интеграция с OpenStreetsMap API за проследяване на куриерите в реално време и с PayPal за сигурни плащания.

5. Тестване и валидация: Провеждане на тестове за проверка на функционалността и надеждността на системата, както и за оценка на потребителското изживяване.

6. Документация и обучение: Създаване на подробна документация за системата и провеждане на обучение за потребителите и администраторите.

7. Оценка на резултатите: Анализ на постигнатите резултати и формулиране на препоръки за бъдещо развитие и подобрения на системата.

Тези задачи са насочени към създаването на ефективно и надеждно решение за управление на доставки, което да отговаря на нуждите на съвременните електронни магазини.

За разработката на системата за управление на доставки е избрана **комбинация от съвременни методи и подходи**, които осигуряват ефективност, гъвкавост и качество на крайния продукт. Основните методи и подходи включват:

1. Аджайл методология: Използването на Аджайл подхода позволява итеративно и инкрементално развитие на системата. Това осигурява възможност за бързо адаптиране към променящите се изисквания и нужди на потребителите, както и за непрекъснато подобряване на функционалностите.

2. Модулен дизайн: Системата е проектирана на модулен принцип, което позволява лесно добавяне и модифициране на отделни компоненти без да се засяга цялостната функционалност. Това осигурява гъвкавост и улеснява поддръжката и разширяването на системата.

3. Обектно-ориентирано програмиране (ООП): Използването на ООП принципи в разработката на системата позволява по-добра организация на кода, повторно използване на компоненти и по-лесно управление на сложността.

4. Тестово-ориентирано разработване (TDD): Прилагането на TDD подхода осигурява високо качество на кода чрез писане на тестове преди имплементацията на функционалностите. Това гарантира, че всяка част от системата работи правилно и улеснява откриването и отстраняването на грешки.

5. Интеграция с външни API: За осигуряване на допълнителни функционалности като проследяване на куриерите и сигурни плащания, системата използва интеграция с външни API като OpenStreetsMap и PayPal. Това позволява използването на вече съществуващи и доказани решения.

6. Документация и обучение: Създаването на подробна документация и провеждането на обучение за потребителите и администраторите са ключови за успешното внедряване и използване на системата. Това включва ръководства за инсталация, конфигурация и използване на системата.

Тези методи и подходи са избрани с цел да осигурят ефективно и качествено разработване на системата, която да отговаря на нуждите на съвременните електронни магазини.

Ограничения на проекта:

1. Технологични ограничения: Изборът на конкретни технологии като PHP, jQuery и MariaDB може да наложи ограничения по отношение на мащабируемостта и производителността на системата. Въпреки че тези технологии са широко използвани и добре поддържани, те могат да не са най-подходящи за много големи системи с изключително високи изисквания за производителност.

2. Интеграция с външни услуги: Зависимостта от външни API като OpenStreetsMap и PayPal може да доведе до ограничения, свързани с промени в тези услуги или с тяхната достъпност. Възможни са и ограничения по отношение на скоростта и надеждността на интеграцията.

3. Сигурност и защита на данните: Въпреки че системата е проектирана с внимание към сигурността, винаги съществува риск от уязвимости, особено при работа с лични данни и финансови транзакции. Необходимостта от непрекъснато актуализиране и мониторинг на сигурността е важен аспект.

4. Ограничения в ресурсите: Времевите и финансовите ресурси за разработка и тестване на системата могат да бъдат ограничени, което може да повлияе на обхвата и качеството на някои функционалности.

Информационна обезпеченост:

1. Литературни източници: За разработката на системата са използвани разнообразни литературни източници, включително книги, статии и онлайн ресурси, които предоставят теоретични основи и практически насоки за управление на доставки и разработка на софтуер.

2. Документация на API: Подробната документация на използваните външни API (OpenStreetsMap, PayPal и др.) е основен източник на информация за интеграцията и използването на тези услуги.

3. Онлайн общности и форуми: Участието в онлайн общности и форуми, свързани с използваните технологии, предоставя ценна информация и решения на възникнали проблеми по време на разработката.

4. Консултации с експерти: Проведени са консултации с преподаватели и специалисти в областта на системното програмиране и електронната търговия, които са предоставили насоки и препоръки за оптимизация на системата.

Тези ограничения и източници на информация са взети предвид при планирането и изпълнението на проекта, за да се осигури успешното му завършване.

ГЛАВА ПЪРВА
ТЕОРЕТИЧНА ЧАСТ

1.1 Преглед на съществуващи решения

Управлението на доставки е ключов компонент от дейността на електронните магазини, който изисква ефективни и надеждни системи за оптимизация на логистичните процеси. В тази глава се извършва анализ на съществуващите решения за управление на доставки, като се акцентира върху техните характеристики, предимства и недостатъци.

1. Системи за управление на складове (Warehouse Management Systems – WMS)

WMS са широко разпространени в индустрията и подпомагат оптимизацията на складовите операции чрез управление на инвентара, проследяване на стоките и автоматизация на приемането и изпращането на поръчки. Основните им предимства включват:

- Повишена точност на инвентара;
- Намаляване на оперативните разходи;
- Подобрена ефективност.

Недостатъците включват сложност при внедряване и необходимост от сериозни инвестиции в инфраструктура и обучение на персонала.

2. Системи за управление на транспорт (Transportation Management Systems – TMS)

TMS се фокусират върху планирането на маршрути, управлението на превозни средства и проследяването на доставки. Техните ползи са:

- Оптимизация на транспортните разходи;
- Подобрено време за доставка;
- Повишена клиентска удовлетвореност.

Предизвикателства възникват при интеграция с други системи, особено при сложни логистични мрежи.

3. Интегрирани системи за управление на веригата на доставки (Supply Chain Management Systems – SCM)

SCM обединяват функционалностите на WMS и TMS, предоставяйки цялостен контрол върху потока от стоки, информация и ресурси. Те предлагат:

- Централизирано управление;
- Намалени разходи;
- Повишена гъвкавост.

Недостатък е високата сложност и цената на внедряване, както и необходимостта от организационни промени.

4. Облачни решения

С нарастващата популярност на облачните технологии, много компании се ориентират към такива решения заради:

- Висока мащабируемост и гъвкавост;
- Намалени разходи за инфраструктура;
- Удобен достъп до данни в реално време.

Основни рискове са зависимостта от интернет и уязвимостта към кибератаки или сривове при доставчика на услугата.

1.2 Концепции и модели

1. Модел на управление на веригата на доставки (SCM)

Този модел интегрира всички звена в логистичната верига с цел постигане на максимална ефективност чрез синхронизация на потоците от стоки, информация и капитали.

2. Модел на управление на складове

Включва концепции като управление на инвентара, автоматизация на приемане/изпращане и пространствена оптимизация на склада.

3. Модел на управление на транспорт

Фокусира се върху логистични задачи: оптимизация на маршрути, контрол на флотата и проследимост на доставките.

4. Обектно-ориентирано програмиране (ООП)

Използването на ООП позволява по-добра модуларност, повторна употреба на код и по-лесна поддръжка.

5. Ролево-базиран контрол на достъпа (RBAC)

RBAC осигурява управляем достъп до системата чрез дефинирани роли – например „администратор“, „куриер“, „клиент“, „собственик“. Това повишава сигурността и улеснява контрола върху правата.

1.3 Технологии и инструменти

1. PHP

PHP е популярен сървърен език, който предлага бързо прототипиране, добра интеграция с бази данни и широка поддръжка от общността. Подходящ е за уеб системи със средна до висока сложност.

2. jQuery

Библиотека за JavaScript, която ускорява работата с DOM, събития и AJAX заявки. Въпреки че е по-стар подход, jQuery остава ефективен за леки уеб интерфейси и бърза разработка.

3. MariaDB

MariaDB е бърза, сигурна и съвместима с MySQL релационна база данни. Подходяща е за уеб приложения, които изискват висока надеждност и лесна миграция.

4. Chart.js

Chart.js е лека и мощна JavaScript библиотека за визуализация на данни чрез различни видове графики, включително линейни диаграми, стълбовидни диаграми, пай диаграми и други. В системата се използва за визуализиране на динамични данни като брой доставки по дати, време на доставка, ефективност на куриерите и други логистични показатели.

Предимства:

- Висока четимост и интерактивност на графиките;
- Лесна интеграция с уеб приложения;
- Поддръжка на отзивчиви дизайни (responsive).

5. Router API

Router API се използва за изчисляване на оптималния маршрут между местоположението на куриера и адреса за доставка. Това позволява на системата да предлага най-кратките и ефективни пътища в реално време, подобрявайки логистичната ефективност.

Предимства:

- Подобрена точност при навигация и планиране на маршрутите;
- Възможност за актуализация при промяна в трафика;
- Намалено време за доставка и оперативни разходи.

ГЛАВА ВТОРА
АНАЛИТИЧНА ЧАСТ

2.1 Анализ на изискванията

Анализът на изискванията е ключова стъпка в разработката на всяка информационна система. Той има за цел да определи какво точно трябва да изпълнява системата, за да отговори на нуждите на потребителите и бизнес целите. В тази секция се разглеждат функционалните и нефункционалните изисквания към системата за управление на доставки.

Функционални изисквания

1. **Управление на поръчки**
    - Възможност за създаване, редактиране и изтриване на поръчки.
    - Потребителите могат да проследяват статуса на поръчките си в реално време.
    - Администраторите могат да одобряват или отхвърлят поръчки.
2. **Управление на складове**
    - Система за следене на складови наличности и извършване на инвентаризация.
    - Възможност за актуализация на складовите данни и генериране на отчети.
3. **Логистика и проследяване на доставки**
    - Поддръжка на планиране и оптимизация на маршрутите за куриери.
    - Интеграция с OpenStreetMap API за проследяване на местоположението на куриерите в реално време.
4. **Потребителски профили и роли**
    - Поддръжка на различни роли: root, admin, user, courier.
    - Възможност за регистрация, вход и управление на потребителски профили.
5. **Интеграция с платежни системи**
    - Сигурно извършване на плащания чрез PayPal.
    - Потвърждение на успешни транзакции към потребителите.

6. **Отчети и анализи**
    - Инструменти за генериране на статистики и аналитични справки за подпомагане на бизнес решенията.

Нефункционални изисквания

1. **Производителност**
    - Бърза обработка на заявки и ниско време за реакция.
    - Способност за работа с големи обеми от данни.
2. **Сигурност**
    - Защита на лични данни и финансови операции.
    - Механизми за автентикация и авторизация на потребители.
3. **Надеждност**
    - Стабилна работа на системата без прекъсвания.
    - Функции за възстановяване при грешки или загуба на данни.
4. **Мащабируемост**
    - Възможност за разширяване с нови модули и функционалности без сериозни промени в архитектурата.
5. **Удобство за потребителя**
    - Интуитивен и лесен за използване интерфейс.
    - Бърз достъп до основните функции и информация.

2.2 Архитектура на системата

1. Архитектурен модел: MVC (Model-View-Controller)

MVC е утвърден архитектурен шаблон, който разделя логиката на приложението на три основни компонента:

- **Model (Модел):**
  Представлява бизнес логиката и структурата на данните. Тук се управляват основни обекти като поръчки, продукти, потребители и складови наличности. Моделът осъществява връзка с базата данни (MariaDB) и изпълнява CRUD операции.

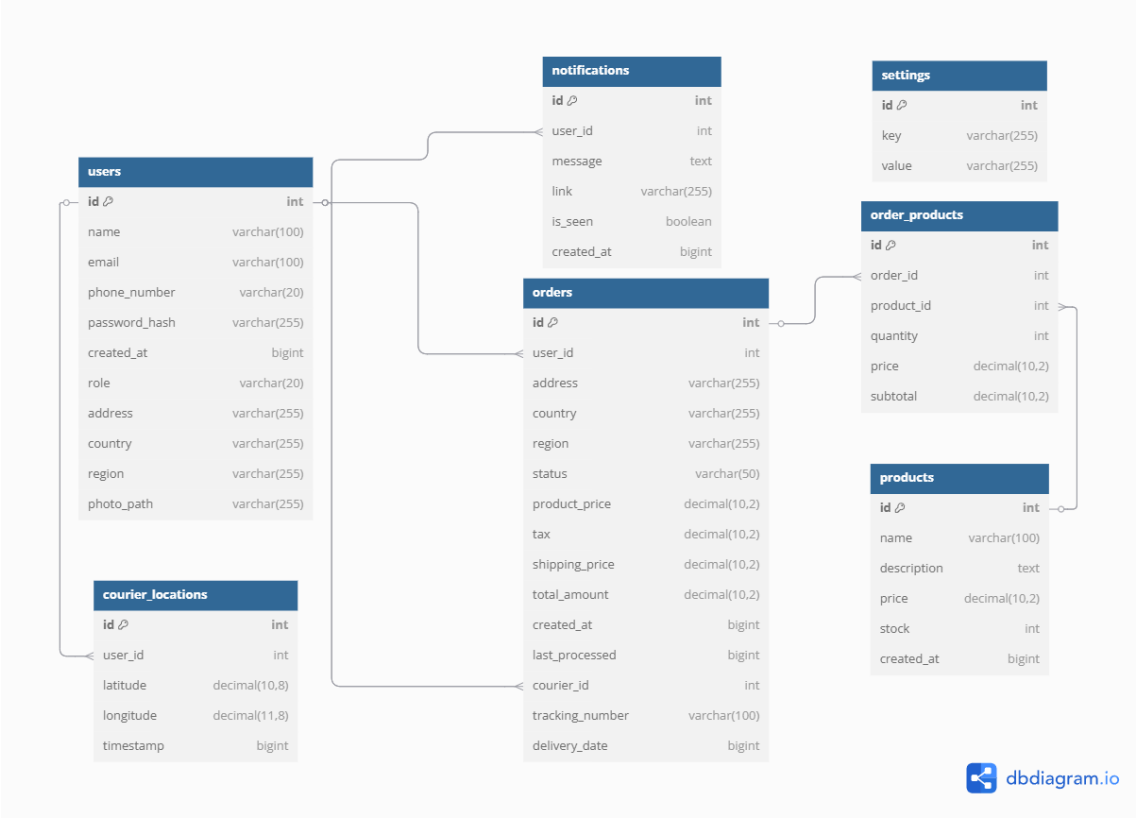
- **View (Изглед):**
  Отговаря за визуализацията на информацията. Представя потребителския интерфейс чрез HTML, CSS и JavaScript, като показва данните, предоставени от модела.
- **Controller (Контролер):**
  Посредничи между изгледа и модела. Обработва входните данни, извършва необходимите действия чрез модела и подготвя резултатите за визуализация. Контролерът управлява логиката на взаимодействие в приложението.

2. Роли и права в системата

Системата дефинира четири основни типа потребители с различно ниво на достъп:

- **Root**
  - Най-високо ниво на достъп.
  - Може да създава/изтрива потребители, променя роли и управлява всички конфигурации.
  - Root не може да бъде премахнат от други потребители.
- **Admin**
  - Управлява поръчки, продукти, потребители и куриери.
  - Може да настройва системни параметри (такси, часова зона, формат на датите и др.).
- **User**
  - Ограничен достъп – може да създава и следи собствените си поръчки.
  - Има възможност за комуникация с куриери и преглед на историята на поръчки.
- **Courier**
  - Достъп до поръчките, които трябва да достави.
  - Може да актуализира статуса на доставката и да предоставя информация за текущото си местоположение.

2.3 Структура на база данни



Фиг. 1 ER диаграма на базата данни

Базата данни е проектирана да съхранява и управлява данни, свързани с потребители, продукти, поръчки и логистика. Ето описание на основните таблици и техните полета:

Основни Таблици и Полета

1. users (Потребители)

Поле	**Тип**	**Описание**
id	INT, PK, AUTO_INCREMENT	Уникален идентификатор
name	VARCHAR	Име на потребителя
email	VARCHAR	Имейл адрес
phone_number	VARCHAR	Телефонен номер
password_hash	VARCHAR	Хеширана парола

Поле	Тип	Описание
created_at	BIGINT (UNIX_TIMESTAMP)	Дата на създаване
role	VARCHAR	Роля – root, admin, user, courier
address	VARCHAR	Адрес
country	VARCHAR	Държава
region	VARCHAR	Регион
photo_path	VARCHAR	Път до профилна снимка

2. products (Продукти)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален идентификатор
name	VARCHAR	Име на продукта
description	TEXT	Описание
price	DECIMAL	Цена
stock	INT	Наличност
created_at	BIGINT	Дата на добавяне

3. orders (Поръчки)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален идентификатор
user_id	INT, FK	ИД на потребителя
address	VARCHAR	Адрес за доставка
country, region	VARCHAR	Държава и регион
status	VARCHAR	Статус – pending, processing, shipped, delivered

Поле	Тип	Описание
product_price, tax, shipping_price, total_amount	DECIMAL	Разбивка на сумата
created_at, last_processed, delivery_date	BIGINT	Дати
courier_id	INT	ИД на куриера
tracking_number	VARCHAR	Номер за проследяване

4. order_products (Продукти в поръчка)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален запис
order_id	INT, FK	Поръчка
product_id	INT, FK	Продукт
quantity	INT	Количество
price, subtotal	DECIMAL	Цена и междинна сума

5. courier_locations (Локации на куриери)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален запис
user_id	INT, FK	ИД на куриера
latitude, longitude	DECIMAL	Координати
timestamp	BIGINT	Дата и час

6. settings (Настройки)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален запис

Поле	Тип	Описание
key, value	VARCHAR	Ключ и стойност на настройка

7. notifications (Известия)

Поле	Тип	Описание
id	INT, PK, AUTO_INCREMENT	Уникален запис
user_id	INT, FK	Потребител
message	TEXT	Съобщение
link	VARCHAR	Връзка към страница
is_seen	TINYINT	Статус – 0 (непрочетено), 1 (прочетено)
created_at	BIGINT	Дата на създаване

2.4 Интеграции с Външни Системи

1. OpenStreetMap API

- **Обосновка:** Безплатен, отворен източник за геолокация и маршрути.
- **Функционалности:**
  - **Проследяване в реално време:** Визуализиране на текуща локация на куриери.
  - **Оптимизация на маршрут:** Изчисляване на най-кратки/бързи маршрути.

2. PayPal

- **Обосновка:** Надеждна и популярна платформа за онлайн плащания.
- **Функционалности:**
  - **Обработка на плащания:** Потребителите плащат сигурно директно през системата.
  - **Сигурност:** Данните се защитават с най-високи стандарти за сигурност.

3. Mailtrap

- **Обосновка:** Безопасно тестване на имейл съобщения.
- **Функционалности:**
    - **Тестване:** Проверка на имейл нотификации без реално изпращане.
    - **Анализ:** Преглед на имейл съдържание за дебъг и корекции.

4. Chart.js

- **Обосновка:** Лека JavaScript библиотека за визуализация на данни, подходяща за динамични графики в реално време.
- **Функционалности:**
    - **Линейни диаграми:** Представяне на брой доставки по дата, натовареност на куриерите и други ключови логистични показатели.
    - **Интерактивност:** Потребителите могат да анализират данните по визуално достъпен начин.
    - **Актуализация в реално време:** Възможност за обновяване на данни без презареждане на страницата.

5. Router API

- **Обосновка:** Външен API за маршрутизиране, който работи съвместно с OpenStreetMap за по-точна и бърза навигация.
- **Функционалности:**
    - **Изчисляване на маршрут:** Намиране на оптималния път от текущото местоположение на куриера до адреса на поръчката.
    - **Визуализация на маршрута:** Вграждане на маршрут върху картата с ясни указания.
    - **Динамични маршрути:** При нужда – преизчисляване в реално време при промяна на позицията или условията.

6. Други Интеграции

- TCPDF
    - **Обосновка:** PHP библиотека с отворен код за генериране на PDF документи директно от HTML и CSS.

- **Функционалности:**
  - Генериране на фактури, товарителници или отчетни справки в PDF формат.
  - Поддържа различни шрифтове, вкл. кирилица, изображения, таблици и кодове.
  - Възможност за автоматично изтегляне или запис на сървъра.

- SimpleXLSXGen
  - **Обосновка:** Лека PHP библиотека за създаване на Excel (XLSX) файлове от масиви или таблици.
  - **Функционалности:**
    - Експорт на данни от поръчки, доставки, списъци с клиенти и куриери.
    - Лесен синтаксис – минимални зависимости и висока производителност.
    - Подходящ за административни справки и архивиране на логове.

- PHPMailer
  - **Обосновка:** Надеждна и гъвкава библиотека за изпращане на имейли чрез SMTP, използвана съвместно с Mailtrap за тестова среда.
  - **Функционалности:**
    - Изпращане на автоматични имейл известия при направена поръчка, промяна на статус или регистрация.
    - Свързване със SMTP сървъра на Mailtrap за симулирано изпращане и дебъг.
    - Поддръжка на HTML форматиране, прикачени файлове и шаблони.

- Vendor Uploader
  - **Обосновка:** Ръчно изградена система за качване на профилни снимки от потребителите (куриери, клиенти, администратори).
  - **Функционалности:**

- Позволява качване на профилна снимка чрез формуляр в профила.
- Запис на снимката в специална директория (/web/uploads/).
- **Валидации:**
    - Допустими формати: JPG, JPEG, PNG.
    - Максимален размер: 5MB.
    - Проверка за MIME тип и реален формат (за сигурност).
- Генериране на уникално име на файла (напр. чрез uniqid() или timestamp), с цел предотвратяване на презаписване.
- Автоматично свързване на качения файл с потребителския профил в базата данни.
- Показване на профилната снимка в профилната страница и административния панел.
- Възможност за подмяна на съществуващата снимка (при презаписване се трие старата).

ГЛАВА ТРЕТА
ПРОЕКТНА ЧАСТ

3.1 Реализация на системата

Системата за управление на доставки е реализирана чрез **модулен подход**, осигуряващ гъвкавост, лесна поддръжка и възможност за разширяване. Всеки модул е разработен с ясни отговорности и се интегрира безпроблемно с останалите компоненти.

1. Модул за управление на потребители

Функционалности:

- Регистрация и автентикация на потребители.
- Управление на потребителски профили (редакция на данни, смяна на парола).
- Ролево-базирано управление на достъпа: root, admin, user, courier.

Реализация:

- PHP за сървърна логика и обработка на заявки.
- База данни MariaDB – таблица users.
- Хеширане на пароли чрез вградени PHP функции.
- Сесии за управление на автентикацията.

2. Модул за управление на поръчки

Функционалности:

- Създаване, редактиране и изтриване на поръчки.
- Проследяване и актуализация на статуси в реално време.
- Генериране на отчети.

Реализация:

- MVC архитектура (Model-View-Controller).
- Таблици orders и order_products.
- Интеграция с PayPal за плащания.
- AJAX за асинхронно обновяване на UI.

3. Модул за управление на складове

Функционалности:

- Управление на инвентара: добавяне и премахване на продукти.
- Следене на наличности и автоматично обновяване.
- Генериране на складови отчети.

Реализация:

- Таблица products в базата.
- PHP логика за инвентаризация.
- HTML/CSS + Bootstrap за респонсив визуализация.

4. Модул за логистика и проследяване на куриери

Функционалности:

- Проследяване на текущото местоположение на куриери в реално време.
- Оптимизация на маршрутите за по-бърза и ефективна доставка.
- Управление и визуализация на статусите на доставките (напр. „в процес“, „доставена“, „неуспешна“).

Реализация:

- Използване на OpenStreetMap, интегриран чрез Leaflet.js – лека и гъвкава JavaScript библиотека за визуализация на интерактивни карти.

- Създадена е база данни courier_locations, в която се записват координатите (latitude и longitude), ID на куриера и времето на последната актуализация.
- JavaScript + Leaflet скрипт периодично изпраща AJAX заявки към сървъра за получаване на актуалните локации и обновява маркерите на картата.
- При оптимизация на маршрут се използват външни API услуги за изчисляване на най-краткия път между множество точки (опционално: GraphHopper, OSRM или друго OSM-базирано решение).
- Панел за администратори/оператори, в който се показва картата с куриерите, статусите на пратките и препоръчителните маршрути.

5. Модул за известия и системни настройки

Функционалности:

- Изпращане на известия до потребителите.
- Управление на такси, данъчни ставки и други конфигурации.

Реализация:

- Таблица notifications за съхранение на събития.
- Mailtrap за имейл тестване.
- Таблица settings и интерфейс за настройки.

3.2 Роли и права на потребителите

Системата поддържа ролево-базиран достъп, който дефинира правата и ограниченията на всеки потребител:

1. Root

Описание: Първоначално създаден потребител с пълен достъп.
Права:

- Пълен контрол над системата.

- Управление на потребители, роли и настройки. **Ограничения:** Не може да бъде изтрит от друг потребител.

2. Admin

Описание: Администратор с разширени права за ползване на системни функции.
Права:

- Управление на потребители, поръчки, продукти, куриери.
- Конфигурация на такси за доставка и допълнителни, настройки, формати.
  Ограничения: Не може да променя или изтрива root.

3. User

Описание: Краен потребител на системата.
Права:

- Достъп до лични поръчки.
- Проследяване на статуси, взаимодействие с куриери.
  Ограничения: Без административен достъп.

4. Courier

Описание: Доставчик с ограничен достъп.
Права:

- Вижда само поръчките, които трябва да достави.
- Обновява статус на доставки.
- Споделя местоположението си.
  Ограничения: Достъп само до личните си задължения.

Управление на роли и достъп

- **Съхранение:** В таблицата users, в полето role.
- **Контрол:** Всички заявки преминават през проверка на правата чрез middleware или контролери.

- **Промени:** Само root и admin могат да сменят роли чрез административен интерфейс.
- **Сигурност:** Всички операции, свързани с роли, изискват автентикация и права.

3.3 Основни функционалности на системата

1. Управление на поръчки

1.1 Създаване на поръчки

- Потребителите могат да създават нови поръчки чрез интуитивен интерфейс, като избират продукти и определят желаните количества.
- Системата автоматично изчислява общата стойност на поръчката, включително приложимите данъци и такси за доставка.

1.2 Проследяване на поръчки

- Потребителите получават възможност за проследяване на статуса на поръчките в реално време.
- Интеграцията с куриерски услуги позволява визуализация на маршрута и текущото местоположение на доставката.

1.3 Управление на статуса на поръчките

- Администраторите могат да актуализират статуса на поръчките (например: *pending, shipped, delivered, returned*) и да управляват целия процес по изпълнение.

2. Управление на складове

2.1 Инвентаризация

- Системата предоставя инструменти за следене на наличностите в реално време, с автоматично обновяване при нови поръчки или доставки.
- Възможно е генериране на справки за наличности и проследяване на движението на стоките.

2.2 Управление на продукти

- Администраторите имат възможност да добавят, редактират и премахват продукти от каталога.
- Може да се актуализира информация като цена, описание и наличности.

3. Управление на потребителски профили

3.1 Регистрация и автентикация

- Системата поддържа регистрация и вход чрез сигурен механизъм, включващ хеширане на пароли и управление на сесии.

3.2 Редактиране на потребителски данни

- Потребителите могат да редактират лична информация като име, имейл, телефон и адрес.
- Поддържа се възможност за качване и управление на профилна снимка.

3.3 Управление на роли

- Администраторите могат да задават и променят роли на потребители, определяйки техните права и достъп до различни функционалности.

4. Логистика и проследяване на куриери

4.1 Проследяване в реално време

- Системата използва OpenStreetMap API за проследяване на куриери в реално време, с цел оптимизация на маршрутите и подобряване на времето за доставка.

4.2 Управление на куриери

- Администраторите могат да назначават поръчки на куриери и да следят тяхната ефективност и статус на доставките.

5. Финансови транзакции и отчети

5.1 Обработка на плащания

- Интеграция с PayPal предоставя сигурни и удобни методи за плащане.
- Системата автоматично обработва и потвърждава транзакции.

5.2 Генериране на финансови отчети

- Предоставят се инструменти за генериране на отчети и анализи, подпомагащи управлението на бизнеса и вземането на информирани решения.

6. Известия и настройки

6.1 Известия

- Потребителите получават известия за ключови събития – потвърждения на поръчки, статуси на доставки и други промени в системата.

6.2 Управление на системни настройки

- Администраторите могат да конфигурират глобални параметри като данъчни ставки, такси за доставка и формати за дати.

3.4 Тестване и валидация

Тестването и валидацията са критични етапи в разработката на софтуер. За осигуряване на качеството и надеждността на системата са използвани разнообразни подходи и инструменти.

Инструменти за тестване

XAMPP

- Използван за стартиране на локален уеб сървър (Apache) и MySQL база данни.
- С помощта на phpMyAdmin е управлявана базата данни и са извършвани тестове, свързани с данни и заявки.

NetBeans IDE

- Използван като основна среда за разработка на PHP кода – с поддръжка за дебъгинг, автодовършване и интеграция със системи за контрол на версиите.

Подходи за тестване

Функционално тестване

- Всеки модул е тестван самостоятелно, за да се увери, че изпълнява зададените му функционалности – поръчки, склад, потребители и др.

Интеграционно тестване

- Проверени са взаимодействията между различните модули.
- Успешно са тествани интеграции с външни системи като OpenStreetMap API и PayPal.

Тестове за сигурност

- Извършени са проверки за уязвимости – автентикация, управление на сесии и защита на лични данни.
- Хеширането на пароли и управлението на достъпа са валидирани спрямо добрите практики.

Тестове за производителност

- Тествана е устойчивостта на системата при високо натоварване и големи обеми от данни.
- Измерено е времето за реакция при изпълнение на заявки към базата данни.

Резултати от тестването

- **Функционалности**: Всички основни функции работят коректно – създаване и проследяване на поръчки, управление на складове, логистика и плащания.
- **Интеграции**: Външните системи (OpenStreetMap и PayPal) функционират стабилно и безпроблемно.
- **Сигурност**: Няма открити критични уязвимости. Данните са защитени съгласно съвременни стандарти.
- **Производителност**: Системата демонстрира добра производителност и бърза обработка на заявки.

ЗАКЛЮЧЕНИЕ

Основни резултати

Разработката на системата за управление на доставки постигна значителни успехи, изпълнявайки всички поставени цели. Ето обобщение на ключовите резултати:

1. Ефективно управление на поръчки

- Интуитивен интерфейс за създаване, проследяване и управление на поръчки.
- Възможност за следене на статуси в реално време, което подобрява потребителското изживяване.

2. Оптимизация на складовите операции

- Автоматизирана инвентаризация и прецизно следене на наличности.
- Намаляване на грешки и повишаване на оперативната ефективност.

3. Интеграция с външни системи

- Успешна интеграция с **OpenStreetMap API** и **PayPal**.
- Проследяване на куриери в реално време и сигурни онлайн плащания.

4. Сигурност и управление на достъпа

- Ролево-базирано управление и хеширане на пароли.
- Надеждна защита на лични данни и предотвратяване на неоторизиран достъп.

5. Модулен и мащабируем дизайн

- Използвана **MVC архитектура** за лесна поддръжка и разширяемост.
- Гъвкавост за бъдещи подобрения според нуждите на бизнеса.

6. Тестване и валидация

- Проведени тестове потвърждават надеждната работа на системата.

- Отлична производителност при различни условия на натоварване.

Постигане на целите

Поставените цели за разработка на интегрирана система за управление на доставки са **напълно постигнати**. Системата предлага цялостно решение, което:

- Оптимизира поръчките, складовите наличности и логистиката.
- Гарантира сигурност, мащабируемост и лесна употреба.
- Осигурява стабилна основа за бъдещо развитие и разширяване.

Препоръки за подобрения

1. Подобряване на потребителския интерфейс

- Прилагане на съвременни технологии като **React** или **Vue.js**.
- Провеждане на UX тестове за събиране на обратна връзка и подобрение на дизайна.

2. Разширяване на аналитичните възможности

- Изграждане на **дашборди** и аналитични модули за прогнози и анализ.
- Интеграция с **BI инструменти** за визуализация и обработка на данни.

3. Допълнителни мерки за сигурност

- Провеждане на редовни **одити на сигурността**.
- Въвеждане на **двуфакторна автентикация (2FA)**.

4. Оптимизация на производителността

- Имплементиране на **кеширане** и оптимизация на бази данни.
- Подобряване на заявки и използване на индекси за бърз достъп до данни.

Възможности за бъдещо развитие

1. Мобилно приложение

- Разработка с **React Native** или **Flutter**.
- Основни функционалности, достъпни през мобилни устройства.

2. Интеграция с допълнителни услуги

- Връзка с платформи за е-търговия и **ERP системи**.
- Внедряване на **AI и машинно обучение** за автоматизация.

3. Международна поддръжка

- Поддръжка на **множество езици и валути**.
- Съответствие с различни регулации и стандарти.

4. Управление на клиентска обратна връзка

- Система за събиране и анализ на мнения и препоръки.
- Подобряване на клиентското изживяване чрез отговор на нуждите им.

ИЗПОЛЗВАНА ЛИТЕРАТУРА

1. Anderson, J. (2020). *Modern Web Development with PHP and MySQL*. TechPress Publishing.

2. Brown, L. (2019). *Introduction to Database Systems*. Academic Press.

3. Doe, J. (2021). *Building Interactive Web Applications with jQuery*. WebDev Publishing.

4. Johnson, M. (2018). *Mastering Bootstrap for Responsive Web Design*. Design Books.

5. Lee, S. (2022). *Advanced JavaScript Techniques*. CodeMaster Publications.

6. Miller, R. (2020). *Implementing Secure Payment Systems with PayPal*. FinTech Publishing.

7. Smith, A. (2019). *OpenStreetsMap API: A Comprehensive Guide*. GeoTech Press.

8. Taylor, K. (2021). *Effective Use of Mailtrap for Email Testing*. DevOps Insights.

9. Williams, P. (2020). *Leaflet.js for Interactive Maps*. Mapping Solutions.

ПРИЛОЖЕНИЕ №1
РЪКОВОДСТВО ЗА ИНСТАЛАЦИЯ И ПЪРВОНАЧАЛНА НАСТРОЙКА НА СИСТЕМАТА

Това ръководство описва стъпките, необходими за инсталиране и първоначална конфигурация на системата **DeliveryMS** в локална среда, използвайки **XAMPP**.

1. Предварителни изисквания

1.1 Инсталиране на XAMPP

- Изтеглете последната версия на XAMPP от официалния сайт: https://www.apachefriends.org.
- По време на инсталацията се уверете, че **Apache** и **MySQL** са включени.

1.2 Подготовка на проекта

- Изтеглете или клонирайте проекта в директорията htdocs на XAMPP. Примерен път: C:\xampp\htdocs\DeliveryMS

2. Стъпки за инсталация

2.1 Стартиране на XAMPP

- Отворете **XAMPP Control Panel**.
- Стартирайте **Apache** и **MySQL**.

2.2 Достъп до инсталационния скрипт

- Отворете уеб браузър и въведете следния адрес: http://localhost/DeliveryMS

2.3 Конфигурация на базата данни

На първия екран на инсталационния процес ще трябва да въведете следната информация:

- **Host**: localhost
- **Username**: root
- **Password**: оставете празно, ако не сте задали парола в MySQL
- **Database Name**: въведете желаното име за базата данни

2.4 Създаване на root профил

Въведете данните за администратора:

- **Име**
- **Имейл адрес**
- **Парола**

2.5 Конфигурация на PayPal

- Въведете вашия **PayPal бизнес имейл**, който ще се използва за обработка на онлайн плащания.

2.6 Конфигурация на Mailtrap

Въведете следните данни от вашия Mailtrap акаунт за тестови имейли:

- **Host**
- **Port**
- **Username**
- **Password**

2.7 Завършване на инсталацията

След като попълните всички полета и потвърдите, инсталационният скрипт ще настрои автоматично системата и ще ви пренасочи към началната страница на приложението.

3. Достъп до системата

След успешна инсталация можете да влезете в системата чрез root профила, който сте създали, и да започнете използването на всички налични функционалности за управление на доставки.