Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *Новікова Наталія Олександрівна*
(ПІБ)

академічної групи *121-20зск-1*
(шифр)

спеціальності *121 Інженерія програмного забезпечення*
(код і назва спеціальності)

освітньої програми *Комп'ютерні науки*
(назва освітньої програми)

на тему: *Web-додаток автоматизації ведення обліку доставки відправлень перевізником*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*проф.Алексєєв М.О.*			
розділів:				
спеціальний	*проф.Алексєєв М.О.*			
економічний	*доц. Касьяненко Л.В.*			
Рецензент				
Нормоконтролер	*Мартиненко А. А.*			

Дніпро
2023

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » _____ 2023 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студента *121-20зск-1* *Новікова Н.О.*
(група) (прізвище та ініціали)

тема кваліфікаційної роботи *Web-додаток автоматизації ведення обліку доставки відправлень перевізником*

затверджена наказом ректора НТУ «ДП» від *11.04.2023* № *256-с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів проєктно-технологічної практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі. Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***13.05.2023 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***27.05.2023 р.***

Завдання видав _____ *проф. Алексєєв М.О.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Новікова Н.О.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2023 р.*

Термін подання кваліфікаційної роботи до ЕК: *12,06.2023 р.*

ЗМІСТ

РЕФЕРАТ

Пояснювальна записка: 108 с., 16 рис., 3 дод., 19 джерел.

Об'єкт розробки: розробка web-додатоку автоматизації ведення обліку доставки відправлень перевізником.

Мета кваліфікаційної роботи: розробити web-додаток автоматизації ведення обліку доставки відправлень перевізником, зручниий для використання та корисний для клієнтів.

У вступі розглядається важливість сайтів на сьогоднішній день та в чому полягають їх недоліки.

У першому розділі було охоплено загальні знання, що стосуються предметної області, включаючи значимість завдання, визначену проблему, яка потребує вирішення, визначені умови та критерії реалізації програми, а також задіяні технології та програмні засоби.

У другому розділі була проведена експертиза існуючих рішень з подальшим вибором платформи розробки. Потім була спроектована та розроблена програма, що охоплює опис алгоритму та схему операційної структури програми. Визначено вхідні та вихідні дані, а технічні параметри охарактеризовано за їх складом. Крім того, було з’ясовано процес запуску та завантаження програми, а також вичерпний опис її функціональності.

В економічному розділі було визначено оцінку трудомісткості розробленої інформаційної системи, розрахунок вартості створення програми та оцінку часу, необхідного для її виконання.

Важливість розробки веб-сайту полягає в її здатності сприяти зростанню бізнесу, поширювати інформацію, сприяти спілкуванню з цільовою аудиторією та використовувати можливості, які надає Інтернет.

Список ключових слів: ДОДАТОК, АВТОМАТИЗАЦІЯ, АЛГОРИТМ, КОМП’ЮТЕР, ІНФОРМАЦІЙНА СИСТЕМА, ОБЛІК, ПРОЄКТУВАННЯ, МЕНЮ, ВКЛАДКА.

ABSTRACT

Explanatory note: 83 pp., 16 figures, 3 appendices, 19 sources.

Object of development: development of a web application for the automation of accounting for the delivery of shipments by the carrier.

Meta-qualification work: to develop a web-based application for the automation of accounting for the delivery of shipments by the carrier, which is convenient to use and useful for customers.

The introduction examines the importance of sites today and what their shortcomings are.

In the first section, general knowledge related to the subject area was covered, including the significance of the task, the identified problem that needed a solution, the defined conditions and criteria for the implementation of the program, as well as the technologies and software tools involved.

In the second section, an examination of existing solutions was carried out, followed by the selection of a platform development. Then the program was designed and developed, covering the description of the algorithm and the diagram of the operational structure of the program. Input and output data are defined, and technical parameters are characterized by their composition. In addition, the process of launching and downloading the program was clarified, as well as a comprehensive description of its functionality.

In the economic section, the assessment of the labor intensity of the developed information system, the calculation of the cost of creating the program and the assessment of the time required for its implementation were determined.

The importance of website development lies in its ability to promote business, expand information, facilitate communication with target audiences, and utilize the opportunities provided by the Internet.

List of keywords: APPLICATION, AUTOMATION, ALGORITHM, COMPUTER, INFORMATION SYSTEM, ACCOUNTING, DESIGN, MENU, TAB.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

АС – автоматизована система;

БД – база даних;

ЕОМ – електронно-обчислювальна машина;

ПЗ – програмне забезпечення;

ТЗІ - технічний захист інформації;

URL – Uniform Resource Locator;

HTTP – HyperText Transfer Protocol;

WWW – World Wide Web.

ВСТУП

Веб-сайти служать багатьом цілям і пропонують ряд переваг для окремих осіб, підприємств, організацій і спільнот. Ось кілька причин, чому веб-сайти важливі:

– присутність в Інтернеті: веб-сайти забезпечують цифрову присутність, дозволяючи окремим особам, компаніям або організаціям створити онлайн-ідентичність і продемонструвати свою інформацію, продукти або послуги глобальній аудиторії;

– інформація та комунікація: веб-сайти є ефективним способом обміну інформацією, новинами, оновленнями та оголошеннями з відвідувачами. Вони служать центральним центром для спілкування та взаємодії, надаючи контактну інформацію, форми та канали для прямої взаємодії;

– доступність і зручність: веб-сайти доступні 24/7 з будь-якого місця, де є підключення до Інтернету, що забезпечує користувачам зручність доступу до інформації або послуг у власному темпі та зручності;

– брендінг і довіра: добре розроблений веб-сайт із постійним брендингом допомагає встановити довіру та професіоналізм. Це дозволяє підприємствам і організаціям формувати імідж свого бренду, ділитися своїми цінностями та зміцнювати довіру відвідувачів;

– маркетинг і просування: веб-сайти служать потужними маркетинговими інструментами, дозволяючи компаніям просувати свої продукти чи послуги, проводити цільові рекламні кампанії та залучати потенційних клієнтів за допомогою пошукової оптимізації (SEO) та інших стратегій цифрового маркетингу;

– електронна комерція та онлайн-продажі: веб-сайти дозволяють компаніям продавати продукти чи послуги в Інтернеті, охоплюючи ширшу клієнтську базу за межами фізичних магазинів. Веб-сайти електронної комерції полегшують транзакції, перегляд продуктів, безпечну обробку платежів і керування замовленнями;

– інформаційний ресурс: веб-сайти можуть функціонувати як сховища знань, надаючи навчальний вміст, навчальні посібники, документацію та ресурси з конкретних тем. Вони пропонують платформу для обміну досвідом, результатами досліджень і розумінням галузі;

– спільнота та соціальна взаємодія: веб-сайти можуть сприяти розвитку онлайн-спільнот, дискусійних форумів або платформ соціальних мереж, об'єднуючи людей зі спільними інтересами та сприяючи взаємодії, обміну знаннями та співпраці;

– ефективність і автоматизація: веб-сайти можуть оптимізувати процеси та автоматизувати завдання, такі як планування зустрічей, підтримка клієнтів, онлайн-бронювання та відстеження замовлень. Це підвищує ефективність і економить час як для компаній, так і для клієнтів;

– аналітика та статистика: веб-сайти надають цінні дані та аналітику про поведінку користувачів, трафік веб-сайту та коефіцієнти конверсії. Ця інформація допомагає компаніям приймати зважені рішення, оптимізувати свої стратегії та покращити взаємодію з користувачем.

Це лише деякі з багатьох причин, чому веб-сайти мають вирішальне значення в сучасну цифрову епоху. Вони дають змогу окремим особам і організаціям охоплювати ширшу аудиторію, спілкуватися з клієнтами, надавати цінну інформацію та вести бізнес у все більш взаємопов'язаному світі.

Загалом наявність веб-сайту дає можливість підприємцям створити сильну присутність в Інтернеті, залучити клієнтів, зміцнити довіру та розвивати свій бізнес у все більш цифровому світі. Це центральний центр для маркетингу, продажів, взаємодії з клієнтами та розвитку бізнесу.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1. Загальні відомості з предметної галузі

Веб-сайт — це цифрова платформа, доступна через Інтернет, яка відображає інформацію, надає послуги, полегшує транзакції або дозволяє користувачам взаємодіяти та спілкуватися онлайн.

Перші сайти були створені на початку 1990-х років. Точну дату часто пов’язують із запуском Всесвітньої мережі (WWW) Тімом Бернерсом-Лі, британським комп’ютерним вченим, у 1991 році. Бернерс-Лі розробив базові технології та протоколи, такі як HTML (Hypertext Markup Language), HTTP (протокол передачі гіпертексту) та URL-адреси (уніфіковані покажчики ресурсів), які дозволяли створювати та ділитися веб-сторінками через Інтернет.

Перший веб-сайт, який коли-небудь створювався, був присвячений наданню інформації про сам проект World Wide Web. Він послужив базовим вступом до концепції гіпертексту та надав ресурси для веб-розробки. URL-адреса веб-сайту була http://info.cern.ch, і він запрацював 20 грудня 1990 року.

Після запуску першого веб-сайту більше людей і організацій почали створювати власні веб-сайти, що призвело до швидкого зростання Інтернету та мережі. Перші веб-сайти були переважно текстовими, з обмеженою графікою та інтерактивністю порівняно з сучасними стандартами. Однак вони заклали основу для веб-сайтів, які ми бачимо сьогодні, розвиваючись із часом завдяки розвитку веб-технологій, дизайну та функціональності.

Ось деякі ключові аспекти та компоненти, пов’язані з веб-сайтами:

1. Структура та компоненти.

– доменне ім’я: унікальна адреса, яка використовується для доступу до веб-сайту (наприклад, www.example.com);

– веб-сторінки: окремі документи, що містять вміст веб-сайту, зазвичай написані мовою HTML (мова розмітки гіпертексту) ;

– навігаційне меню: посилання або кнопки, які допомагають користувачам переходити між різними розділами чи сторінками веб-сайту;

– верхній і нижній колонтитули: узгоджені елементи, що з'являються у верхній і нижній частині кожної сторінки, часто містять бренд, навігаційні посилання та контактну інформацію;

– області вмісту: розділи, де відображаються текст, зображення, відео та інші медіафайли;

– форми: інтерактивні елементи, які дозволяють користувачам вводити та надсилати дані (наприклад, контактні форми, форми входу).

2. Дизайн і взаємодія з користувачем.

– макет: розташування елементів на веб-сторінці, включаючи текст, зображення та інші візуальні компоненти;

– візуальні елементи: графіка, кольори, типографіка та інші елементи дизайну, які сприяють загальному вигляду веб-сайту;

– швидкість реагування: здатність веб-сайту адаптуватися та забезпечувати зручність для користувачів на різних пристроях і розмірах екрана;

– зручність використання: забезпечення простоти навігації, розуміння та взаємодії на веб-сайті з інтуїтивно зрозумілим інтерфейсом користувача та чіткими інструкціями;

– доступність: розробка веб-сайту, щоб він був зручним і інклюзивним для людей з обмеженими можливостями, дотримуючись інструкцій і стандартів доступності.

3. Функціональність і особливості.

– інтерактивні елементи: кнопки, посилання, спадні меню, повзунки та інші елементи, які забезпечують взаємодію та взаємодію користувача;

– функція пошуку: дозволяє користувачам шукати певний вміст на веб-сайті;

– мультимедійна інтеграція: вбудовування зображень, відео, аудіо та інших медіа для покращення взаємодії з користувачем;

– онлайн-форми та збір даних: збір інформації про користувачів,

наприклад контактних даних або відгуків, за допомогою форм;

– функціональні можливості електронної комерції: можливість онлайн-покупок, списків продуктів, кошиків для покупок і безпечної обробки платежів;

– система керування вмістом (CMS): серверна система, яка дозволяє власникам веб-сайтів створювати, редагувати та керувати вмістом без знання програмування;

– інтеграція зі сторонніми службами: включення зовнішніх інструментів і послуг, таких як аналітика, платформи соціальних мереж і маркетинг електронною поштою.

4. Розробка сайту.

– front-end розробка: створення інтерфейсу користувача та взаємодії за допомогою HTML, CSS і JavaScript;

– внутрішня розробка: створення логіки на стороні сервера та інтеграція бази даних для обробки даних і генерації динамічного вмісту;

– веб-хостинг: зберігання файлів веб-сайту та надання доступу до них через сервери, підключені до Інтернету;

– безпека: впровадження заходів для захисту веб-сайту та його користувачів від несанкціонованого доступу, витоку даних і зловмисного програмного забезпечення.

Веб-сайти можуть служити різним цілям, від обміну інформацією та продажу продуктів до спілкування людей і надання онлайн-послуг. Дизайн і функціональність кожного веб-сайту адаптовані до конкретних цілей і цільової аудиторії.

1.2. Призначення розробки та галузь застосування

Серед великої кількості сайтів, їх тематики, дизайнів та створення структури, було прийняте рішення зупинитися на створенні web-додатоку автоматизації ведення обліку доставки відправлень перевізником.

На сьогоднішній момент є велика кількість підприємств, які займаються відправками. Щоб усі відправки були виконанні вчасно та без помилок, необхідно надати працівникам зручний та сучасний web-додаток, який зможе суттєво полегшить їхню роботу.

Web-додаток створено для автоматизації роботи менеджера в веденні обліку відправлень перевізником, а також перегляду клієнтом інформації щодо відправлень. Також Web-додаток виконує функцію формування звітності.

1.3. Підстави для розробки

Підставою для розробки дипломного проекту на тему «Web-додаток автоматизації ведення обліку доставки відправлень перевізником» є наказ по Національному технічному університету «Дніпровська політехніка» від 11 .04. 2023р. № 256-с.

1.4. Постановка завдання

Web-сайт автоматизації ведення обліку доставки відправлень перевізником має бути зручними, інформативними та вирішувати проблеми потенційних та існуючих клієнтів.

Дизайн — це компонент веб-сайту, який формує перше враження про компанію та її пропозиції. Вміст стосується всієї необхідної інформації на web-сайті, включаючи послуги, ціни, акції, рекомендації та контактну інформацію. Функціональність – це елемент web-сайту, який дозволяє користувачам легко отримувати доступ до бажаних дій, таких як створення відділень, ведення інформації, перегляд поточних відправлень тощо.

Поганий дизайн, вміст і функціональність можуть призвести до втрати клієнтів.

Веб-сайти шиномонтажних послуг є важливими інструментами для просування бренду, залучення клієнтів та покращення обслуговування.

Покращення якості веб-сайтів може мати позитивний вплив на репутацію компанії та її прибутки.

Метою цього дослідження є розробка та оцінка прототипу веб-сайту для доставки відправлень перевізником. Завданнями дослідження є:

– проаналізувати наукову літературу, яка стосується стандартів, які використовуються для оцінки якості веб-сайту, розробленого для послуг доставки перевізником;

– визначити потреби та очікування потенційних та існуючих клієнтів компанії перевізників;

– розробити прототип веб-сайту на основі аналізу літератури та потреб клієнтів;

– провести експертну оцінку прототипу за допомогою методу експертного огляду;

– провести користувацьку оцінку прототипу за допомогою методу сценарного тестування;

– проаналізувати результати оцінки та запропонувати рекомендації щодо покращення прототипу.

1.5. Вимоги до програми або програмного виробу

1.5.1. **Вимоги до функціональних характеристик**

В межах даного дипломного проєкту потрібно організувати базу даних для збереження інформації про відділення, відправлення, клієнтів та менеджерів, розробити web-додаток, який дозволяє працювати з розробленою базою даних.

Програма повинна виконувати такі функції по веденню бази даних:

– створення особистого кабінету менеджера та адміністратора;

– додавання, редагування та видалення відділень;

– ведення інформації щодо відправлень;

– додавання та редагування клієнтів;

– додавання, редагування та видалення менеджерів;

– перегляд поточних відправлень;

– фільтрація відділень за містами;

– формування звітності.

Web-додаток повинен перевіряти усі введені дані на коректність та видавати повідомлення у випадку помилкового вводу.

Про відправлення повинна зберігатися така інформація:

– ТТН — товарно-транспортна накладна;

– менеджер;

– відправник;

– відділення;

– отримувач;

– адреса призначення;

– тариф;

– ознака крихкості;

– оцінена вартість;

– опис;

– сума до сплати;

– ознака, що сплачено.

Завдання дипломного проєкту повинно бути реалізоване з використанням таких ролей користувачів: «Адміністратор», «Менеджер» та «Клієнт».

Для ролі користувача «Адміністратор» повинні бути реалізовані функції створення, редагування та видалення облікових записів менеджерів та інформації щодо відділень, формування звітності з вказаного діапазону.

Для ролі користувача «Менеджера» повинні бути реалізовані функції для редагування статусу відправлення, зворотній зв’язок та оформлення відправлення.

Для ролі користувача «Клієнт» повинні бути реалізовані функції для надання перегляду інформації щодо статусу відправлення, відділень та зворотнього зв’язку.

Для покращення пошуку інформації на web-додатку повинні надаватися функції пошуку та фільтрації.

Web-додаток повинен бути сумісним з основними браузерами, а саме з Chrome, Firefox, Safari та Edge. Повинен мати високу швидкість завантаження та реагування, а також простий та інтуїтивний інтерфейс користувача.

1.5.2. **Вимоги до інформаційної безпеки**

Додаток має надавати такий захист даних:

– безпечне зберігання та обробка дані клієнтів використовуючи базу даних, розташовану на захищеному сервері;

– захист даних клієнтів від несанкціонованого доступу та зловживання шляхом впровадження заходів шифрування та автентифікації;

– відповідати нормам захисту даних, таким як GDPR, CCPA, CPRA та іншим чинним законам;

– можливість доступу, зміни та видалення своїх даних через особистий обліковий запис.

Додаток має надавати такий захист трафіку:

– використання протоколу HTTPS для шифрування трафіку між клієнтом та сервером;

– дійсний та надійний SSL-сертифікат, який підтверджує справжність та безпеку додатку;

– використання сучасних алгоритми шифрування, такі як SHA256, та вимикати небезпечні шифри, такі як RC4;

– використання HTTP Strict Transport Security (HSTS) для заборони незашифрованого з’єднання;

– використання HTTP Public Key Pinning (HPKP) для запобігання атакам типу man-in-the-middle;

Додаток має надавати такий захист веб-застосунку

– перевірка та фільтрування дані, які вводяться користувачами через форми або URL-параметри, для запобігання атакам типу SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF) тощо.

– обмеження кількісті спроб авторизації або замовлення послуг для запобігання атакам типу brute force або denial-of-service (DoS).

– використовування безпечних налаштувань веб-сервера, такі як вимкнення відображення помилок, приховування версії та інформації про сервер, встановлення прав доступу до файлів та папок тощо.

– використовування безпечних налаштувань куків, такі як HttpOnly, Secure, SameSite, Expire тощо.

1.5.3. **Вимоги до складу та параметрів технічних засобів**

Рекомендовані вимоги до програмного та апаратного забезпечення хостингу (серверу):

– пропускна здатність у місяць — 1Гб;

– операційна система: Windows Server 2012;

– підтримка протоколів: HTTP;

– підтримувані бази даних: MySQL;

– об'єм жорсткого диску від 1 Гб;

Рекомендованими вимогами до програмного забезпечення клієнта програми є:

– наявність браузеру з підтримкою HTML5;

– наявність доступу до мережі Інтернет.

1.5.4. **Вимоги до інформаційної та програмної сумісності**

Додаток повинен надавати користувачам чітку та зрозумілу інформацію про те, які саме дані збираються, зберігаються, обробляються та передаються третім сторонам. Користувачам повинна надаватися можливість погоджуватися

або відмовлятися від збору та використання їх даних, а також змінювати або видаляти свої дані за запитом. Веб-сайт повинен надавати користувачам належну інформацію про свої права та обов'язки щодо використання веб-сайту, таку як політика конфіденційності, умови користування, відмова від відповідальності тощо.

Веб-сайт повинен бути сумісним з основними браузерами, такими як Chrome, Firefox, Safari та Edge. Сайт повинен бути адаптивним.

Додаток повинен використовувати стандартні та сучасні технології для розробки веб-застосунків, такі як HTML, CSS, JavaScript, PHP, MySQL тощо.

Веб-сайт повинен використовувати безпечні та надійні програмні засоби для забезпечення функціонування веб-застосунку, такі як веб-сервер, база даних, CMS, SSL-сертифікат тощо.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РОЗРОБКА ІНФОРМАЦІЙНОЇ СИСТЕМИ

2.1. Функціональне призначення системи

Функціональне призначення додатку доставки відправлень перевізником – це створення онлайн сервісу, який дозволить перевізникам створювати та переглядати відправлення.

Онлайн сервіси – це сайти, які надають різноманітні послуги, що значно полегшує роботу і дозволяє істотно заощадити час. За допомогою таких сайтів в Інтернеті можна робити що завгодно: проводити грошові операції, спілкуватися, шукати, зберігати, редагувати, пересилати та розповсюджувати інформацію та багато іншого.

Незважаючи на всю різноманітність сайтів і послуг, які вони надають, онлайн сервіси можна розділити на наступні основні групи, в залежності від їх тематики і призначення:

– Інформаційно-пошукові. Дані ресурси призначені для пошуку необхідної інформації. До них відносяться не тільки пошукові бази, а й онлайн-бібліотеки з книгами, музикою або фільмами, різні каталоги і довідники, а також онлайн-перекладачі.

– Соціальні. До цієї групи відносяться всілякі соціальні мережі, онлайн-комунікатори, а також електронні поштові служби. За допомогою таких сервісів можна не тільки спілкуватися з людьми, незалежно від їх дислокації, а й вести ділову переписку, пересилати файли і багато іншого.

– Сервіси покупок. Дані ресурси призначені для придбання речей або замовлення послуг. До цієї категорії відносяться не тільки онлайн-магазини, але і сервіси на замовлення і бронювання квитків (починаючи від театральних і закінчуючи залізничними), готелів і багато чого іншого.

– Банківські сервіси. За допомогою таких ресурсів можна оплатити покупку або послугу, наприклад, замовлення в інтернет-магазині, або навіть

заплатити за комунальні послуги, без необхідності особистого відвідування банку.

– Дорожні сервіси дають можливість швидко прокласти маршрут незалежно від способу пересування, оцінити транспортну завантаженість на дорогах, отримати інформацію про тривалість поїздки тощо.

Мета дизайну конкретного веб-сайту полягає в тому, щоб ефективно передати користувачам його призначення, функціональність і можливі дії. Дизайн служить засобом спілкування, що дозволяє розробникам зрозуміти потреби та вподобання людей, з якими вони спілкуються, через сам дизайн. Отже, поставлене завдання передбачає розуміння вимог замовника та створення веб-сайту, який буде одночасно зручним і зрозумілим, дозволяючи клієнту легко орієнтуватися та розуміти сайт.

2.2. Опис застосованих математичних методів

В кваліфікаційній роботі математичних методів не було застосовано.

2.3. Опис розроблених функцій

При розробці програмного продукту була створена функці перевірки, щоб відправник та отримувач були вибрані або створені нові з перевіркою на наявність.

В лістингу 2.1 наведено приклад перевірки відправника та отримувача.

Лістинг 2.1 – Фрагмент перевірки відправника та отримувача

```
public function checkNewCustomers($who, $customers, $phone, $name, $error2, $customersPhone, $customersName){
                $q = ($customers == 'sender') ? 'відправника' : 'отримувача';
                if($who === true and !empty($phone) || !empty($name)){
                        $this->_errors[$customers] = 'Неможливо одночасно обрати '.$q.' та створити нового. Оберіть щось одне!';
                        return false;
                }elseif($who === true and empty($phone) || empty($name)){
```

```
                $customers = $this->who;
                return true;
            }elseif($who === false and empty($phone) || empty($name)){
                $this->_errors[$error2] = 'Оберіть '.$q.' зі списку або заповніть цих обидва поля, щоб створити нового.';
                return false;
            }elseif($who === false and !empty($phone) and !empty($name)){
                if(preg_match('/^[0-9]{10}$/', $phone)){
                    $sql = "SELECT * FROM customers WHERE phone = '{$phone}'";
                    $res = $this->_db->sendQuery($sql);
                    if ($res->num_rows > 0){
                        $this->_errors[$error2] = 'Такий користувач вже зареєстровано! Введіть інший номер!';
                        return false;
                    }else{
                        $this->$customersPhone = $phone;
                        $this->$customersName = filter_var($name,FILTER_SANITIZE_STRING);
                        return true;
                    }
                }else{
                    $this->_errors[$error2] = 'Телефон в неправильному форматі! (10 цифр)';
                    return false;}
            }
        }
```

Також було створено функцію зворотнього зв’язку, за допомогою якої менеджер та адміністратор можуть відповідати на повідомлення клієнтів. Цю функцію зображено на лістингу 2.2.

Лістинг 2.2 – Функція зворотнього зв’язку

```
        public function replyMessage(){
            $email = $this->getEmail($this->replyId);
            $sql = "UPDATE feedback SET mark = 1 WHERE id = {$this->replyId}";
            $res = $this->_db->sendQuery($sql);
            if(!$res){
                return false;
            }
            if($email){
                mail($email, "Служба підтримки", $this->message, "From: fastdelivery.support@".$SERVER_NAME);
            }
```

```
        return true;
    }
```

Для звітності було розроблено функцію вигрузки в файл з розширенням .csv, який з легкістю можна відкрити в MS Excel. Цю функцію можно побачити на лістингу 2.3.

Лістинг 2.3 – Функцію вигрузки в файл

```
public function downloadFile(){
        $res = $this->getDispatch();
        $array = [];
        $array[0] = array('Номер накладної',
                                        'Дата створення',
                                        'Дата відправлення',
                                        'Дата прибуття',
                                        'Вага',
                                        'Оголошена вартість',
                                        'Крихкість',
                                        'Вартість доставки',
                                        'Оплата',
                                        'Менеджер',
                                        'Відправник',
                                        'Отримувач',
                                        'Статус',
                                        'Опис');
        for ($i=0; $i < $res->num_rows ; $i++) {
            $q = $res->fetch_assoc();
            foreach ($q as $key => $value) {
                if($key == 'weight'){
                    $array[$i+1][] = str_replace('.', ',', $value);
                }else{
                    $array[$i+1][] = $value;
                }
            }
        }
        $start = ($this->startDate) ? $this->startDate : 'початку';
        $end = ($this->endDate) ? $this->endDate : date('Y-m-d');

        $fileName = 'Звіт з '.$start.' по '.$end.'.csv';

        $fp = fopen($fileName, 'w');
```

```
        foreach ($array as $fields) {
            fputcsv($fp, $fields, ';', '"');
        }
        fclose($fp);

        header('Location: /'.$fileName);
    }
```

На лістингу 2.4 представлено калькулятор вартості відправлення, який розміщено на головній сторінці для клієнтів, які відвідують сайт.

Лістинг 2.4 – Калькулятор вартості віправлення

```
    public function calculator($type = null, $weight = null, $fragility = null, $cost = null){

        // Если задан параметр null то я возвращаю ошибку которая содердыт цену, противном случае везвращаю просто цену доставки

        if(is_null($type) and is_null($weight) and is_null($fragility) and is_null($cost)){
            $fragility = $this->fragility;
            $weight = $this->weight;
            $type = $this->type;
            $cost = $this->cost;
            $fragility = ($fragility == 'on') ? 1 : 0;
        }
        $weight = str_replace(",", ".", $weight);
        $sql = "SELECT t_cost FROM tariff WHERE type = {$type} AND startWeight <= {$weight} AND endWeight >= {$weight}";
        $res = $this->_db->sendQuery($sql);
        if($res->num_rows > 0){
            $tariff = $res->fetch_assoc()['t_cost'];
            $price = $tariff + ($fragility * ($cost * 0.05));
            return $this->_errors['price'] = $price;
        }
        $this->_errors['price'] = 'Ми не приймаємо такий тип вантажу з такою вагою!';
        return false;
    }
```

2.4. Опис використаних технологій та мов програмування

Brackets — безкоштовний редактор з відкритим кодом для web-розробників. Brackets орієнтований на роботу з HTML, CSS і JavaScript. Цій ж технології лежать в основі самого редактора, що забезпечує його кроссплатформенность тобто сумісність з операційними системами Mac, Windows і Linux. Brackets створений і розвивається Adobe Systems під ліцензією MIT License та підтримується на GitHub. На сьогоднішній день співтовариством створено безліч розширень, що додають більшість необхідних інструментів для роботи над кодом, таких як система контролю версій Git, перегляд HTML-коду в браузері в реальному часі (Live Preview).

Хоч Brackets і позиціонується як текстовий редактор, за фактом він все більше нагадує повноцінну IDE. Проте, слід сказати про те, що ми отримуємо при базовій установці цього редактора:

- плагін для Live Preview — працює тільки з Google Chrome. Вносимо будь-які зміни в код в редакторі — у вікні браузера автоматично ви можете бачити різницю;
- підсвічування синтаксису;
- підказки при редагуванні CSS, JS і HTML-файлів;

Саме величезна кількість плагінів дозволяє перетворити даний текстовий редактор в потужний комбайн для WEB-розробки.

PHP — мова, у код якої можна вбудовувати безпосередньо html-код сторінок, які, у свою чергу, коректно оброблюватимуться PHP-інтерпретатором. Обробник PHP просто починає виконувати код після відкриваючого тегу (<?php) і продовжує виконання до того моменту, поки не зустріне закриваючий тег.

Велика різноманітність функцій PHP дає можливість уникати написання багаторядкових функцій, призначених для користувача, як це відбувається в C або Pascal.

У PHP вбудовані бібліотеки для роботи з MySQL, PostgreSQL, SQLite, mSQL, Oracle, dbm, Hyperware, Informix, InterBase, Sybase. Завдяки стандарту

відкритого інтерфейсу зв'язку з базами даних можна підключатися до всіх баз даних, до яких існує драйвер.

Мова PHP здаватиметься знайомою програмістам, що працюють в різних областях. Багато конструкцій мови запозичені з C, Perl. Код PHP дуже схожий на той, який зустрічається в типових програмах мовами C або Pascal. Це помітно знижує початкові зусилля при вивченні PHP. PHP — мова, що поєднує переваги Perl та C і спеціально спрямована на роботу в Інтернеті, мова з універсальним і зрозумілим синтаксисом. І хоча PHP є досить молодою мовою, вона здобула таку популярність серед web-програмістів, що в наш час є найпопулярнішою мовою для створення web-застосунків (скриптів).

Стратегія Open Source, і розповсюдження початкових текстів програм в масах, безсумнівно справили сприятливий вплив на багато проектів, в першу чергу — Linux хоч і успіх проекту Apache сильно підкріпив позиції прихильників Open Source. Сказане відноситься і до історії створення PHP, оскільки підтримка користувачів зі всього світу виявилася дуже важливим чинником в розвитку проекту PHP. Ухвалення стратегії Open Source і безкоштовне розповсюдження початкових текстів PHP надало неоціненну послугу користувачам. Окрім цього, користувачі PHP в усьому світі є свого роду колективною службою підтримки, і в популярних електронних конференціях можна знайти відповіді, навіть на найскладніші питання.

Ефективність є дуже важливим чинником у програмуванні для середовищ розрахованих на багато користувачів, до яких належить і web. Важливою перевагою PHP є те, що ця мова належить до інтерпретованих. Це дозволяє обробляти сценарії з достатньо високою швидкістю. За деякими оцінками, більшість PHP-сценаріїв (особливо не дуже великих розмірів) обробляються швидше за аналогічні їм програми, написані на Perl. Проте хоч би що робили розробники PHP, виконавчі файли, отримані за допомогою компіляції, працюватимуть значно швидше — в десятки, а іноді і в сотні разів. Але продуктивність PHP достатня для створення цілком серйозних web-застосунків.

Було обране середовище СУБД MySQL для реалізації бази даних в дипломному проекті тому, що MySQL є поширеною СУБД у сучасній розробці програмного забезпечення.

MySQL — це система управління реляційними базами даних. У реляційній базі даних дані зберігаються не все скопом, а в окремих таблицях, завдяки чому досягається виграш в швидкості і гнучкості. Таблиці зв'язуються між собою за допомогою стосунків, завдяки чому забезпечується можливість об'єднувати при виконанні запиту дані з декількох таблиць. SQL як частину системи MySQL можна охарактеризувати як мова структурованих запитів плюс найбільш поширена стандартна мова.

Програмне забезпечення MySQL — це ПО з відкритим кодом. ПО з відкритим кодом означає, що застосовувати і модифікувати його може той, хто бажає. Таке ПО можна отримувати по Internet і використовувати безкоштовно. При цьому кожен користувач може вивчити вихідний код і змінити його відповідно до своїх потреб. Використання програмного забезпечення MySQL регламентується ліцензією GPL (GNU General Public License).

Але MySQL також має низьку ефективність при використанні її як сховища даних, це частково пов’язано з нездатністю використовувати декілька процесорів для обробки одного запиту. До того ж, MySQL часто критикують за те, що СКБД має розходження з стандартами SQL, щодо трактування NULL значень і значення за замовчуванням.

2.5. Опис структури програми та алгоритмів її функціонування

Для реалізації програмного продукту було створено базу даних, вона складається з восьми таблиць.

Схему зв’язків між таблицями представлено на рис. 2.1.

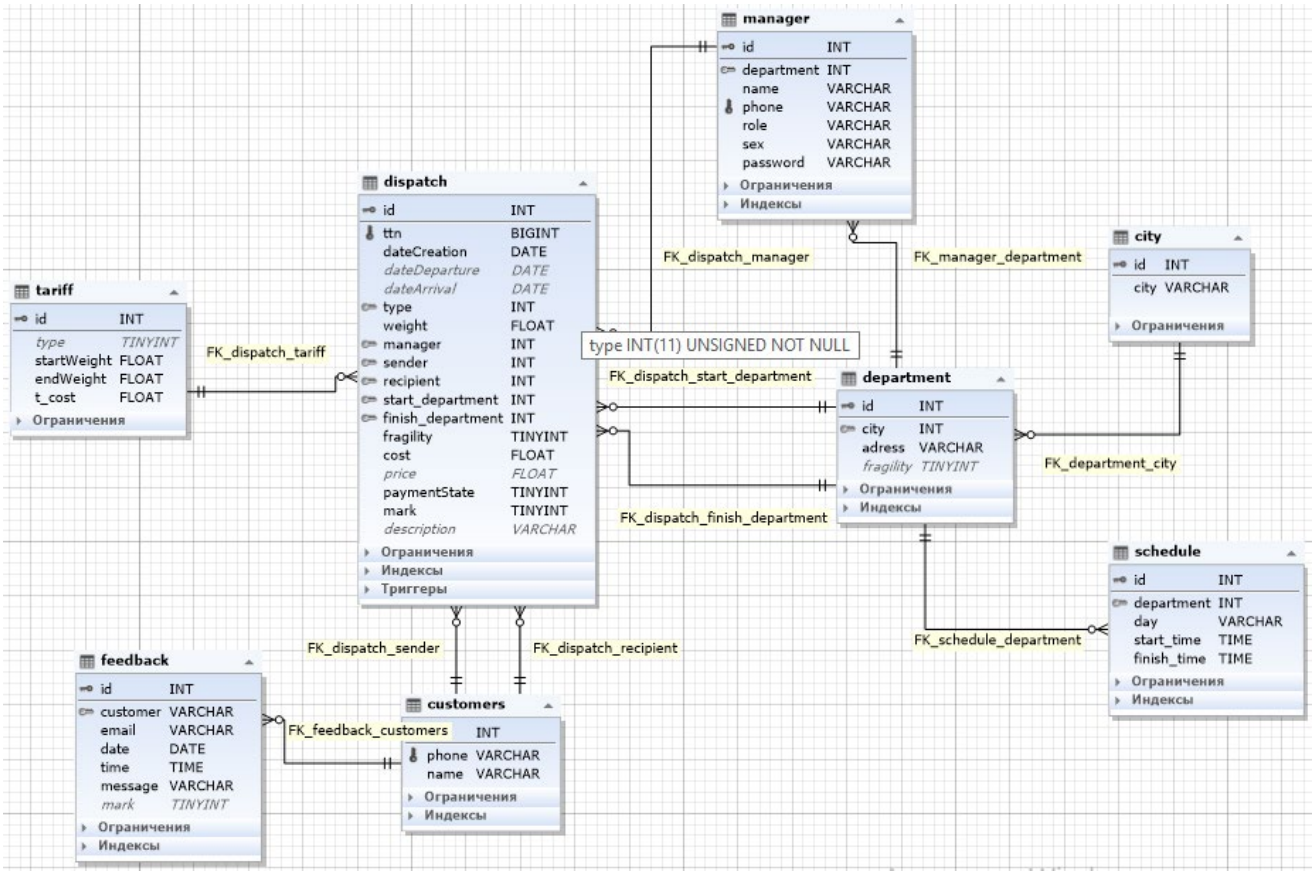

Рис. 2.1 – Схема зв’язків між таблицями

В таблиці «Feedback», яку наведено в таблиці 2.1, знаходиться інформація про зворотній зв’язок.

Таблиця 2.1 – Дані зі зворотнього зв’язку

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер зворотнього зв’язку	Числовий	11	PK
customers	Ідентифікаційний номер клієнта	Числовий	11	FK
email	Пошта відправника	Текстовий	255	
date	Дата відправлення	Дата		
time	Час відправлення	Час		
message	Повідомлення	Текстовий	255	
mark	Ознака обробки	Числовий	1	

В таблиці «Dispatch», яку наведено в таблиці 2.2, знаходиться інформація про відправлення.

Таблиця 2.2 – Дані про відправлення

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
ttn	Номер товарно транспортної накладної	Числовий	20	
dateCreator	Дата створення	Дата		
dateDeparture	Дата відправлення	Дата		
dateArrival	Дата прибуття	Дата		
type	Тип відправлення	Числовий		FK
weight	Вага відправлення	Дійсний		
manager	Ідентифікаційний номер менеджера	Числовий	11	FK
sender	Ідентифікаційний номер відправника	Числовий	11	FK
recipient	Ідентифікаційний номер отримувача	Числовий	11	FK
start_department	Ідентифікаційний номер початкового відділення	Числовий	11	FK
finish_department	Ідентифікаційний номер кінцевого відділення	Числовий	11	FK
fragility	Ознака крихкості	Числовий	1	
cost	Оціночна вартість	Дійсний		
price	Вартість доставки	Дійсний		
paymentState	Ознака сплати	Числовий	1	
mark	Статус	Числовий	1	
description	Примітка відправлення	Текстовий	255	

В таблиці «Customers», яку наведено в таблиці 2.3, знаходиться інформація про клієнтів.

Таблиця 2.3 – Дані про клієнтів

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
phone	Номер телефона	Текстовий	10	
name	ПІБ	Текстовий	255	

В таблиці «Manager», яку наведено в таблиці 2.4, знаходиться інформація про менеджерів.

Таблиця 2.4– Дані про менеджерів

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
department	Ідентифікаційний номер відділення	Числовий	11	FK
name	ПІБ	Текстовий	255	
phone	Номер телефону	Текстовий	10	
role	Права	Текстовий	255	
sex	Стать	Текстовий	255	
password	Пароль	Текстовий	255	

В таблиці «Department», яку наведено в таблиці 2.5, знаходиться інформація про відділення.

Таблиця 2.5 – Дані про відділення

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
city	Місто	Текстовий	255	
adress	Вулиця	Текстовий	255	
fragility	Ознака прийому крихких товарів	Числовий	1	

В таблиці «Tariff», яку наведено в таблиці 2.6, знаходиться інформація про тарифи.

Таблиця 2.6– Дані про тарифи

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
type	Вид документ	Числовий	1	
startWeight	Початкова вага	Дійсний		
endWeight	Кінцева вага	Дійсний		
t_cost	Вартість	Дійсний		

В таблиці «City», яку наведено в таблиці 2.7, знаходиться інформація про тарифи.

Таблиця 2.7– Дані про місто

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер	Числовий	11	PK
city	Найменування міста	Текстовий	255	

В таблиці «Schedule», яку наведено в таблиці 2.8, знаходиться інформація про графік роботи.

Таблиця 2.8 – Дані про графік роботи

Назва поля	Опис	Тип	Розмір	Ключ
id	Ідентифікаційний номер графіка	Числовий	11	PK
department	Ідентифікаційний номер відділення	Числовий	11	FK
day	День тижня	Текстовий	3	
start_time	Початок роботи	Час		
finish_time	Кінець роботи	Час		

На рис. 2.2. представлено структурну схема взаємодії складових програми.

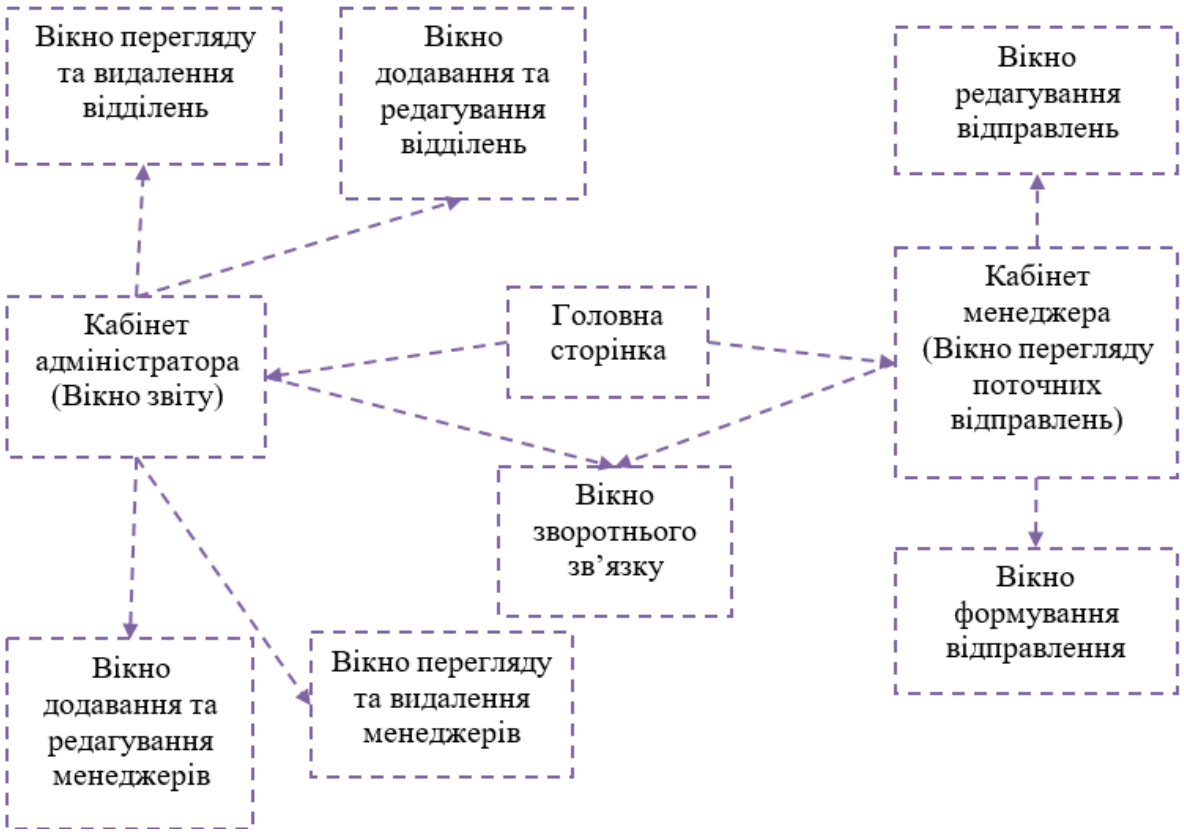

Рис. 2.2 – Схема зв'язку між модулями інтерфейсу

В рамках дипломного проєкту розроблений web-додаток містить 6 модулів, які містять свої методи. Опис розроблених методів представлено в таблиці 2.9.

Таблиця 2.9 — Вміст модулів проєкту

Назва	Перелік класів
base	BaseForm.class — клас для роботи з формами Controller.class — базовий клас контролера View.class
config	db.conf — підключення до БД
controllers	ControllerAdmin.class — контролер для адміністратора ControllerMain.class — головний контролер ControllerManager.class — контролер для менеджера

Продовження таблиці 2.9.

Назва	Перелік класів
library	AccessException.class — клас наслідування виключень Auth.class — клас для роботи з авторизацією Db.class — клас для роботи з БД DbException.class — виключення в БД HttpException.class Request.class — клас для запросів зі сторінки Url.class – клас для роботи з url Validator.class — клас валідатор
models	Calculator.class — клас для розрахунку CreateDepartment.class — створення відправлення CreateManager.class —створення менеджера Departments.class клас відправлень DepartureForm.class — клас для роботи зі сторінкою Feedback.class — клас для зворотнього зв’язку LoginForm.class — клас авторизації Managers.class — клас менеджерів ReportForm.class — клас для звіту SearchDepartment.class — пошук відправлень SearchParcel.class — пошук накладної Support.class — клас для зворотньої підтримки
views	createDepartment — структура форми створення відділення createManager — структура форми створення менеджера departments — структура форми відділень managers — структура форми менеджерів index — структура форми для відвідувачів adminka — структура форми для адміністратора main — структура форми головної сторінки departure — структура форми відправлення messages — структура форми повідомлень

Більш детально призначення класів наведено в таблицях 2.10 – 2.19.

Таблиця 2.10 — Клас «BaseForm»

Назва	Призначення
getErrors()	Отримання помилок
getRules()	Абстрактний метод для роботи правилами
validate()	Перевірка даних
sorting($param = null, $delimiter = null)	сортування
getTable($columns = '*', $table, $data = null)	Виведення даних з таблиць
groupingDepartment($data)	Дані про відділення

Таблиця 2.11 — Клас «View»

Назва	Призначення
setLayout($layout)	Встановлення шапки сторінки
setTitle($str)	Встановлення заголовка
setRole($str)	Встановленя ролі
setName($str)	Встановлення назви
setCss($css)	Встановлення стилів для сторінки
getError($param)	Додавання помилки в масив помилок

Таблиця 2.12 — Клас «ControllerAdmin»

Назва	Призначення
actionIndex	Метод за замовчування (звітність)
actionCreateManager	Створення/редагування менеджера
actionCreateDepartment	Створення/редагування відділення
actionManagers	Метод за замовчування (менеджери)
actionDepartments	Метод за замовчування (відділення)
actionMessages	Метод за замовчування (повідомлення)

Таблиця 2.13 — Клас «Db»

Назва	Призначення
getDb()	Підключення до БД
sendQuery()	Відправка запиту

Таблиця 2.14 — Клас «Auth»

Назва	Призначення
isGuest()	Перевідка на роль відвідувача
canAccess($role)	Перевірка на роль менеджера та адміністратора
login($id, $name, $role)	Перевірка коректності
getUserId()	Виведення id
getUserName()	Виведення ПІБ

Таблиця 2.15 — Клас «Url»

Назва	Призначення
getSegmentsFromUrl	Отримання посилання на сайт
getParam($paramName)	Отримання потрібного параметру
getSegment($n)	Отримання сегменту за номером
getAllSegments()	Отримання всього сегменту

Таблиця 2.16 — Клас «Validator»

Назва	Призначення
required($field)	Перевірка на заповленість поля
email($field)	Перевірка емейлу
selected($field)	Перевірка на порожність випадаючого списку
sex($field)	Перевірка на стать
role($field)	Перевірка на роль
phone($field)	Перевірка на номер телефона
existDetartment($field)	Перевірка на відділення
existCity($field)	Перевірка на місто
confirm($field)	Перевірка на співпадання
date($field)	Перевірка дати
addError($field, $error)	Додавання помилки у масив
getErrors()	Виведення помилок
getError($field)	Виведення помилки
validateThis()	Перевірка на помилки

Таблиця 2.17 — Клас «CreateDepartment»

Назва	Призначення
save()	Збереження відділення
update($departmentId)	Оновлення відділення
getDepartment($param)	Виведення відділення

Таблиця 2.18 — Клас «DepartureForm»

Назва	Призначення
save()	Збереження відправленя
checkCustomers($id)	Перевірка клієнта
newCustomers($phone, $name)	Створення клієнта
checkNewCustomers($who, $customers, $phone, $name, $error2, $customersPhone, $customersName)	Перевірка нового на існуючого клієнта

Таблиця 2.19 — Клас «Support»

Назва	Призначення
isCustomers()	Збереження відправленя
isNameCustomers()	Пошук клієнта
updateNameCustomers()	Оновлення ПІБ клієнта
newCustomers()	Створення клієнта
save()	Збереження відповіді

2.6. Обґрунтування та організація вхідних та вихідних даних програми

Вхідні дані додаток отримує від коричтувача або з інших джерел для виконання певної задачі, які вписуються за допомогою клавіатури та миші, а саме:ПІБ, номер телефону, логін, пароль, e-mail, номер накладної тощо.

Інформація, яка надається користувачеві під час виконання програми є вихідними даними, наприклад: сформований звіт, дані про менеджерів чи відправлення.

2.7. Опис розробленої системи

2.7.1. Використані технічні засоби

Для розробки веб-додатку знадобились наступні технічні засоби:

- Процесор Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz 2.59 GHz;
- 16 ГБ оперативної пам’яті;
- ОС Windows 10;
- Клавіатура;
- Маніпулятор «миша».

2.7.2. Використані програмні засоби

Проєкт був розроблений за допомогою безкоштовного редактору Brackets, мові PHP та програмного забезпечення MySQL.

2.7.3. Виклик та завантаження програми

Для відкриття сайту потрібно в браузері вставити посилання https://fastdelivery2023.000webhostapp.com/.

2.7.4. Опис інтерфейсу користувача

Щоб опинитися на сайт web-додатку потрібно перейти за посиланням https://fastdelivery2023.000webhostapp.com/, після цього з’явиться вікно сайта, що предтавлено на рис. 2.3, на якому зображено головну сторінку.

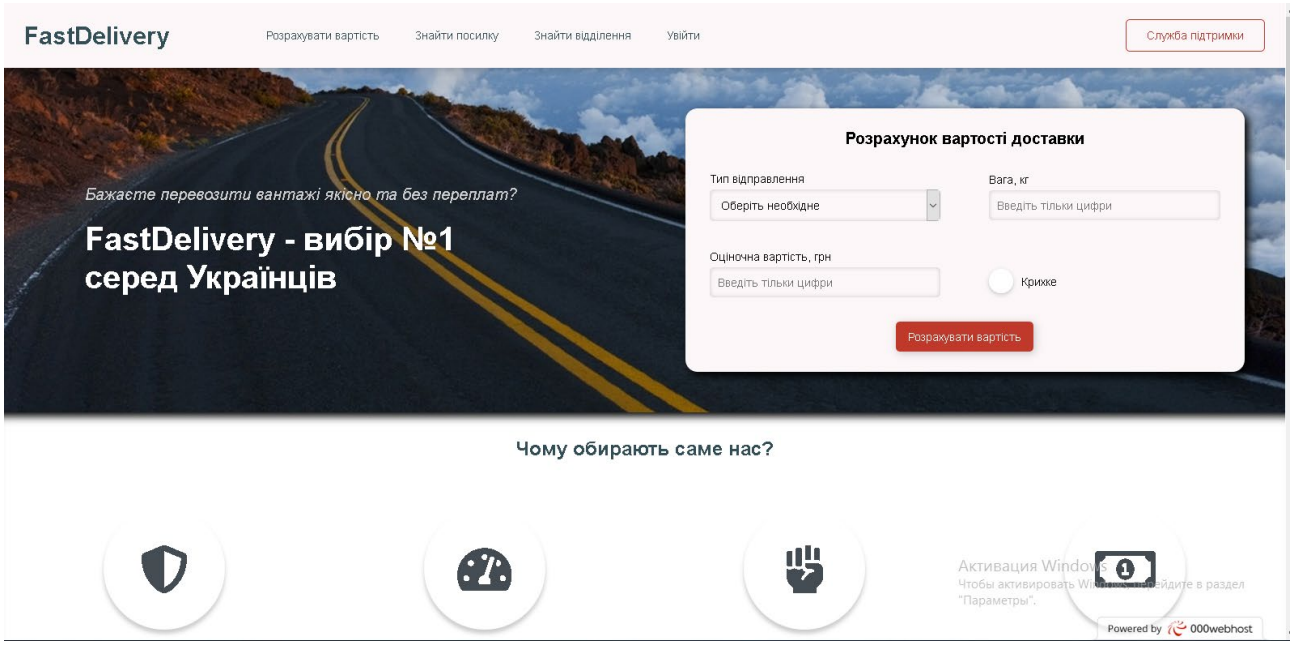

Рис. 2.3. — Головна сторінка

Ця сторінка зроблена для зручності користувача. На ній розміщено розрахунок вартості доставки, блок для пошуку накладної, зображено на рис. 2.4.

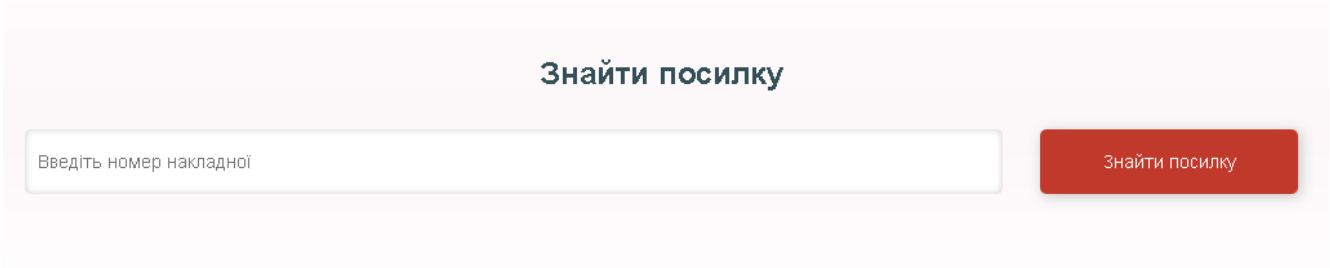

Рис. 2.4. — Пошук накладної

На рис. 2.5 представлено пошук відділень, для обрання зручнішого для клієнта. Для того щоб здійснити пошук потрібно обрати місто, відділення або разом, після чого нижче виведеться інформація стосовно відділення з певним графіком.

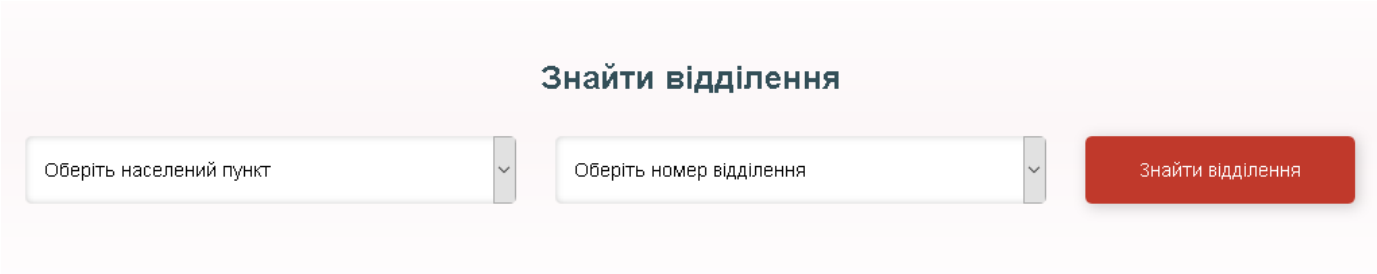

Рис. 2.5. — Пошук відділення

Також на цій сторінці є можливість звернутися в службу підтримки для цього достатньо ввести певні дані, після чого відповіть прийде на введений вами email. Цей блок зображено на рис. 2.6.

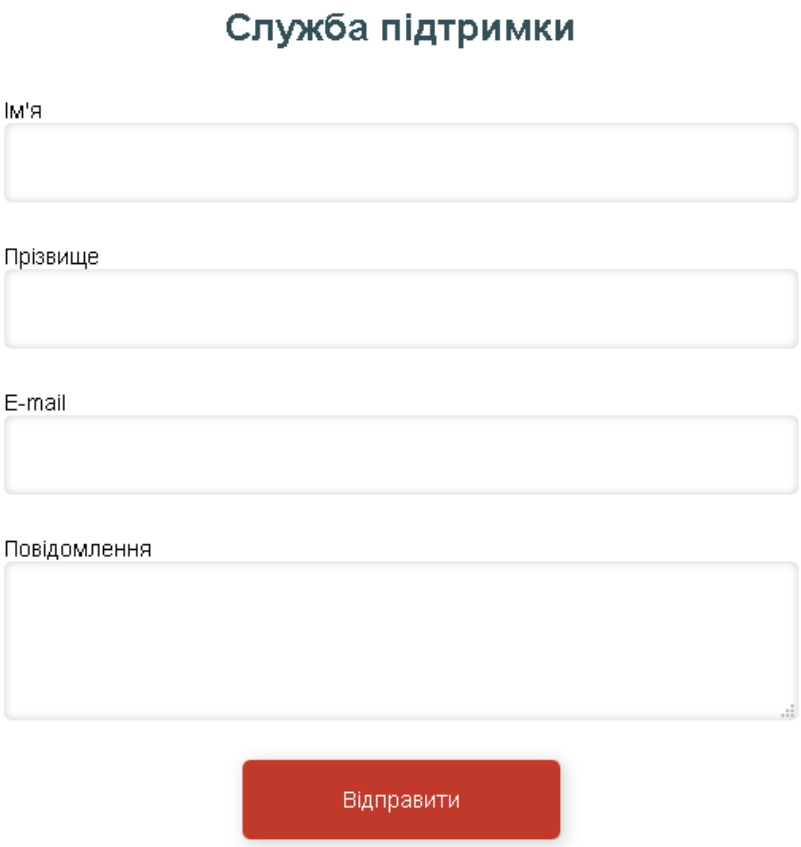

Рис. 2.6. — Служба підтримки

Усіма функціями, які розміщено на головній сторінці з легкістю можна скористатися навіть не авторизувавшись в самому web-додатку. Саме це надає простоти та легкості у використанні для відвідувачів сайту.

З головної сторінки до свого кабінету можуть зайти адміністратор та менеджери. Блок входу представлено на рис. 2.7.

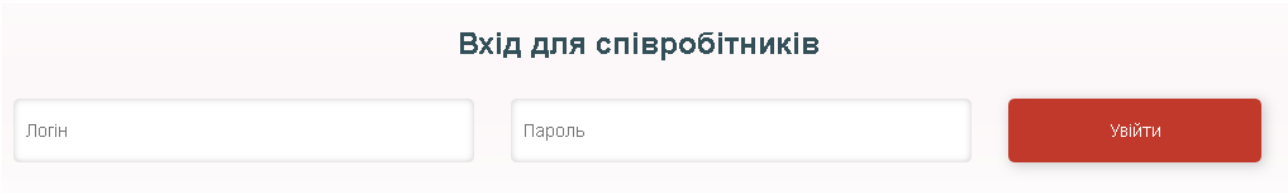

Рис. 2.7. — Вхід для співробітника

При авторизації як адміністратор першою сторінкою буде звітність, яку зображено на рис. 2.8. Тут можно обрати період звітності або відділення, а також здійснити пошук за ТТН.

Також на цій сторінці присутнє меню, що знаходиться зліва. Воно дозволяє переходити на інші сторінки.

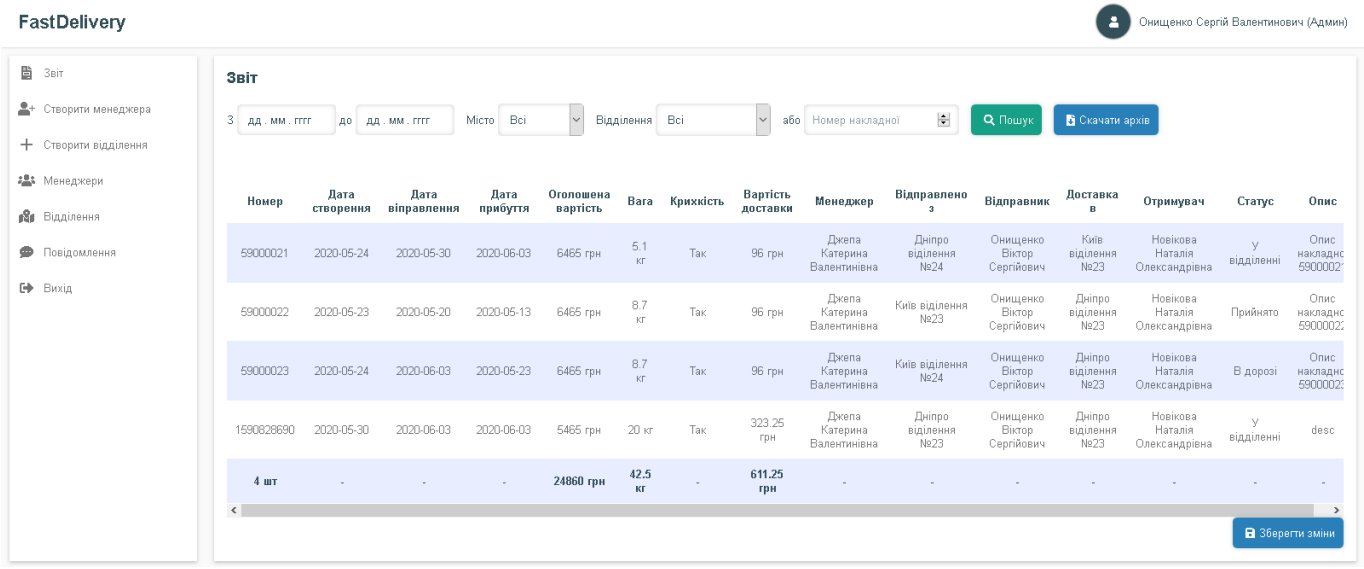

Рис. 2.8. — Звітність

Наступними пунктами меню є створення менеджера та відділеня, що предствлені на рисунках 2.9 – 2.10. При створенні менеджера потрібно ввести всі необхідні поля, а також здійснити підтвердження паролю, а при створенні відділення йому створюється графік роботи.

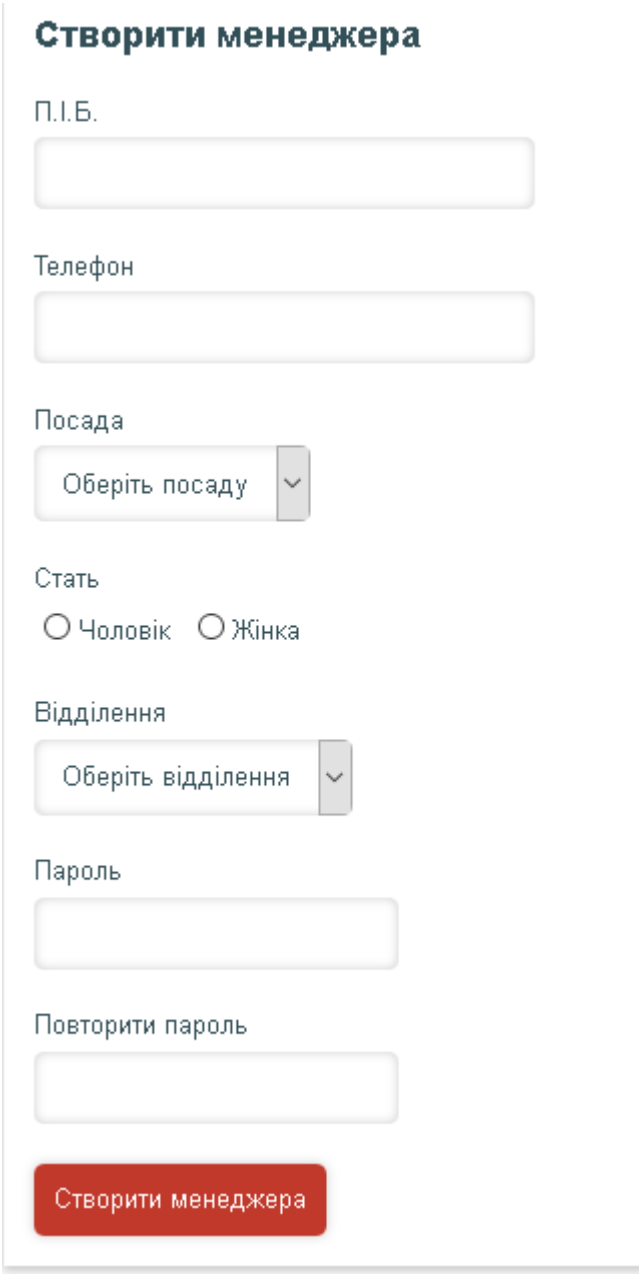

Рис. 2.9. — Додавання менеджера

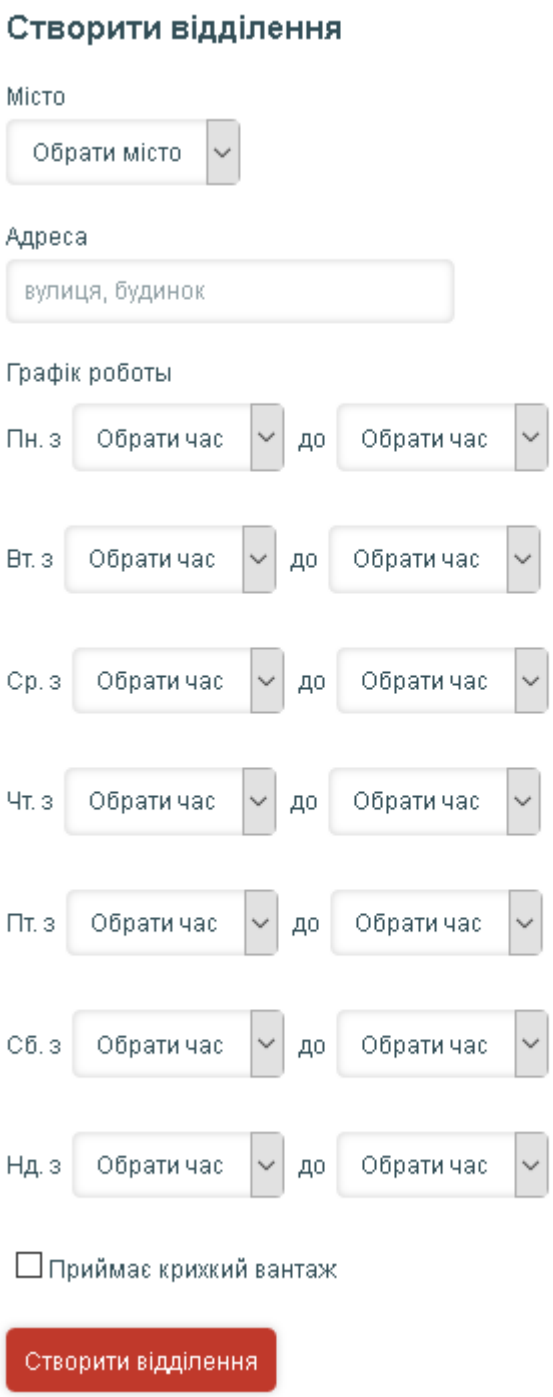

Рис. 2.10. — Додавання відділення

На рисунках 2.11 – 2.12 представлено інформацію щодо менеджерів та відділень з можливістю видалення та редагування. Інформація відділень представлена у вигляді таблиць з можливістю фільтрації за містами та відділеннями, а також видаленням або редагуванням за допомогою кнопок, які можна побачити з правої сторони від інформації.

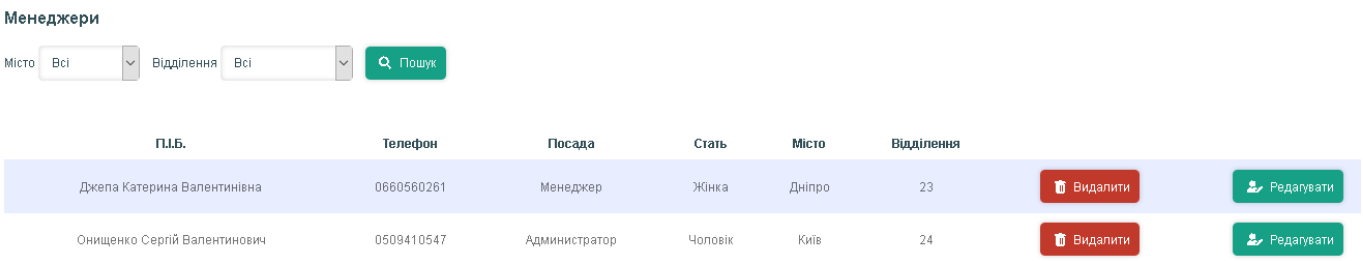

Рис. 2.11. — Сторінка перегляду менеджерів

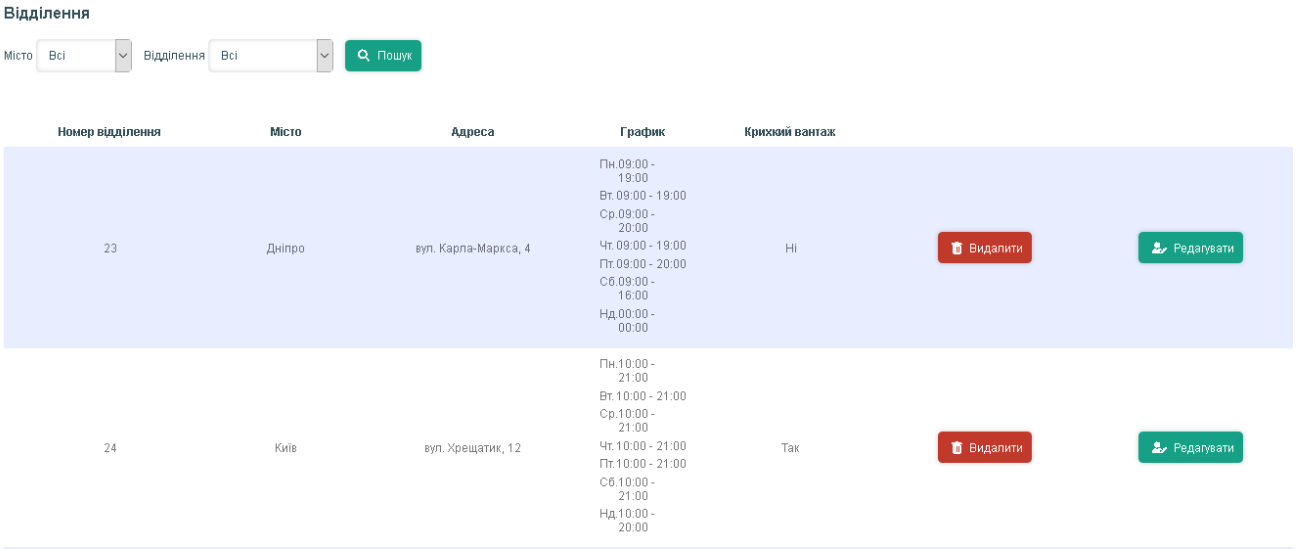

Рис. 2.12. — Сторінка перегляду відділень

На рис. 2.13 представлено сторінку повідомлень з можливістю відповіді. Відповідь на повідомлення клієнта здійснюється, пясля введення повідомлення та натиснення книпки «Відправити».

Рис. 2.13. — Сторінка повідомлення

Після виходу з кабінету адміністратора користувач повернеться на головну сторінку.

З головної сторінки також можна перейти до кабінету менеджера. В цьому кабінеті також є меню.

На першій сторінці кабінету менеджера знаходиться перелік відправлень, що представлені на рис. 2.14. На цій сторінці можна побачити деякі фільтри та пошук, а також кнопку «Зберегти», після зміни статусу.

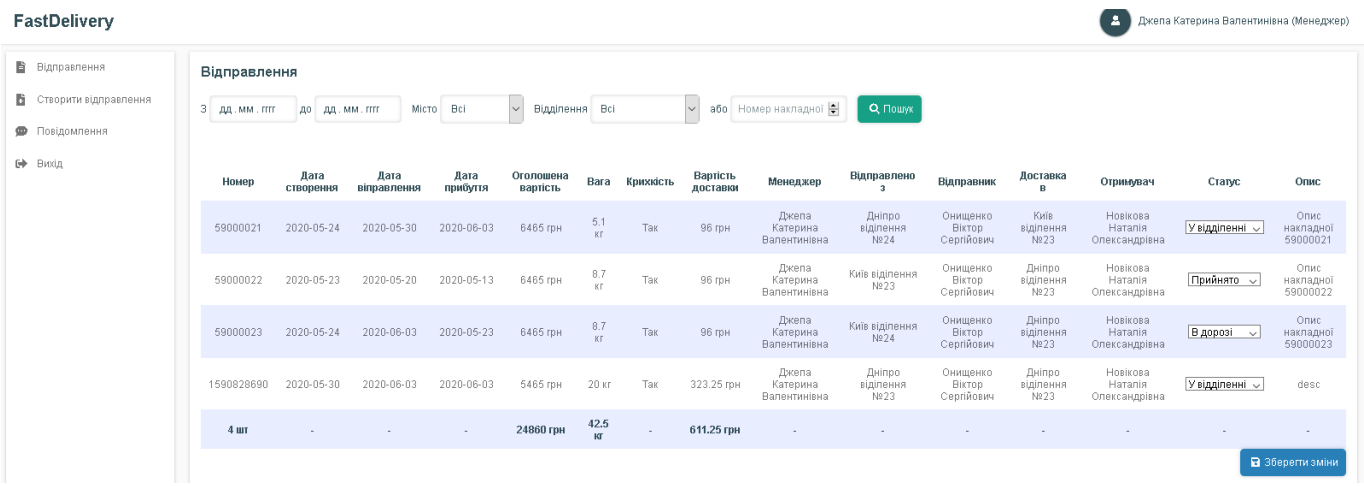

Рис. 2.14 — Відправлення

На рис. 2.15 зображено наступну сторінку – створення відправлення. Під час створення відправлення менеджер може обирати відправника та отримувача з випадаючого списку або створити нового заповнивши поля «Телефон» та «ПІБ», після чого достатньо вказати відділення, вагу та крихкість. Якщо всі поля заповнено вірно, а також не допущено логічних помилок, наприклад відправник та отримувач одна людина тощо, то створюється відправлення.

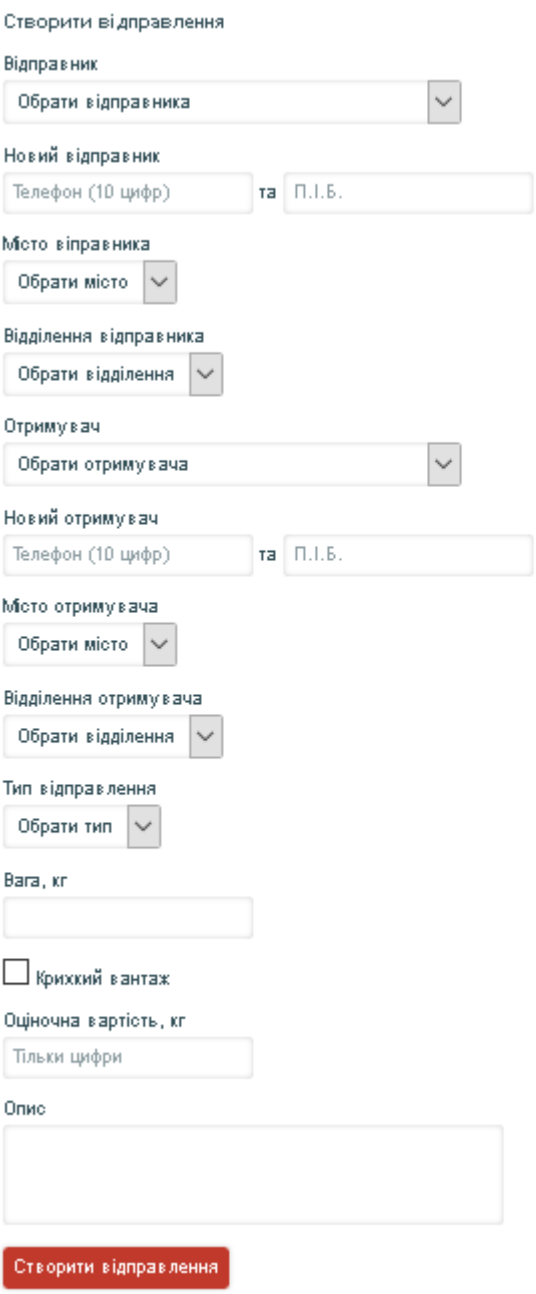

Рис. 2.15 — Створення відправлення

Сторінку повідомлення зображено на рис. 2.16. На цій сторінці менеджер може відповісти на повідомлення, після відправлення відповідь буде надіслана на пошту.

Рис. 2.16 — Створення відправлення

Після виходу з кабінету менеджера користувач повернеться на головну сторінку.

Виходячи з рекомендації користувача можна побачити, що web-додаток має інтуїтивно зрозумілий сучасний інтерфей та необхідний для автоматизації ведення обліку функціонал.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1. Розрахунок трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. Передбачуване число операторів програми – 1 500;

2. Коефіцієнт складності програми – 1,5;

3. Коефіцієнт корекції програми в ході її розробки – 0,4;

4. Годинна заробітна плата програміста – 210 грн/год;

Сайт Work.ua повідомляє, що веб-розробники в Україні отримують у середньому 45 000 грн на основі 143 вакансій за останні 3 місяці. Зарплата веб-розробника може значно різнитися залежно від регіону, кваліфікації, досвіду, мови програмування та інших чинників.

5. Коефіцієнт збільшення витрат праці в наслідок недостатнього опису задачі – 1,5;

6. Коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності – 1,1;

7. Вартість машино-години ЕОМ – 18,5 грн/год.

Мінімальна кількість витрат на інтернет, електроенергію, обладнання за місяць для розробки веб-сайту залежить від того, які параметри ви обираєте для цих ресурсів. Наприклад:

- Інтернет. Середня ціна інтернету в Україні становить близько 150 грн на місяць.
- Електроенергія. Середня ціна електроенергії в Україні становить близько 2,64 грн за кВт*год. Якщо припустити, що ваш комп’ютер споживає 200 Вт на годину і ви працюєте 8 годин на день, то ваша місячна витрата на електроенергію буде близько 97 грн.

- Обладнання. Середня ціна комп’ютера в Україні становить близько 25 000 грн. Якщо припустити, що ваш комп’ютер прослужить вам 5 років і ви не будете оновлювати його комплектуючі, то ваша місячна амортизація обладнання буде близько 417 грн.

Таким чином, мінімальна кількість витрат на інтернет, електроенергію, обладнання за місяць для розробки веб-сайту може скласти близько 664 грн.

Нормування праці в процесі створення ПЗ істотно ускладнено в силу творчого характеру праці програміста. Тому трудомісткість розробки ПЗ може бути розрахована на основі системи моделей з різною точністю оцінки.

Враховуючи всі дані, то виходить що за годину розробник витрачає 11,3 гривні.

Трудомісткість розробки інформаційної системи можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д \text{, людино-годин,} \quad (3.1)$$

де: t_o – витрати праці на підготовку й опис поставленої задачі (приймається 50 людино-годин);

t_u – витрати праці на дослідження алгоритму рішення задачі;

t_a – витрати праці на розробку блок-схеми алгоритму;

t_n – витрати праці на програмування по готовій блок-схемі та монтажу приладу;

$t_{отл}$ – витрати праці на налагодження інформаційної системи на ЕОМ;

$t_д$ – витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у інформаційної системи, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C \cdot (1 + p) \text{ , (3.2)}$$

де: q – передбачуване число операторів (3000);

C – коефіцієнт складності інформаційної системи (1,4);

p – коефіцієнт корекції інформаційної системи в ході її розробки (0,05).

Таким чином, умовне число операторів становить:

$$Q = 1{,}5 \cdot 1\,500 \cdot (1 + 0{,}4) = 3150 \text{ , (3.3)}$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75..85) \cdot k} \text{ , людино-годин , (3.4)}$$

де B – коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі;

k – коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності. За відсутністю стажу роботи та наявністю освіти дорівнює 1,1.

Таким чином, витрати праці на вивчення опису завдання приблизно становлять:

$$t_u = \frac{30150 \times 1.2}{85 \times 1.1} = 40{,}42 \text{ , людино-годин , (3.5)}$$

Витрати праці на розробку алгоритму рішення задачі визначаються за формулою:

$$t_a = \frac{Q}{(20\ldots25) \cdot k} \text{ , людино-годин , (3.6)}$$

де: Q – умовне число операторів інформаційної системи;

k – коефіцієнт кваліфікації програміста.

Таким чином, витрати праці на розробку алгоритму рішення задачі дорівнюють:

$$t_a = \frac{3150}{21 \times 1,1} = 136,36 \text{ , людино-годин , (3.7)}$$

Витрати на складання програми по готовій блок-схемі та монтажу інформаційної системи:

$$t_n = \frac{Q}{(20 \ldots 25) \cdot k} \text{ , людино-годин , (3.8)}$$

Таким чином, витрати на складання програми по готовій блок-схемі та монтажу інформаційної системи дорівнюють:

$$t_n = \frac{3150}{25 \times 1,1} = 114,54 \text{ , людино-годин , (3.9)}$$

Витрати праці на налагодження інформаційної системи на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4..5) \cdot k} \text{ , людино-годин , (3.10)}$$

Таким чином, умови автономного налагодження одного завдання становлять:

$$t_{отл} = \frac{3150}{5 \times 1,1} = 572,72, \text{ людино-годин , (3.11)}$$

- за умови комплексного налагодження завдання:

$$t_{отл}^{k} = 1{,}5 \cdot t_{отл}$$, людино-годин , (3.12)

Таким чином, умови комплексного налагодження завдання становлять:

$t_{\text{отл}}^{k}$= 1,5 × 572,72 = 858, людино-годин , (3.13)

Витрати праці на підготовку документації визначаються за формулою:

$$t_{д} = t_{др} + t_{до}$$, людино-годин , (3.14)

де: $t_{др}$ – трудомісткість підготовки матеріалів і рукопису:

$$t_{др} = \frac{Q}{(15..20) \cdot k}$$, людино-годин , (3.15)

$t_{до}$ – трудомісткість редагування, печатки й оформлення документації:

$$t_{до} = 0{,}75 \cdot t_{др}$$, людино-годин , (3.16)

Таким чином, отримуємо:

$t_{\text{др}}$= $\frac{3150}{20 \times 1{,}1}$ = 143,18 ,людино-годин , (3.17)

$t_{\text{до}}$= 0,75 × 143,18 = 107,38 ,людино-годин , (3.18)

$t_{\text{до}}$= 143,18 + 104,38 = 250,56 ,людино-годин , (3.19)

Повертаючись до формули (3.1), отримуємо кінцеву оцінку трудомісткості розробки програмного забезпечення:

$t = 50 + 40{,}42 + 136{,}36 + 114{,}54 + 572{,}72 + 250{,}56 = 1\,164{,}6$, людино-годин , (3.20)

3.2. Розрахунок витрат на створення інформаційної системи

Витрати на створення інформаційної системи $К_{ПО}$ (3.2) включають витрати на заробітну плату виконавця програми $З_{ЗП}$ і витрат машинного часу, необхідного на налагодження інформаційної системи на ЕОМ:

$$К_{ПО} = З_{ЗП} + З_{МВ}$$, грн, (3.21)

Заробітна плата виконавців визначається за формулою:

$$З_{ЗП} = t \cdot С_{ПР}$$, грн , (3.22)

де: t – загальна трудомісткість, людино-годин;

$С_{ПР}$ – середня годинна заробітна плата програміста, грн/година

Таким чином, заробітна плата виконавців дорівнює:

$З_{ЗП} = 1164{,}6 \cdot 210 = 244\,566$ грн , (3.23)

Вартість машинного часу, необхідного для налагодження інформаційної системи на ЕОМ, визначається за формулою:

$$З_{мв} = t_{отл} \cdot С_{мч}$$, грн , (3.24)

де: $t_{отл}$ – трудомісткість налагодження інформаційної системи на ЕОМ, год;

$C_{мч}$ – вартість машино-години ЕОМ, грн/год (21 грн/год).

Отже, вартість необхідного для налагодження машинного часу становить:

$З_{мв}$ = 572,72 · 18,5 = 10 595,32 грн, (3.25)

Таким чином, витрати на створення інформаційної системи за формулою (3.2) становлять:

$К_{ПО}$ = 244 566 + 10 595,32 = 255 161,32 грн , (3.26)

Очікуваний період створення інформаційної системи:

$$T = \frac{t}{B_k \cdot F_p},$$ міс , (3.27)

де: B_k – число виконавців (дорівнює 1);

F_p – місячний фонд робочого часу (при 40 годинному робочому тижні F_p=176 годин).

Отже, витрати на створення інформаційної системи дорівнюють:

T = 1 164,6 / (1·176) ≈ 6,6 міс , (3.28)

Отже, час розробки даного програмного забезпечення складає 1 164,6 людино-годин. Таким чином, очікувана тривалість розробки складе 6,6 місяця при 40 годинному робочому тижні (місячний фонд робочого часу 176 годин), а витрати на створення програмного забезпечення складатимуть 255 161,32 грн.

ВИСНОВКИ

Метою квалфікаційної роботи була розробка веб-додатку для автоматизації ведення обліку доставки відправлень перевізником.

Проєкт був розроблений за допомогою безкоштовного редактору Brackets, мові PHP та програмного забезпечення MySQL.

Розроблена програма має зручний та зрозумілий інтерфейс для роботи, який відповідає виконуваним функціям.

Розроблений додаток має наступні переваги:

– авторизація в кабінет менеджера та адміністратора;

– захист особистої інформаціїї;

– швидкий пошук інформації про відправлення за ТТН, витрачення меншого часу;

– створення зручного звіту;

– відповідь на питання клієнтів не виходячі з додатку.

Веб-сайт допомагає покращити рівень задоволеності та лояльності існуючих клієнтів, надаючи їм персоналізовану та якісну обслуговування.

Додаток дозволяє оптимізувати роботу компанії, зменшити витрати на заповнення документів та автоматизування необхідних процесів.

В економічному розділі визначено трудомісткість розробленої інформаційної системи (1 164,6 люд-год), проведений підрахунок вартості роботи по створенню програми (255 161,32 грн) та розраховано час на його створення (6,6 міс).

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. [Електронний Ресурс] — Текстовий редактор Brackets, який використовується при написанні web-додатків: https://habr.com/ru/post/242623/.

2. [Електронний Ресурс] — PHP мова програмування back-end частини: https://www.php.net/manual/ru/intro-whatis.php.

3. Веллінг, Люк. MySQL Навчальний посібник [Текст]: Пер. з англ. / Люк Веллінг, Лора Томпсон – М.: Видавничій дім «Віл'ямс», 2005. – 304 с.

4. Пасічник, В.В. Організація баз даних та знань [Текст] / В.В. Пасічник, В.А. Резніченко – К.: Видавнича група BHV, 2006. – 384 с.

5. Berners-Lee, T. The World Wide Web: A very short personal history [Електронний ресурс] / T. Berners-Lee. – URL: https://www.w3.org/People/Berners-Lee/ShortHistory.html. – Дата публікації: 1998.

6. Markup. Історія розвитку HTML [Електронний ресурс] / Markup. – URL:

7. Webtune. Види сайтів та їх функціонал [Електронний ресурс] / Webtune. – URL: https://webtune.com.ua/statti/web-rozrobka/vydy-sajtiv-ta-yih-funkczional/. –

8. Веб-розробник: середня зарплата в Україні [Електронний ресурс] / Work.ua. – URL: https://www.work.ua/salary. – Дата звернення: 8.06.2023.

9. Веб-сайт [Електронний ресурс] // Вікіпедія : вільна енциклопедія. – URL: https://uk.wikipedia.org/wiki/. – Дата звернення: 10.05.2023.

10. Кузнєцов Максим, Симдянов Ігор Об'єктно-орієнтоване

11. програмування на PHP. / М. Кузнєцов, І. Симдянов - Спб.: «БХВ-Петербург», 2007. - 608 с.

12. Л. Аткінсон, З. Сураскін. PHP5. Бібліотека професіонала. / Л. Аткінсон, З.Сураскін - М.: "Вільямс", 2006 - 543 с.

13. Скотт Хокінс. Адміністрування веб-сервера Apache і керівництво поелектронній комерції. / С. Хокінс. - М.: "Вільямс", 2001. - 336 с.

14. Хокінс С. Адміністрація Web-сервера Apache / С. Хокінс - М.:

15. Вільямс, 2001. - 336 с.20. Больє А. - Learning SQL [Електронний ресурс]. - 2005. - Режим доступу:http://shop.oreilly.com/product/9780596007270.do.

16. Роберт Шелдон, Джоффрей Моє MySQL: базовий курс Beginning MySQL./ Р. Шелдон, Д. Мойє - М.: "Діалектика" 2007. - 880 с.

17. MySQL. Справочник. MySQL AB. - М: "Вільямс", 2006 - 521 с.

18. Кузнєцов Максим, Симдянов Ігор MySQL на прикладах. / М. Кузнєцов, І.Симдянов - Спб.: "БХВ-Петербург", 2008. - 952 с.

19. Леон А. Г. PHP 5. Бібліотека професіонала / А. Г. Леон - М.: Вільямс,2006. - 944 с.

ДОДАТОК А

ЛІСТИНГ ПРОГРАМИ

```
//BaseForm
<?php
namespace base;
use library\Db;
use library\Validator;
abstract class BaseForm{
	protected $_db;
	protected $_errors = [];
	protected $_data;
	protected $_validator = null;
	protected $_sorting = '';
	public function __construct(){
		$this->_db = Db::getDb();
	}
	public function getErrors(){
		return $this->_errors;
	}
	abstract public function getRules();
	public function validate(){
		$validator = new Validator($this->_data, $this->getRules());
		if (!$validator->validateThis()){
			$this->_errors = $validator->getErrors();
			return false;
		}
		return true;
	}

	//
	public function load($data){
		foreach ($data as $propName => $propValue){
			if (property_exists(static::class, $propName)){
				$propValue = $this->_db->getSaveData($propValue);
				$this->$propName = $propValue;
				$this->_data[$propName] = $propValue;
			}else{
				return false;
			}
		}
		return true;
	}

	public function sorting($param = null, $delimiter = null){


		if($param != 'report'){
			$start = null;
			if(!is_null($this->city) and $this->city != '-'){
				$start = (!is_null($delimiter)) ? $delimiter : 'AND';
				$this->_sorting .= $start." department.city_id = ".$this->city;
			}

			if(!is_null($this->department) and $this->department !== '-'){
				$middle = (!empty($this->_sorting) and !is_null($start)) ? ' AND' :
$delimiter;
				$this->_sorting .= $middle." department.id = ".$this->department;
			}
		}else{
```

```
                        if(empty($this->ttn)){
                                $startDate = (empty($this->startDate)) ? '0000-00-00' : $this->startDate;
                                $endDate = (empty($this->endDate)) ? date('Y-m-d') : $this->endDate;

                                if(!is_null($this->city) and $this->city != '-'){
                                        $this->_sorting .= "AND (dispatch.start_city = '{$this->city}' OR
dispatch.finish_city = '{$this->city}') ";
                                }

                                if(!is_null($this->department) and $this->department != '-'){
                                        $this->_sorting .= "AND (dispatch.start_department = '{$this-
>department}' OR dispatch.finish_department = '{$this->department}') ";
                                }

                                $this->_sorting .= "AND (DATE(dispatch.dateCreation) BETWEEN
'{$startDate}' AND '{$endDate}')";
                        }else{
                                $this->_sorting .= "AND dispatch.ttn = ".$this->ttn;
                        }
                }

                return $this->_sorting;
        }

        public function getTable($columns = '*', $table, $data = null){

                if(!is_null($data)){
                        $where = 'WHERE '.$data;
                }

                $sql = "SELECT {$columns} FROM {$table} ".$where;

                $res = $this->_db->sendQuery($sql);

                if(!$res){
                        return \Exception($this->_db->error);
                }

                return $res;
        }

        public function groupingDepartment($data){
                $departments = [];

                for($i = 0; $i < $data->num_rows; $i++){
                        $departments[] = $data->fetch_assoc();
                }

                $newDepartments = [];

                foreach($departments as $key){

                    $id = $key['id'];
                    $adress = $key['adress'];
                    $fragility = $key['fragility'];
                    $city = $key['city'];
                    $day = $key['day'];
                    $startTime = $key['startTime'];
                    $endTime = $key['endTime'];

                    if(!isset($newDepartments[$id])){
```

```
                        $newDepartments[$id] = [
                            'id' => $id,
                            'adress' => $adress,
                            'fragility' => $fragility,
                            'city' => $city
                        ];

                    }

                    $newDepartments[$id]['schedule'][] = [
                        'day' => $day,
                        'startTime' => substr($startTime, 0, 5),
                        'endTime' => substr($endTime, 0, 5),
                    ];
                }

                if(!$newDepartments){
                        return false;
                }

                return $newDepartments;
        }
    }
?>
//Controller
<?php
namespace base;

use library\Auth;
  /**
   *
   */
abstract class Controller{

        protected $_view;

        public function __construct(){
                $this->_view = new View();
                $this->_view->setLayout('main');
        }

        abstract public function actionIndex();

        public function actionLogout(){
                Auth::logout();
                header('Location: ../');
        }
    }
?>
//conf.db
<?php
return [
        'host' => 'localhost',
        'user' => 'root',
        'password' => '',
        'db_name' => 'fastdelivery',
        'charset' => 'utf8'
    ];
?>
//ControllerMain
<?php
namespace controllers;
```

```
use base\Controller;

use library\Auth;
use library\Request;
use library\HttpException;

use models\SearchParcel;
use models\Calculator;
use models\Departments;
use models\Support;
use models\LoginForm;


  /**
 * Главный контроллер
   */
class ControllerMain extends Controller{
        public function actionIndex(){
                $model = new Departments();
                if(Request::getPost('formName') == 'calculator'){
                        $model = new Calculator();
                        if($model->load(Request::getPost()) and $model->validate()){
                                $model->calculator();
                        }
                }

                if(Request::getPost('formName') == 'searchParcel'){
                        $model = new SearchParcel();
                        if($model->load(Request::getPost()) and $model->validate()){
                                $model->search();
                        }
                }
                if(Request::getPost('formName') == 'searchDepartment'){
                        if($model->load(Request::getPost()) and $model->validate()){
                                $model->getDepartments();
                        }
                }
                if(Request::getPost('formName') == 'support'){
                        $model = new Support();
                        if($model->load(Request::getPost()) and $model->validate()){
                                if($model->isCustomers()){
                                        if($model->isNameCustomers()){
                                                $model->save();
                                        }else{
                                                if($model->updateNameCustomers()){
                                                        $model->save();
                                                }
                                        }
                                }else{
                                        if($model->newCustomers()){
                                                $model->save();
                                        }
                                }
                        }
                }
                if(Request::getPost('formName') == 'loginForm'){
                        $model = new LoginForm();
                        if($model->load(Request::getPost()) and $model->validate()){
                                if($model->doLogin()){
                                        if(Auth::canAccess('manager')){
                                                header('Location: /manager');
                                        }elseif(Auth::canAccess('admin')){
                                                header('Location: /admin');
```

```
                                        }
                                    }
                                }
                            }
                            $this->_view->setTitle('FastDelivery - доставка твоєї мрії');
                            $this->_view->render('guest/index', $model);
                    }
            }
?>
//ControllerManager
<?php

namespace controllers;

use base\Controller;
use base\View;

use library\Auth;
use library\AccessException;
use library\Request;


use models\ReportForm;
use models\DepartureForm;
use models\StatusForm;
use models\Feedback;
  /**
 * Контроллер менеджера
   */
class ControllerManager extends Controller{
        public function __construct(){
                if (Auth::isGuest() || !Auth::canAccess('manager')){
                        throw new AccessException('Доступ заборонено!', 403);
                }
                $this->_view = new View();
                $this->_view->setRole(Auth::getUserRole());
                $this->_view->setName(Auth::getUserName());
                $this->_view->setLayout('adminka');
        }
        public function actionIndex(){
                $model = new ReportForm();
                if(Request::getPost('formName') == 'saveStatus'){
                        $model->saveStatus();
                }
                if(Request::isPost()){
                        $model->load(Request::getPost());
                        $model->validate();
                }
                $this->_view->setTitle('Відправлення');
                $this->_view->render('departures', $model);
        }
        public function actionDeparture(){
                $model = new DepartureForm();
                if(Request::isPost()){
                        if($model->load(Request::getPost()) and     $model->validate()){
                                $model->save();
                        }
                }
                $this->_view->setTitle('Створити відправлення');
                $this->_view->render('manager/departure', $model);
        }
        public function actionMessages(){
                $model = new Feedback();
```

```
                if(Request::isPost()){
                        if($model->load(Request::getPost()) and $model->validate()){
                                $model->replyMessage();}
                }
                $this->_view->setTitle('Повідомлення');
                $this->_view->render('messages', $model);
        }
    }
?>
//ControllerAdmin
<?php

namespace controllers;

use base\Controller;
use base\View;

use library\Auth;
use library\AccessException;
use library\Request;
use library\Url;

use models\ReportForm;
use models\CreateManager;
use models\CreateDepartment;
use models\Managers;
use models\Departments;
use models\Feedback;

  /**
 * Контроллер админа
   */
class ControllerAdmin extends Controller{
        public function __construct(){
                if (Auth::isGuest() || !Auth::canAccess('admin')){
                        throw new AccessException('Доступ заборонено!', 403);
                }
                $this->_view = new View();
                $this->_view->setRole(Auth::getUserRole());
                $this->_view->setName(Auth::getUserName());
                $this->_view->setLayout('adminka');
        }
        public function actionIndex(){
                $model = new ReportForm();
                if(Request::isPost()){
                        $model->load(Request::getPost());
                        $model->validate();
                        if(Request::getPost('formName') == 'downloadFile'){
                                $model->downloadFile();
                        }
                }
                $this->_view->setTitle('Звіт');
                $this->_view->render('departures', $model);
        }
        public function actionCreateManager(){
                $model = new CreateManager();
                if(Request::isPost()){
                        if($model->load(Request::getPost()) and $model->validate()){
                                if(Url::getSegment(2)){
                                        $model->update(Url::getSegment(2));
                                        header('Location: /admin/managers');
                                }else{
                                        if($model->save()){
```

```
                                        header('Location: /admin/managers');
                                    }
                                }
                            }

                        }
                        if(Url::getSegment(2)){
                            $this->_view->setTitle('Редагувати менеджера');
                        }else{
                            $this->_view->setTitle('Створити менеджера');
                        }
                        $this->_view->render('admin/createManager', $model);
                    }
                    public function actionCreateDepartment(){
                        $model = new CreateDepartment();
                        if(Request::isPost()){
                            if($model->load(Request::getPost()) and $model->validate()){
                                if(Url::getSegment(2)){
                                    $model->update(Url::getSegment(2));
                                    header('Location: /admin/departments');
                                }else{
                                    if($model->save()){
                                        header('Location: /admin/departments');
                                    }
                                }
                            }
                        }
                        if(Url::getSegment(2)){
                            $this->_view->setTitle('Редагувати відділення');
                        }else{
                            $this->_view->setTitle('Створити відділення');
                        }

                        $this->_view->render('admin/createDepartment', $model);
                    }
                    public function actionManagers(){
                        $model = new Managers();
                        if(Request::getPost('formName') == 'deleteManager'){
                            $model->deleteManager();
                                header('Location: managers');
                        }

                        if(Request::isPost()){
                            $model->load(Request::getPost());
                            $model->validate();
                        }

                        $this->_view->setTitle('Менеджери');
                        $this->_view->render('admin/managers', $model);
                    }
                    public function actionDepartments(){
                        $model = new Departments();
                        if(Request::getPost('formName') == 'deleteDepartment'){
                            $model->deleteDepartment();
                            header('Location: departments');
                        }
                        if(Request::isPost()){
                            $model->load(Request::getPost());
                            $model->validate();
                        }
                        $this->_view->setTitle('Відділення');
                        $this->_view->render('admin/departments', $model);
                    }
```

```
        public function actionMessages(){
                $model = new Feedback();
                if(Request::isPost()){
                        if($model->load(Request::getPost()) and $model->validate()){
                                $model->replyMessage();
                        }
                }
                $this->_view->setTitle('Повідомлення');
                $this->_view->render('messages', $model);
        }
    }
?>
//Db
<?php

namespace library;

use library\DbException;
  /**
 * Класс для работы с базой данных
    */
class Db{
        // Изначально подключение к базе отсутствует
        private static $_db = null;
        private $_link;
        private function __construct(){
                if(!file_exists(__DIR__.'/../config/db.conf.php')){
                        throw new \Exception('Config db not found!');
                }
                $config = require_once __DIR__.'/../config/db.conf.php';
                // Соединяюсь с базой данных
                $this->_link = @new \mysqli($config['host'],
                                                                        $config['user'],
                                                                        $config['password'],
                                                                        $config['db_name']);
                if ($this->_link->connect_error) {
                        throw new DbException($this->_link->connect_error);
                }
                $this->_link->set_charset($config['charset']);
        }
        public static function getDb(){
                if (is_null(self::$_db)) {
                        // Создаю новое соединение
                        self::$_db = new self();
                }
                return self::$_db;
        }
        public function sendQuery($sql){
                $res = $this->_link->query($sql);

                return $res;
        }
        public function getSaveData($data){
                return $this->_link->escape_string($data);
        }
        public function getInsertId(){
                return $this->_link->insert_id;
        }
    }
?>
//Auth
<?php
namespace library;
```

```
  /**
 * Класс для работи с авторизацией
   */
class Auth{
	public static function isGuest(){
		if(empty($_SESSION['user'])){
			return true;
		}
		return false;
	}
	public static function canAccess($role){
		if($_SESSION['user']['role'] == $role){
			return true;
		}
		return false;
	}
	public static function login($id, $name, $role){
		$_SESSION['user']['id'] = $id;
		$_SESSION['user']['name'] = $name;
		$_SESSION['user']['role'] = $role;
	}
	public static function logout(){
		session_unset();
		session_destroy();
	}
	public static function getUserId(){
		return $_SESSION['user']['id'];
	}
	public static function getUserName(){
		return $_SESSION['user']['name'];
	}
	public static function getUserRole(){
		return $_SESSION['user']['role'];
	}
   }
?>
//Request
<?php
namespace library;
  /**
   *
   */
class Request{
	public static function isPost(){
		if($_SERVER['REQUEST_METHOD'] == 'POST'){
			return true;
		}
		return false;
	}
	public static function getPost($param = null){
		if(is_null($param)){
			return $_POST;
		}else{
			return $_POST[$param];
		}
	}
   }
?>
//Url
<?php
namespace library;
```

```
// Клас для работы с URL
class Url{
	protected static function getSegmentsFromUrl(){
		$segments = explode('/', $_GET['url']);
		if (empty($segments[count($segments)-1])) {
			unset($segments[count($segments)-1]);
		}
		$segments = array_map(function($v){
			return preg_replace('/[\'\\\*]/','',$v);
		}, $segments);
		return $segments;
	}
	public static function getParam($paramName){
		return urlencode($_GET[$paramName]);
	}
	public static function getSegment($n){
		$segments = self::getSegmentsFromUrl();
		return $segments[$n];
	}
	public static function getAllSegments(){
		return self::getSegmentsFromUrl();
	}
   }
?>
//Validator
<?php

namespace library;

  /**
    *
    */
 class Validator{
	// Храним все ошибки в виде масива (ключ - имя поля, значение - ошибка)
	protected $_errors = [];
	// Правила - храним правила по которым нужно проверять поля
	protected $_rules = [];
	// Внутринее поле чтоба напрямую работать с полями (елементами форм)
	protected $_fields = [];
	// Данные которые приходят от пользователя с формы
	protected $_data = [];

	public function __construct($data, $rules){
		// Получаю вс необходимые данные для работы
		$this->_rules = $rules;
		$this->_data = $data;
		// Из правил достаю необходимы поля чтобы  дальнейшем с ними работать
		$this->_fields = array_keys($rules);
	}

	// Проверяю заполнено поле или нет
	protected function required($field){
		// Если не заполнено то возвращаю ошибку
		if(empty($this->_data[$field])){
			$this->addError($field, 'Поле обов`язкове до заповнення!');
		}
	}

	// Проверка емейла на коректность
	protected function email($field){

		// Если проверка не пройжена то возвращаю ошибку
		if(!preg_match('/^([\w\-.])+@+([\w\-]{2}+.+[a-zA-Z]{2})$/', $this->_data[$field])){
```

```
                $this->addError($field, 'Email в неправильному форматі!');
            }
        }

        // Проверка чтобы был выбран селект
        protected function selected($field){
            // Если проверка не пройжена то возвращаю ошибку
            if($this->_data[$field] == 'none'){
                $this->addError($field, 'Оберіть варіант!');
            }
        }

        // Проверка что вібрана существующая роль
        protected function sex($field){
            // Если проверка не пройжена то возвращаю ошибку
            if($this->_data[$field] != 'man' and $this->_data[$field] != 'woman'){
                $this->addError($field, 'Оберіть стать співробітника!');
            }
        }


        // Проверка что вібрана существующая роль
        protected function role($field){

            // Если проверка не пройжена то возвращаю ошибку
            if($this->_data[$field] != 'manager' and $this->_data[$field] != 'admin'){
                $this->addError($field, 'Оберіть посаду співробітника!');
            }
        }

        // Проверка номера на коректность
        protected function phone($field){

            // Если проверка не пройжена то возвращаю ошибку
            if(!preg_match('/^[0-9]{10}$/', $this->_data[$field])){
                $this->addError($field, 'Телефон в неправильному форматі! (10 цифр)');
            }
        }

        // Проверка номера на коректность
        protected function number($field){

            // Если проверка не пройжена то возвращаю ошибку
            if(!preg_match('/^[0-9]{1,10}$/', $this->_data[$field]) and !preg_match('/^[\-]{1}$/', $this->_data[$field])){
                $this->addError($field, 'Має бути тільки цифра!');
            }
        }

        // Проверяюна уникальность
        protected function existDetartment($field){
            $sql = "SELECT * FROM department WHERE id = '{$this->_data[$field]}'";

            $res = Db::getDb()->sendQuery($sql);

            if ($res->num_rows == 0){
                $this->addError($field, 'Оберіть відділення!');
            }
        }

        // Проверяюна уникальность
        protected function existCity($field){
            $sql = "SELECT * FROM city WHERE id = '{$this->_data[$field]}'";
```

```
                $res = Db::getDb()->sendQuery($sql);

                if ($res->num_rows == 0){
                        $this->addError($field, 'Оберіть місто!');
                }
        }

        // Проверяю чтобы поля совпадали
        protected function confirm($field){
                if ($this->_data[$field] != $this->_data[$field.'_confirm']){
                        $this->addError($field, 'Поля не співпадають');
                }
        }

        protected function date($field){
                if(!preg_match('/^([0-9]{4})+-+([0-9]{2}+-+[0-9]{2})$/', $this->_data[$field]) and $this->_data[$field] != ''){
                        $this->addError($field, 'Дата вказана в неправильному форматі!'.$this->_data[$field]);
                }
        }
        // Функция для добавления новой ишибки в массив
        public function addError($field, $error){
                $this->_errors[$field] = $error;
        }

        // Получаю все ошибки
        public function getErrors(){
                return $this->_errors;
        }

        // Получаю только одну ошибку
        public function getError($field){
                return $this->_errors[$field];
        }

        //
        public function validateThis(){
                // Прохожусь в цикле по всем правилам
                foreach($this->_rules as $field => $rules) {
                        foreach ($rules as $rule) {
                                // Проверяю если такой метод существует
                                if(method_exists($this, $rule)){
                                        // Если нету ошибки на данном поле
                                        if(is_null($this->getError($field))) {
                                                $this->$rule($field);
                                        }
                                }else{
                                        // В противном слечае возвращаю исключение
                                        throw new \Exception('Невідоме правило перевірки: '.$rule);
                                }
                        }
                }

                // Если ошибки есть то возаращаю false
                if (!empty($this->_errors)){
                        return false;
                }

                // Если ошибки нету то возвращаю true
                return true;
        }
```

```
    }
?>
//Calculator
<?php
namespace models;

use base\BaseForm;

  /**
   *
   */
class Calculator extends BaseForm{
        public $type;
        public $weight;
        public $cost;
        public $fragility;
        public $formName;
        public function getRules(){
                return [
                        'type' => ['required', 'number'],
                        'weight' => ['required'],
                        'cost' => ['required', 'number'],
                        'fragility' => [],
                        'formName' => ['required'],
                ];
        }
        public function calculator($type = null, $weight = null, $fragility = null, $cost = null){
                if(is_null($type) and is_null($weight) and is_null($fragility) and is_null($cost)){
                        $fragility = $this->fragility;
                        $weight = $this->weight;
                        $type = $this->type;
                        $cost = $this->cost;
                        $fragility = ($fragility == 'on') ? 1 : 0;
                }
                $weight = str_replace(",", ".", $weight);
                $sql = "SELECT t_cost FROM tariff WHERE type = {$type} AND startWeight <= {$weight}
AND endWeight >= {$weight}";
                $res = $this->_db->sendQuery($sql);
                if($res->num_rows > 0){
                        $tariff = $res->fetch_assoc()['t_cost'];
                        $price = $tariff + ($fragility * ($cost * 0.05));
                        return $this->_errors['price'] = $price;
                }
                $this->_errors['price'] = 'Ми не приймаємо такий тип вантажу з такою вагою!';
                return false;
        }
    }
?>
//CreateDepartment
<?php
namespace models;

use base\BaseForm;

  /**
   *
   */
class CreateDepartment extends BaseForm{

        public $day = ['Пн.','Вт.','Ср.','Чт.','Пт.','Сб.','Нд.'];
        public $timeTitle  = ['з','до'];
        public $name =  ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday'];
```

```
        public $city;
        public $adress;

        public $Monday0;
        public $Monday1;
        public $Tuesday0;
        public $Tuesday1;
        public $Wednesday0;
        public $Wednesday1;
        public $Thursday0;
        public $Thursday1;
        public $Friday0;
        public $Friday1;
        public $Saturday0;
        public $Saturday1;
        public $Sunday0;
        public $Sunday1;

        public $fragility = null;

        public function getRules(){
            return [
                'city' => ['required', 'existCity'],
                'adress' => ['required'],
                'Monday0' => ['required', 'number'],
                'Monday1' => ['required', 'number'],
                'Tuesday0' => ['required', 'number'],
                'Tuesday1' => ['required', 'number'],
                'Wednesday0' => ['required', 'number'],
                'Wednesday1' => ['required', 'number'],
                'Thursday0' => ['required', 'number'],
                'Thursday1' => ['required', 'number'],
                'Friday0' => ['required', 'number'],
                'Friday1' => ['required', 'number'],
                'Saturday0' => ['required', 'number'],
                'Saturday1' => ['required', 'number'],
                'Sunday0' => ['required', 'number'],
                'Sunday1' => ['required', 'number'],
                'fragility' => []
            ];
        }

        public function save(){

            $fragility = ($this->fragility == 'on') ? 1 : 0;

            $sql = "INSERT INTO department (city_id, adress, fragility) VALUES ('{$this->city}', '{$this-
>adress}', '{$fragility}')";

            $res = $this->_db->sendQuery($sql);

            $department_id = $this->_db->getInsertId();

            $sql = "INSERT INTO schedule (department_id,
                                                day,
                                                startTime,
                                                endTime)
                                VALUES ('{$department_id}', '{$this->day[0]}', '{$this->Monday0}:00:00',
'{$this->Monday1}:00:00'),
                                        ('{$department_id}', '{$this->day[1]}', '{$this-
>Tuesday0}:00:00', '{$this->Tuesday1}:00:00'),
```

```
                                                ('{$department_id}',    '{$this->day[2]}',    '{$this-
>Wednesday0}:00:00', '{$this->Wednesday1}:00:00'),
                                                ('{$department_id}',    '{$this->day[3]}',    '{$this-
>Thursday0}:00:00', '{$this->Thursday1}:00:00'),
                                                ('{$department_id}',    '{$this->day[4]}',    '{$this-
>Friday0}:00:00', '{$this->Friday1}:00:00'),
                                                ('{$department_id}',    '{$this->day[5]}',    '{$this-
>Saturday0}:00:00', '{$this->Saturday1}:00:00'),
                                                ('{$department_id}',    '{$this->day[6]}',    '{$this-
>Sunday0}:00:00', '{$this->Sunday1}:00:00')";

                        $res = $this->_db->sendQuery($sql);

                        if(!$res){
                                $this->_errors['CreateDepartment'] = 'Error!';
                                return false;
                        }

                        return true;
                }

                public function update($departmentId){
                        $fragility = ($this->fragility == 'on') ? 1 : 0;

                        $sql = "UPDATE department SET city_id = '{$this->city}', adress = '{$this->adress}', fragility
= '{$fragility}' WHERE id = ".$departmentId;

                        $this->_db->sendQuery($sql);

                        $sql = "DELETE FROM schedule WHERE department_id = ".$departmentId;

                        $this->_db->sendQuery($sql);

                        $sql = "INSERT INTO schedule (department_id,
                                                                day,
                                                                startTime,
                                                                endTime)
                                                VALUES ('{$departmentId}', '{$this->day[0]}', '{$this->Monday0}:00:00',
'{$this->Monday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[1]}',    '{$this-
>Tuesday0}:00:00', '{$this->Tuesday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[2]}',    '{$this-
>Wednesday0}:00:00', '{$this->Wednesday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[3]}',    '{$this-
>Thursday0}:00:00', '{$this->Thursday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[4]}',    '{$this-
>Friday0}:00:00', '{$this->Friday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[5]}',    '{$this-
>Saturday0}:00:00', '{$this->Saturday1}:00:00'),
                                                ('{$departmentId}',     '{$this->day[6]}',    '{$this-
>Sunday0}:00:00', '{$this->Sunday1}:00:00')";

                        $res = $this->_db->sendQuery($sql);

                        if(!$res){
                                $this->_errors['CreateDepartment'] = 'Error!';
                                return false;
                        }

                        return true;
                }

                public function getDepartment($param){
```

```
                        $sql = "SELECT department.id,
                    department.city_id AS city,
                    department.adress,
                    department.fragility,
                    schedule.day,
                    schedule.startTime,
                    schedule.endTime
                FROM department AS department
                LEFT JOIN schedule AS schedule ON schedule.department_id = department.id WHERE department.id
= ".$param;

                        $res = $this->_db->sendQuery($sql);

                        $departments =  $this->groupingDepartment($res);

                        if(!$departments){
                                return false;
                        }

                        foreach($departments as $department){
                                $department = $department;
                        }

                        return $department;
                }

        }
?>
//SearchDepartment
<?php
namespace models;

use base\BaseForm;

  /**
   *
   */
class SearchDepartment extends BaseForm{
        public $city;
        public $department;
        public $formName;

        public function getRules(){
                return [
                        'city' => ['number'],
                        'department' => ['number'],
                        'formName' => ['required']
                ];
        }

        public function search(){

                $this->sorting = $this->sorting(null, 'WHERE');

                $sql = "SELECT department.id,
                    department.adress,
                    department.fragility,
                    city.city,
                    schedule.day,
                    schedule.startTime,
                    schedule.endTime
                FROM department
                LEFT JOIN city ON city.id = department.city_id
```

```
                LEFT JOIN schedule ON schedule.department_id = department.id ".$this->sorting;

                        $res = $this->_db->sendQuery($sql);

                        $departments =  $this->groupingDepartment($res);

                        if($res->num_rows > 0){
                                $this->_errors['SearchDepartment'] = $departments;
                                return true;
                        }
                        $this->_errors['SearchDepartment'] = 'Відділення не знайдено!';

                        return false;
                }
        }
?>
//main
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="utf-8">
    <title><?= $this->title ?></title>
    <link rel="stylesheet" href="/assets/css/style.css">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.6.1/css/all.css">
  </head>
  <body>
    <?php include __DIR__.'/../'.$tplName.'.php'; ?>
<script type="text/javascript">
    const city = document.querySelector('select[name="city"]');
    const department = document.querySelector('select[name="department"]');
    city.onchange = function() {
      department.querySelectorAll('option').forEach(opt => {
        if (opt.dataset.id != city.value) opt.hidden = true;
        else opt.hidden = false;
        opt.selected = false;
      });
    }
</script>
  </body>
</html>
//admin.css
/*@import
url('https://fonts.googleapis.com/css2?family=Roboto:ital,wght@0,300;0,500;0,900;1,400&display=swap');*/

*{
margin: 0;
padding: 0;
font-family: sans-serif;
box-sizing: border-box;
font-size: 13px;
}
body, a{
color: #334f58;
background-color: rgb(250,250,250);
}
ul{
list-style: none;
}
ul li{

}
a{
```

```
text-decoration: none;
}
input[type="text"],
input[type="date"],
input[type="password"],
input[type="number"],
.select,
textarea,
button,
.button{
display: inline-block;
border: none;
outline: none;
box-shadow: inset 0 0 5px rgba(0,0,0,0.2);
background-color: white;
padding: 10px;
font-family: inherit;
color: inherit;
border-radius: 5px;
}
textarea{
width: 100%;
max-width: 500px;
height: 100px;
resize: none;
}
button, .button{
cursor: pointer;
background-color: #C0392B;
color: white;
box-shadow: 0 0 5px rgba(0,0,0,0.2);
}
button i, .button i{
margin: 0 5px 0 5px;
}
.btn-info{
background-color: #16a085
}
.btn-download{
background-color: #2980b9;
}
button:hover, .button:hover{
box-shadow: inset 0 0 5px rgba(0,0,0,0.2);
}
/*
Header
*/
header{
/*position: fixed;*/
background-color: white;
width: 100%;
height: 60px;
display: flex;
align-items: center;
justify-content: space-between;
-webkit-box-shadow: 0 0 3px rgba(0,0,0,0.2);
-moz-box-shadow:    0 0 3px rgba(0,0,0,0.2);
box-shadow:         0 0 3px rgba(0,0,0,0.2);
}
.header__logo,
.header__profile{
height: 100%;
display: flex;
```

```
align-items: center;
padding: 0 20px;
}
.header__profile{
cursor: pointer;
}
.header__logo h3{
font-size: 22px;
font-weight: 700;
cursor: pointer;
margin: 0;
}
/*
Burger
*/
.btn-burger{
width: 30px;
height: 30px;
display: none;
justify-content: center;
align-items: center;
cursor: pointer;
margin-right: 20px;
}
.btn-burger i{
font-size: 20px;
}
#input__check{
display: none;
}

#input__check:checked ~ .sidebar{
display: block;
}
#input__check:checked ~ .content{
width: calc(100% - 220px);
display: none;
}
#input__check:checked ~ .stub{
display: block;
}
.stub{
position: absolute;
top: 62px;
right: 0;
bottom: 0;
left: 0;
background-color: rgba(0,0,0,0.8);
z-index: 998;
display: none;
}
/*
End burger
*/
.header__profile--img{
min-width: 40px;
min-height: 40px;
background-color: #334f58;
border-radius: 50%;
display: flex;
justify-content: center;
align-items: center;
color: white;
```

```
box-shadow: 0 0 2px black
}
.header__profile--name{
margin-left: 10px;
}
.header__profile--name span{
font-weight: normal;
}

/*
 Main
*/
main{
display: flex;
justify-content: space-between;
padding: 10px;
}


/*
Sidebar
*/
.sidebar{
background-color: white;
width: 220px;
-webkit-box-shadow: 0 2px 4px 0 rgba(0,0,0,.22);
-moz-box-shadow:    0 2px 4px 0 rgba(0,0,0,.22);
box-shadow:                 0 2px 4px 0 rgba(0,0,0,.22);
   transition: 1s;
   position: relative;
   z-index: 999
}
.sidebar ul li a{
/*padding: 10px;*/
position: relative;
display: flex;
align-items: center;
background-color: white;
color: rgba(0,0,0,.54);
font-weight: 400;
margin-bottom: 2px;
}
.sidebar ul li a:hover{
/*color: red;*/
color: black;
background-color: #e8edff;
}
.sidebar i{
width: 40px;
height: 40px;
font-size: 16px;
display: inline-block;
line-height: 40px;
text-align: center;
}

/*
Content
*/
.content{
padding: 15px;
width: calc(100% - 240px);
background-color: white;
```

```
-webkit-box-shadow: 0 2px 4px 0 rgba(0,0,0,.22);
-moz-box-shadow:    0 2px 4px 0 rgba(0,0,0,.22);
box-shadow:                 0 2px 4px 0 rgba(0,0,0,.22);
}
.content h4{
font-size: 18px;
margin-bottom: 20px;
}


/*
 Table
*/
.scroll{
max-height: 500px;
overflow: auto;
}
table{
border-collapse: collapse;
width: 100%;
}
table th {
  padding: 10px 10px;
  font-weight: bold;
}
table td {
  color: rgba(0,0,0,.54);
  border-top: 1px solid #e8edff;
  padding: 10px 10px;
  text-align: center;
}
table tr:nth-child(2n) {background: #e8edff;}



/*
 Form
*/

.form__label, .select, .form__container{
display: block;
margin-bottom: 20px;
}
.input__name{
margin-bottom: 5px;
display: block;
/*font-weight: bold;*/
}
form input[type="text"]{
width: 100%;
max-width: 250px;
}
.form__container div{
display: inline-block;
margin: 5px 5px
}


/*
Schedule
*/
.schedule__container{
text-align: left;
```

```
width: 100px;
}
.schedule{
display: flex;
justify-content: space-between;
padding-bottom: 5px;
width: 100px;
}
.schebule span:last-child(){
padding-right: 5px;
}

/*
Sorting
*/
.sorting__container{
display: inline-block;
margin-right: 10px;
margin-bottom: 10px;
}
.sorting__container button{
margin-right: 10px;
}

.w400{
max-width: 400px;
min-width: 200px;
}
.sidebar.active{
    display: block;
    position: absolute;
    margin: -10px;
    width: 400px;
}
.error{
font-size: 12px;
color: tomato;
}
@media (max-width: 1210px) {
  .sidebar{
    display: none;
  }
  .content{
  width: 100%;
  }
  .header__logo{
    cursor: unset;
  }
  .btn-burger{
  display: flex;
  }
}
@media (max-width: 450px) {
  .header__logo h3{
    display: none;
  }
}
@media (min-width: 1210px) {
  .stub{
    display: none;
  }
}
@media (min-width: 1920px) {
```

```
    *{
        font-size: 18px;
    }
.sidebar{
        width: 250px;
    }
.content{
        padding: 15px;
        width: calc(100% - 270px);
    }
.schedule{
        width: 150px;
    }
.header__logo h3{
        font-size: 27px;
    }
}
//style.css
*{
  margin: 0;
  padding: 0;
  font-family: "montserrat",sans-serif;
  font-size: 14px;
  box-sizing: border-box;
}
ul li{
  display: inline-block;
  list-style: none;
}
a{
  text-decoration: none;
  color: #C0392B;
}
input, select, textarea{
  border-style: none;
  background-color: white;
  padding: 10px;
  font-family: inherit;
  color: inherit;
  outline: none;
  border-radius: 5px;
  border-radius: 5px;
  box-shadow: inset 0 0 5px rgba(0,0,0,0.2);
}
textarea{
  resize: none;
}
input[type=checkbox]{
  display: none;
}
input[type=checkbox] + span{
  width: 30px;
  height: 30px;
  border-radius: 50%;
  background-color: white;
  display: inline-block;
  position: absolute;
  bottom: 5px;
  display: flex;
  justify-content: center;
  align-items: center;
  box-shadow: 0px 2px 5px rgba(0,0,0,0.2);
}
```

```
input[type=checkbox]:checked + span{
  box-shadow: inset 0px 2px 5px rgba(0,0,0,0.2);
}
input[type=checkbox]:checked + span:after{
  content: '\2714';
  position: absolute;
  font-weight: bold;
}
button{
  border: none;
  background-color: #C0392B;
  border-radius: 5px;
  padding: 10px 15px;
  color: white;
  box-shadow: 2px 2px 10px rgba(0,0,0,0.2);
  margin-bottom: 25px;
  outline: none;
  cursor: pointer;
}
button:hover{
  box-shadow: inset 2px 2px 10px rgba(0,0,0,0.2);
}
/*
  ********** Header **********
*/
header{
  position: relative;
  max-height: 700px;
  max-width: 1824px;
  margin: 0 auto;
  overflow: hidden;
  background-image: url(../img/bg-header.jpg);
  background-position: left;
  background-repeat: no-repeat;
  background-size: cover;
}

/*
  header menu
*/
.header__menu{
  position: fixed;
  top: 0;
  width: 100%;
  max-width: 1824px;
  height: 80px;
  background-color: #FCF7F8;
  padding: 0 25px;
  box-shadow: 0 2px 10px rgba(0,0,0,0.2);
  z-index: 999;
  display: flex;
  align-items: center;
  justify-content: space-between;
}
.header__menu h2{
  font-size: 30px;
  color: #334f58;
}

.header__menu .menu{
  width: 100%;
  display: flex;
  justify-content: space-between;
```

```
  align-items: center;
}
.header__menu .menu ul{
  margin-left: 100px;
}
.header__menu .menu ul li a{
  color: #334f58;
  margin: 0 20px;
}
.header__menu .menu ul li a:hover{
  color: #C0392B
}
.header__menu .menu i{
  display: none;
  width: 40px;
  height: 30px;
  font-size: 18px;
  line-height: 30px;
  text-align: center;
}
.btn-support{
  border: 1px solid #C0392B;
  padding: 0 25px;
  height: 40px;
  border-radius: 5px;
  display: flex;
  align-items: center;
  color: #C0392B;
}
.btn-support:hover{
  background-color: #C0392B;
  color: #FCF7F8;
}

/*
  header body
*/
.header__body{
  margin-top: 80px;
  height: calc(100% - 80px);
  display: flex;
}
.header__body--container{
  flex: 1 1;
}
.header__body--container:first-child{
  display: flex;
  flex-direction: column;
  justify-content: center;
}
.title--description{
  font-style: italic;
  font-size: 20px;
  line-height: 23px;
  color: white;
  margin-bottom: 20px;
  color: #ccc;
}
.title--main{
  font-weight: bold;
  font-size: 40px;
  line-height: 47px;
  color: white;
```

```
  max-width: 500px;
}
.title--description,
.title--main{
  margin-left: 100px;
}
.header__body--container:last-child{
  display: flex;
  justify-content: center;
  align-items: center;
}
.calculator__container{
  width: 100%;
  max-width: 700px;
  background-color: #FCF7F8;
  border-radius: 15px;
  box-shadow: 5px 5px 10px black;
  margin: 50px;
}
.calculator__container h3{
  text-align: center;
  padding: 25px;
  font-size: 20px;
}
.input__container{
  display: flex;
}
.input__container div{
  flex: 1 1;
  padding: 0 30px 30px 30px;
}
.input__container label{
  display: block;
  height: 100%;
  display: flex;
  flex-direction: column;
  justify-content: center;
  position: relative;
}
.input__container label span{
  padding: 5px 0;
}
.input__container input[type="text"],
.input__container select{
  width: 100%;
  background-color: transparent;
}
.check__span{
  margin: 25px 0 0 40px;
}


/*
  ********** Main **********
*/
main{
  width: 100%;
  max-width: 1824px;
  margin: 0 auto;
  box-shadow: inset 0px 15px 10px -10px black;
}
/*
  cards
```

```
*/
.section__name{
  font-weight: bold;
  font-size: 24px;
  text-align: center;
  padding: 30px;
  color: #334f58;
}
.section__cards{
  display: flex;
  flex-wrap: wrap;
  justify-content: space-around;
}
.card__container{
  width: 25%;
  min-width: 250px;
  margin: 30px auto;
  text-align: center;
  padding: 0 25px;
}
.card__container--item{
  width: 150px;
  height: 150px;
  border-radius: 50%;
  box-shadow: 0 5px 5px rgba(0,0,0,0.2);
  display: flex;
  justify-content: center;
  align-items: center;
  margin: 0 auto;

}
.card__container--item i{
  font-size: 60px;
  color: #444C53;

}
.card__title{
  font-weight: bold;
  margin: 10px;
  display: inline-block;
}
/*
  section
*/
section{
  padding: 80px 25px;
}
.search, .section__auth{
  background: linear-gradient(white, #FCF7F8, white);
}
section input,
section select{
  height: 50px;
  margin-right: 25px;
  margin-bottom: 25px;
}
section button{
  height: 50px;
}
.search button,
.section__support button,
.section__auth button{
  width: 200px;
```

```
}
.section__search--ttn input{
  width: calc(100% - 230px);
}
.section__search--branch select,
.section__auth input{
  width: calc(50% - 130px);
}
p{
  margin: 15px 0;
  color: #334f58;
}
.section__support--container{
   display: flex;
   flex-direction: column;
   align-items: center;
}
.section__support--container div{
   width: 100%;
   max-width: 500px;
}
.section__support input,
.section__support textarea{
  display: block;
  width: 100%;
}
.section__support textarea{
  margin-bottom: 25px;
  height: 100px;
}
.error{
  font-size: 12px;
  color: tomato;
}

@media (max-width: 1140px) {
  .header__body{
    flex-direction: column;
  }
  .header__body--container:first-child{
    display: none;
  }
  .header__menu .menu ul{
    margin-left: 50px;
  }
  .header__menu .menu ul li a{
    margin: 0 10px;
  }
}

@media (max-width: 940px) {
  .header__menu .menu span{
    display: none;
  }
  .header__menu .menu i{
    display: block;
  }
}

@media (max-width: 650px) {
  .header__menu{
    padding: 10px;
    height: 60px;
```

```
  }
  .header__menu h2{
    font-size: 20px;
  }
  .btn-support{
    padding: 0;
  }
  .header__menu .menu i{
    font-size: 14px;
  }
  .section__search--ttn input{
    width: 100%;
  }
  .section__search--branch select,
  .section__auth input{
    width: 100%;
  }
  section{
    padding: 40px 25px;
  }
}

@media (max-width: 550px) {
  .calculator__container h3{
    font-size: 18px;
  }
  .input__container{
    flex-direction: column;
  }
  .input__container div{
    padding: 0 15px 10px 15px;
  }
  input[type=checkbox] + span{
    bottom: 0;
  }
  .check__span{
    margin: 10px 0 0 40px;
  }
}

@media (max-width: 420px) {
  .header__menu .menu ul{
    margin-left: 0;
  }
}
```

ДОДАТОК Б

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

ДОДАТОК В

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
121-20зск-1_Диплом_Новікова Н.О.doc	Пояснювальна записка до кваліфікаційної роботи. Документ Word.
121-20зск-1_Диплом_Новікова Н.О.pdf	Пояснювальна записка до кваліфікаційної роботи в форматі PDF
Програма	
xfastDelivery.rar	Архів. Містить коди програми і откомпільовану програму
Презентація	
121-20зск-1_Презентація_Новікова Н.О.ppt	Презентація кваліфікаційної роботи