

Міністерство науки і освіти України
Житомирський державний технологічний університет

Кафедра програмного забезпечення систем

КУРСОВИЙ ПРОЕКТ

з дисципліни «Бази даних»

на тему: _____

Студента (ки) _____ курсу _____ групи
спеціальності 7.05010301 «Програмне забезпечення
систем»

(прізвище та ініціали)

Керівник: _____

Національна шкала _____
Кількість балів: _____ Оцінка: ECTS _____

Члени комісії:

(підпис)

(прізвище та ініціали)

(підпис)

(прізвище та ініціали)

(підпис)

(прізвище та ініціали)

м. Житомир – 2015 рік

Зміст

Зміст	2
Вступ	3
1. ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ	4
1.1. Аналіз інформаційних потреб та визначення предметної області дослідження	4
1.2. Обґрунтування вибору засобів реалізації.	5
2. ПРОЕКТУВАННЯ БАЗИ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ РОБОТИ	8
2.1. Аналіз інформаційних процесів.....	8
2.2. Проектування структури бази даних.....	9
2.3. Алгоритми обробки даних в системі.....	13
2.3.1. Розглянемо процес додавання даних до таблиці «workers»:	13
2.3.2. Розглянемо процес додавання даних до таблиці «packages»:	13
2.3.3. Розглянемо процес відстеження прострочених відправлень.	14
2.3.4. Розглянемо функцію генерування номера накладної.....	14
3. РЕАЛІЗАЦІЯ СИСТЕМИ ОБРОБКИ ДАНИХ СЛУЖБИ ДОСТАВКИ	15
3.1. Проектування інтерфейсу обробки даних	15
Рис 3.1.1 Форма авторизації.....	15
3.2. Реалізація операцій обробки даних в БД	21
3.3. Організація звітності системи.....	29
4. АДМІНІСТРУВАННЯ БАЗ ДАНИХ	31
4.1. Розробка заходів захисту інформації в БД	31
4.2. Налаштування параметрів роботи MySQL-сервера.....	32
ВИСНОВКИ	33
ЛІТЕРАТУРА.....	34
ДОДАТКИ	35

Вступ

В наш час інформаційні технології розвиваються дуже швидко, зокрема, у категорії збереження і обробки даних. Стрімко росте кількість нових технологій, що дозволяють створювати програмні продукти які мають бути достатньо швидкими, надійними та гнучкими для розширення і масштабування. Вони мають надавати можливість для швидкого оперування даними та пошуку.

Метою курсового проекту є дослідження особливостей проектування та реалізації бази даних служби доставки. Розглянути основні засоби, підходи та технології, що дозволяють створити систему для швидкого доступу до даних, та додавання нових даних до бази даних.

Об'єктом дослідження є підходи, взаємодії, методи та засоби проектування системи управління базою даних служби доставки.

Завданням на курсову роботу є:

- аналіз теоретичних засад проектування та реалізації систем на основі баз даних;
- визначення інформаційних потреб предметної області дослідження;
- проектування бази даних за визначеною предметною областю;
- реалізація БД та графічних засобів інформаційної системи.

						Арк.
						3
Змн.	Арк.	№ докум.	Підпис	Дата		

1. ТЕОРЕТИЧНИЙ АНАЛІЗ ІНФОРМАЦІЙНИХ ПОТОКІВ ТА ОСОБЛИВОСТЕЙ ПРЕДМЕТНОЇ ОБЛАСТІ ДОСЛІДЖЕННЯ

1.1. Аналіз інформаційних потреб та визначення предметної області дослідження

Щоб створити продукт, який зможе швидко оперувати великими об'ємами даних, бути масштабованим та легко розширюваним, потрібно правильно проаналізувати предметну область та правильно вибрати основні засоби для реалізації кінцевого програмного продукту.

Система управління базою даних служби доставки призначена для збору, обробки та відображення даних необхідні для реєстрації та пошуку інформації про відправлення. Система дозволяє реєструвати унікальних клієнтів для можливості відновлення історії відправлень. Система надає можливість реєструвати нові відправлення лише робітникам відділення. Також в системі реалізована можливість переглядати та зберігати інформація про відділення, зокрема усіх робітників, найпродуктивнішого робітника, доходи відділення за кожен місяць, та відображення інформації про відправлення. Також в системі присутня авторизація користувачів, для того, щоб надати різні можливості та права працівникам служби доставки.

Основні функції, які надає наш програмний продукт є:

- можливість дистанційної роботи з робочих станцій локальної та глобальної мережі;
- постійний доступ користувачів до БД;
- аутентифікацію користувачів та захист інформації від несанкціонованого доступу;
- надійне збереження даних та можливість відновлення даних у випадку непередбачуваних збоїв системи;
- створення архівів даних, що не використовуються;
- Реєстрацію відправлень;

						Арк.
						4
Змн.	Арк.	№ докум.	Підпис	Дата		

- Перевірку вхідних даних;
- Збереження вхідних даних;
- Розрахунок вартості доставки;
- Пошук та відбір відправлень, відділень та клієнтів за мультикритеріями (місто, номер накладної, мобільний телефон);
- Формування звітності щодо відділень та відправлень;

Користувачі(в залежності від рівня доступу) мають можливість додавати нових людей до черги отримання закордонного паспорту, зареєструвати новий закордонний паспорт, та перегляд необхідної інформації для прийняття подальших рішень.

Всі права на редагування таблиць бази даних, та маніпулювання інформацією, яка не належить до таблиць, що не відносяться до реєстрації закордонних паспортів належить лише адміністратору бази даних.

1.2. Обґрунтування вибору засобів реалізації.

Аналіз та вибір СУБД проведемо з урахуванням того, що нам потрібна серверна СУБД з високою надійністю, можливістю розмежень прав доступу та підтримкою багатокористувацького доступу до БД. Обчислювальна техніка працюватиме під керівництвом ОС Windows. В таблиці наведено дані порівняння СУБД Microsoft SQL Server та Oracle.

Характеристика	Microsoft SQL Server	Oracle	MySQL
Адміністративне керування	Добре	Відмінно	Добре
Графічні інструменти	Відмінно	Добре	Добре
Простота обслуговування	Відмінно	Відмінно	Відмінно
Механізм даних	Добре	Відмінно	Добре
Робота с декількома ЦП	Задовільно	Відмінно	Добре
Функції з'єднання і вибір індексів	Відмінно	Відмінно	Відмінно
Одночасний доступ декількох користувачів	Добре	Відмінно	Відмінно
Обробка даних мультимедіа	Плохо	Відмінно	Задовільно

Підключення к Web	Задовільно	Відмінно	Відмінно
Повнотекстовий пошук	Добре	Відмінно	Відмінно
Функціональна сумісність	Добре	Добре	Добре
Інтеграція з іншими СУБД	Добре	Добре	Добре
Єдина реєстрація	Добре	Добре	Добре
Робота під керівництвом ОС	Задовільно	Добре	Добре
Можливості програмування	Задовільно	Відмінно	Добре
Процедури, що зберігаються та тригери	Добре	Відмінно	Добре
Вбудована мова програмування	Задовільно	Відмінно	Добре
Побудова БД	Добре	Відмінно	Добре
Мова SQL	Відмінно	Відмінно	Відмінно
Підтримка об'єктно-орієнтованої парадигми	Задовільно	Відмінно	Добре
Робота у режимі віддаленого доступу	Відмінно	Відмінно	Добре
Тиражування	Відмінно	Відмінно	Відмінно
Розподілена обробка транзакцій	Відмінно	Відмінно	Відмінно
Дистанційне адміністрування	Добре	Відмінно	Відмінно
Організація сховищ даних і підготовка звітів	Відмінно	Відмінно	Відмінно
Засоби завантаження	Відмінно	Відмінно	Відмінно
Засоби аналізу	Відмінно	Відмінно	Відмінно

Таблиця 1.2.1

Порівняння з вимогами ТЗ

Необхідні умови	Microsoft Sql Server	Oracle	MySQL
Підтримка технології клієнт-сервер	+	+	+
Контроль цілісності БД	+	+	+
Наявність графічного інтерфейсу для адміністраторів БД	+	+	+
Імпорт та експорт таблиць БД	+	+	+
Контроль доступу до даних. Аутентифікація засобами СУБД	+	+	+
Відлагоджений механізм реплікації даних	+	+	+
Підтримка утиліт резервування даних	+	+	+
Імпорт та експорт таблиць БД	+	+	+

Таблиця 1.2.2

Враховуючи дані в таблиці порівняння, ми можемо зробити висновок, що усі представлені СУБД підходять для вирішення поставленої задачі. Але, було

						Арк.
						6
Змн.	Арк.	№ докум.	Підпис	Дата		

вирішено використовувати MySQL.

Переваги MySQL:

1. MySQL має зручний інтерфейс взаємодії з БД, завдяки додатку MySQL Workbench.
2. MySQL має API для багатьох сучасних мов програмування, як приклад, C/C++, Java(JDBC), Python, PHP (PDO).
3. Так як розробник має досвід розробки БД завдяки MySQL, то швидкість розробки системи та якість буде значно вищою в порівнянні з іншими СУБД.

Для створення додатку будемо використовувати PHP в якості мови програмування та MVC фреймворк Laravel.

						Арк.
						7
Змн.	Арк.	№ докум.	Підпис	Дата		

2. ПРОЕКТУВАННЯ БАЗИ ДАНИХ ЗА НАПРЯМКОМ КУРСОВОЇ РОБОТИ

2.1. Аналіз інформаційних процесів

Вхідними даними для системи є:

- Вхідними даними є дані про відділення, особу відправника та отримувача, інформація про відправлення(вага, тип відправлення, тип пакування), дані співробітника відділення.

Вихідними даними є:

- Зареєстроване відправлення та інформація про нього;
- Відомості про відділення;
- Формування накладної про відправлення;
- Формування звіту за відділенням;

Проведений аналіз предметної області дослідження дозволив сформувати структурну схему функціональних модулів системи керування відправленнями служби доставки. рис. 2.1.1.

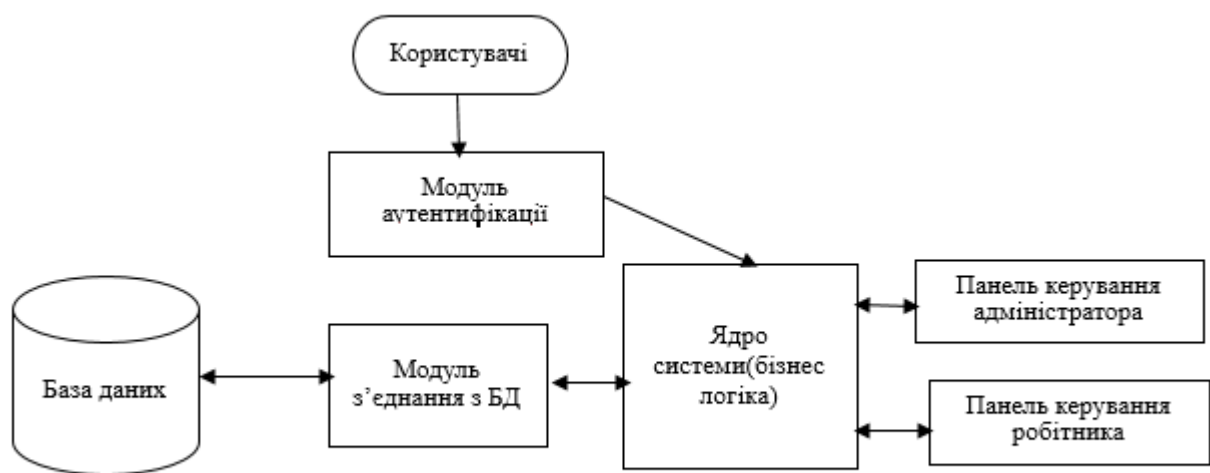


Рис. 2.1.1. Структура системи керування відправленнями служби доставки

2.2. Проектування структури бази даних

Внаслідок проектування до бази даних включено таблиці:

- packages;
- constants;
- distance;
- departments;
- packing_types;
- package_types;
- clients;
- workers;
- users;
- expired_packages;

Для збереження основних даних про відправлення використовується таблиця «packages». В дану таблицю вводяться такі дані як ПІБ відправника та отримувача, назва відділень отримувача та відправника, номер телефону відправника та отримувача, платник доставки (відправник або отримувач), тип відправлення, тип упакування, вага. Структура таблиці наведена нижче:

Таблиця 2.2.1

Структура таблиці "packages"

Назва	Тип даних	ПК	ЗК	Опис поля
id	int	+	-	Ідентифікатор
ttn	Varchar(45)	-	-	Номер накладної
sender_id	Int	-	+	Дані відправника
receiver_id	Int	-	+	Дані отримувача
sender_department_id	Int	-	+	Дані відділення відправника
Receiver_department_id	Int	-	+	Дані відділення отримувача
packing_type_id	Int	-	+	Тип упакування
package_type_id	Int	-	+	Тип відправлення
shipping_cost	float	-	-	Вартість доставки
weight	float	-	-	Вага
payer	Enum	-	-	Платник доставки
Registered_at	Datetime	-	-	Дата реєстрації відправлення
Arrived_at	Datetime	-	-	Дата прибуття відправлення
Package_status	Enum	-	-	Статус відправлення
Given_att	Datetime	-	-	Дата видачі відправлення
Orient_date	Datetime	-	-	Орієнтована дата прибуття

						Арк.
						9
Змн.	Арк.	№ докум.	Підпис	Дата		

Для збереження констант, які необхідні для оперування з базою даних використовується таблиця “constants”. Структура таблиці наведена нижче:

Таблиця 2.2.2

Структура таблиці " constants "

Назва	Тип даних	ПК	ЗК	Опис поля
name	Varchar(20)	-	-	Назва константи
value	Int	-	-	Значення константи

Таблиця «departments» використовується для зберігання даних відділення. Структура таблиці наведена нижче:

Таблиця 2.2.3

Структура таблиці "departments"

Назва	Тип даних	ПК	ЗК	Опис поля
id	Int	+	-	Ідентифікатор відділення
city_id	Int	-	+	Інформація про місто
name	Varchar(45)	-	-	Назва
address	Varchar(45)	-	-	Вулиця
phone	Varchar(12)	-	-	Телефон
weight_limit	Int	-	-	Обмеження ваги

Таблиця «workers» використовується для збереження інформації про робітників. Структура таблиці наведена нижче:

Таблиця 2.2.4

Структура таблиці "workers "

Назва	Тип даних	ПК	ЗК	Опис поля
id	Int	+	-	Ідентифікатор робітника
fio	Varchar(255)	-	-	ПІБ
department_id	Int	-	+	Ключ відділення
count_packages	Int	-	-	Кількість оброблених відправлень
auth_id	Int	-	+	Обліковий запис входу в систему
work_since	Datetime	-	-	Дата прийому на роботу

Таблиця «clients» використовується для зберігання даних про клієнтів. Структура таблиці наведена нижче:

							Арк.
							10
Змн.	Арк.	№ докум.	Підпис	Дата			

Таблиця 2.2.5

Структура таблиці "clients"

Назва	Тип даних	ПК	ЗК	Опис поля
Id	int	-	+	Ідентифікатор клієнта
name	Varchar(120)	-	-	ПІБ
phone	Int	-	-	Номер клієнта (моб.)

Таблиця «packing_type» зберігає дані про тип упакування відправлення. Структура таблиці наведена нижче:

Таблиця 2.2.6

Структура таблиці "packing_type"

Назва	Тип даних	ПК	ЗК	Опис поля
Id	Int	+	-	Ідентифікатор типу
Name	Varchar(120)	-	-	Опис типу
Extra_cost	Float	-	-	Додаткова вартість

Таблиця «package_type» зберігає дані про тип відправлення. Структура таблиці наведена нижче:

Таблиця 2.2.7

Структура таблиці "package_type"

Назва	Тип даних	ПК	ЗК	Опис поля
Id	int	+	-	Ідентифікатор типу
Name	Varchar(120)	-	-	Опис типу
Extra_cost	Float	-	-	Додаткова вартість

Таблиця «expired_packages» зберігає дані про відправлення, у яких пройшов термін безкоштовного зберігання на складі. Структура таблиці наведена нижче:

Таблиця 2.2.8

Структура таблиці " expired_packages "

Назва	Тип даних	ПК	ЗК	Опис поля
id	Int	+	-	Ідентифікатор відправлення
Expired_from	Datetime	-	-	Дата

Таблиця «distance» зберігає дані про відстань між містами, та назву міста, представлена у вигляді матриці відстаней. Структура таблиці наведена нижче:

Таблиця 2.2.9

Структура таблиці "distance"

Назва	Тип даних	ПК	ЗК	Опис поля
Id	Int identity	+	-	Id міста
Region_name	Varchar(30)	-	-	Назва міста
Reg1	Int	-	-	Значенн відстані
Reg2				
Reg3				
Reg4				
Reg5				

Таблиця «users» зберігає дані, що потрібні для авторизації, реєстрації користувачів та роль користувача (за замовчуванням користувач адміністратор). Структура таблиці наведена нижче:

Таблиця 2.2.10

Структура таблиці "users"

Назва	Тип даних	ПК	ЗК	Опис поля
Id	Int	+		Ідентифікатор облікового запису
Login	Varchar(90)		-	Логін
Password	Varchar(60)		-	пароль
Worker_id	Int	-	+	Дані робітника

2.3. Алгоритми обробки даних в системі

2.3.1. Розглянемо процес додавання даних до таблиці «workers»:

- Адміністратор заповнює усі основні дані про робітника (ПІБ, відділення, логін та пароль входу в систему) і підтверджує його створення;
- Відбувається виклик процедури для вставки даних до таблиці, в якості параметрів передаються основні дані про робітника;
- Процедура реалізує транзакцію, де спочатку ми додаємо дані про робітника у таблицю “workers”, далі додаємо логін та пароль у таблицю облікових записів “users” та зв’язуємо їх за ідентифікатором робітника.
- Під час вставки даних спрацьовує тригер, який встановлює дату прийому робітника;

2.3.2. Розглянемо процес додавання даних до таблиці «packages»:

- Робітник заповнює всі основні дані необхідні для реєстрації відправлення;
- Відбувається виклик процедури для реєстрації відправлення;
- У процедурі реалізовано механізм транзакцій для коректного збереження даних. У процедурі ми знаходимо відстань між відділеннями, та знаходимо вартість перевезення (помноживши відстань на вартість перевезення за кожних 10 кілометрів). Далі, на основі вибраних способів пакування та типу відправлення формується остаточна вартість доставки. Потім перевіряємо наявність клієнтів, якщо клієнт існує, то ми отримуємо його ідентифікатор та передаємо на вхід в таблицю відправлень, інакше ми реєструємо клієнта та отримуємо його ідентифікатор. Після всіх перевірок вставляємо дані в таблицю відправлень.
- Під час вставки даних спрацьовує тригер, який перевіряє вхідні дані, у якому

						Арк.
						13
Змн.	Арк.	№ докум.	Підпис	Дата		

ми викликаємо функцію генерування накладної, перевіряємо коректність ваги, встановлюємо дату реєстрації, та статус “Зареєстровано”.

2.3.3. Розгляне процес відстеження прострочених відправлень.

В базі даних реалізовано MySQL подію яка спрацьовує кожен. Дана подія перевіряє наявність відправлень які прибули, і зберігаються на складі довше зазначеного терміну. Якщо такі відправлення виявлені, подія занесе ідентифікатор відправлення в таблицю “expired_packages”, в таблиці “expired_packages” спрацює тригер який встановить дату від якої відправлення вважається простроченим.

2.3.4. Розглянемо функцію генерування номера накладної.

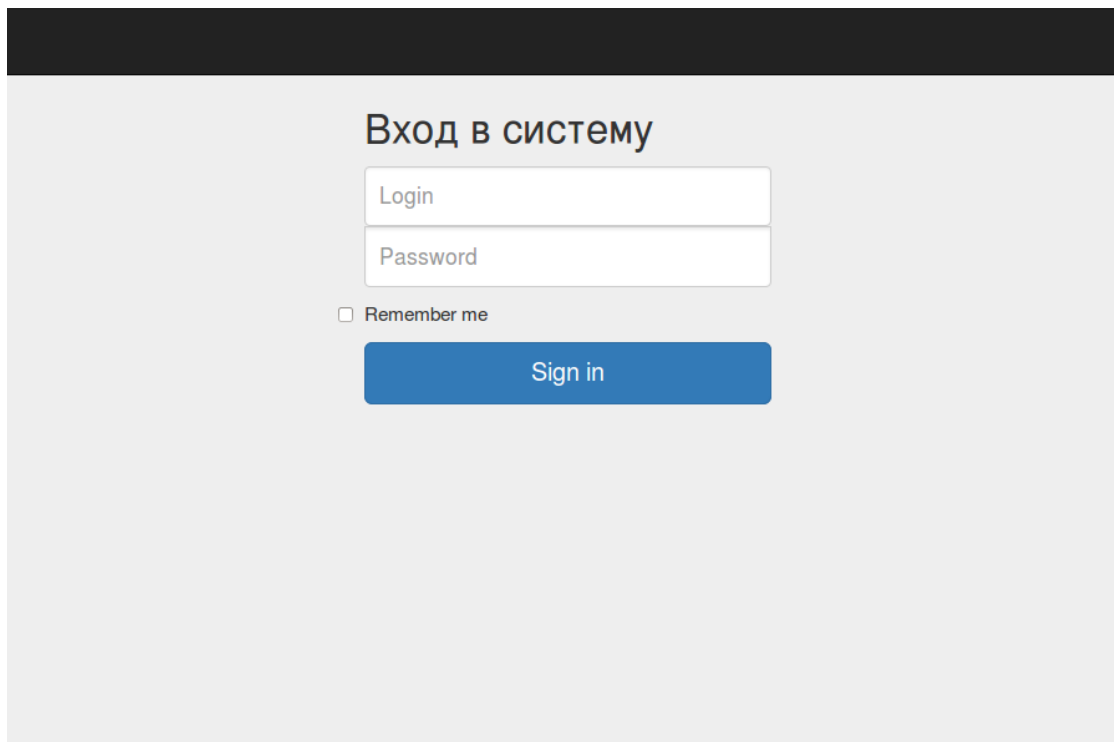
Вхідними параметрами функції є ідентифікатор відправлення, ідентифікатор відділення відправника та ідентифікатор відділення отримувача. Результатом виконання функції є рядок який містить по дві літери міста відправника та отримувача, ідентифікатор відділень отримувача та відправника та ідентифікатор відправлення.

						Арк.
						14
Змн.	Арк.	№ докум.	Підпис	Дата		

3. РЕАЛІЗАЦІЯ СИСТЕМИ ОБРОБКИ ДАНИХ СЛУЖБИ ДОСТАВКИ

3.1. Проектування інтерфейсу обробки даних

Після запуску програми з'являється вікно для аутентифікації користувача (рис. 3.1.1).



Вход в систему

Login

Password

☐ Remember me

Sign in

Рис 3.1.1 Форма авторизації

Після того, як ми успішно пройшли авторизацію, ми потрапляємо до головного вікна програми. В залежності від ролі користувача, ми побачимо вікно з панеллю керування адміністратора із відкритим списком відділень (Рис 3.1.2) або з панеллю керування робітника із відкритою формою реєстрації відправлення.(Рис 3.1.7)

						Арк.
						15
Змн.	Арк.	№ докум.	Підпис	Дата		

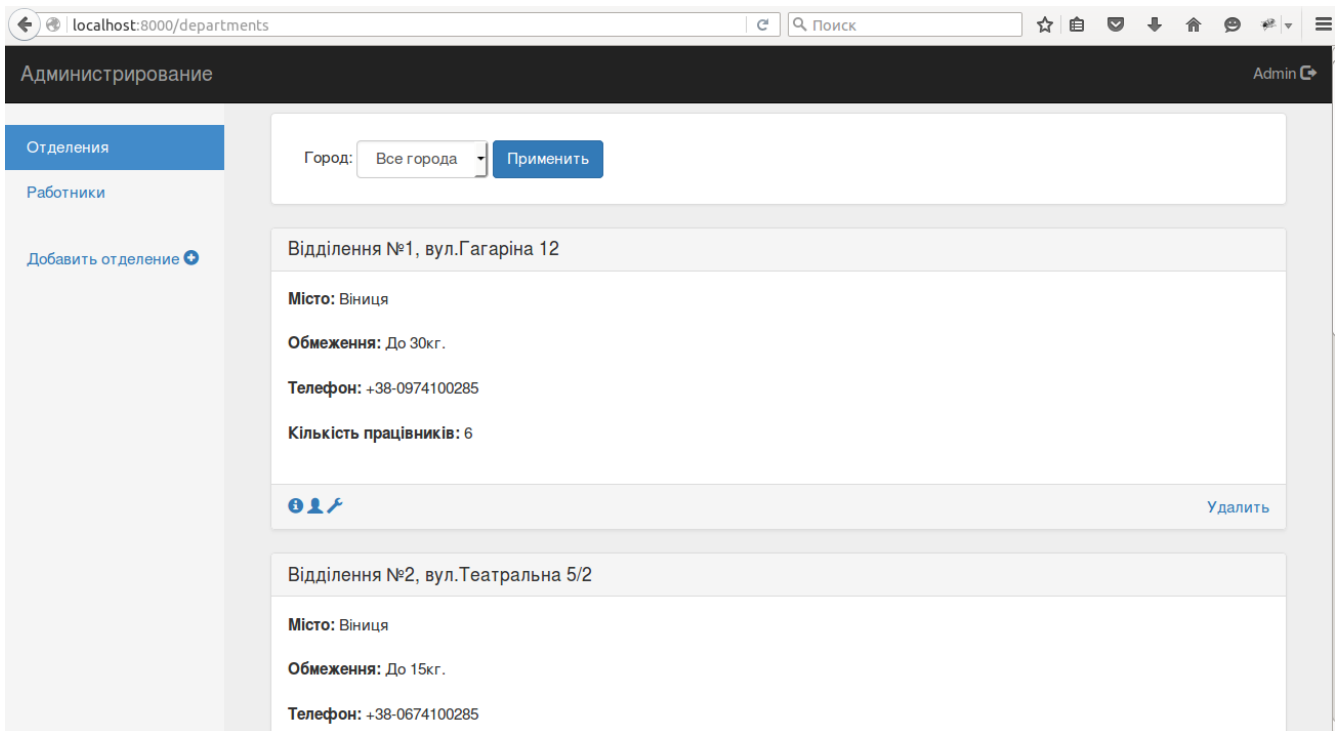


Рис. 3.1.2. Стартова сторінка адміністратора

Стартова сторінка адміністратора містить інформацію про усі відділення, передбачена можливість фільтрації відділень за містом. Також користувач може переглянути детальну інформацію про відділення. (Рис 3.1.3)

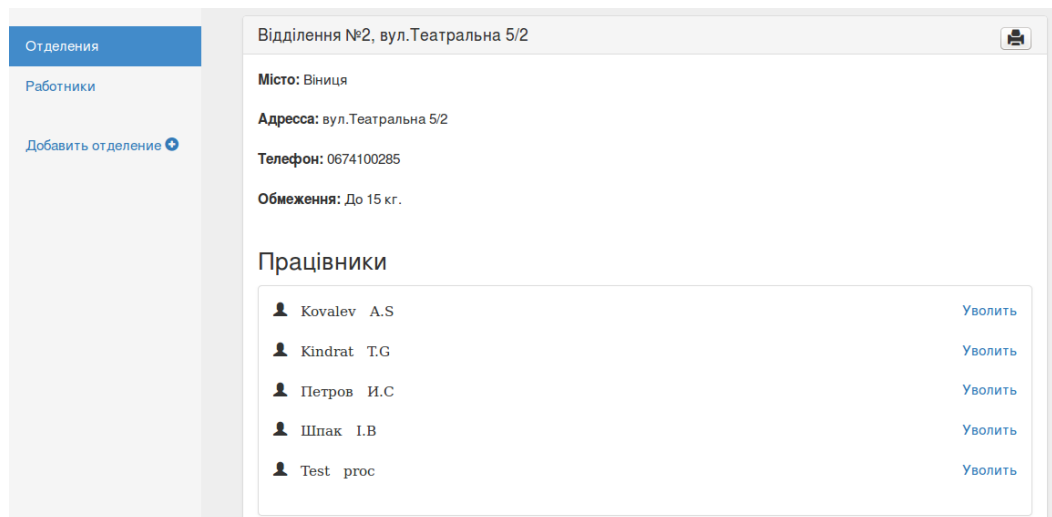


Рис. 3.1.3. Детальна інформація про відділення

На сторінці детальної інформації розміщені імена робітників, що працюють на відділенні, та реалізована можливість звільнення робітника.

Також на сторінці відображається діаграма прибутків відділення за кожен

						Арк.
						16
Змн.	Арк.	№ докум.	Підпис	Дата		

місяць.

Вибравши пункт “робітники” адміністратор перейде до списку усіх працівників. (Рис 3.1.4)

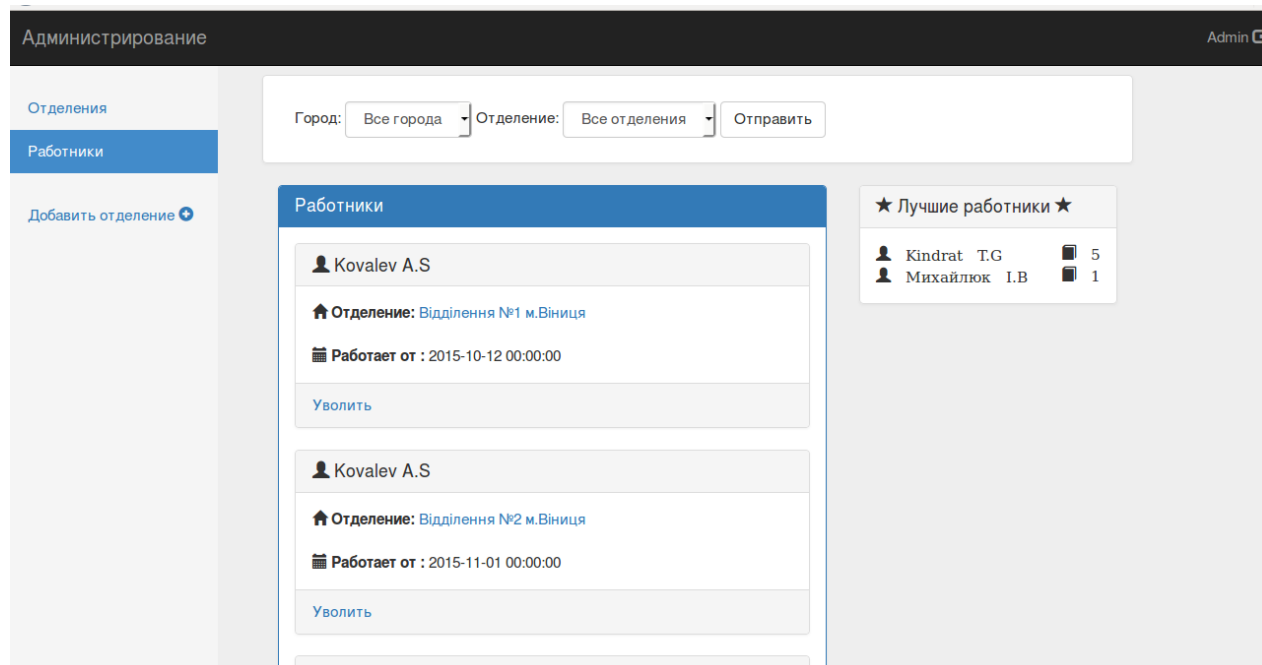


Рис 3.1.4 Список робітників

У даному вікні буде відображатись інформація про робітника, зокрема його ім'я, відділення на якому працює та дата початку роботи. Також передбачено відображення найкращих робітників усієї системи.

Натиснувши на іконку певного відділення “назначити працівника”, адміністратор потрапить на форму реєстрації нового робітника.(Рис 3.1.5)

Обравши пункт “Додати відділення” адміністратор потрапить до форми додавання нового відділення.(Рис.3.1.6)

					Арк. 17
Змн.	Арк.	№ докум.	Підпис	Дата	

Регистрация работника

Ф.И.О	<input type="text" value="Иванов И.И"/>
Логин	<input type="text" value="Логин"/>
Пароль	<input type="password"/>
<input type="button" value="Зарегистрировать"/>	

Рис 3.1.5 Форма реєстрації нового працівника

Администрирование Admin

Отделения
Работники
Добавить отделение +

Регистрация отделения

Название	<input type="text" value="Отделение №"/>
Віниця	<input type="text"/>
Вулиця	<input type="text" value="Вулиця"/>
Телефон	<input type="text" value="Телефон"/>
Ограничение веса	<input type="text" value="0"/>
<input type="button" value="Зарегистрировать"/>	

localhost:8000/workers

Рис 3.1.6 Форма реєстрації нового відділення

						Арк.
						18
Змн.	Арк.	№ докум.	Підпис	Дата		

Якщо користувач увійде як робітник, йому відобразиться вікно де буде відображено назву відділення та ПІБ робітника. При старті робітнику буде відображено форму реєстрації нового відправлення. (Рис.3.1.7)

Регистрация посылки

Інформація про відправника

Телефон:

Назва фірми або П.І.Б особи:

☐ Плательщик

Інформація про отримувача

Місто:

Філія одержувач:

Телефон:

Назва фірми або П.І.Б особи:

☐ Плательщик

Інформація про відправлення

Тип відправлення

☐ Інше

☐ Електроніка

☐ Цінні папери

☐ Крихіт предмети

Тип пакування

☐ Пакет фірмовий

☐ Картонна коробка мала

☐ Картонна коробка велика

☐ Пакувальна стрічка

☐ Дерев'яний ящик

☐ Упаковано відправником

Вага :

Зареєструвати

Рис 3.1.7 Форма реєстрації нового відправлення

При введенні номера телефону клієнта система здійснить пошук клієнта в базі даних, і заповнить відповідне поле іменем знайденого клієнта. Також система відобразить у випадяючому списку тільки ті міста у яких присутні відділення після чого у випадяючому списку відділень відобразяться відділення обраного міста.

Перейшовши у категорію “відправлення” робітнику відобразиться список усіх відправлень що прийшли на відділення, де він зможе відмітити відправлення як видане, або дізнатись детальну інформація про відправлення. Та роздрукувати або зберегти накладну. (Рис.3.1.8)

						Арк.
						19
Змн.	Арк.	№ докум.	Підпис	Дата		

Відправлення №: 1232112
Відділення відправник: м.Житомир, Відділення 12, вул.Миру 12
Отримувач: Самойленко В.А

Відправлення №: 1232112
Відділення відправник: м.Житомир, Відділення 12, вул.Миру 12
Отримувач: Самойленко В.А

Відправлення №: 1232112
Відділення відправник: м.Житомир, Відділення 12, вул.Миру 12
Отримувач: Самойленко В.А

Рис 3.1.8 Список вхідних відправлень

Також робітнику реалізована можливість швидкого пошуку посилок за номером накладної. В результаті успішного пошуку відобразиться детальна інформація про відправлення. (Рис.3.1.9)

Посилка №BI223ДН
Зареєстровано: 2015-12-27 14:41:13
Орієнтована дата прибуття: 2015-12-16
Інспектор: Kindrat T.G
Статус:REGISTRED
Вага: 0 Кг.
Вартість доставки: 213.5
Платник: SENDER

Інформація про отправителя

Клиент Отправитель 2
+380-123123123
Вінниця
Відділення №2, вул.Театральна 5/2

Інформація про получателя

Клиент 2 Плучатель
+380-453453453
Житомир
Відділення №11, Черняковського 103

Рис 3.1.9 Детальна інформація про відправлення

						Арк.
						20
Змн.	Арк.	№ докум.	Підпис	Дата		

3.2. Реалізація операцій обробки даних в БД

Для доступу до бази даних в Laravel можна використовувати декілька підходів. У нашому проєкті ми будемо використовувати фасад DB та Eloquent ORM.

Для прикладу реалізації виконання запитів між базою даних та додатком використовуючи Eloquent ORM розглянемо відображення, пошук, оновлення, вставку та видалення даних з таблиці «departments»:

Відображення усіх даних:

```
public function index()
{
    $dept = Department::all();
    return View::make('department.all', ['departments'=>$dept, 'selectedCity'=>0, 'category'=>1]);
};
```

Відображення детальної інформації про відділення:

```
public function show($id)
{
    $moneyPerMonth = DB::select('CALL
department_money_per_month_statistic(?)', array($id));
    $packagesCountStatistic = DB::select('CALL
sended_received_stats(?)', array($id));
    $best5Clients = DB::select('CALL best_5_clients(?)', array($id));
    $bestWorker = Worker::where('department_id', $id)-
>orderBy('packages_count', 'desc')->take(1)->get();
    $monthsSet = array('January', 'February',
    'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October',
    'November', 'December');
    $values = array(0,0,0,0,0,0,0,0,0,0,0,0);
    foreach($moneyPerMonth as $stat) {
        $values[array_search($stat->month, $monthsSet)]=$stat->value;
    };
    #return json_encode($months);
    $dept = Department::find($id);
    $workers = $dept->workers();
    $objMoney = json_encode($values);
    $objMonth = json_encode($monthsSet);
    $params = ['department'=>$dept, 'workers'=>$workers, 'months'=>$objMonth,
'money'=>$objMoney, 'category'=>4, 'packagesStatistic'=>array_shift($packagesCountSta
tistic),
'moneyPerMonth'=>$moneyPerMonth, 'bestWorkers'=>$bestWorker,
'bestClients'=>$best5Clients, 'category'=>1];
    return View::make('department.detail', $params);
}
```

						Арк.
						21
Змн.	Арк.	№ докум.	Підпис	Дата		

Оновлення відділення:

```
public function postEdit($id) {
    $rules =
array('name'=>'required|min:6|max:100', 'address'=>'required|min:6|max:100',
    'weight_limit'=>'required', 'city_id'=>'required', 'phone'=>'required');
    $validator = Validator::make(Input::all(), $rules);
    if($validator->fails()) {
        return Redirect::route('new-department')->withErrors($validator);
    }

    $department = Department::find($id);
    $department->name = Input::get('name');
    $department->phone = Input::get('phone');
    $department->address = Input::get('address');
    $department->city_id = intval(Input::get('city_id'));
    $department->weight_limit = intval(Input::get('weight_limit'));
    $department->save();

    return Redirect::route('departments', ['category'=>1]);
}
```

Вставка:

```
public function postCreate() {
    $rules =
array('name'=>'required|min:6|max:100', 'address'=>'required|min:6|max:100',
    'weight_limit'=>'required', 'city_id'=>'required', 'phone'=>'required');
    $validator = Validator::make(Input::all(), $rules);
    if($validator->fails()) {
        return Redirect::route('new-department')->withErrors($validator);
    }

    $department = new Department();
    $department->name = Input::get('name');
    $department->phone = Input::get('phone');
    $department->address = Input::get('address');
    $department->city_id = intval(Input::get('city_id'));
    $department->weight_limit = intval(Input::get('weight_limit'));
    $department->save();

    return Redirect::route('departments');
}
```

Видалення:

```
public function destroy($id)
{
    $deleted = Department::find($id);
    $deleted->delete();
}
```

Розглянемо приклад використання фасаду DB для виклику процедур та виконання запитів.

					Арк. 22
Змн.	Арк.	№ докум.	Підпис	Дата	

Виклик процедури реєстрації відправлення:

```
public function packageNew() {
    $params = null;
    $rules =
array('sender_name'=>'required', 'sender_phone'=>'required|min:9|max:9',
    'receiver_name'=>'required', 'receiver_phone'=>'required|min:9|max:9',
    'payer'=>'required', 'packing_type'=>'required', 'package_type'=>'required',
    'department'=>'required', 'weight'=>'required');
    $validator = Validator::make(Input::all(), $rules);
    if($validator->fails()){
        return Redirect::route('packages')->withErrors($validator);
    }

    $usr = Auth::user();
    if($usr->isWorker()){

        $currentDepartment = $usr->worker->department->id;
        $currentWorker = $usr->worker->id;
        $params = array((int)Input::get('sender_phone'), Input::get('sender_name'),
            (int)Input::get('receiver_phone'), Input::get('receiver_name'),
            $currentDepartment, Input::get('department'), (int)Input::get('weight'),
            (int)Input::get('packing_type'), (int)Input::get('package_type'),
            $currentWorker, (int)Input::get('payer'));
        DB::statement('CALL regis_package(?, ?, ?, ?, ?, ?, ?, ?, ?, ?
        ?)', $params);
    }
    return Redirect::route('departments');
}
```

Виконання запитів використовуючи фасад:

```
public function getJSONDepartments($city_id){

    $query = "
select departments.id as ID, departments.name as VALUE
from departments WHERE city_id = ?";
    $cities = DB::select($query, array($city_id));
    $obj = json_encode($cities);
    return $obj;
}

public function getJSONRegions(){
    $query = "
select distance.id as ID, distance.region_name as VALUE
from distance";
    $cities = DB::select($query);
    $obj = json_encode($cities);
    return $obj;
}
```

Для отримання інформації про прибуток відділень була реалізована процедура

“ department_money_per_month_statistic”

```
DELIMITER //
CREATE PROCEDURE `department_money_per_month_statistic` (IN dept_id
INTEGER)
BEGIN
```

						Арк.
						23
Змн.	Арк.	№ докум.	Підпис	Дата		

```

select sum(packages.shipping_cost),
elt(month(registred_at), 'January', 'February',
'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October',
'November', 'December') as MonthName
from packages where sender_department_id = dept_id group by
month(registred_at);
END//

```

Для отримання статистики відправлених та отриманих відправлень було створено процедуру “ department_send_receiv_statistic”

```

DELIMITER //
CREATE PROCEDURE `department_send_receiv_statistic` (IN dept_id INTEGER)
BEGIN
    SELECT (
        SELECT COUNT(*)
        FROM packages where sender_department_id = dept_id
    ) AS sended,
    (
        SELECT COUNT(*)
        FROM packages where receiver_department_id = dept_id
    ) AS received;
END//
DELIMITER //

```

Для реєстрації робітників було створено відповідну процедуру:

```

DELIMITER //
CREATE PROCEDURE worker_registration(IN n_fio varchar(150),
IN n_department INT, n_login VARCHAR(80), IN n_password VARCHAR(60))
BEGIN
    DECLARE nworker_id INT;
    DECLARE exit handler for sqlexception
    BEGIN
        -- ERROR
        ROLLBACK;
    END;
    DECLARE exit handler for sqlwarning
    BEGIN
        -- WARNING
        ROLLBACK;
    END;
    START TRANSACTION;
    SET nworker_id = (SELECT id from workers order by id desc limit 1);
    IF nworker_id IS NULL THEN
    SET nworker_id = 1;
    ELSE SET nworker_id = nworker_id + 1;
    END IF;
    INSERT INTO workers(fio,department_id,registred_at)
    VALUE(n_fio, n_department, current_date());
    INSERT INTO users(login, password,worker_id)
    VALUE(n_login, n_password, nworker_id);
    COMMIT;

```

						Арк.
						24
Змн.	Арк.	№ докум.	Підпис	Дата		

END//

Процедура реєстрації нових відправлень:

```
DELIMITER //
```

```
CREATE PROCEDURE `regis_package`  
(IN sender_phone INTEGER(9), IN sender_name VARCHAR(45),  
  IN receiver_phone INTEGER(9), IN receiver_name VARCHAR(45),  
  IN sender_dept INTEGER, IN receiver_dept INTEGER,  
  IN package_weight FLOAT, IN pking_type_id INTEGER,  
  IN pkg_type_id INTEGER, IN worker_id INTEGER, IN payer_code INTEGER(1))  
  
BEGIN  
  DECLARE total_cost FLOAT;  
  DECLARE packing_cost FLOAT;  
  DECLARE extra_cost FLOAT;  
  DECLARE depts_distance INTEGER;  
  DECLARE transporting_cost FLOAT;  
  DECLARE client_sender INTEGER;  
  DECLARE client_receiver INTEGER;  
  DECLARE prognos_date DATETIME;  
  DECLARE prognos_day INT;  
  
  DECLARE exit_handler_for_sqlexception  
  BEGIN  
    -- ERROR  
    ROLLBACK;  
  END;  
  
  DECLARE exit_handler_for_sqlwarning  
  BEGIN  
    -- WARNING  
    ROLLBACK;  
  END;  
  
  START TRANSACTION;  
  
  SET depts_distance = get_distance((SELECT departments.city_id FROM  
  departments  
  WHERE departments.id = sender_dept), (SELECT departments.city_id FROM  
  departments  
  WHERE departments.id = receiver_dept));  
  
  SET prognos_day = ROUND(depts_distance / (SELECT constants.const_value FROM  
  constants WHERE name = 'KILOMETRS_PER_DAY' limit 1));  
  
  IF (prognos_day = 0)  
  THEN  
    SET prognos_date = DATE_ADD(current_date(), INTERVAL 1 DAY);  
  ELSE  
    SET prognos_date = DATE_ADD(current_date(), INTERVAL prognos_day DAY);  
  END IF;  
  SET packing_cost = (SELECT packing_type.cost FROM packing_type  
                      WHERE packing_type.id = pking_type_id);
```

						Арк.
						25
Змн.	Арк.	№ докум.	Підпис	Дата		

```

SET extra_cost = (SELECT package_type.extra_price FROM package_type
                  WHERE package_type.id = pkg_type_id);

SET transporting_cost = (SELECT constants.const_value
FROM constants WHERE name = "MONEY_PER_10_KM" limit 1) *(depts_distance /
10);

SET total_cost = transporting_cost + extra_cost + packing_cost;

SET client_sender = (SELECT clients.id FROM clients WHERE clients.phone =
sender_phone);

IF(client_sender IS NULL) THEN
INSERT INTO clients(name, phone)VALUE(sender_name, sender_phone);
SET client_sender = (SELECT clients.id FROM clients WHERE clients.phone =
sender_phone);
END IF;

SET client_receiver = (SELECT clients.id FROM clients
                      WHERE clients.phone = receiver_phone);

IF(client_receiver IS NULL) THEN
INSERT INTO clients(name, phone)VALUE(receiver_name, receiver_phone);
SET client_receiver = (SELECT clients.id FROM clients
                      WHERE clients.phone = receiver_phone);
END IF;

INSERT INTO packages(orient_date,weight, shipping_cost, payer, sender_id,
receiver_id,
sender_department_id, receiver_department_id, package_type_id,
packing_type_id, inspector_id)
VALUE(prognos_date,package_weight, total_cost, payer_code, client_sender,
client_receiver,
      sender_dept,receiver_dept,pkg_type_id, pking_type_id, worker_id);

COMMIT;
END//

```

Для забезпечення цілісності даних, система реалізує перевірку даних. Наприклад, під час вставки даних до таблиці «packages» відбуваються перевірки: перевірка належності робітника до відділення відправника, перевірка валідності, ваги відправлення, та встановлення дати відправки. Якщо одна із умов виконується, то вставка даних до таблиці не відбувається. Приклад триггеру:

```

DELIMITER //
CREATE TRIGGER before_insert_packages
BEFORE INSERT ON packages
FOR EACH ROW
BEGIN

```

						Арк.
						26
Змн.	Арк.	№ докум.	Підпис	Дата		

```

DECLARE msg VARCHAR(255);
DECLARE dep_wlimit FLOAT;
DECLARE worker_dept_id INTEGER;
DECLARE package_id INT;

SET worker_dept_id = (SELECT workers.department_id
FROM workers WHERE workers.id = NEW.inspector_id);

SET dep_wlimit = (SELECT departments.weight_limit FROM departments
WHERE departments.id = NEW.receiver_department_id);

SET package_id = (SELECT id from packages GROUP BY id DESC LIMIT 1);
IF package_id IS NULL then
SET package_id = 1;
ELSE SET package_id = package_id + 1;
END IF;
SET NEW.ttn = generate_TTN(package_id, NEW.sender_department_id,
NEW.receiver_department_id);

SET NEW.registred_at = NOW();
SET NEW.package_status = "REGISTRED";

IF ((NEW.weight > dep_wlimit) OR (NEW.weight < 0)) THEN
SET msg = "Not correct weight!";
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
END IF;
IF (NEW.sender_department_id != worker_dept_id) THEN
SET msg = "You are not from this department!";
SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = msg;
END IF;
UPDATE workers SET packages_count = packages_count + 1 WHERE id =
NEW.inspector_id;
END//

```

Для контролю прострочених відправлень було реалізовано відповідну подію:

```

CREATE EVENT expire_packages_checker
ON SCHEDULE
EVERY 1 DAY
STARTS '2015-12-31 00:00:00'
DO
INSERT INTO expired_packages(package_id)
(select packages.id from packages
left join expired_packages
on packages.id = expired_packages.fackage_id
where expired_packages.package_id IS NULL
AND
TIMESTAMPDIFF(DAY,packages.registred_at, curdate()) >
(SELECT const_value from constants WHERE name = 'EXPIRED_AFTER_DAYS')
AND packages.givent_at IS NULL);

```

						Арк.
						27
Змн.	Арк.	№ докум.	Підпис	Дата		

Щоб оптимально використовувати ресурси збереження, було реалізовано подію видалення застарілих відправлень:

```
CREATE EVENT clear_old_packages
ON SCHEDULE
    EVERY 1 DAY
    STARTS '2015-12-30 00:00:00'
DO
    DELETE FROM packages WHERE
TIMESTAMPDIFF(DAY,packages.registred_at, curdate()) >
(SELECT const_value from constants WHERE name = 'DELETE_AFTER_DAYS' LIMIT
1)
AND (packages.givent_at IS NOT NULL);
```

Для відзначення прибувщикх посилок реалізовано подію, яка міняє статус посилки в залежності від орієнтованої дати прибуття:

```
CREATE EVENT mark_packages_as_received
ON SCHEDULE
    EVERY 1 DAY
    STARTS '2015-12-31 00:00:00'
DO
    UPDATE packages SET package_status = 2
WHERE orient_date < curdate() and package_status = 1;
```

Для отримання відстані між містами була реалізована відповідна функція:

```
DELIMITER //
CREATE FUNCTION `get_distance` ( from_city INT, to_city INT)
RETURNS INT
LANGUAGE SQL
BEGIN
return (select
elt(to_city,reg1,reg2,reg3,reg4,reg5,
    reg6,reg7,reg8,reg9,reg10,
    reg11,reg12,reg13,reg14,reg15,
    reg16,reg17,reg18,reg19,reg20,
    reg21,reg22,reg23,reg24,reg25)
from distance where id = from_city);
END//
```

						Арк.
						28
Змн.	Арк.	№ докум.	Підпис	Дата		

3.3. Організація звітності системи

Кожен адміністратор може отримати детальний звіт за обраним ним відділенням. До звіту буде включено назву відділення, кількість відправлених та отриманих посилок, доходи за кожен місяць, найпродуктивнішого робітника та постійних клієнтів.(Рис.3.3.1)

м.Віниця,Відділення №2, вул.Театральна 5/2

Отправлено посилок:5

Получено посилок:0

Доходи :

December: 898.7999877929688

Лучший работник

Kindrat T.G обробчено = 3

Постоянные клиенты

Рис 3.3.1 Приклад звіту за відділенням

Кожен робітник має можливість роздрукувати накладну за обраним відправленням. У накладну буде включено інформація про відправника, отримувача, вартість доставки, представник відділення що реєстрував посилку, дату реєстрації.(Рис.3.3.2)

						Арк.
						29
Змн.	Арк.	№ докум.	Підпис	Дата		

Накладна № ВІ223ДН

Зареєстровано: 2015-12-27 14:41:13

Сотрудник: Kindrat T.G

Вес: 0

Стоимость доставки: 213.5

Информация отправителя

ФИО: Клиент Отправитель 2

Телефон: 123123123

Город: Віниця

Отделение: Відділення №2,вул.Театральна 5/2

Информация получателя

ФИО: Клиент 2 Плучатель

Телефон: 453453453

Город: Житомир

Отделение: Відділення №11,Черняхівського 103

Рис 3.3.2 Вигляд сформованої накладної за відправленням

						Арк.
						30
Змн.	Арк.	№ докум.	Підпис	Дата		

4. АДМІНІСТРУВАННЯ БАЗ ДАНИХ

4.1. Розробка заходів захисту інформації в БД

Отже, розглянемо категорії користувачів служби доставки: адміністратор, робітник та клієнти.

Найбільші права доступу до даних має адміністратор, що необхідні для організації роботи системи. Він може додавати, оновлювати та видаляти усі відділення, може звільняти та призначати робітників, має контроль над усіма процедурами, функціями та представленнями в системі. Також, адміністратор може переглядати статистику по усім відділенням.

Робітники мають право переглядати інформацію про відправлення, реєструвати відправлення, видавати відправлення (змінювати статус).

Клієнти мають право лише на перегляд інформації про відправлення.

Більш детально перелік об'єктів БД, доступ до яких надано для кожної групи користувачів наведено в табл. 4.1.1. На перетині рядків і стовпців зазначено дії, які може виконувати користувач даної категорії з заданими сутностями.

Для сутності “Відправлення”, “Відділення” та “Робітники”: 0 — доступ заборонено, 1 — дозволено перегляд, 2 — повний доступ до даних

Таблиця 4.1.1.

Матриця доступу для груп користувачів

	Відправлення	Робітники	Відділення
Адміністратор	2	2	2
Робітник	1	0	1
Клієнт	1	0	0

Додамо трьох користувачів для управління доступом до даних, використавши наступний скрипт:

```
CREATE USER poshtaAdmin  
IDENTIFIED BY 'secretpassword';
```

```
CREATE USER poshtaWorker  
IDENTIFIED BY 'secretpassword';
```

						Арк.
						31
Змн.	Арк.	№ докум.	Підпис	Дата		

```
CREATE USER poshtaClient
  IDENTIFIED BY 'secretpassword';

GRANT ALL ON poshta.* TO poshtaAdmin;

GRANT SELECT, INSERT, UPDATE ON poshta.packages TO poshtaWorker;

GRANT EXECUTE ON PROCEDURE client_package_info TO poshtaClient;
```

4.2. Налаштування параметрів роботи MySQL-сервера

По замовчуванню mysql server використовує кодування latina, перевірити це можна виконавши в консольному клієнті наступну команду:

```
SHOW VARIABLES LIKE 'char%';
```

Для того щоб сервер завантажувався із потрібним кодування, потрібно відредагувати файл /etc/mysql/my.cnf. В секцію [mysqld] додати наступні параметри:

```
skip-character-set-client-handshake
character-set-server = utf8
init-connect='SET NAMES utf8'
collation-server=utf8_general_ci
```

Також бажано встановити кодування для клієнта і mysqldump. Для цього в секціях [client] і [mysqldump] даємо параметр:

```
default-character-set=utf8
```

Перезапускаємо сервер.

						Арк.
						32
Змн.	Арк.	№ докум.	Підпис	Дата		

ВИСНОВКИ

Результатом виконання даного курсового проекту є надійна система керування відправленнями служби доставки.

Завдяки правильному підбору інструментів розробки бази даних, перед нами не виникло ніяких труднощів у реалізації.

Система є легко розширюваною, вона легко може бути доповнена різними засобами для роботи, вона проста у користування і не потребує високих характеристик апаратного забезпечення. Клієнт даної системи коректно працює у всіх сучасних браузерях.

Завдяки системі обробки помилок, транзакціям та тригерам бази даних, ми гарантуємо забезпечення збереження цілісності даних.

Розроблена система є готовим програмним продуктом, що містить всі необхідні модулі для зручної роботи. Зокрема додавання, оновлення, видалення, пошуку даних за різними критеріями та можливістю друкування звітності.

						Арк.
						33
Змн.	Арк.	№ докум.	Підпис	Дата		

ЛІТЕРАТУРА

1. Шварц Б., Зайцев П., Ткаченко В. и др. - MySQL. Оптимизация производительности (2-е издание) 2010 – 823 с.
2. Робин Никсон - Создаем динамические веб-сайты с помощью PHP, MySQL и JavaScript 2015 - 688с
3. <http://ruseller.com/lessons.php?id=1189>
4. http://ruseller.com/lessons.php?rub_id=28&id=692
5. <https://laravel.ru/docs/v5>

						Арк.
						34
Змн.	Арк.	№ докум.	Підпис	Дата		

ДОДАТКИ

						Арк.
						35
Змн.	Арк.	№ докум.	Підпис	Дата		