**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ІВАНА ФРАНКА**

**ФАКУЛЬТЕТ УПРАВЛІННЯ ФІНАНСАМИ ТА БІЗНЕСУ**

**Кафедра цифрової економіки та бізнес-аналітики**

**КУРСОВА РОБОТА**

**з дисциплін  професійної та практичної підготовки**

на тему:

Інформаційна система для бізнесу з оренди складських боксів

**спеціальність:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**051 «Економіка»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                              (код та найменування спеціальності)

**спеціалізація:** \_\_\_\_\_\_\_\_\_«Інформаційні технології в бізнесі» \_\_\_\_\_\_\_\_\_\_\_

                             (найменування спеціалізації)

**освітній ступінь: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**бакалавр**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

              (бакалавр/магістр)

|  |  |  |
| --- | --- | --- |
| **Науковий керівник:**  к.е..доцент кафедри **Депутат Б.Я**.    (науковий ступінь, посада, прізвище, ініціали)  \_\_\_\_\_\_\_\_\_\_  **“\_\_\_\_”** \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.       (підпис) |  | **Виконавець:**  студент групи УФЕ-21с Мазур Н.Ю                         (прізвище, ініціали)  \_\_\_\_\_\_\_\_\_\_\_\_  **“\_\_\_\_”** \_\_\_\_\_\_\_\_\_\_\_ 20\_\_ р.      (підпис) |

**Загальна**

**кількість балів**  \_\_\_\_\_\_\_  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                                                                            (підпис, ПІП членів комісії)

**ЛЬВІВ 2023**

**ЗМІСТ**

**ВСТУП**

**Актуальність теми.** Оренда складських приміщень є популярною послугою в багатьох галузях бізнесу, таких як логістика, електронна комерція, виробництво тощо. Багато компаній, звичайних людей, людей які займаються підприємництвом потребують спеціального сервісу, який допоможе спростити процес оренди складських площ, де вони могли б зберігати свої речі ( особисті речі, товари, техніку , документи і т.д).

**Мета та завдання дослідження:** розробити інформаційну систему для бізнесу з оренди складських боксів. З’ясувати зміст та роль цієї системи для данного бізнесу.

Для досягнення мети в роботі поставлено й вирішено такі теоретичні та практичні *завдання*:

* Проаналізувати вимоги до інформаційної системи
* Вибір технологій ( мов програмування, фреймворків, серверного середовища, система управління базами даних), які будуть використовуватись для розробки веб-додатку та серверної частини і бази данних
* Розробка структури бази даних, яка буде використовуватись для зберігання інформації про складські бокси, клієнтів тощо
* Реалізувати функціоналість для клієнтів: cтворити сайт, що дозволить клієнтам здійснювати оренду складських боксів, а також переглядати інформацію про доступність та розташування цих боксів
* Обґрунтувати та розробити серверну логіку, яка дозволить обробляти запити від веб-додатку, взаємодіяти з базою даних, забезпечит автентифікацію та авторизацію користувачів, здійснюватиме операції з боксами. А також її тестування та налагодження, перевірка продуктивності та стабільності
* Розгорнути розроблену системи на сервері, налаштувати інфраструктуру, встановити необхідні залежності та конфігурації.

*Об'єкт дослідження***:** теоретичні та практичні основи організації бізнесу з оренди складських боксів та процес розробки веб-сайту з серверною частиною та базою данних

*Предмет дослідження***:** вимоги до інформаційної системи, конкурентні рішення на ринку, база даних, технології для розробки, веб-додаток(веб-сайт), серверна частина.

**Практичне значення отриманих результатів**. Веб-сай який буде допомагати клієнтам отримати інформацію та орендувати складську площу. Інформаційна система (веб-сайт, сервер, база данних) яка допоможе підприємству автоматизувати та спростити частину процесів.

**Використане програмне забезпечення**. Для створення програмного продукту використовувалось середовище розробки Intellij IDEA та мови програмування – Kotlin, JavaScript, мова розмітки - HTML, мова стилів – CSS, VirtualBox, Windows 10, SQL Server Management Studio 19, Azure Data Studio, Google Chrome, Microsoft Edge, Gradle (менеджер залежностей, збірка і компіляція проекту ), система керування версіями Git

**Структура роботи.** Курсова робота складається з чотирьох розділів («Проблеми автоматизації бізнес-процесів залізниці», «Функціональні інтерфейси в JAVA», «Adapter шаблон проектування програмного забезпечення» та «Розробка мобільного додатку»), висновків, списку використаних джерел та додатків. Загальний обсяг роботи − 71 сторінка.

**РОЗДІЛ 1 . АНАЛІЗ ВИМОГ**

* 1. **Постановка завдання**

Перш за все необхідно проаналізувати вимоги до інформаційної системи: Провести детальний аналіз вимог до функціональності та характеристик системи. Визначити, які можливості має мати система для оренди складських боксів та перегляду інформації про їх доступність та розташування. Врахувати потреби клієнтів та бізнесові вимоги.

Технічне завдання яке було сформоване в процесі аналізу можна подивтися у додатку A.

По технічному завданню за допомогою інструменту Office 365 Visio була складена USE CASE діаграма(рис.1.1).

A picture containing text, screenshot, diagram, font

Description automatically generatedРис.1.1 Use Case діаграма

Після дослідження бізнесових аспектів можна перейти безпосередньо до ІТ аспектів.

Насамперед, необхідно обрати, які технології будуть використовуватись для розробки веб-додатку, серверної частини та бази даних. Вибрати мову програмування, фреймворки, серверне середовище та систему управління базами даних, які найкраще підходять для реалізації функціональності та вимог проекту.

Для розробки структури бази даних потрібно визначити, яка інформація зберігатися в БД про складські бокси, клієнтів та інші відповідні дані. Далі слід створити оптимальну структуру бази даних, включаючи таблиці, поля, обмеження, зв'язки та індекси, які відповідають вимогам системи.

З метою реалізації клієнтського функціоналу потрібно, розробити веб-додаток, що дозволить клієнтам здійснювати процес оренди складських боксів та переглядати інформацію про їх доступність та розташування. Основним завданням є розробка необхідних функцій, створення інтерфейсу користувача та інших складових системи.

Після розробки функціональності клієнтської частини, наступним кроком є обґрунтування та розробка серверної логіки. Сервер буде відповідати за обробку запитів, що надходять від веб-додатку, взаємодію з базою даних, а також забезпечуватиме механізми автентифікації та авторизації користувачів, а також виконання різних операцій з боксами.

Важливим етапом є проведення тестування та налагодження серверної логіки. Це дозволить перевірити правильність роботи системи, виявити та виправити можливі помилки та недоліки. Також важливо провести перевірку продуктивності та стабільності системи.

І на сам кінець, необхідно розгорнути розроблену систему на сервері (забезпечити функціонування бази данних та серверу для web-додатку). Також потрібно виконати налаштування інфраструктури, встановити всі необхідні залежності та конфігурації, щоб гарантувати правильну роботу системи. Не менш важливо забезпечити належну інтеграцію між сервером та всіма необхідними засобами для її нормального функціонування.

* 1. **Розробка моделі варіантів використання веб-сайту**

Веб-сайт повинен мати реєстрацію користувачів, щоб нові користувачі могли створити обліковий запис. Після реєстрації, користувачі повинні мати можливість авторизуватися на сайті, використовуючи свої облікові дані. Це дозволяє отримати доступ до особистого та інформації про бронювання.

Крім того, на веб-сайті користувачі повинні мати можливість переглядати список доступних для бронювання складських боксів. Інформація про кожен бокс, така як його розмір, вартість та місцезнаходження, повинна бути доступна для перегляду. Затим, користувачі можуть обрати необхідний бокс і забронювати його, вказавши дати початку та закінчення бронювання. Після успішного бронювання, користувач отримує підтвердження та додаткову інформацію.

Також, на веб-сайті повинна бути функція авторизації для працівників. Це дозволяє їм мати доступ до спеціалізованої функціональності. Працівники можуть переглядати нові замовлення, які надійшли на сайт. Вони мають можливість підтверджувати або скасовувати ці замовлення, щоб вирішувати процес бронювання та оновлювати статус замовлень.

Отже, основні функції для користувачів включають перегляд доступних складських боксів, їх бронювання, авторизацію на сайті та перегляд власних бронювань.

Відповідно, основні функції працівників, включають авторизацію, перегляд нових замовлень та можливість їх підтвердження або скасування. Ці функції сприяють зручному та ефективному використанню веб-сайту як для користувачів, так і для працівників.

**1.3 Аналіз засобів реалізації (техніко-економічне обґрунтування**

**вибору)**

Декілька основних засобів, які можуть бути використані в процесі розробки, включають Kotlin, Spring Boot, Spring MVC, Thymeleaf, HTML, CSS, JS, jQuery, бібліотеку js-snackbar для JS та SQL (зокрема MS SQL).

Kotlin є сучасною мовою програмування, яка працює на віртуальній машині Java (JVM). Вона дозволяє писати чистий, експресивний і безпечний код, що спрощує розробку та підтримку інформаційної системи.[1]

Spring Boot є фреймворком для розробки веб-додатків на основі Spring Framework. Він надає швидку та просту конфігурацію, автоматичне управління залежностями і підтримку вбудованого веб-сервера. Spring Boot сприяє швидкому старту проекту та забезпечує ефективну розробку інформаційної системи.[2]

Spring MVC є реалізацією шаблону проектування Model-View-Controller (MVC) для розробки веб-додатків. Вона забезпечує поділ додатка на модель (дані), представлення (відображення) і контролер (логіку обробки запитів). Spring MVC дозволяє ефективно обробляти HTTP-запити і відповіді, що допомагає розробити функціональну інформаційну систему.

Thymeleaf, HTML, CSS, JS і jQuery є основними засобами для розробки інтерфейсу користувача та взаємодії з ним. Thymeleaf є шаблонним двигуном для створення динамічних веб-сторінок з використанням HTML, CSS і JavaScript. HTML, CSS і JS використовуються для структуризації, стилізації та програмування веб-сторінок, а jQuery допомагає спростити маніпулювання HTML-елементами та роботу з подіями на сторінці.

Бібліотека js-snackbar для JS є додатковим інструментом для відображення сповіщень (snackbar) на веб-сторінці з використанням JavaScript. Вона дозволяє відображати інформаційні повідомлення або попередження для користувачів інформаційної системи.

SQL (MS SQL) використовується для управління базами даних. MS SQL є системою управління базами даних (СУБД), яка надає потужні можливості для зберігання, зчитування, оновлення та видалення даних. Вона є важливим елементом в розробці інформаційної системи, оскільки дозволяє зберігати та обробляти необхідну інформацію.

Також, можуть бути використані такі програмні рішення як IntelliJ IDEA, SQL Server Management Studio 19, Azure Data Studio, Gradle, Git, VirtualBox, Windows 10, Google Chrome і Microsoft Edge які допоможуть оптимізувати, спростити та пришвидчити процес розробки.

Середовище розробки, таке як IntelliJ IDEA, надає зручність та підтримку для розробки програмного забезпечення на Kotlin і Java. Воно включає такі функції як автодоповнення, налагодження, управління залежностями та багато інших функцій, які сприяють швидкій і ефективній розробці інформаційної системи.

SQL Server Management Studio 19 є графічним інструментом для управління базами даних SQL Server. Він надає зручний спосіб створення та управління базами даних, таблицями, запитами і процедурами. SSMS допомагає ефективно працювати з базою даних.

Azure Data Studio є інтегрованою розробкою середовищем для роботи з базами даних, включаючи SQL Server та Azure SQL Database. Воно надає зручні інструменти для створення, налагодження та управління базами даних, запитами і збереженими процедурами. Azure Data Studio може бути корисним для розробки та керування базою даних. Також , на мою думку, інтерфейс цієї програми є більш зручним у користуванні ніж інтерфейс SQL Server Management Studio 19 , тому що він заснований на новому дизайні, в той час інтерфейс SSMS 19 все ще наслідує попередні версії ( дизайн інтерфейсу в яких був розроблений для старих операційних систем Windows).

Gradle є сучасним інструментом для автоматизації процесу збірки, тестування та розгортання програмного забезпечення. Він використовуєся для керування залежностями проекту, компіляції коду, виконання тестів, створення пакетів і багато інших задач розробки. Однією з головних переваг Gradle є його гнучкість і потужний механізм конфігурації, що дозволяє налаштовувати процес збірки під конкретні потреби проекту.

Git є розподіленою системою керування версіями, яка дозволяє відстежувати зміни в кодовій базі та спільно працювати над проектом в команді. Використання Git у розробці інформаційної системи з оренди складських боксів допоможе зберігати історію змін, створювати гілки для розвитку нових функцій і виправлення помилок, а також спрощувати процес об'єднання змін з різних джерел.

Додатково, такі програми як VirtualBox, Windows 10, Google Chrome і Microsoft Edge також можуть бути використані під час розробки інформаційної системи. VirtualBox дозволяє створювати віртуальні машини для запуску інших операційних систем. Windows 10 є операційною системою, яка може використовуватись для розробки інформаційних систем. Google Chrome і Microsoft Edge є веб-браузерами, які можуть використовуватись для тестування та перегляду розробленої веб-сторінки.

Ці засоби реалізації є потужними інструментами для розробки інформаційної системи з оренди складських боксів. Вони надають необхідні можливості для розробки функціональності системи, взаємодії з базою даних, розробки користувацького інтерфейсу та тестування додатку.

Економічне обґрунтування вибору засобів реалізації може включати наступні аспекти:

Один з факторів, які можуть вплинути на економічні показники проекту, - це вартість використовуваних засобів. У даному випадку, Kotlin, Spring Boot, Spring MVC, Thymeleaf, HTML, CSS, JS, jQuery, js-snackbar, SQL (MS SQL) є відкритими та безкоштовними засобами, що дозволяє знизити витрати на придбання ліцензійного програмного забезпечення. Однак, варто враховувати, що деякі інструменти, такі як IntelliJ IDEA або SQL Server Management Studio 19, можуть мати платні версії з додатковими функціями.

Використання ефективних засобів реалізації може позитивно вплинути на продуктивність розробки та розгортання проекту. Kotlin та Spring Boot є модульними і мають просту конфігурацію, що дозволяє розробникам швидко створювати та розгортати функціонал. Засоби розробки, такі як IntelliJ IDEA, надають розширені функціональні можливості, що можуть сприяти підвищенню продуктивності розробників.

Використання потужних інструментів та фреймворків, таких як Kotlin, Spring Boot і Spring MVC, може зменшити час, необхідний для розробки проекту. Ці засоби надають широкий набір готових компонентів і функціоналу, що дозволяє прискорити процес розробки. Крім того, використання IntelliJ IDEA або іншої інтегрованої середовища розробки (IDE) може полегшити написання коду, відлагодження та тестування програмного забезпечення, що впливає на загальні строки розробки.

Kotlin, Spring Boot і Spring MVC мають активні спільноти розробників та широку підтримку. Вже не кажучи про величезну спільноту (мільйони) веб-розробників. Це означає, що ви зможете знайти велику кількість документації, підручників, статей та форумів, які допоможуть вам вирішити проблеми які виникають у процесі розробки.

**РОЗДІЛ 2. РОЗРОБКА БАЗИ ДАННИХ**

**2.1 Опис моделі данних**

Модель даних – це деяка абстракція, яка, будучи застосовна до конкретних даних, дозволяє користувачам трактувати їх уже як інформацію, тобто відомості, що містять не тільки дані, але й взаємозв’язок між ними.[3]

В нашому випадку можна сказати що модель данних – це база данних, таблиці в цій базі, та взаємозв’язок між ними.

Основні данні які потрібні для того щоб забронювати бокс , це данні про сам бокс, данні про бронювання, та данні про клієнта . Також для кращої роботи інформаціїної системи потрібно мати деякі данні про працівників. Таки чином було складено наступні макети таблиць:

* BOXES: містить інформацію про бокси. Клонки: ID (унікальний ідентифікатор бокса), CODE (код бокса, щоб полегшити його розпізнавання), SIZE (розмір покса), BOOKED (позначає чи бокс зараз знаходиться під бронюванням чи використанням), Price ( місячна вартість оренди бокса), LocationID (ідентифікатор який посилається на місця рощташування бокса).
* CLIENTS: містить інформацію про клієнтів. Колонки ID ( унікальний ідентифікатор клієнта), NAME (ім'я клієнта), SURNAME (прізвище клієнта), PHONE (номер телефону клієнта), REGULAR ( позначає, чи є клієнт постійним), EMAIL (адреса електронной скриньки клієнта).
* EMPLOYEE: містить інформацію про співробітників. Колонки: ID (унікальний ідентифікатор співробітника), NAME (ім'я співробітника), SURNAME (прізвище співробітника), POSITION (посада співробітника), PHONE (телефонний номер співробітника), SALARY (заробітна плата співробітника), ManagerID (ідентифікатор керівника співробітника), PASSWORD (пароль співробітника, використовується для роботи на сайті).
* Locations: містить інформацію приміщеяння з боксами. Колонки: ID (унікальний ідентифікатор приміщення ), BOXES\_COUNT (кількість скриньок у приміщенні), LOCATION\_Address (адреса приміщення)

Також за допомогою онлайн інструменту DBdesigner[3] було створено макет БД (рис. 2.1).

A screenshot of a computer

Description automatically generated with low confidence

Рис. 2.1 Макет майбутньої бази данних

**2.2 Нормалізація відношень**

Нормалізація бази даних або нормалізація SQL допомагає нам згрупувати пов'язані дані в одну єдину таблицю. Будь-які атрибутивні дані або опосередковано пов’язані дані розміщуються в різних таблицях, і ці таблиці пов’язані логічним взаємозв’язком між батьківською та дочірньою таблицями.

1NF (перша нормальна форма)- за визначенням, сутність, яка не має жодних повторюваних стовпців або груп даних, може бути названа Першою нормальною формою. У першій нормальній формі кожен стовпець унікальний.

2NF (друга нормальна форма) - за визначенням, сутність, яка дорівнює 1NF, та один з її атрибутів визначається як первинний ключ, а решта атрибутів залежать від первинного ключа.

3NF (третя нормальна форма) - за визначенням, таблиця вважається третьою нормаллю, якщо таблиця (сутність) вже у другій нормальній формі, а стовпці таблиці (сутності) неперехідно залежать від первинного ключа.

Також існують інші нормальні форми такі як: Звичайна форма Бойса-Кодда (3,5 нормальна форма), 4NF (4 нормальна форма) та 5NF (5 нормальна форма) [4].

Так як база данних яку потрібно нормалізувати є не великою (за кільікстю таблиць та зв’язків між ними, 5 таблиць) можна обмежетися 3NF.

Основні кроки нормалізації включають перевірку кожної таблиці на відповідність нормальним формам.

Перша нормальна форма (1NF): У наших таблицях ми використовуєте атомарні значення для кожної клітинки, а також встановлюємо первині ключі для кожної таблиці. Таким чином, таблиці вже задовольняють першу нормальну форму.

Друга нормальна форма (2NF): Для досягнення другої нормальної форми ми повинні забезпечити, щоб кожен стовпець в таблицях залежав від усього первинного ключа, а не від його частини. Якщо подивитися на наші таблиці, то можна зробити висновок, що всі стовпці залежать від цілого первинного ключа кожної таблиці. Тому дані таблиці задовольняють другу нормальну форму.

Третя нормальна форма (3NF): Третя нормальна форма вимагає, щоб кожен стовпець в таблиці залежав від первинного ключа, а не від інших незалежних стовпців. Проаналізувавши таблиці, стає очевидно, що вони також задовольняють третю нормальну форму.

Отже, можна сказати що всі таблиці задовольняють першу, другу і третю нормальні форми (1NF, 2NF і 3NF).

**2.3 Визначення типів данних**

В SQL Server кожній колонці, локальній змінній, виразу та параметру присвоєно певний тип даних. Тип даних є атрибутом, що визначає, якого типу дані можуть зберігатися в об'єкті: цілі числа, символи, дані у грошовому форматі, мітки часу та дати, двійкові рядки тощо.

SQL Server надає набір системних типів даних, які визначають всі можливі типи даних для використання в SQL Server.

Типи даних в SQL Server об'єднані в наступні категорії:

* Точні числа
* Символьні рядки в Юнікоді
* Приблизні числа
* Двійкові дані
* Дата і час
* Інші типи даних
* Символьні рядки [5]

Основні типи які можна використати для нашої бази данних це:

* int – для цілочисельних значень
* nchar – для зберігання юнікод символів (для коректного відображення українських символів)
* varchar – для зберігання латинських символів
* binary – для зберігання булевих значень
* date – для зберігання дати

Отже, тепер можна проаналізувати таблиці та вибрати необхідні типи данних для їх стовпців.

Для всіх ID (зокрема LocationID з таблиці BOXES; ManagerID з таблиці EMPLOYEE; CLIENTID, EMPLOYEEID, BOXID, з таблиці ORDERS) можна використати тип int

Таблиця BOXES:

* CODE, SIZE: тип nchar(20) для українського тектсу до 20 символів.
* BOOKED: тип binary(1) для зберігання булевого значення (так/ні)
* Price: тип int

Таблиця CLIENTS:

* NAME: тип nchar(30)
* SURNAME: тип nchar(30)
* PHONE: тип varchar(15) (для зберігання номеру - телефону нам не прийдеться проводити ніяких математичних операцій з ним тому його можна зберігати як текст, щоб не нароком під час роботи інформаційної системи він був використаний як число)
* REGULAR: тип binary(1) для зберігання булевого значення (так/ні)
* EMAIL: тип varchar(50)

Таблиця EMPLOYEE:

* NAME: тип nchar(30)
* SURNAME: тип nchar(30)
* POSITION: тип nchar (30) (Для зберіганя назви посади українською мовою)
* PHONE: тип varchar(15) (так само як PHONE у таблиці CLIENTS)
* SALARY: тип int
* PASSWORD: тип varchar(32) (для зберігання паролю зашифрованого за допомогою md5)

Таблиця Locations:

* BOXES\_COUNT: int(4)
* LOCATION\_Address: тип nchar(100) (для зберігання адреси українською мовою

Таблиця ORDERS:

* START\_DATE, END\_DATE: тип date (для зберігання дати)
* SUM: тип int
* STATUS: тип nchar(20) ( для зберігання статусу замовлення українською мовою)

Ці типи даних були вибрані на підставі природи даних, які будуть зберігатися в кожному стовпці таблиці, та врахування вимог до обмежень довжини або типу даних.

**2.4 Обмеження цілісності даних**

Терміном цілісність даних позначають достовірність і точність інформації, що зберігається в базі. Цілісність досягається забезпеченням відповідності даних певним додатковим обмеженням, крім тих, які накладаються схемою бази на структуру даних та їхні типи.

Обмеження цілісності — це правила, які обмежують усі можливі стани бази даних, а також переходи з одного стану в інший. Таким чином, обмеження цілісності визначають множину «допустимих» станів і переходів між ними. База даних перебуває в цілісному стані, якщо вона відповідає всім визначеним для неї вимогам цілісності.[6]

Не будемо розглядати всі обмеження, поглянемо тількі на ті які будемо використовувати.

Обмеження PRIMARY KEY (первинний ключ) унікально ідентифікує кожен запис у таблиці. Первинні ключі повинні містити унікальні значення і не можуть містити значення NULL. У таблиці може бути лише один первинний ключ, і в цьому ключі можуть бути одиничні або кілька стовпців (полів).

Обмеження UNIQUE забезпечує, що всі значення в стовпці були різними. Як обмеження UNIQUE, так і обмеження PRIMARY KEY гарантують унікальність для стовпця або набору стовпців. Обмеження PRIMARY KEY автоматично має обмеження UNIQUE. Отже, ви можете мати багато обмежень UNIQUE для однієї таблиці, але лише одне обмеження PRIMARY KEY для однієї таблиці.

За замовчуванням, стовпець може містити значення NULL. Обмеження NOT NULL забезпечує, що стовпець не може приймати значення NULL. Це вимагає, щоб поле завжди містило значення, що означає, що ви не можете додати новий запис або оновити існуючий запис без додавання значення до цього поля.

Зовнішній ключ (FOREIGN KEY) використовується для запобігання діям, які можуть знищити зв'язки між таблицями. Зовнішній ключ (FOREIGN KEY) є полем (або набором полів) у одній таблиці, яке посилається на первинний ключ (PRIMARY KEY) в іншій таблиці. Таблиця з зовнішнім ключем називається дочірньою таблицею (child table), а таблиця з первинним ключем називається посиланою або батьківською таблицею (referenced or parent table). [7]

Подамо обмеження які ми будемо застосовувати у вигляді таблиці(рис.2.2).

Рис.2.2. Обмеження для таблиць

Обмеження унікальності UNIQUE: Використовується для стовпців CODE у таблиці BOXES, для стовпців PHONE у таблицях CLIENTS та EMPLOYEE, для стовпця EMAIL у таблиці CLIENTS та LOCATION\_ADRESS у таблиці LOCATIONS. Ці обмеження гарантують, що значення у відповідних стовпцях будуть унікальними.

Первинний ключ (PRIMARY KEY): Використовується для стовпців ID у таблицях BOXES, CLIENTS, EMPLOYEE, Locations та ORDERS. Ці обмеження встановлюють первинний ключ для відповідних таблиць, що гарантує унікальність та ідентифікацію записів.

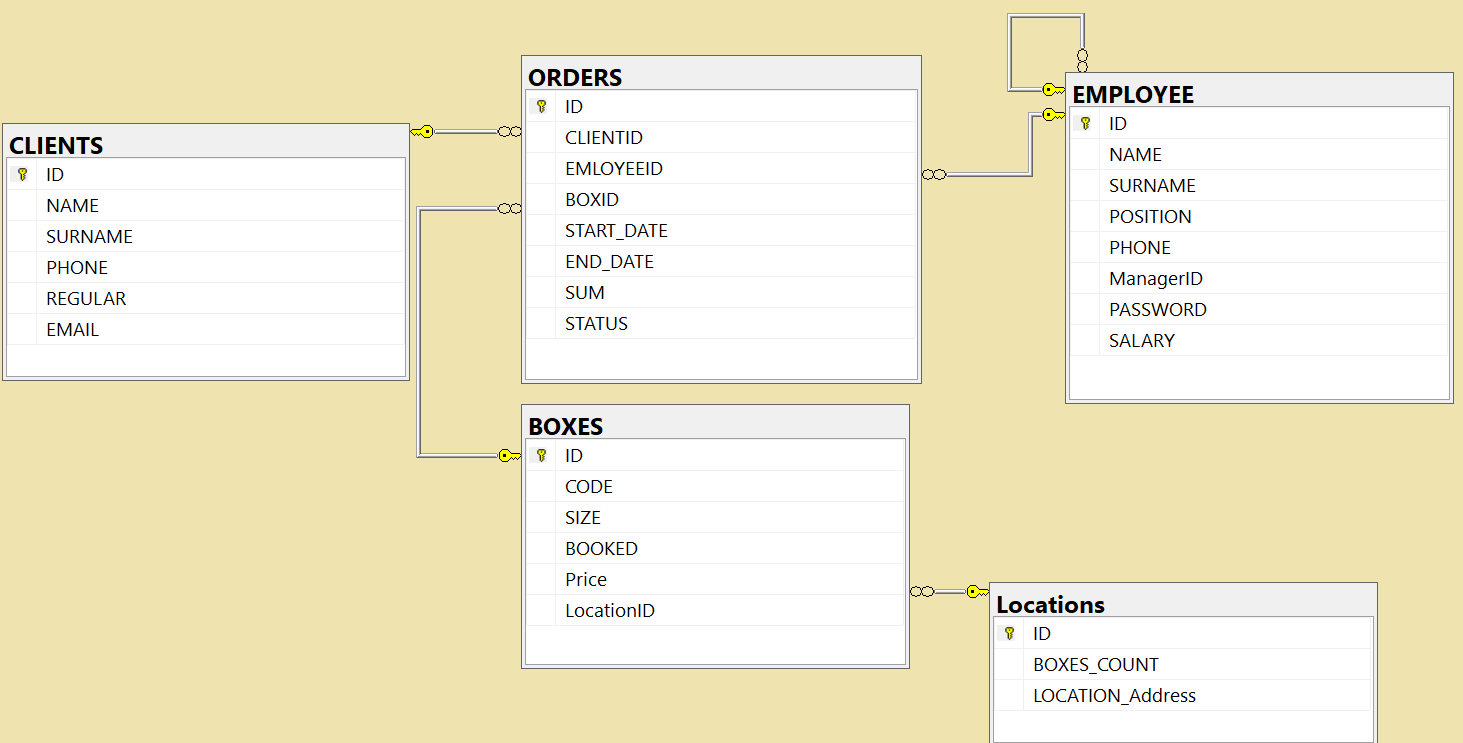
NOT NULL використовується для всіх ствопців у всіх таблицях, крім двох стовпців у EMPLOYEE - це стовпці MANAGERID та PASSWORD (деякі працівники можуть не мати керівника; не у всіх працівників може бути пароль, бо вони можуть не використовувати частину інформаційної системи де передбачена авторизація). Обмеження NOT NULL гарантує, що значення у цих стовпцях завжди будуть присутніми.

**2.5 Реалізація SQL-скрипту**

Скрипт для створення бази данних можна переглянути в Додатку TODO(“додаток”)

Остаточна ER-діагарма подана на рисунку 2.3

Скрипт для наповнення БД данними, та скрипти які перекривають бізнес потреби ( USE CASE) можна знайти на віддаленому репозиторії у гілці sql [8]. Там також є дуплікат скрипта створення БД.

Рис.2.3 ER-діаграма остаточного вигляду БД

**РОЗДІЛ 3. РОЗРОБКА ВЕБ-ДОДАТКУ**

**3.1 Структура веб-сайту**

Структура веб-сайту грає ключову роль у забезпеченні зручного та ефективного використання сайту. В даному розділі ми розглянемо основне призначення сайту і перелічимо основні сторінки, а також коротко описуємо їх функціональне призначення.

Основне призначення веб-сайту: головна мета цього веб-сайту полягає у наданні інформації про компанію та надійному бронюванні складських боксів. Він дозволяє відвідувачам ознайомитися з інформацією про компанію та забезпечує простий спосіб бронювання боксів через онлайн-форму. Крім того, сайт надає можливість зареєстрованим користувачам переглянути свої замовлення та працівникам компанії обробити нові замовлення.

Основні сторінки веб-сайту та їх функціональне призначення:

1. Головна сторінка: Головна сторінка є стартовою точкою для відвідувачів, які переходять на домен сайту. Вона містить коротку інформацію про компанію, включає логотип та слоган, щоб відобразити її ідентичність. На цій сторінці також знаходиться центральна кнопка або посилання, яке спрямовує відвідувачів на сторінку бронювання.
2. Сторінка бронювання: Ця сторінка містить форму, яку клієнт повинен заповнити для перегляду доступних для бронювання складських боксів. Форма може включати поля для введення розміру боксу, тривалості оренди та інших додаткових параметрів. Після заповнення форми клієнт може надіслати запит на бронювання.
3. Сторінка мої замовлення: Ця сторінка призначена для зареєстрованих користувачів, які мають можливість авторизуватися на сайті. Після успішної авторизації користувач може переглянути свої замовлення на складські бокси. Інформація про кожне замовлення може включати деталі, такі як розмір боксу, тривалість оренди та статус замовлення.
4. Сторінка співробітника (Employee): Ця сторінка призначена для співробітників компанії, які відповідають за обробку замовлень. На цій сторінці співробітник може переглядати нові замовлення, які ще не оброблені, і виконувати необхідні дії, такі як підтвердження бронювання або надсилання сповіщень клієнтам.

Ця структура веб-сайту з 4 основними сторінками забезпечує зручний та ефективний спосіб взаємодії з веб-сайтом. Вона дозволяє відвідувачам отримати інформацію про компанію та легко здійснити бронювання, а зареєстрованим користувачам та співробітникам компанії забезпечує додаткові можливості управління замовленнями та обробки.

* 1. **Макети сторінок веб-сайту**

Макети сторінок було зроблені за допомогою HTML, CSS, та фреймворка Bootstrap.

Bootstrap - це безкоштовний фреймворк для швидкого та простого cтворення веб-інтерфейсів. Bootstrap включає HTML- та CSS-шаблони для дизайну, такі як типографіка, форми, кнопки, таблиці, навігація, модальні вікна, каруселі зображень та багато інших, а також необов'язкові плагіни JavaScript. Bootstrap також надає можливість легко створювати адаптивні дизайни, які добре виглядають на різних пристроях і розмірах екранів.

Bootstrap був розроблений Марком Отто і Джейкобом Торнтоном в компанії Twitter і був випущений як відкрите програмне забезпечення у серпні 2011 року на платформі GitHub. У червні 2014 року Bootstrap став найпопулярнішим проектом на GitHub.

Переваги Bootstrap:

* Легкість використання: Будь-хто з базовими знаннями HTML і CSS може почати використовувати Bootstrap.
* Адаптивні можливості: Адаптивні CSS-стилі Bootstrap адаптуються до мобільних телефонів, планшетів і настільних комп'ютерів.
* Мобільний підхід: У Bootstrap мобільно-орієнтовані стилі є частиною основного фреймворку.
* Сумісність з браузерами: Bootstrap сумісний з усіма сучасними браузерами (Chrome, Firefox, Internet Explorer, Edge, Safari та Opera).[9]

Макети сторінок веб-сайту розроблені в стилі сучасного та мінімалістичного дизайну. Вони мають збалансовану структуру і привертають увагу користувачів до ключової інформації.

Кожна сторінка має адаптивну розмітку, що забезпечує коректне відображення на різних пристроях і розмірах екранів. Верхній колонтитул (або header) містить компанії та навігаційне меню. Основний вміст розташовується у центрі сторінки, а нижній колонтитул (footer) містить додаткову інформацію та посилання.

Для фону сторінок веб-сайту використані темнуваті і нейтральні кольори, наприклад. Це створює чисте та сучасне візуальне середовище для презентації інформації.

Тепер можна проаналізувати кожну сторінку окремо.

Головна сторінка.

Як і в кожної сторінки в неї header та footer.

У верхній частині сторінки розміщені логотип компанії та слоган. Логотип представлений зображенням, яке відображається на сторінці у формі круга. Слоган написаний у вигляді двох заголовків, що розташовані нижче логотипу.

Заднє фонове зображення сторінки встановлено з використанням CSS. Воно розтягнуте на весь екран та додає візуальний ефект. Також використовується прозорий темний шар над зображенням, що дозволяє зробити текст читабельним.

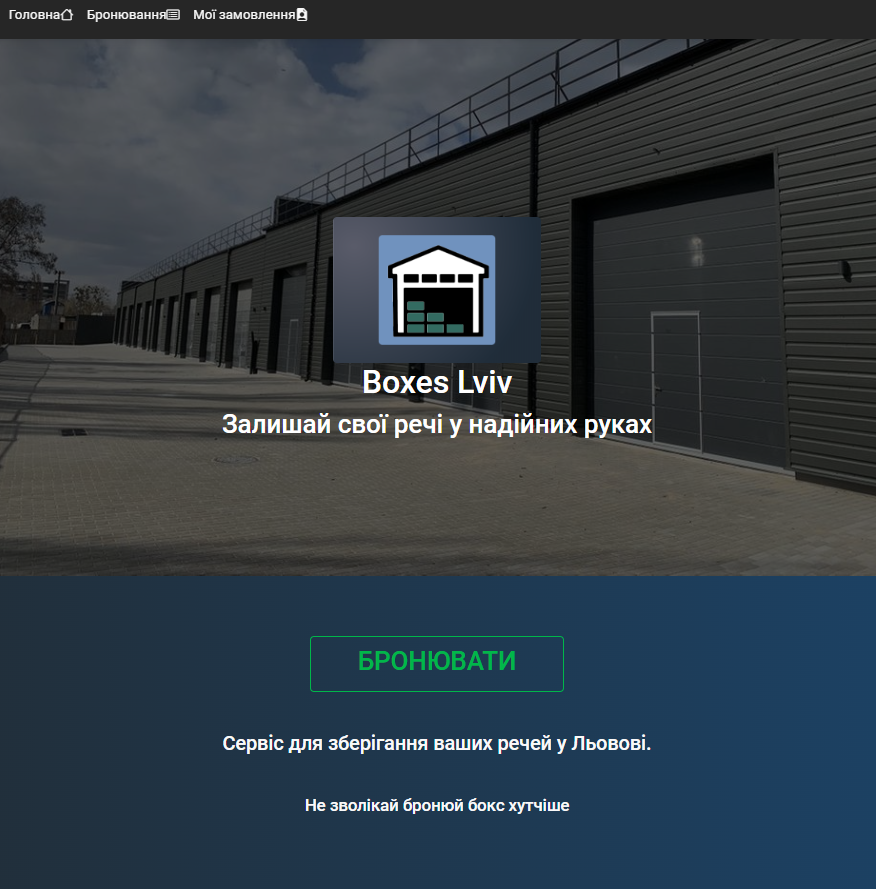
Нижче слогану розміщена кнопка з посиланням "Бронювати". Вона виглядає як прозора обведена зеленою рамкою кнопка з великим текстом. Ця кнопка спрямовує користувача на сторінку бронювання. Нижче кнопки розташований короткий опис сервісу. Він містить заголовок "Сервіс для зберігання ваших речей у Львові" та додатковий підзаголовок із закликом до дії “Не зволікай бронюй бокс хутчіше”. Це все можна побачити на рисунку 2.1

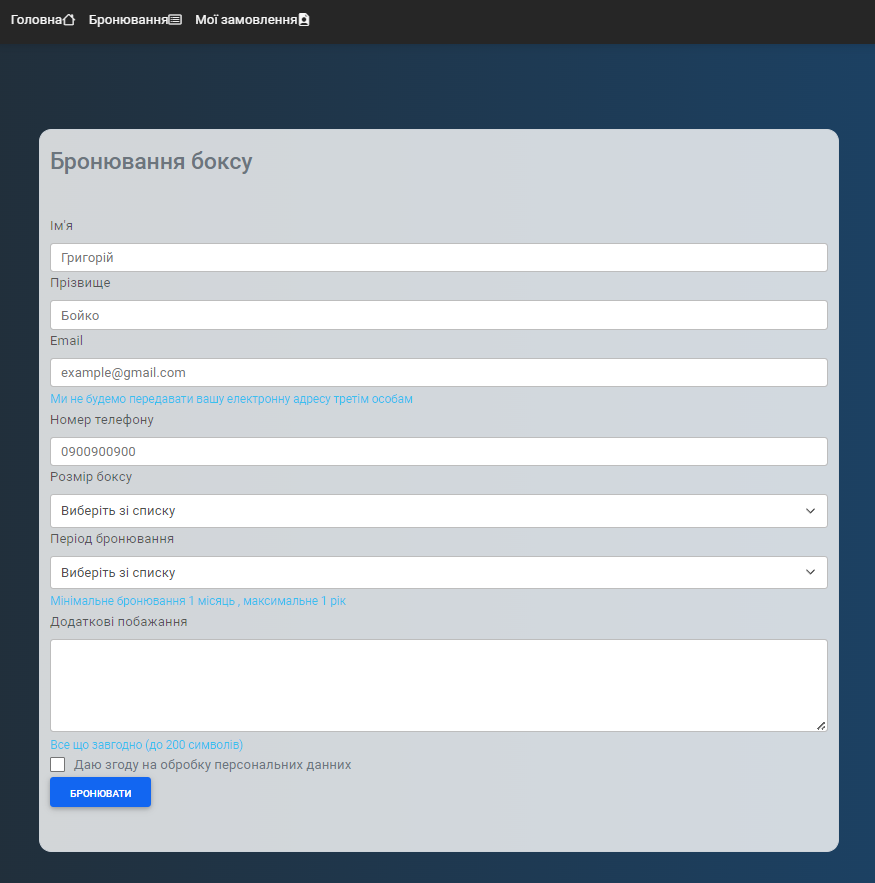
Рис. 2.1 Головна сторінка

Сторінка Бронювання

У верхній частині сторінки розташований header з меню як і всюди. Посередені сторінки розташований контейнер або блок з основним контеном. заголовок "Бронювання боксу", який представлений у вигляді заголовка третього рівня . На сторінці розташована форма бронювання, яка складається з наступних полів:

* Поля для введення імені, прізвища, електронної адреси та номера телефону: Ці поля дозволяють користувачеві ввести свої персональні дані.
* Поля для вибору розміру боксу та періоду бронювання: Користувач може обрати розмір боксу зі списку опцій, а також вибрати тривалість бронювання зі списку опцій.
* Поле для додаткових побажань: Користувач може ввести додаткові побажання щодо бронювання.
* Поле з підтвердженням згоди на обробку персональних даних: Користувач може позначити це поле, щоб дати згоду на обробку своїх персональних даних.

Після заповнення форми, користувач може натиснути кнопку "Бронювати", яка розміщена нижче форми. Ця кнопка ініціює процес бронювання. І в низу сторінки міститься footer. Вигляд сторінки на рисунку 3.2

Рис.3.2 Сторінка Бронювання

Після того як користувач натисне кнопку Бронювати вигляд сторінки зміниться . В контейнері замість форми з’явиться таблиця з доступними боксами для бронювання (зверху шапка таблиці, знизу по троє в ряд доступні бокси з кнопкою для бронювання, очевидно що кількість рядів залежить від кількості боксів ) . Змінена сторінка показана на рисунку 3.3

A screenshot of a computer

Description automatically generatedРис. 3.3 Змінена сторінка бронювання

Сторінка Мої замовлення

Сторінка Мої замовлення як і всі містить header і footer. Посередені сторінки також розташований головний контейнер , а у ньому тепер тільки оде поле для вводу – це мейл і кнопка “Отримати Код” ,при натисканні на яку користувача на пошту відправляється код для входу (рис. 3.4) .

A screenshot of a computer

Description automatically generated with medium confidenceРис. 3.4 Сторінка Мої замовлення

A screenshot of a computer

Description automatically generated with medium confidenceПісля натискання на кнопку сиорінка змінюється. З’являється додаткове поле для вводу Код, також поле емейл стає не активним та обводиться зеленим кольором, колір кнопки зміюється на зелену а текст в ній на “Вхід”. (рис.3.5).

Рис. 3.5 Змінена Мої замовлення

A picture containing text, screenshot, electric blue, line

Description automatically generatedПісля того як користувач залогінився сторінка знову змінюється. В головному контейнері з’являється інформація про бронювання користувача у вигляді списку. (рис. 3.6)

Рис. 3.6 Бронювання користувача на сторнці мої замовлення

І на сам кінець сторінка Співробітник.

A screenshot of a computer

Description automatically generatedСторінка співробтника дуже подібна на сторінку Мої замовлення. Вона містить header та footer , основний контейнер посередині в якому є уже два поля для вводу Телефон та Пароль і кнопку “Вхід”.(рис.3.7)

Рис.3.7 Сторінка співробітника

Після входу, контент в основному контейнері змінюється. Там з’являються нові замолення також у вигляді списку та після кожного замовлення з’являється дві кнопки червона “Скасувати” та зелена “Прийняти”.(рис.3.8)

A screenshot of a computer

Description automatically generatedРис.3.8 Сторінка Співробітника після входу

Важливо сказати, звичайні користувачі не зможуть потрапити на цю сторінку через інтерфейс сайту, бо на неї можна зайти тільки через URL.

Html розмітку всіх сторінок можна знайти на віддаленому репозиторії

* 1. **Програмування серверної частини**

Розробка серверної частини використовувала фреймворк Spring Boot, Spring MVC та шаблонізатор Thymeleaf. Основні модулі, що були створені при розробці серверної частини, включають:

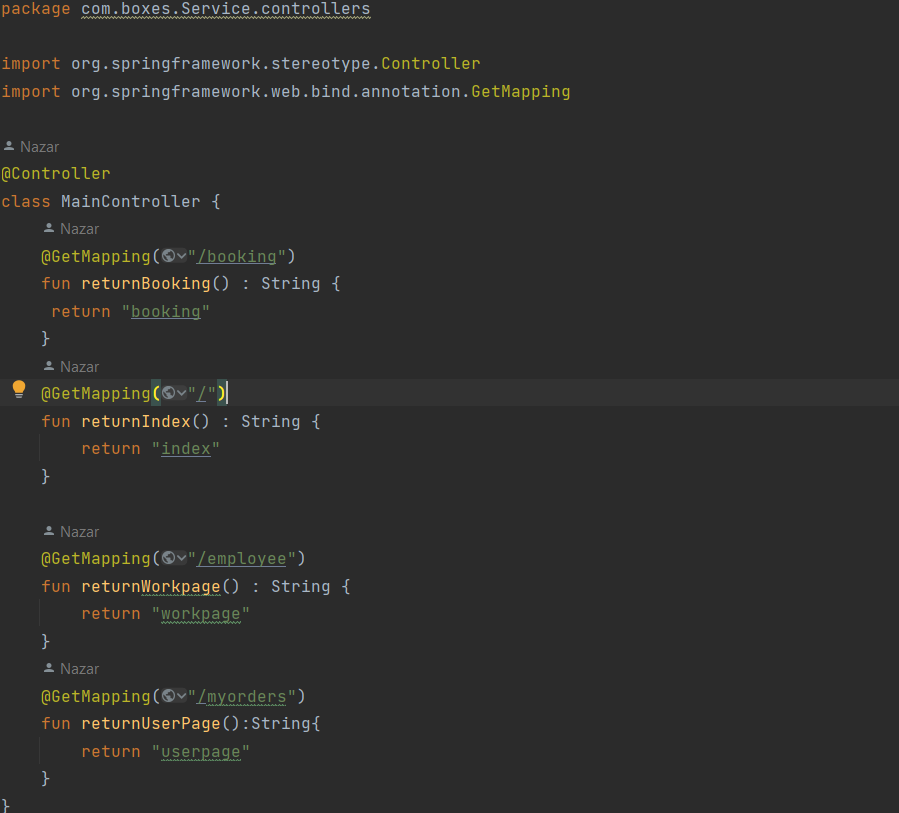
1. Контролери (Controllers): Контролери в Spring MVC відповідають за обробку HTTP-запитів та управління відповідями. Вони визначають методи, які будуть виконуватися при отриманні певного запиту. Наприклад, для обробки запиту на бронювання боксу може бути створено контролер з методом, що приймає дані форми бронювання та виконує необхідну логіку для збереження бронювання. Приклад контроллеру на рисунку 3.8.

Рис.3.9 Головний контроллер , він відповідає за маршрутизацію сторінок

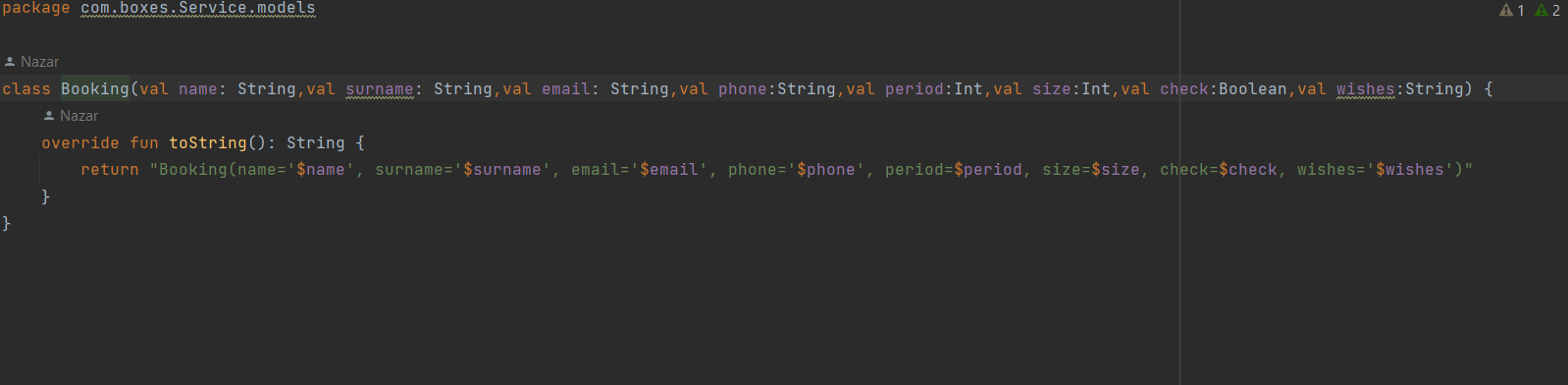
1. Модель (Model): Модель відображає дані, з якими працює програма. У нашому випадку модель може містити класи, що представляють бронювання, користувачів, розміри боксів тощо. Ці класи можуть мати анотації для валідації даних та забезпечення правильної мапінгування з форми бронювання. Приклад моделі можна побачити на рисунку 3.10

Рис.3.10 Модель Booking, що відповідає за бронювання

1. Перегляди (Views) з використанням Thymeleaf: Thymeleaf є шаблонізатором, який дозволяє створювати динамічні сторінки з використанням HTML та вбудовувати дані з моделі у ці сторінки. У них можуть бути використані вирази Thymeleaf для ітерації по списку бронювань, відображення даних користувача тощо. Розмітку цих сторінок не озброєним оком важко відрізнити від звичайних html сторінок.

Це лише декілька основних модулів, що були створені при розробці серверної частини з використанням Spring Boot, Spring MVC та Thymeleaf. Також були створенні такі модулі як сервіси , та компоненти.

Увесь код серверної частини з усіма модулями можна преглянути на віддаленому репозиторії у гілці dev\_server.

* 1. **Програмування клієнтської частини**

При розробці клієнтської частини сайту (frontend) для даного проекту використовувалися JavaScript та ряд бібліотек, таких як jQuery та js-snackbar.

JavaScript є мовою програмування, яка використовується для створення інтерактивних елементів на веб-сторінках. В даному проекті він використовувався для реалізації різних функцій, таких як валідація форм, маніпуляції з DOM (Document Object Model) та здійснення AJAX-запитів до сервера.

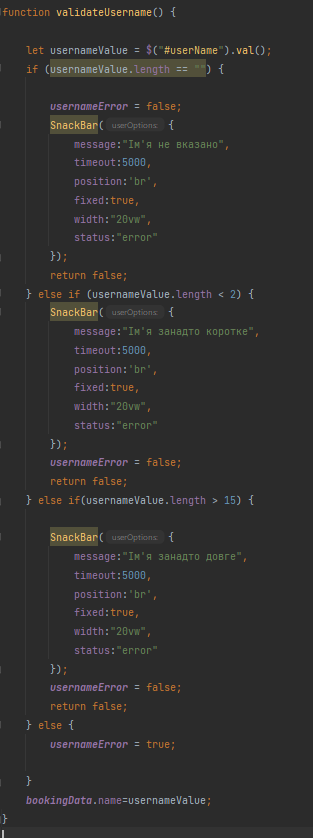
Одна з бібліотек, яку використовували для спрощення роботи з JavaScript, - це jQuery. Вона надає набір готових функцій та методів, що спрощують маніпулювання DOM, роботу з подіями, валідацію форм та здійснення AJAX-запитів.

Наприклад, для валідації форм можна використовувати методи jQuery, які дозволяють перевіряти правильність заповнення полів форми перед їх надсиланням на сервер. Також, з використанням jQuery можна змінювати вміст HTML-сторінки динамічно, без перезавантаження всієї сторінки. Приклад використання jQuery для валідації форми на рисунку 3.11.

Крім того, була використана бібліотека js-snackbar для виведення сповіщень про валідацію форм та запити до сервера. Ця бібліотека надає простий спосіб створення сповіщень (snackbar) - маленьких повідомлень, які з'являються знизу екрану. Вона дозволяє зручно виводити повідомлення про помилки валідації полів форми або повідомлення про результати запитів до сервера.

Загалом, використання JavaScript та бібліотек, таких як jQuery та js-snackbar, дозволяють реалізувати інтерактивність та зручність взаємодії користувача з веб-сайтом, забезпечуючи швидкі та плавні динамічнізміни на сторінці, валідацію форм та виведення сповіщень.

Приклад валідації форми за дромогою JQuery та js-snacbar можна побачити на рисунках 3.11-3.12

Рис.3.11 Функція для валідації поля Ім’я користувача

A screenshot of a computer

Description automatically generated

Рис.3.12 Сповіщення від Js-snackbar про те що форма не правильно заповнена

* 1. **Розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті**

Для розміщення веб-сайту, розробленого на фреймворку Spring Boot, на локальному віртуальному середовищі або в Інтернеті, необхідно виконати деякі кроки. Переконайтеся, що у вас вже встановлена Java, оскільки вона є передумовою для коректної роботи Spring Boot. Крім того, вам потрібно відкрити порт 8080 для віртуальної машини Java, щоб вона могла приймати вхідні HTTP або HTTPS запити.

Ось коротка інструкція по рзгортанню та запуску проекту Spring Boot який використовує Gradle:

* 1. Для початку потрібно завантажтити проект Spring Boot на віртуальну машину чи сервер. (можна клонувати його з віддаленого репозиторію, чи завантажити як архів через ftp і т.п.)
  2. Відкрийте командний рядок або термінал та виконайте команду ./gradlew build all для збирання проекту та створення JAR-файлу.
  3. Виконайте команду java -jar назва\_файлу.jar, де назва\_файлу - назва JAR-файлу вашого проекту (зазвичай він знаходиться по такому шляху: ”коронева\_папка\_проекту/buid/libs/назва\_файлу.jar”)( Після успішного запуску, веб-сайт буде доступний за адресою localhost:8080)
  4. Якщо ви використовуєте операційну систему Windows, перейдіть до налаштувань брандмауера та додайте правило для дозволу вхідних з'єднань на порт 8080.
  5. Якщо ви використовуєте операційну систему Linux, скористайтеся відповідними командами, такими як iptables, для відкриття порту 8080.

Після успішного завершення цих кроків, ваш веб-додаток, розроблений на Spring Boot, буде розміщено на локальному віртуальному середовищі або буде доступний у мережі Інтернет за адресою сервера на порту 8080.

Також потрібно розгорнути сервер бази данних ( в нашому випадку MS SQL)

Опис кроків для інсталяції Microsoft SQL Server:

* 1. Завантажте встановлювач Microsoft SQL Server з офіційного веб-сайту Microsoft.
  2. Запустіть встановлювач SQL Server і слідуйте інструкціям майстра установки.
  3. Оберіть тип установки, налаштування сервера та баз даних відповідно до вашого випадку використання.
  4. Під час установки, зверніть увагу на порт, на якому запущений SQL Server. За замовчуванням, це зазвичай порт 1433. (його потрібно відкрити таким самим способом як і порт 8080 для Spring Boot)
  5. У вашому проекті Spring Boot налаштуйте параметри підключення до бази даних, потрібно вказати URL підключення до SQL Server, ім'я користувача та пароль для доступу до бази данних.

Після успішного виконання цих кроків, веб-аплікація буде атоматично підключатися до бази данних при необхідності.

**ВИСНОВКИ**

Після проведення аналізу вимог та розробки інформаційної системи для бізнесу з оренди складських боксів, що складається з веб-додатку, серверної частини, а також бази даних, можна зробити наступні розгорнуті висновки:

Аналіз вимог є першим кроком у розробці будь-якої інформаційної системи. Постановка завдання визначає основні мети та функціональні вимоги до системи. У цьому випадку, головним завданням було створення інформаційної системи для бізнесу з оренди складських боксів. Це вимагало аналізу потреб користувачів та розробки моделі варіантів використання веб-сайту, яка враховує основні функціональні вимоги системи.

Другим кроком була розробка бази даних. Вона починається з опису моделі даних, де були визначені сутності та їх взаємозв'язки. Нормалізація відношень забезпечує оптимальну структуру бази даних та цілісність інформації. Визначення типів даних відображає специфіку даних, що будуть зберігатися, та впливає на ефективність операцій з базою даних. Також були встановлені обмеження цілісності даних, що допомагають забезпечити правильність та унікальність інформації. Було створенно багато запитів до бази данних для задоволення певних бізнес потреб . Ці запити стали одним із фундаментальних елементів інформаційної системи.

Після розробки бази даних було приступлено до розробки веб-додатку. Була створена структура веб-сайту, яка включала основні сторінки та їх взаємозв'язки. Макети сторінок веб-сайту були розроблені, щоб відповідати вимогам та дизайну системи.

Програмування серверної частини веб-додатку було здійснено за допомогою Spring Boot та Spring MVC. Це забезпечило можливість реалізації основної функціональності системи та взаємодії з базою даних. Важливою складовою було програмування клієнтської частини, що дозволило взаємодіяти з сервером та відображати дані на сторінках веб-сайту.

Останнім кроком було розміщення веб-сайту на локальному віртуальному середовищі або в Інтернеті. Це дозволило доступ до системи користувачам та бізнесу з будь-якого пристрою, що має підключення до мережі.

У цілому, розробка інформаційної системи для бізнесу з оренди складських боксів вимагала комплексного аналізу вимог, розробки бази даних та веб-додатку. Використання Spring Boot та Spring MVC сприяло швидкій реалізації серверної частини системи, а розміщення веб-сайту забезпечило доступність інформації з будь-якого пристрою, також багато бізнесових потреб вдалося перекрити за рахунок правильно SQL запитів до бази данних . Результатом роботи є інформаційна система, яка допомагає бізнесу ефективно керувати процесами оренди складських боксів та забезпечує зручний інтерфейс для користувачів.

Список використаних джерел

1. https://kotlinlang.org

2. https://spring.io/projects/spring-boot

3. dbdesigner.net

4. Підручник з нормалізації бази даних: Приклади 1NF 2NF 3NF BCNF – Інший URL: <https://uk.myservername.com/database-normalization-tutorial>

5.Типи данних (Transact-SQL) URL: <https://learn.microsoft.com/ru-ru/sql/t-sql/data-types/data-types-transact-sql?view=sql-server-ver16>

6. Організація баз даних та знань Тема 8 - Цілісність та безпека даних Конспект лекції URL: <https://elearning.sumdu.edu.ua/free_content/lectured:89b3d175c06a6b137e410cb14821d0e94549ad5a/20151030211833/44700/index.html>

7. <https://www.w3schools.com/sql>

8. Віддалений репозиторій <https://github.com/NazarMazur0/Boxes>

9. <https://www.w3schools.com/bootstrap/bootstrap_get_started.asp>

10.