

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)

магістра

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ МЕТОДІВ

(назва теми)

ПЛАНУВАННЯ ДІЯЛЬНОСТІ ПІЦЕРІЇ

RESEARCH AND SOFTWARE IMPLEMENTATION OF METHODS
FOR PLANNING THE ACTIVITY OF A PIZZERIA

Виконав(ла): студент(ка) 2 курсу, групи КНТ-123м

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

СЛАВНИЙ Б.Ю.

(ПРІЗВИЩЕ та ініціали)

Керівник

СКРУПСЬКИЙ С.Ю.

(ПРІЗВИЩЕ та ініціали)

Рецензент

ПОЛЯКОВ М.О.

(ПРІЗВИЩЕ та ініціали)

2024_____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет _____ КНТ _____
Кафедра _____ програмних засобів _____
Ступінь вищої освіти _____ магістр _____
Спеціальність _____ 121 Інженерія програмного забезпечення _____
(код і найменування)
Освітня програма (спеціалізація) _____ Інженерія програмного забезпечення _____
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф. _____
Сергій СУББОТІН
« _____ » _____ 2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

СЛАВНОГО Богдана Юрійовича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів планування діяльності піцерії. Research and Software Implementation of Methods for Planning the Activity of a Pizzeria

керівник проєкту (роботи) к.т.н., доцент СКРУПСЬКИЙ Степан Юрійович _____,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 7 » жовтня 2024 року № 398

2. Строк подання студентом проєкту (роботи) 01 грудня 2024 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз проблеми та постановка завдань дослідження. 2. Матеріали і методи. 3. Проєктування програмного забезпечення. 4. Основні рішення щодо реалізації програмного забезпечення. 5. Експлуатація, тестування та експериментальне дослідження програми.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів, плакатів)
Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	СКРУПСЬКИЙ С.Ю., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання « 06 » вересня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області	2 тиждень	Розділ 1
3	Дослідження проблеми	3-5 тижні	Розділ 2
4	Проектування та розробка архітектури програми	6 тиждень	Розділ 3
5	Розробка програми	7-8 тижні	Розділ 4
6	Тестування та експериментальне дослідження	9-10 тижні	Розділ 5
7	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	11 тиждень	Додатки
8	Захист роботи	12 тиждень	

Студент(ка)

(підпис) Богдан СЛАВНИЙ
(Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Степан СКРУПСЬКИЙ
(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи магістра:
119 с., 26 рис., 64 табл., 3 дод., 9 джерел.

ПІЦЕРІЯ, ПЛАНУВАННЯ, ЗАМОВЛЕННЯ, ЧАСОВИЙ РЯД, ARIMA.

Об'єкт роботи – процеси підтримки та планування діяльності піцерії.

Предмет роботи – методи планування діяльності піцерії.

Метою роботи є розробка програмного забезпечення підтримки діяльності піцерії для забезпечення планування діяльності піцерії на основі прогнозування кількості проданих піц за певними групами.

Матеріали, методи та технічні засоби: методи прогнозування часового ряду, ARIMA, мова програмування Python, фреймворк Django, система керування базами даних MySQL.

Результати. Розроблене програмне забезпечення призначене для забезпечення замовлення піц, додатків до піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами. Для цього надаються засоби категоризації піц, додаткових продуктів, пошуку піц, вподобання, оцінювання піц та додаткових продуктів. У програмне забезпечення, створене як вебзастосунок піцерії, було інтегровано модель прогнозування кількості проданих піц за групами інгредієнтів.

Висновки. Наукова новизна роботи полягає в методі прогнозування кількості проданих піц за групами інгредієнтів, що характеризується розподілом піц на групи за конкретними інгредієнтами, застосуванням моделі ARIMA, що дозволяє визначати кількість піц з вмістом відповідного інгредієнта, яка буде продана в піцерії на певну дату.

Галузь використання. Окремі піцерії стосовно підтримки роботи з замовленнями та планування подальших замовлень.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 119 pages, 26 figures, 64 tables, 3 appendixes, 9 sources.

PIZZERIA, PLANNING, ORDERING, TIME SERIES, ARIMA.

The object of the work is the processes of support and planning of pizzeria activities.

The subject of the work is methods of planning pizzeria activities.

The purpose of the work is the development of software to support pizzeria activities to ensure planning of pizzeria activities based on forecasting the number of pizzas sold in certain groups.

Materials, methods and technical means: time series forecasting methods, ARIMA, Python programming language, Django framework, MySQL database management system.

The results. The developed software is designed to ensure the ordering of pizzas, applications for pizzas and additional goods in a pizzeria, planning the activity of this pizzeria by forecasting the number of sold pizzas in certain groups. To do this, the means of categorizing pizzas, additional products, searching for pizzas, liking, rating pizzas and additional products are provided. A model for predicting the number of pizzas sold by ingredient groups was integrated into the software, created as a pizzeria web application.

Conclusions. The scientific novelty of the work is in the method of forecasting the number of sold pizzas in groups of ingredients, which is characterized by the distribution of pizzas into groups by specific ingredients, the application of the ARIMA model, which allows to determine the number of pizzas with the content of the corresponding ingredient that will be sold in the pizzeria on a certain date.

Field of use. Individual pizzerias regarding order support and planning of further orders.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз проблеми та постановка завдань дослідження	10
1.1 Функціональні можливості та концепція існуючих програмних застосунків забезпечення діяльності піцерій.....	10
1.2 Визначення рівня створюваного вебзастосунку піцерії.....	13
1.3 Висновки за розділом 1	14
2 Матеріали і методи.....	16
2.1 Задача прогнозування кількості проданих піц за групами інгредієнтів	16
2.2 Метод прогнозування кількості проданих піц за групами інгредієнтів	17
2.3 Висновки за розділом 2	19
3 Проектування програмного забезпечення	21
3.1 Визначення сценаріїв роботи користувачів з вебзастосунком піцерії ..	21
3.2 Вибір засобів розробки вебзастосунку піцерії.....	27
3.3 Проектування структури бази даних.....	28
3.4 Висновки за розділом 3	40
4 Основні рішення щодо реалізації програмного забезпечення	41
4.1 Загальний порядок роботи з замовленнями піц у піцерії	41
4.2 Реалізовані програмні одиниці у складі вебзастосунку піцерії	42
4.3 Висновки за розділом 4	56
5 Експлуатація, тестування та експериментальне дослідження програми	57
5.1 Основне призначення програми	57
5.2 Виконання програми.....	57
5.3 Експериментальне дослідження прогнозування кількості проданих піц за групами інгредієнтів	61
5.4 Висновки за розділом 5	82
Висновки	83
Перелік джерел посилання	84

Додаток А Технічне завдання	85
Додаток Б Текст програми	91
Додаток В Слайди презентації.....	112

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

ARIMA – AutoRegressive Integrated Moving Average;

MAE – Mean Absolute Error;

MSE – Mean Squared Error;

RMSE – Root Mean Squared Error.

ВСТУП

Формування замовлень піц у піцерії через програмні засоби є фактично обов'язковою умовою існування піцерії для досягнення успішних бізнес-результатів. Деякі піцерії можуть використовувати соціальні мережі для представлення меню, можливо формування замовлень, однак для контролю замовлень піц піцерією це достатньо незручно, відповідно можуть виникати проблеми з обслуговуванням, особливо коли таких замовлень піц стає багато. Тому створення вебзастосунку піцерії є важливим. Проте і розвиток такого програмного засобу з врахуванням тих можливостей, які б надали можливість опрацювання наявних даних, є також важливим.

Наявні дані у такій програмі звісно дозволяють проаналізувати, скільки яких саме піц було продано за кожен день роботи піцерії. Також відповідно можна і проаналізувати кількість піц з певними інгредієнтами, які було продано. Однак окрім аналізу статистики за цими параметрами на основі цих даних можна також і виконати прогнозування кількості проданих піц за певними групами. Наявність цього засобу дозволяє виконати планування діяльності піцерії. Тому важливо під час розробки вебзастосунку піцерії надати можливості і замовлення піц у піцерії, і планування діяльності піцерії на основі прогнозування кількості проданих піц за певними групами.

Метою роботи є розробка програмного забезпечення підтримки діяльності піцерії для забезпечення планування діяльності піцерії на основі прогнозування кількості проданих піц за певними групами.

1 АНАЛІЗ ПРОБЛЕМИ ТА ПОСТАНОВКА ЗАВДАНЬ ДОСЛІДЖЕННЯ

1.1 Функціональні можливості та концепція існуючих програмних застосунків забезпечення діяльності піцерій

У світі існує багато піцерій, при цьому не кожна з них представлена відповідним вебзастосунком. Деякі з них детальніше описані нижче.

Мережа піцерій Celentano має відповідний вебзастосунок, де можна обрати відповідну піцерію в залежності від міста, після чого відкривається вебзастосунок конкретної піцерії (рис. 1.1) [1].

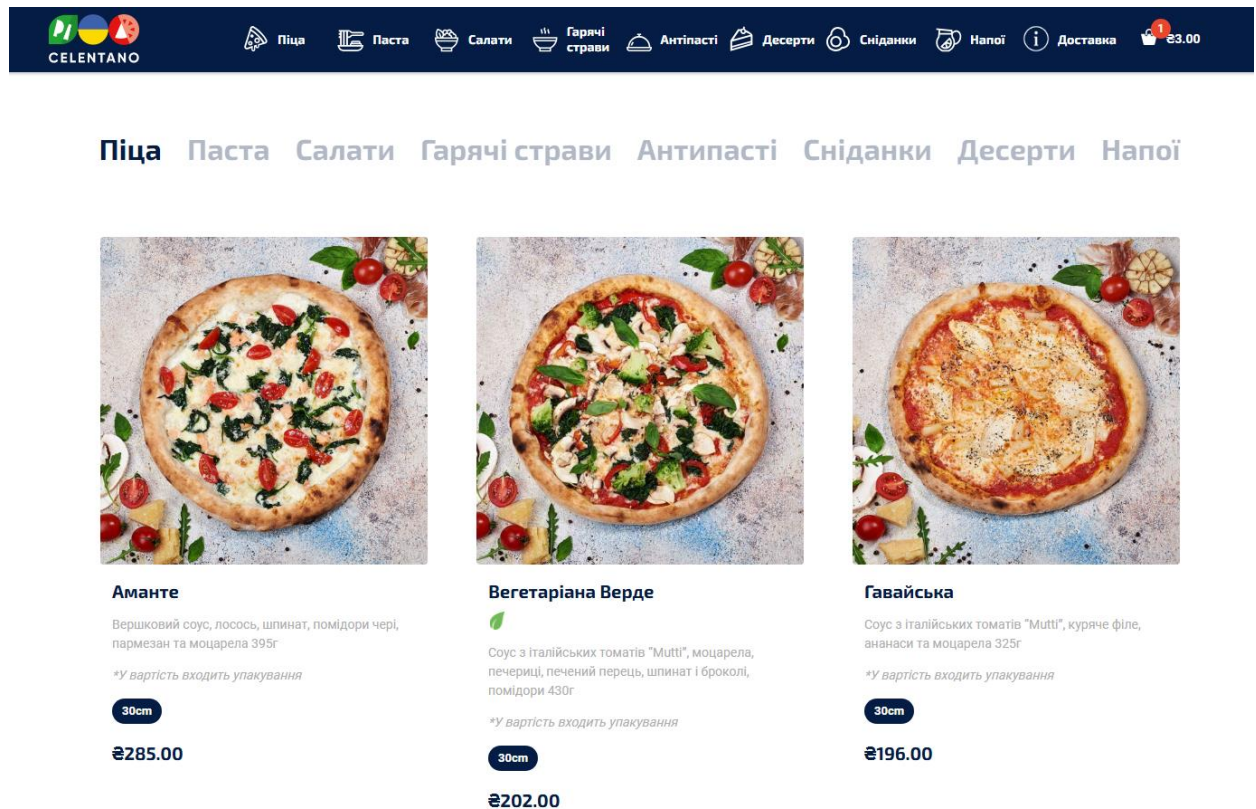


Рисунок 1.1 – Вебсайт піцерії з мережі Celentano

Піцерія Celentano надає можливість придбати не тільки піцу, але і різні додаткові страви, включаючи пасти, салати, гарячі страви, сніданки, десерти, напої. Піци представлені єдиним переліком, звідки їх можна одразу замовити, відкрити детальну інформацію про потрібну піцу. Піци при цьому визначені своїм розміром, а фактично кожна піца визначена одним розміром. Інших

варіантів додаткові до піци або інших варіантів піци не надається. Відповідно користувачі можуть замовляти піци або інші додаткові страви, але не можуть оцінювати ті піци, які вони спробували, не можуть вподобати піци. Тобто функціональність присвячена виключно замовленням та наданню відповідного меню піцерії.

Мережа піцерій La П'єц (рис. 1.2) має також піци, представлені єдиним варіантом, але при цьому можна додатково до піци додавати соуси та інші додатки обирати [2].

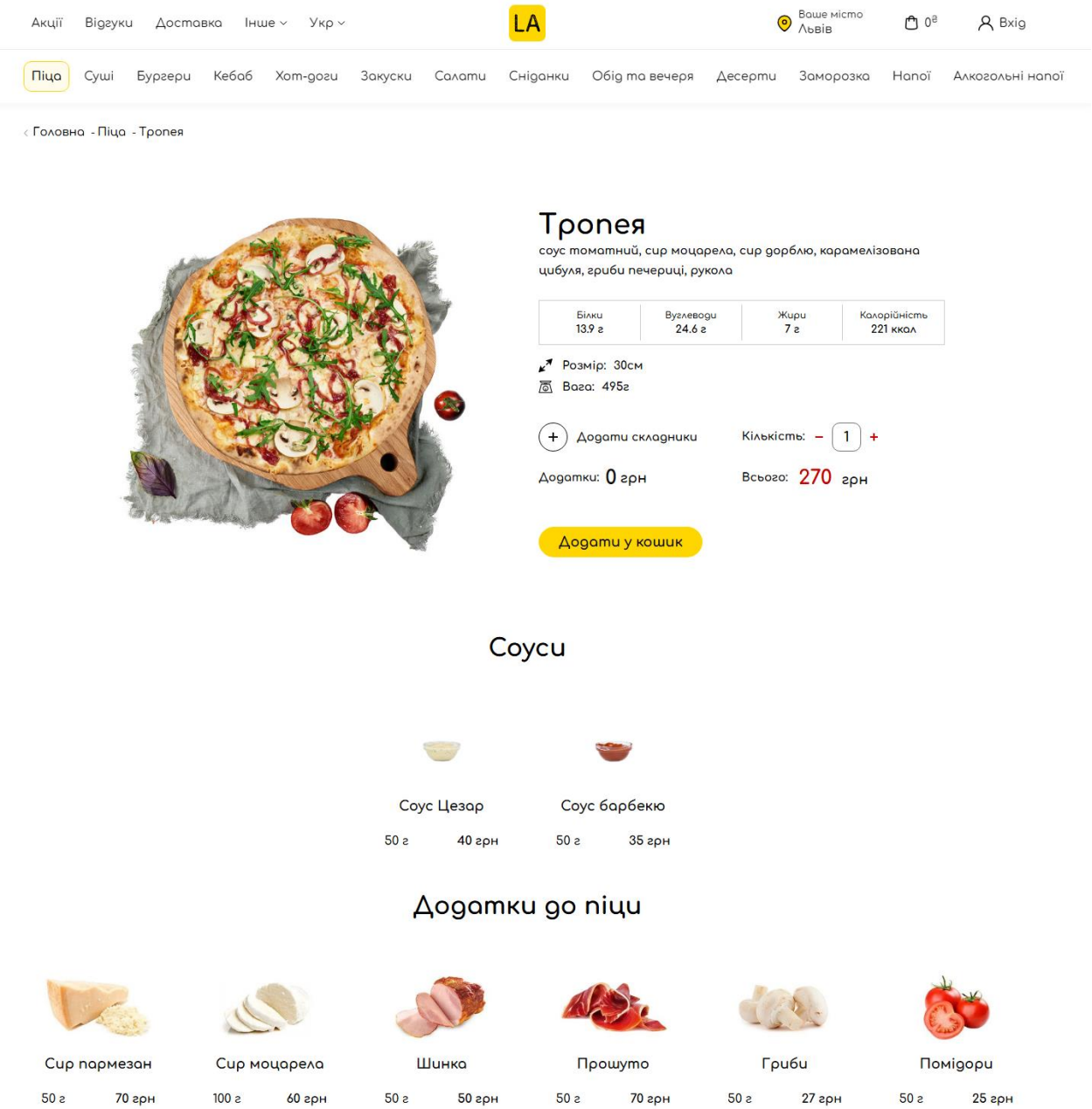


Рисунок 1.2 – Вебсайт піцерії з мережі La П'єц

При цьому також користувачі не можуть оцінювати ті піци, які вони спробували, не можуть вподобати піци.

Піцерії з мережі Domino's (рис. 1.3) мають вебзастосунок, який дозволяє за кожною піцою обирати її розмір, варіант тіста, борту піци [3]. Окрім того можна визначати потрібні додатки, зокрема соуси для піци. Користувачі вебзастосунка не можуть оцінювати ті піци, які вони спробували, не можуть вподобати піци.

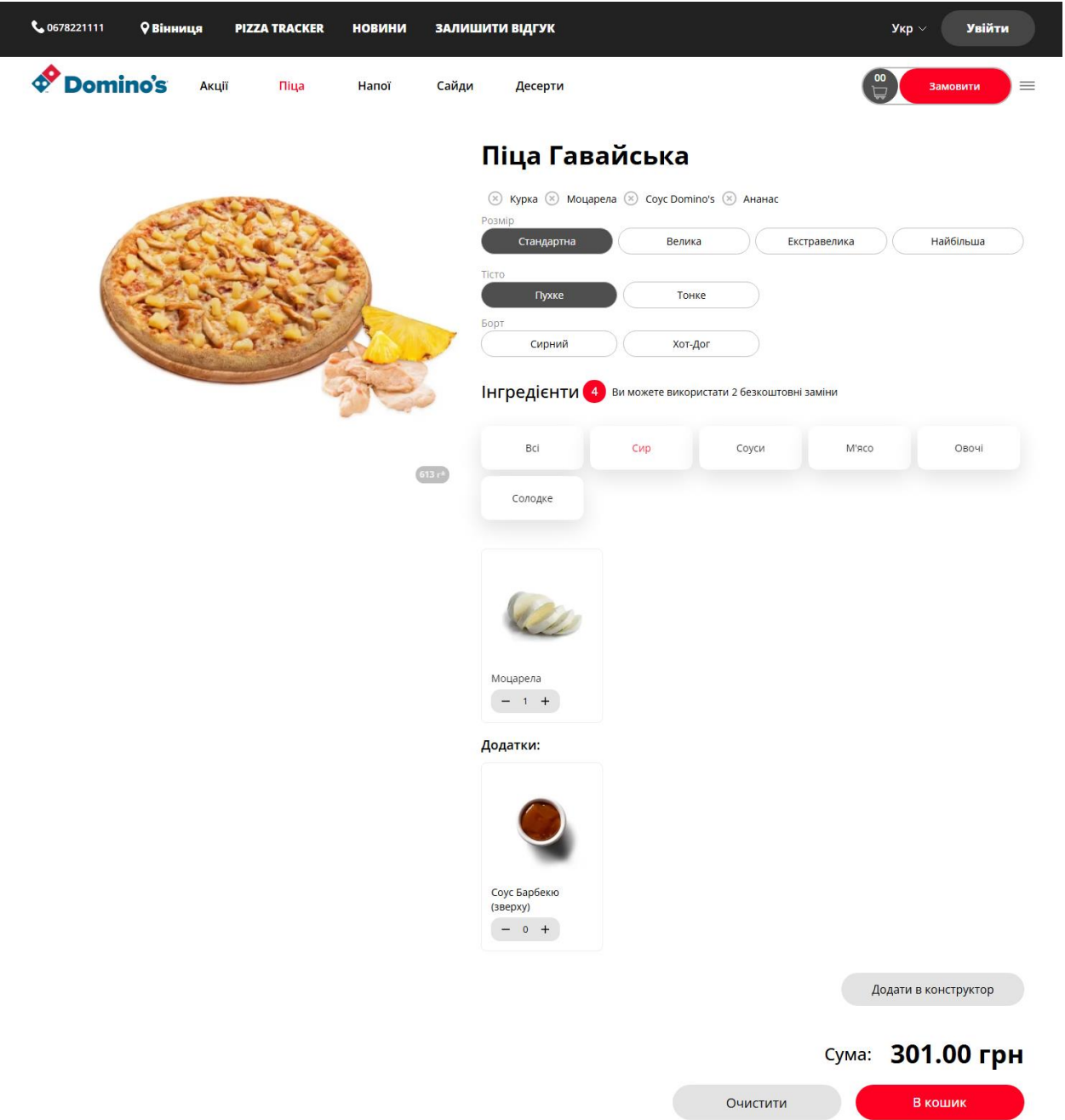


Рисунок 1.3 – Вебсайт піцерії з мережі Domino's

Піцерія Keste (рис. 1.4) виготовляє неаполітанську піцу в Нью Йорку [4].

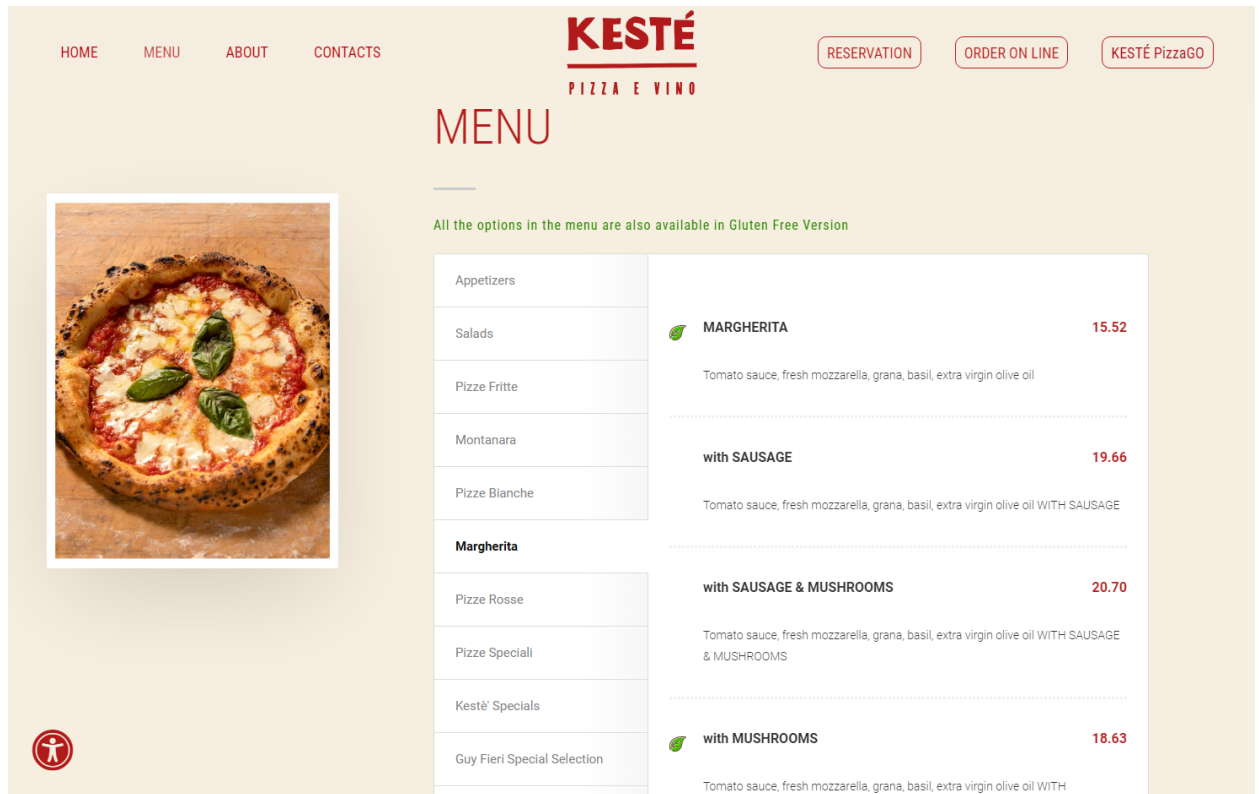


Рисунок 1.4 – Вебзастосунок піцерії Keste

Вебзастосунок надає основне меню з певним описом піц, включає також ресторанне меню, але окремих сторінок піц, детальних описів при цьому немає. Надаються можливості замовлення піц у піцерії, проте користувачі не можуть оцінювати ті піци, які вони спробували, не можуть вподобати піци у піцерії.

1.2 Визначення рівня створюваного вебзастосунку піцерії

Аналоги розробки (табл. 1.1) зокрема включають Celentano, La П'єц, Domino's.

Таблиця 1.1 – Порівняння можливостей аналогів розробки вебзастосунку піцерії

Показник	Celentano	La П'єц	Domino's	Розробка
Замовлення піц	+	+	+	+
Вибір доступних варіантів піц	—	—	+	+
Вибір додатків до піц	—	+	+	+
Уподобання піц	—	—	—	+
Оцінювання піц	—	—	—	+
Прогнозування кількості проданих піц за групами	—	—	—	+

Порівняно з аналогами розробка повинна надавати можливості прогнозування кількості проданих піц за групами, оцінювання піц, уподобання піц, що відсутні у всіх інших програм. У той же час вибір доступних варіантів піц надається тільки в Domino's, а також і в розробці. Також вибір додатків до піц відсутній у вебзастосунку Celentano.

1.3 Висновки за розділом 1

Розділ 1 включає огляд існуючих програмних застосунків забезпечення діяльності піцерій, зокрема Celentano, La П'єц, Domino's. Для виконання розробки виділено основні функції, виходячи з результатів огляду, встановлено відмінні можливості програми, що включають можливості прогнозування кількості проданих піц за групами, оцінювання піц, уподобання піц, що відсутні у всіх інших програм. Для розробки такого вебзастосунку піцерії визначено завдання:

- виконати аналіз програмних застосунків забезпечення діяльності піцерій;
- розробити метод прогнозування кількості проданих піц за групами;
- виконати проєктування вебзастосунку піцерії;
- реалізувати вебзастосунок піцерії;
- виконати експериментальне дослідження прогнозування кількості проданих піц за групами.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Задача прогнозування кількості проданих піц за групами інгредієнтів

Задача прогнозування кількості проданих піц за групами інгредієнтів полягає у прогнозування того, скільки саме піц буде продано у конкретній піцерії, для якої виконується прогнозування, на певну дату. Ця дата представлена певною добою відносно поточної, фактично її номером. У подальшому в практичній частині під час експериментального дослідження була використана тридцята доба. Тобто прогнозування відбувається на 30-ту добу відносно поточної.

Прогнозування кількості проданих піц за групами інгредієнтів виконується на основі даних про попередні дні. Це послідовність значень кількості проданих піц за ряд послідовних днів. Відповідно тоді на вхід подається послідовність значень за ряд послідовних днів, а на відповідний останній день відбувається прогнозування кількості проданих піц за групами інгредієнтів на відповідний номер дня відносно останнього дня. Тобто вхідними даними є послідовність значень кількості проданих піц за послідовність днів (наприклад, 30), а вихідним значенням – кількість проданих піц на відповідний день (наприклад, 30-й після 30-го, що є частиною вхідних даних).

Відповідно необхідно мати навчальні дані, які відповідають спостереженням за роботою піцерії. Спостереження полягають у кількості проданих піц на день у цій піцерії. При цьому піци розглядаються з певної групи, але це описано нижче. Відповідно тоді є послідовність днів, яка є значно ширшою за послідовність вхідних даних і яка може використовуватися для формування часових рядів на основі навчальних даних. Для формування таких часових рядів необхідно, починаючи з першого дня, виконувати зміщення на основі певного вікна. Вікно має ширину, що дорівнює кількості значень у вхідних даних.

При цьому особливістю цієї постановки задачі прогнозування кількості проданих піц за групами інгредієнтів є те, що розглядаються піци тільки певної групи інгредієнтів. Це означає, що кожна постановка задачі існує в межах окремо виділених інгредієнтів, таким чином об'єднуючи піци, які містять цей інгредієнт, у окрему групу.

Таким чином, виходить, що постановка задачі прогнозування кількості проданих піц за групами інгредієнтів полягає в тому, щоб за заданим інгредієнтом виконати прогнозування кількості проданих піц у піцерії, включаючи до них тільки ті піци, які містять заданий інгредієнт, на певний день відносно заданого (поточного), знаючи послідовність значень кількості піц, що містять цей інгредієнт, у піцерії протягом послідовних днів, включаючи заданий день.

Тоді для розв'язання цієї задачі прогнозування кількості проданих піц за групами інгредієнтів необхідно створити окрему модель прогнозування, але ця модель відповідає одній групі піц, що містить заданий інгредієнт. Відповідно в залежності від кількості можливих основних інгредієнтів, які використовуються у піцерії при приготуванні піц, кількість створених моделей у підсумку має відповідати кількості цих інгредієнтів. Окрім того можуть розглядатися не абсолютно всі інгредієнти, а ті, що вважаються найбільш значимими при плануванні діяльності піцерії, тобто за якими мають бути зроблені висновки.

2.2 Метод прогнозування кількості проданих піц за групами інгредієнтів

Нижче представлений метод прогнозування кількості проданих піц за групами інгредієнтів, який дозволяє описану постановку задачі прогнозування кількості проданих піц за групами інгредієнтів розв'язати.

Оскільки дані про продажі піц у піцерії зазвичай представлені замовленнями піц у конкретній піцерії, то на першому етапі фактично має

відбуватися підготовка до роботи з задачею прогнозування кількості проданих піц за групами інгредієнтів. Для цього має бути встановлено перелік інгредієнтів, за якими необхідно згрупувати піци. Цей перелік інгредієнтів, якщо інакше він не був заданий, може бути створений шляхом виділення всіх інгредієнтів, які використовуються при приготуванні піц у заданій піцерії. Тобто тоді цей перелік є всеохоплюючим.

Тоді далі на цьому етапі для такого переліку інгредієнтів для кожного замовлення має бути встановлено, чи входить кожен такий інгредієнт у відповідну піцу, що входить до замовлення. В залежності від того, яка кількість піц у замовленні, відповідно для інгредієнта, який входить у приготування відповідної піци, має бути встановлено значення, що дорівнює кількості таких піц у замовленні.

Якщо у замовленні одразу декілька піц, то виконується об'єднання даних за замовленням у одне спостереження. Тоді для кожної піци для кожного її інгредієнта встановлюється число, що відповідає кількості таких піц. Відповідно в результаті знаходиться сума таких значень за кожним інгредієнтом.

На другому етапі відбувається формування часових рядів за кожним інгредієнтом. Для цього визначається проміжок доступних дат, за якими було накопичено замовлення піц у піцерії. Тоді для кожної дати у цьому проміжку визначається кількість піц з відповідним інгредієнтом, яку було продано. Тобто виділяються всі замовлення, які підпадають під задану дату, а тоді знаходиться сума у позиції, що відповідає заданому інгредієнту. Таким чином відбувається робота зі всіма датами у проміжку. При цьому забезпечується нормування цих даних, тобто вони зводяться до інтервалу від 0 до 1. Відповідно за кожним інгредієнтом створюється набір екземплярів даних. Кожен такий екземпляр представляє послідовність значень розміром з кількість вхідних даних (наприклад, 30) та значення, що відповідає вихідній ознаці (наприклад, 60-й день). Наступний екземпляр тоді формується зсуванням значень на 1 добу. Тобто вхідними значеннями вже є кількість

проданих піц за 2-й, 3-й, ..., 31-й дні. Вихідним значенням є кількість проданих піц на 61-й день. За таким принципом формується кожен наступний екземпляр.

При експериментальному дослідженні 80 % всіх доступних даних використовується в якості навчальної вибірки, а 20 % – в якості тестової вибірки. При створенні моделі для інтеграції в програму всі дані використовуються в якості навчальної вибірки.

На третьому етапі для кожного інгредієнта створюється модель AutoRegressive Integrated Moving Average (ARIMA) [5] та реалізується навчання кожної такої моделі на основі відповідної навчальної вибірки даних.

Загалом в якості моделі прогнозування часового ряду можуть використовуватися окрім моделі ARIMA, наприклад:

- дерево рішень;
- метод опорних векторів.

Саме ці методи і були застосовані під час експериментального дослідження прогнозування кількості проданих піц за групами інгредієнтів.

На четвертому етапі формуються екземпляри даних, які характеризують новий день. Для цього відповідно для кожного інгредієнта, виділеного на першому етапі, формується окремий часовий ряд. Якщо довжина вхідного ряду дорівнює 30, то відповідно 29 попередніх нормованих значень відносно нового дня та нормоване значення кількості піц відповідного інгредієнта за новий день формують такий часовий ряд.

Далі кожен такий часовий ряд подається на вхід відповідної моделі прогнозування кількості проданих піц у піцерії за відповідним інгредієнтом та обчислюються відповідні значення кількості піц за інгредієнтами на виході, що і є результатами методу.

2.3 Висновки за розділом 2

У розділі подано задачу прогнозування кількості проданих піц за групами інгредієнтів та представлено метод для її розв'язання. Наукова

новизна роботи полягає в методі прогнозування кількості проданих піц за групами інгредієнтів, що характеризується розподілом піц на групи за конкретними інгредієнтами, застосуванням моделі ARIMA, що дозволяє визначати кількість піц з вмістом відповідного інгредієнта, яка буде продана в піцерії на певну дату.

3 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Визначення сценаріїв роботи користувачів з вебзастосунком піцерії

Основні сценарії клієнта у вебзастосунку піцерії:

- перегляд категорій піц;
- перегляд піц за категорією;
- перегляд інформації про піцу в меню піцерії;
- оформлення замовлення піц у піцерії;
- авторизація;
- перегляд категорій додаткових продуктів у піцерії за типом;
- перегляд додаткових продуктів у піцерії за категорією;
- перегляд інформації про додатковий продукт в меню піцерії;
- перегляд типів додаткових продуктів у піцерії;
- перегляд власних замовлень піц у піцерії;
- перегляд станів за замовленням піц у піцерії;
- уподобання піци;
- уподобання додаткового продукту в піцерії;
- перегляд вподобаних піц та додаткових продуктів;
- перегляд профілю користувача вебзастосунку піцерії;
- оцінювання піци в піцерії;
- оцінювання додаткового продукту в піцерії;
- пошук піци;
- перегляд власного замовлення піц у піцерії;
- перегляд оцінених піц та додаткових продуктів користувачем

(рис. 3.1).

Альтернативні сценарії клієнта у вебзастосунку піцерії:

- редагування власних даних користувача вебзастосунку піцерії, що є альтернативним для перегляду профілю користувача вебзастосунку піцерії для випадку, коли дані користувача вебзастосунку піцерії в профілі застаріли;

– скасування вподобання піци, що є альтернативним для вподобання піц, коли вподобання цієї піци цим користувачем виконано та потрібно його скасувати.

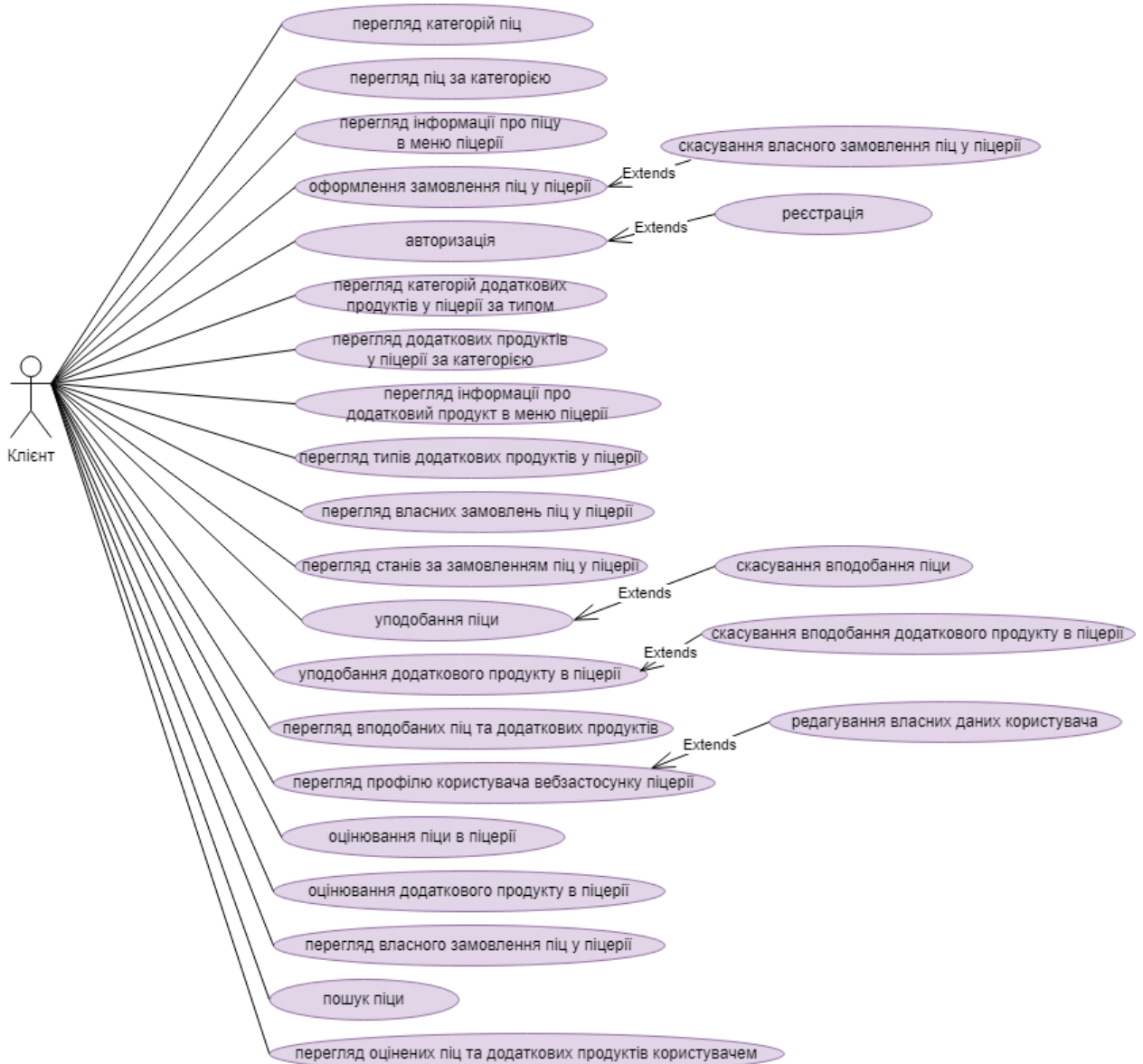


Рисунок 3.1 – Діаграма прецедентів для клієнта у вебзастосунку піцерії

Додаткові альтернативні сценарії клієнта у вебзастосунку піцерії:

– скасування вподобання додаткового продукту в піцерії, що є альтернативним для вподобання додаткового продукту в піцерії, коли вподобання цього додаткового продукту в піцерії цим користувачем виконано та потрібно його скасувати;

– скасування власного замовлення піц у піцерії, що є альтернативним для оформлення замовлення піц у піцерії, коли замовлення піц у піцерії лише створене, але вже не потрібне клієнту;

– реєстрація, що є альтернативним сценарієм для авторизації, коли клієнт у вебзастосунку піцерії не має облікового запису.

Основні сценарії оператора у вебзастосунку піцерії:

- перегляд категорій піц;
- перегляд піц за категорією;
- перегляд інформації про піцу в меню піцерії;
- оформлення замовлення піц у піцерії;
- авторизація;
- перегляд всіх варіантів вартості піци, які існували або використовуються;

- пошук піци;
- перегляд замовлення піц у піцерії;
- перегляд категорій додаткових продуктів у піцерії за типом;
- перегляд додаткових продуктів у піцерії за категорією;
- перегляд інформації про додатковий продукт в меню піцерії;
- перегляд типів додаткових продуктів у піцерії;
- перегляд всіх варіантів вартості додаткового продукту, які існували або використовуються;

- перегляд замовлень піц у піцерії за період та за станами;
- підтвердження замовлення піц у піцерії;
- визначення початку виготовлення піц у піцерії за замовленням;
- визначення готовності замовлення піц у піцерії;
- визначення початку доставки за замовленням піц у піцерії;
- визначення завершення доставки за замовленням піц у піцерії;
- визначення видачі замовлення піц у піцерії;
- перегляд вподобаних піц та додаткових продуктів користувачем;
- перегляд станів за замовленням піц у піцерії;

- перегляд профілю користувача вебзастосунку піцерії;
 - перегляд оцінених піц та додаткових продуктів користувачем
- (рис. 3.2).

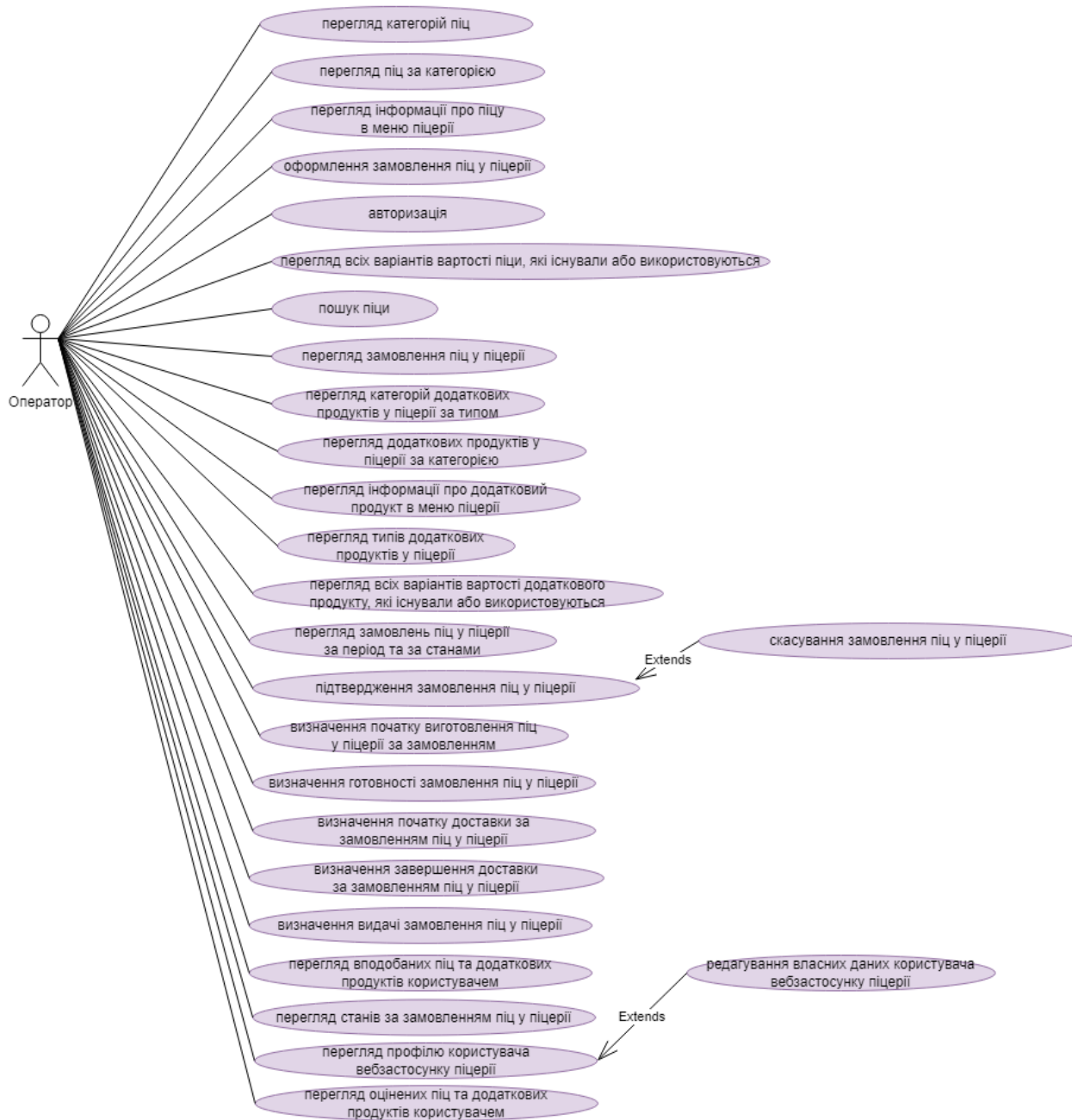


Рисунок 3.2 – Діаграма прецедентів для оператора у вебзастосунку піцерії

Альтернативні сценарії клієнта у вебзастосунку піцерії:

- редагування власних даних користувача вебзастосунку піцерії, що є альтернативним для перегляду профілю користувача вебзастосунку піцерії, коли дані користувача вебзастосунку піцерії в профілі застаріли;

- скасування замовлення піц у піцерії, що є альтернативним для підтвердження замовлення піц у піцерії, коли замовлення для цього клієнта не має виконуватися.

Основні сценарії менеджера у вебзастосунку піцерії додатково до сценаріїв оператора у вебзастосунку піцерії:

- додавання піци до меню піцерії;
- реєстрація операторів у вебзастосунку піцерії;
- додавання варіанта піци за існуючим пунктом меню;
- встановлення вартості піци на період;
- додавання додаткового продукту до меню піцерії;
- додавання варіанта обсягу додаткового продукту за існуючим пунктом меню;
- встановлення вартості додаткового продукту на період;
- перегляд кількості замовлень піц у піцерії за період за піцою в меню;
- перегляд профілю користувача вебзастосунку піцерії;
- перегляд оцінених піц та додаткових продуктів користувачем.

Альтернативні сценарії клієнта у вебзастосунку піцерії:

- редагування даних піци в меню піцерії, що є альтернативним для додавання піци до меню піцерії, коли дані піци в меню піцерії мають бути змінені;
- архівування піци з меню піцерії, що є альтернативним для додавання піци до меню піцерії, коли піца більше не продається в піцерії;
- вилучення варіанта піци за існуючим пунктом меню, що є альтернативним для додавання варіанта піци за існуючим пунктом меню, коли такий варіант піци вже більше не має надаватися клієнтам;

- редагування додаткового продукту в меню піцерії, що є альтернативним для додавання додаткового продукту до меню піцерії, коли дані додаткового продукту в меню піцерії мають бути змінені;
- архівування додаткового продукту з меню піцерії, що є альтернативним для додавання додаткового продукту до меню піцерії, коли додатковий продукт з меню піцерії більше не продається в піцерії;
- вилучення варіанта обсягу додаткового продукту за існуючим пунктом меню, що є альтернативним для додавання варіанта обсягу додаткового продукту за існуючим пунктом меню, коли такий варіант обсягу додаткового продукту за існуючим пунктом меню більше не доступний в піцерії;
- вилучення пропозиції вартості піци на період, що є альтернативним для встановлення вартості піци на період, коли така пропозиція вартості піци на період більше не має використовуватися;
- вилучення пропозиції вартості додаткового продукту на період, що є альтернативним для встановлення вартості додаткового продукту на період, коли така пропозиція вартості додаткового продукту на період більше не має використовуватися (рис. 3.3).

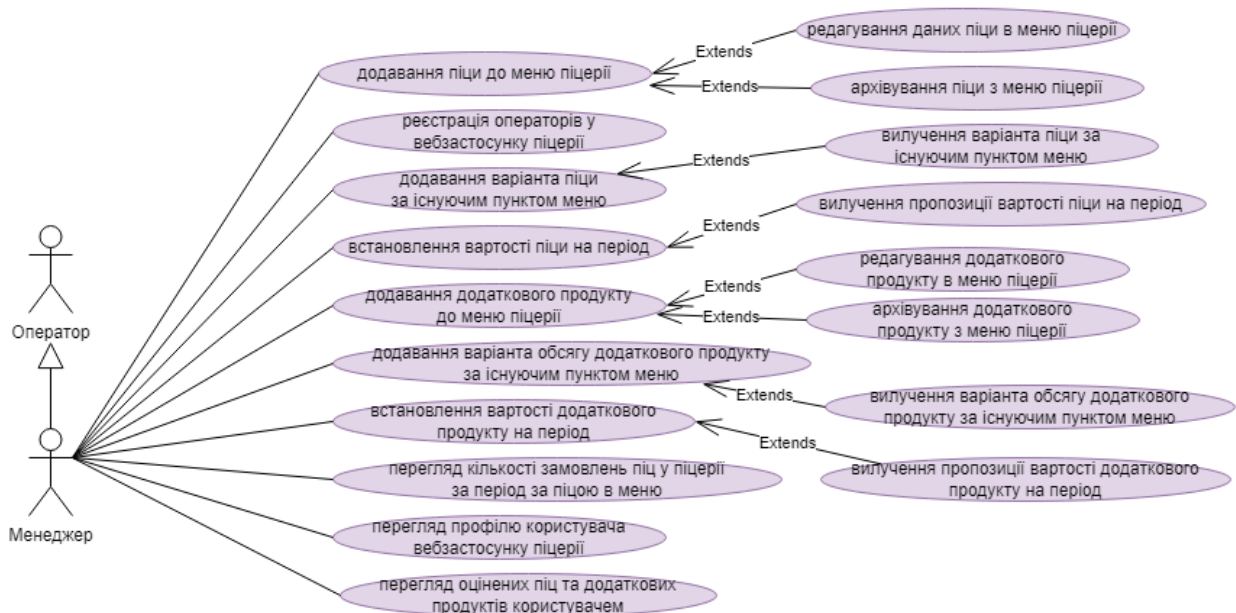


Рисунок 3.3 – Діаграма прецедентів для менеджера у вебзастосунку піцерії

3.2 Вибір засобів розробки вебзастосунку піцерії

Враховуючи одночасно і необхідність створення саме вебзастосунку піцерії, і інтеграції методу прогнозування кількості проданих піц за групами інгредієнтів до складу програмного забезпечення, вибір рішень для розробки було зведено до фреймворку Django [6] і середовища Node.js [7], які до того ж представляють різні мови програмування (табл. 3.1).

Остаточний вибір зроблений на користь Django з врахуванням того, що мова Python має кращий рівень засобів машинного навчання, роботи з даними, що є важливим, а фреймворк має кращий рівень безпеки та швидкості самої розробки. Django поступається Node.js за продуктивністю, проте враховуючи, що Django все ж має продуктивність готових вебзастосунків, прийнятну для практичного використання в тому числі за великого трафіку, то це не є проблемою.

Таблиця 3.1 – Порівняння можливих рішень для реалізації вебзастосунку піцерії

	Django	Node.js
Базова мова програмування	Python	JavaScript
Продуктивність	Висока для практичного використання	Вища
Час, що витрачається на розробку	Менше	Більше
Рівень безпеки	Вищий	Дещо нижчий
Рівень засобів машинного навчання, роботи з даними	Вищий	Нижчий

3.3 Проєктування структури бази даних

База даних на етапах як проєктування, так і реалізації відповідної роботи виконана за допомогою системи керування MySQL, відомої як безкоштовної, такої, що зарекомендувала себе в багатьох проєктах на практиці.

Таблиця Pizzacategory в базі даних визначає категорії представлення піц користувачу через стовпці:

- id – ідентифікатор категорії піци;
- pcateg_name – назва категорії піци;
- position – порядковий номер категорії піци в переліку категорій для відображення на відповідній сторінці.

Таблиця Pizzamenu в базі даних визначає піци з меню піцерії через стовпці:

- id – ідентифікатор піци з меню піцерії;
- fullpizzaname – повна назва піци з меню піцерії;
- shortpizzaname – коротка назва піци з меню піцерії;
- description – опис піци з меню піцерії.

Оскільки кожна піца з меню піцерії може входити в різні перелік для відображення в меню користувачу, то було створено таблицю Pizzacategorymenu, що в базі даних визначає входження піц до категорій через стовпці:

- id – ідентифікатор відношення піци з меню піцерії до категорії представлення піц користувачу;
- pizzacategory – категорія представлення піц користувачу, до якої входить конкретна піца з меню піцерії, визначена зовнішнім ключем до таблиці Pizzacategory;
- pizzamenu – піца з меню піцерії, яка відноситься до цієї категорії представлення піц користувачу, визначена зовнішнім ключем до таблиці Pizzamenu.

Таблиця `Pizzaingredient` в базі даних визначає можливі інгредієнти для виготовлення піц у піцерії через стовпці:

- `id` – ідентифікатор можливого інгредієнта для виготовлення піц у піцерії;
- `ingredient_name` – назва можливого інгредієнта для виготовлення піц у піцерії.

Таблиця `Pizzamenuingredient` в базі даних визначає інгредієнт, який використовується для виготовлення піци з меню піцерії, через стовпці:

- `id` – ідентифікатор інгредієнта, який використовується для виготовлення піци з меню піцерії;
- `pizzaingredient` – інгредієнт, який використовується для приготування цієї піци з меню піцерії, визначена зовнішнім ключем до таблиці `Pizzaingredient`;
- `pizzamenu` – піца з меню піцерії, при приготуванні якої використовується цей інгредієнт, визначена зовнішнім ключем до таблиці `Pizzamenu`.

Таблиця `Pizzamenuphoto` в базі даних визначає фотографії виготовлених піц піцерією через стовпці:

- `id` – ідентифікатор фотографії виготовленої піци піцерією;
- `pizzamenu` – піца з меню піцерії, яка показана на фотографії, визначена зовнішнім ключем до таблиці `Pizzamenu`;
- `photo` – фотографія піци з меню піцерії.

Таблиця `Pizzasize` в базі даних визначає можливі розміри піц через стовпці:

- `id` – ідентифікатор можливого розміру піци, який може бути застосований для конкретної піци в піцерії;
- `sizemarker` – маркер розміру піци, тобто напис, який визначає цей варіант розміру піци.

Таблиця `Pizzamenu size` в базі даних визначає конкретний розмір, доступний для конкретної піци з меню піцерії, через стовпці:

– id – ідентифікатор конкретного розміру, доступного для конкретної піци з меню піцерії;

– pizzamenu – піца з меню піцерії, для якої доступний цей розмір, визначена зовнішнім ключем до таблиці Pizzamenu;

– pizzasize – можливий розмір піци, який і фактично доступний для цієї піци, визначений зовнішнім ключем до таблиці Pizzasize;

– sizeweight – вага піци цього розміру в грамах;

– sizediameter – діагональ піци цього розміру в сантиметрах;

– proteins – кількість білків у піці цього розміру;

– calories – кількість калорій у піці цього розміру;

– fats – кількість жирів у піці цього розміру;

– carbohydrates – кількість вуглеводів у піці цього розміру.

Таблиця Pizzadough в базі даних визначає можливий варіант тіста піци, який може бути доступний у піцерії, через стовпці:

– id – ідентифікатор можливого варіанта тіста піци, який може бути доступний у піцерії;

– doughmarker – маркер варіанта тіста піци, тобто напис, який визначає цей варіант варіанта тіста піци.

Таблиця Pizzamenudough в базі даних визначає конкретний варіант тіста конкретної піци з меню піцерії через стовпці:

– id – ідентифікатор конкретного варіанта тіста конкретної піци з меню піцерії;

– pizzamenu – піца з меню піцерії, для якої доступний цей варіант тіста, визначена зовнішнім ключем до таблиці Pizzamenu;

– pizzadough – можливий варіант тіста піци, який і фактично доступний для цієї піци, визначений зовнішнім ключем до таблиці Pizzadough.

Таблиця Pizzaboard в базі даних визначає можливий варіант бортів піци через стовпці:

- id – ідентифікатор можливого варіанта борта піци, який може бути доступний у піцерії;

- boardmarker – маркер можливого варіанта борта піци, який може бути доступний у піцерії.

Таблиця Pizzamenuboard в базі даних визначає конкретний варіант борта конкретної піци з меню піцерії через стовпці:

- id – ідентифікатор конкретного варіанта борта конкретної піци з меню піцерії;

- pizzamenu – піца з меню піцерії, для якої доступний цей варіант тіста, визначена зовнішнім ключем до таблиці Pizzamenu;

- pizzaboard – можливий варіант борта піци, який і фактично доступний для цієї піци, визначений зовнішнім ключем до таблиці Pizzaboard.

Таблиця Pizzavariant в базі даних визначає конкретний варіант піци в піцерії в конкретний період часу через стовпці:

- id – ідентифікатор конкретного варіанта піци в піцерії;

- pizzamenu size – конкретний розмір піци в цьому варіанті піци в піцерії, визначений зовнішнім ключем до таблиці Pizzamenu size;

- pizzamenu dough – конкретний варіант тіста піци в цьому варіанті піци в піцерії, визначений зовнішнім ключем до таблиці Pizzamenu dough;

- pizzamenu board – конкретний варіант борта піци в цьому варіанті піци в піцерії, визначений зовнішнім ключем до таблиці Pizzamenu board.

Таблиця Pizzaprice в базі даних визначає вартість конкретного варіанті піци в піцерії в конкретний період часу через стовпці:

- id – ідентифікатор вартості конкретного варіанта піци в піцерії в конкретний період часу;

- pizzavariant – конкретний варіант піци в піцерії, для якого визначена вартість в конкретний період часу, визначений зовнішнім ключем до таблиці Pizzavariant;

- activefrom – час, з якого ця пропозиція вартості конкретного варіанта піци в піцерії доступна;

– `activeto` – час, до якого ця пропозиція вартості конкретного варіанта піци в піцерії доступна;

– `price` – вартість конкретного варіанта піци в піцерії в заданий період часу.

Таблиця `Sidetype` в базі даних визначає типи додаткових продуктів, які продаються разом з піцою в піцерії, через стовпці:

– `id` – ідентифікатор додаткового продукту, який продається разом з піцою в піцерії;

– `sidetype_name` – назва додаткового продукту, який продається разом з піцою в піцерії.

Прикладами таких типів можуть бути зокрема напої, сайди, десерти.

Таблиця `Sidecategory` в базі даних визначає категорії додаткових продуктів через стовпці:

– `id` – ідентифікатор категорії додаткових продуктів до піц у меню піцерії;

– `sidetype` – додатковий продукт, який продається разом з піцою в піцерії, визначений зовнішнім ключем до таблиці `Sidetype`;

– `scateg_name` – назва категорії додаткових продуктів до піц у меню піцерії.

Таблиця `Sidecategorymenu` в базі даних визначає конкретні позиції додаткових продуктів до піц у меню піцерії через стовпці:

– `id` – ідентифікатор конкретної позиції додаткових продуктів до піц у меню піцерії;

– `sidecategory` – категорія додаткових продуктів, до якої належить ця конкретна позиція додаткових продуктів до піц у меню піцерії, визначена зовнішнім ключем до таблиці `Sidecategory`;

– `sidemenu` – позиція додаткових продуктів до піц у меню піцерії, що належить до цієї категорії, визначена зовнішнім ключем до таблиці `Sidemenu`.

Таблиця `Sidemenu` в базі даних визначає конкретну позицію додаткових продуктів до піц у меню піцерії через стовпці:

- id – ідентифікатор конкретної позиції додаткових продуктів до піц у меню піцерії;
- fullsidename – повна назва конкретної позиції додаткових продуктів до піц у меню піцерії;
- shortsidename – коротка назва конкретної позиції додаткових продуктів до піц у меню піцерії;
- description – опис конкретної позиції додаткових продуктів до піц у меню піцерії;
- photo – фотографія конкретної позиції додаткових продуктів до піц у меню піцерії.

Таблиця Volumemarker в базі даних визначає можливий варіант обсягу додаткового продукту до піц у меню піцерії через стовпці:

- id – ідентифікатор можливого варіанта обсягу додаткового продукту до піц у меню піцерії;
- volumemarker_name – назва можливого варіанта обсягу додаткового продукту до піц у меню піцерії.

Таблиця Sidevolume в базі даних визначає конкретний варіант обсягу конкретного додаткового продукту в меню піцерії через стовпці:

- id – ідентифікатор конкретного варіанта обсягу конкретного додаткового продукту в меню піцерії;
- sidemenu – конкретна позиція додаткових продуктів до піц у меню піцерії, для якої визначено доступність цього обсягу, визначена зовнішнім ключем до таблиці Sidemenu;
- volumemarker – можливий варіант обсягу, визначений і фактично доступний для конкретного додаткового продукту до піц у меню піцерії, визначений зовнішнім ключем до таблиці Volumemarker;
- volume_type – тип обсягу конкретного додаткового продукту в піцерії (варіант обсягу представлений міткою, а цей тип вказує, що розмір, наприклад, в літрах або інших одиницях);

– volume_size – значення обсягу конкретного додаткового продукту в піцерії.

Таблиця Sideprice в базі даних визначає вартість конкретного варіанта додаткового продукту в піцерії в конкретний період часу через стовпці:

– id – ідентифікатор вартості конкретного варіанта додаткового продукту в піцерії в конкретний період часу;

– sidevolume – конкретний варіант обсягу конкретного додаткового продукту в меню піцерії, для якого встановлена вартість, визначений зовнішнім ключем до таблиці Sidevolume;

– price – вартість цього конкретного варіанта додаткового продукту в піцерії;

– activefrom – час, з якого ця пропозиція вартості конкретного додаткового продукту в піцерії доступна;

– activeto – час, до якого ця пропозиція вартості конкретного додаткового продукту в піцерії доступна.

Таблиця Pizzeriaorder в базі даних визначає замовлення піц у піцерії через стовпці:

– id – ідентифікатор замовлення піц у піцерії;

– pizzeriauser – користувач вебзастосунку піцерії, який сформував замовлення піц у піцерії, визначена зовнішнім ключем до таблиці Pizzeriauser;

– order_saved – час оформлення замовлення піц у піцерії;

– order_planned – час, коли має бути виконано замовлення піц у піцерії;

– price – сумарна вартість замовлення піц у піцерії;

– notation – примітка до замовлення піц у піцерії;

– delivery_address – адреса доставки замовлення піц у піцерії, якщо така доставка здійснюється;

– phone – номер телефону для зв'язку за замовленням піц у піцерії;

– toconnect – необхідність зв'язку з клієнтом за замовленням піц у піцерії для уточнення інформації або відсутність такої необхідності.

Таблиця Pizzeriaorder_pizza в базі даних визначає вміст піц у замовленні піц у піцерії через стовпці:

- id – ідентифікатор вмісту піц у замовленні піц у піцерії;
- pizzeriaorder – замовлення піц у піцерії, до якого входять ці піци, визначена зовнішнім ключем до таблиці Pizzeriaorder;
- pizzavariant – варіант піци в піцерії, який внесено до замовлення піц у піцерії, визначений зовнішнім ключем до таблиці Pizzavariant;
- quantity – кількість цього варіанта піц у піцерії, яку внесено до замовлення.

Таблиця Pizzeriaorder_side в базі даних визначає вміст додаткових продуктів у замовленні піц у піцерії через стовпці:

- id – ідентифікатор вмісту додаткових продуктів у замовленні піц у піцерії;
- pizzeriaorder – замовлення піц у піцерії, до якого входять ці додаткові продукти, визначене зовнішнім ключем до таблиці Pizzeriaorder;
- sidevolume – конкретний варіант обсягу конкретного додаткового продукту в меню піцерії, який входить до замовлення піц у піцерії, визначений зовнішнім ключем до таблиці Sidevolume;
- quantity – кількість конкретного варіанта обсягу конкретного додаткового продукту в меню піцерії в замовленні піц у піцерії.

Таблиця Pizzeriaorder_status в базі даних визначає стани, в яких знаходиться або знаходилося замовлення піц у піцерії, через стовпці:

- id – ідентифікатор стану, в якому знаходиться або знаходилося замовлення піц у піцерії;
- pizzeriaorder – замовлення піц у піцерії, до якого належить цей стан, визначене зовнішнім ключем до таблиці Pizzeriaorder;
- pizzeriauser – користувач вебзастосунку піцерії, який створив замовлення піц у піцерії, визначений зовнішнім ключем до таблиці Pizzeriauser;

– pizzeriaorder_status – стан, який було збережено для замовлення піц у піцерії;

– pizzeriaorder_time – час внесення змін про цей стан для замовлення піц у піцерії.

Таблиця Pizzamenu_like в базі даних визначає вподобання піци у піцерії користувачем вебзастосунку піцерії через стовпці:

– id – ідентифікатор уподобання піци у піцерії користувачем вебзастосунку піцерії;

– pizzamenu – піца у меню піцерії, яку вподобав користувач вебзастосунку піцерії, визначений зовнішнім ключем до таблиці Pizzamenu;

– pizzeriauser – користувач вебзастосунку піцерії, який вподобав піцу в меню піцерії, визначений зовнішнім ключем до таблиці Pizzeriauser;

– like_time – час вподобання піци у піцерії користувачем вебзастосунку піцерії.

Таблиця Sidemenu_like в базі даних визначає вподобання додаткового продукту в піцерії користувачем вебзастосунку піцерії через стовпці:

– id – ідентифікатор уподобання додаткового продукту в піцерії користувачем вебзастосунку піцерії;

– sidemenu – додатковий продукт у піцерії, який було вподобано користувачем вебзастосунку піцерії, визначений зовнішнім ключем до таблиці Sidemenu;

– pizzeriauser – користувач вебзастосунку піцерії, який вподобав додатковий продукт у піцерії, визначений зовнішнім ключем до таблиці Pizzeriauser;

– like_time – час уподобання додаткового продукту в піцерії користувачем вебзастосунку піцерії.

Таблиця Pizzamenu_choice в базі даних визначає оцінювання піц у піцерії користувачем вебзастосунку піцерії через стовпці:

– id – ідентифікатор оцінювання піц у піцерії користувачем вебзастосунку піцерії;

– pizzamenu – піца у піцерії, яку було оцінено користувачем вебзастосунку піцерії, визначений зовнішнім ключем до таблиці Pizzamenu;

– pizzeriauser – користувач вебзастосунку піцерії, який виконав оцінювання цієї піци у піцерії, визначений зовнішнім ключем до таблиці Pizzeriauser;

– choice_mark – виставлена оцінка піци у піцерії користувачем вебзастосунку піцерії;

– choice_text – текст оцінки піци у піцерії користувачем вебзастосунку піцерії;

– choice_time – час виставлення оцінки піци у піцерії користувачем вебзастосунку піцерії.

Таблиця Sidemenu_choice в базі даних визначає оцінювання додаткових продуктів у піцерії користувачем вебзастосунку піцерії через стовпці:

– id – ідентифікатор оцінювання додаткових продуктів у піцерії користувачем вебзастосунку піцерії;

– sidemenu – додатковий продукт у піцерії, який було оцінено користувачем вебзастосунку піцерії, визначений зовнішнім ключем до таблиці Sidemenu;

– pizzeriauser – користувач вебзастосунку піцерії, який виконав оцінювання цього додаткового продукту в піцерії, визначений зовнішнім ключем до таблиці Pizzeriauser;

– choice_mark – виставлена оцінка додатковому продукту в піцерії користувачем вебзастосунку піцерії;

– choice_text – текст оцінки додаткового продукту в піцерії користувачем вебзастосунку піцерії;

– choice_time – час виставлення оцінки додатковому продукту в піцерії користувачем вебзастосунку піцерії.

Таблиця Pizzeriauser в базі даних визначає користувача вебзастосунку піцерії через стовпці:

– id – ідентифікатор користувача вебзастосунку піцерії;

– user – обліковий запис користувача Django, який використовує користувач вебзастосунку піцерії;

– base_delivery_address – базова адреса доставки, яка використовується користувачем вебзастосунку піцерії;

– base_phone – базовий номер телефону, який використовується користувачем вебзастосунку піцерії;

– photo – фотографія користувача вебзастосунку піцерії.

Таблиця Pizzaaddition_type в базі даних визначає типи додатків до піци через стовпці:

– id – ідентифікатор типу додатків до піци;

– addition_name – назва типу додатків до піци.

Таблиця Pizzaaddition в базі даних визначає існуючі додатки до піци через стовпці:

– id – ідентифікатор існуючого додатка до піци;

– pizzaaddition_type – тип додатка до піци, до якого належить цей додаток, визначений зовнішнім ключем до таблиці Pizzaaddition_type;

– addition_name – назва додатка до піци;

– price – вартість додатка до піци;

– volume – обсяг додатка до піци.

Таблиця Pizzamenu_addition в базі даних визначає додатки до конкретної піци з меню через стовпці:

– id – ідентифікатор додатка до конкретної піци з меню піцерії;

– pizzamenu – піца з меню в піцерії, до якої пропонується цей додаток, визначений зовнішнім ключем до таблиці Pizzamenu;

– pizzaaddition – існуючий додаток до піци, який пропонується до цієї піци з меню піцерії, визначений зовнішнім ключем до таблиці Pizzaaddition.

Таблиця Pizzeriaorder_addition в базі даних визначає замовлені додатки до піци через стовпці:

– id – ідентифікатор замовленого додатку до піци в замовленні піц у піцерії;

Таблиця Pizzamenuingredientstat в базі даних визначає статистику замовлень піц за днями у піцерії за інгредієнтами через стовпці:

- id – ідентифікатор запису про статистику замовлень піц за днями у піцерії за інгредієнтами;
- pizzaingredient – інгредієнт при приготуванні піци, за яким визначено цю кількість замовлень, визначений зовнішнім ключем до таблиці Pizzaingredient;
- orderdate – дата, за якою визначено кількість замовлень;
- ordernumber – фактична кількість замовлень піц у піцерії на цей день;
- prediction – прогнозована кількість замовлень піц у піцерії на цей день.

3.4 Висновки за розділом 3

Основними користувачами вебзастосунку піцерії є клієнт, оператор та менеджер, для яких і було розглянуто всі потрібні сценарії для забезпечення їхньої роботи, в результаті чого створено діаграми прецедентів. Мову Python, фреймворку Django і систему MySQL обрано для реалізації вебзастосунку піцерії. Детально описано створену структуру бази даних для вебзастосунку піцерії.

4 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

4.1 Загальний порядок роботи з замовленнями піц у піцерії

Замовлення піц у піцерії як центральна сутність вебзастосунку піцерії потребує деталізації станів, тому побудовано відповідну діаграму (рис. 4.1).

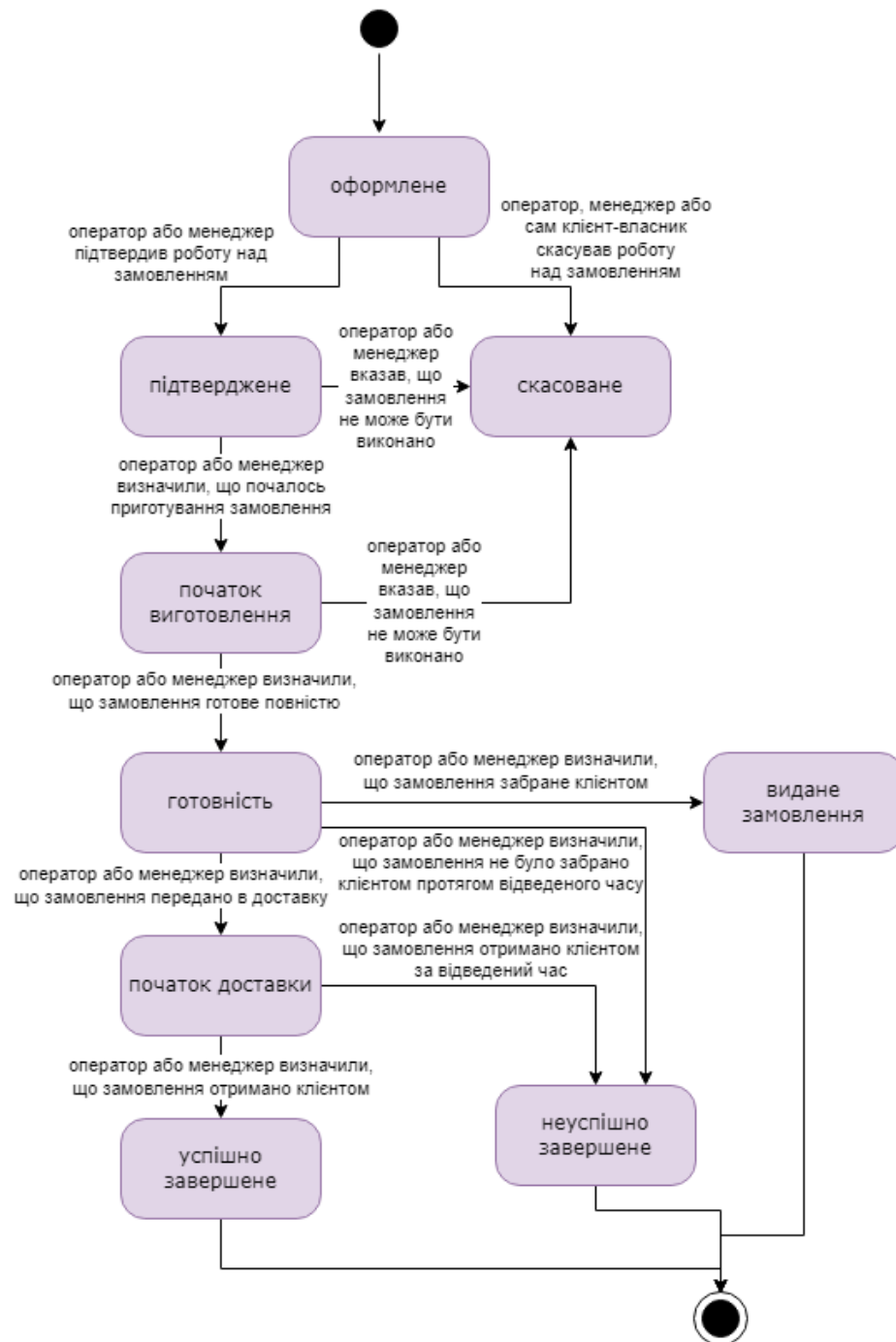


Рисунок 4.1 – Діаграма станів замовлення піц у піцерії

Перший стан замовлення піц у піцерії, який проявляється при роботі з ними, є оформлений, що створюється, коли клієнт оформив замовлення піц у піцерії через вебзастосунок. Це також може зробити і оператор вебзастосунку піцерії, і менеджер. Далі замовлення може бути або скасовано будь-якою стороною, або підтверджене оператором або менеджером.

Якщо замовлення піц у піцерії підтверджене, то далі воно може бути або скасовано оператором або менеджером, або для нього буде визначено початок виготовлення, коли стартує приготування піц з замовлення. Якщо приготування не буде з якихось причин завершено успішно, то оператор або менеджер скасують це замовлення. Якщо замовлення успішно підготовлене, то оператор або менеджер встановлять готовність замовлення.

Якщо готове замовлення було забрано самим клієнтом, то воно є виданим. Якщо замовлення передано для доставки, то оператор або менеджер переводять замовлення до початку доставки. Після цього замовлення може бути або неуспішно завершеним, якщо клієнт не отримає замовлення піц у піцерії у відведений час, або успішно завершеним, якщо клієнт таке замовлення отримав.

4.2 Реалізовані програмні одиниці у складі вебзастосунку піцерії

Вебзастосунок піцерії в першу чергу створений як вебзастосунок на основі фреймворку Django, а тому передбачено та реалізовано моделі, представлення, шаблони вебсторінок та інші більш дрібні програмні одиниці.

Представлення перегляду категорій піц реалізовано функцією `pizza_categories_view` з шаблоном вебсторінки `pizza_categories.html`. Реалізацію всіх шаблонів вебсторінок виконано з використанням безкоштовного шаблону [8].

Представлення перегляду піц за категорією реалізовано функцією `pizza_category_menus_view` з шаблоном вебсторінки `pizza_category_menus.html`. Функція має параметр `pizzacategory_id`, який задає

ідентифікатор категорії піц, за якою мають бути вибрані піци. Розділення відбувається на порції сторінками по 10 піц. Визначаються вподобання піц користувачем.

Представлення перегляду інформації про піцу в меню піцерії реалізовано функцією `pizza_menu_view` з шаблоном вебсторінки `pizza_menu.html`. Функція має параметр `pizzamenu_id`, який задає ідентифікатор піци в меню піцерії. За піцою вибираються її інгредієнти, варіанти тіста, бортів, додатків до піци, з яких можна виконати вибір, її фотографії, доступні варіанти піци, вартість цих варіантів, виконане вподобання та оцінювання піци користувачем, який авторизувався, а також доступність можливості оцінити піцу на поточний момент. Можливість оцінити піцу з'являється тоді, коли користувач замовив піцу, але з моменту замовлення не створив нове оцінювання піци.

Представлення перегляду всіх варіантів вартості піци, які існували або використовуються реалізовано функцією `pizza_prices_view` з шаблоном вебсторінки `pizza_prices.html`. Параметр `pizzamenu_id` задає ідентифікатор самої піци, для якої мають бути виділені варіанти вартості.

Представлення перегляду категорій додаткових продуктів у піцерії за типом та перегляду типів додаткових продуктів у піцерії реалізовано функцією `side_type_categories_view` з шаблоном вебсторінки `side_type.html`. Функція має параметр `sidetype_id`, значення якого може бути і не встановлено. Параметр визначає ідентифікатор типу додаткових продуктів у піцерії. Якщо він не встановлений, то сторінка заповнюється тільки типами додаткових продуктів у піцерії. Якщо значення параметра задано, то за ним як ідентифікатором виділяється тип додаткових продуктів, а за цим типом виділяються вже категорії додаткових продуктів, наявні у піцерії.

Представлення перегляду додаткових продуктів у піцерії за категорією реалізовано функцією `sidecategory_sides_view` з шаблоном вебсторінки `side_category.html`. Сама категорія додаткових продуктів у піцерії задається параметром функції `sidecategory_id`. Розділення відбувається на порції

сторінками по 10 додаткових продуктів у піцерії. Визначаються вподобання додаткових продуктів у піцерії користувачем.

Представлення перегляду інформації про додатковий продукт в меню піцерії реалізовано функцією `side_menu_view` з шаблоном вебсторінки `side_menu.html`. Параметр `sidemenu_id` задає ідентифікатор додаткового продукту в меню піцерії. Витягаються всі варіанти обсягів додаткових продуктів у піцерії, актуальної вартості цих продуктів, визначаються вподобання користувачем цього додаткового продукту в меню піцерії, оцінювання додаткового продукту в меню піцерії користувачем, який авторизувався, а також доступність можливості оцінити цей додатковий продукт в меню піцерії на поточний момент. Можливість оцінити додатковий продукт в меню піцерії з'являється тоді, коли користувач замовив цей додатковий продукт в меню піцерії, але з моменту замовлення не створив нове оцінювання цього додаткового продукту в меню піцерії.

Представлення перегляду всіх варіантів вартості додаткового продукту, які існували або використовуються, реалізовано функцією `side_prices_view` з шаблоном вебсторінки `side_prices.html`. Параметр `sidemenu_id` задає ідентифікатор додаткового продукту в меню піцерії.

Представлення перегляду замовлень піц у піцерії за період та за станами реалізовано функцією `pizzeria_orders_view` з шаблоном вебсторінки `pizzeria_orders.html`. Параметри `startperiod` та `endperiod` функції задають початок та кінець періоду, за який мають відображатися замовлення піц у піцерії. Параметр `state` задає конкретний стан замовлень, які потрібно відображати. Його значення може бути і не встановлено.

Представлення перегляду власних замовлень піц у піцерії реалізовано функцією `own_pizzeria_orders_view` з шаблоном `opizzeria_orders.html`.

Представлення перегляду станів за замовленням піц у піцерії реалізовано функцією `pizzeriaorder_statuses_view` з шаблоном `pizzeria_order_statuses.html`. Параметр `pizzeriaorder_id` задає номер замовлення піц у піцерії.

Представлення перегляду самого замовлення піц у піцерії реалізовано функцією `pizzeriaorder_view` з шаблоном `pizzeria_order.html`. Разом з загальними даними замовлення піц у піцерії за номером `pizzeriaorder_id`, що надаються тільки операторам, менеджерам вебзастосунку піцерії, а також клієнтам-власникам самого замовлення, виділяються 3 останні стани замовлення, склад замовлення, включаючи піци, додаткові продукти, додатки до піц, розміри піц, борти, тісто.

Представлення підтвердження замовлення піц у піцерії реалізовано функцією `pizzeriaorder_approve_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли саме замовлення тільки оформлене.

Представлення скасування замовлення піц у піцерії реалізовано функцією `pizzeriaorder_cancel_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії, а також для клієнтам-власника замовлення у випадку, коли саме замовлення тільки оформлене, підтверджене або розпочався початок виготовлення піц за замовленням.

Представлення підтвердження замовлення піц у піцерії реалізовано функцією `pizzeriaorder_start_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли саме замовлення тільки оформлене.

Представлення визначення готовності замовлення піц у піцерії реалізовано функцією `pizzeriaorder_done_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли за самим замовленням визначено початок виготовлення.

Представлення визначення початку доставки за замовленням піц у піцерії реалізовано функцією `pizzeriaorder_delivery_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли за самим замовленням визначено готовність замовлення.

Представлення визначення успішного або неуспішного завершення доставки за замовленням піц у піцерії реалізовано функцією `pizzeriaorder_end_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли за самим замовленням визначено готовність замовлення або початок доставки. Успішність або неуспішність завершення доставки за замовленням піц у піцерії визначається параметром `endvariant`.

Представлення визначення видачі замовлення піц у піцерії реалізовано функцією `pizzeriaorder_got_view`. Ідентифікатор самого замовлення піц у піцерії задається параметром `pizzeriaorder_id`. Виконання цих дій для замовлення піц у піцерії допустиме тільки для оператора або менеджера вебзастосунку піцерії у випадку, коли за самим замовленням визначено готовність замовлення.

Представлення перегляду оцінених піц та додаткових продуктів користувачем реалізовано функцією `pizza_and_sides_likes_view`. Параметр `pizzeria_user` задає логін користувача, якщо авторизований користувач є оператором або менеджером, а якщо він не заданий, то вибираються дані користувача вебзастосунку піцерії, що авторизувався.

Представлення уподобання піци та скасування вподобання піци реалізовано єдиною функцією `pizzamenu_tolike_view` з параметром `pizzamenu_id`, який задає номер піци з меню піцерії. Якщо уподобання піци цим користувачем присутнє, то реалізується скасування, а якщо ні, то вподобання.

Представлення уподобання додаткового продукту в піцерії та скасування вподобання додаткового продукту в піцерії реалізовано єдиною функцією `sidemenu_tolike_view` з параметром `sidemenu_id`, який задає номер додаткового продукту в піцерії. Якщо уподобання додаткового продукту в піцерії цим користувачем присутнє, то реалізується скасування, а якщо ні, то вподобання.

Аналогічно влаштовані та реалізовані інші представлення у відповідності з заданим початково набором сценаріїв. Нижче ж описані також створені в результаті моделі.

Модель `Pizzacategory` є класом, який визначає категорії представлення піц користувачу.

Поле `pcateg_name` має тип `CharField` та задає назву категорії піци.

Поле `position` має тип `PositiveIntegerField` та задає порядковий номер категорії піци в переліку категорій для відображення на відповідній сторінці.

Поле `pizzamenus` має тип `ManyToManyField` на основі класу `Pizzamenu` та задає піци з меню піцерії, які відносяться до цієї категорії представлення піц користувачу.

Модель `Pizzamenu` є класом, який визначає піци з меню піцерії.

Поле `fullpizzaname` має тип `CharField` та задає повну назву піци з меню піцерії.

Поле `shortpizzaname` має тип `CharField` та задає коротку назву піци з меню піцерії.

Поле `description` має тип `TextField` та задає опис піци з меню піцерії.

Модель `Pizzamenuingredient` є класом, який визначає можливі інгредієнти для виготовлення піц у піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню піцерії, яка містить цей інгредієнт.

Поле `pizzaingredient` має тип `ForeignKey` на основі класу `Pizzaingredient` та задає інгредієнт, який міститься в піці.

Модель `Pizzamenudought` є класом, який визначає конкретний варіант тіста конкретної піци з меню піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню піцерії, для якої доступний цей варіант тіста.

Поле `pizzadought` має тип `ForeignKey` на основі класу `Pizzadought` та задає можливий варіант тіста піци, який і фактично доступний для цієї піци.

Модель `Pizzamenuboard` є класом, який визначає конкретний варіант борта конкретної піци з меню піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню піцерії, для якої доступний цей варіант тіста.

Поле `pizzaboard` має тип `ForeignKey` на основі класу `Pizzaboard` та задає можливий варіант борта піци, який і фактично доступний для цієї піци.

Модель `Pizzamenu_addition` є класом, який визначає додатки до конкретної піци з меню.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню в піцерії, до якої пропонується цей додаток.

Поле `pizzaaddition` має тип `ForeignKey` на основі класу `Pizzaaddition` та задає існуючий додаток до піци, який пропонується до цієї піци з меню піцерії.

Модель `Pizzaingredient` є класом, який визначає можливі інгредієнти для виготовлення піц у піцерії.

Поле `ingredient_name` має тип `CharField` та задає назву можливого інгредієнта для виготовлення піц у піцерії.

Модель `Pizzamenuphoto` є класом, який визначає фотографії виготовлених піц піцерією.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню піцерії, яка показана на фотографії.

Поле `photo` має тип `ImageField` та задає фотографію піци з меню піцерії.

Модель `Pizzasize` є класом, який визначає можливі розміри піц.

Поле `sizemarker` має тип `CharField` та задає маркер розміру піци, тобто напис, який визначає цей варіант розміру піци.

Модель `Pizzamenu` є класом, який визначає конкретний розмір, доступний для конкретної піци з меню піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу з меню піцерії, для якої доступний цей розмір.

Поле `pizzasize` має тип `ForeignKey` на основі класу `Pizzasize` та задає можливий розмір піци, який і фактично доступний для цієї піци.

Поле `sizeweight` має тип `DecimalField` та задає вагу піци цього розміру в грамах.

Поле `sizediameter` має тип `DecimalField` та задає діагональ піци цього розміру в сантиметрах.

Поле `proteins` має тип `DecimalField` та задає кількість білків у піці цього розміру.

Поле `calories` має тип `DecimalField` та задає кількість калорій у піці цього розміру.

Поле `fats` має тип `DecimalField` та задає кількість жирів у піці цього розміру.

Поле `carbohydrates` має тип `DecimalField` та задає кількість вуглеводів у піці цього розміру.

Модель `Pizzadough` є класом, який визначає можливий варіант тіста піци, який може бути доступний у піцерії.

Поле `doughmarker` має тип `CharField` та задає маркер варіанта тіста піци, тобто напис, який визначає цей варіант варіанта тіста піци.

Модель `Pizzaboard` є класом, який визначає можливий варіант бортів піци. Поле `boardmarker` має тип `CharField` та задає маркер можливого варіанта борта піци, який може бути доступний у піцерії.

Модель `Pizzavariant` є класом, який визначає конкретний варіант піци в піцерії в конкретний період часу.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає конкретний розмір піци в цьому варіанті піци в піцерії.

Поле `pizzamenudought` має тип `ForeignKey` на основі класу `Pizzamenudought` та задає конкретний варіант тіста піци в цьому варіанті піци в піцерії.

Поле `pizzamenuboard` має тип `ForeignKey` на основі класу `Pizzamenuboard` та задає конкретний варіант борта піци в цьому варіанті піци в піцерії.

Модель `Pizzaprice` є класом, який визначає вартість конкретного варіанті піци в піцерії в конкретний період часу.

Поле `pizzavariant` має тип `ForeignKey` на основі класу `Pizzavariant` та задає конкретний варіант піци в піцерії, для якого визначена вартість в конкретний період часу.

Поле `activefrom` має тип `DateTimeField` та задає час, з якого ця пропозиція вартості конкретного варіанта піци в піцерії доступна.

Поле `activeto` має тип `DateTimeField` та задає час, до якого ця пропозиція вартості конкретного варіанта піци в піцерії доступна.

Поле `price` має тип `DecimalField` та задає вартість конкретного варіанта піци в піцерії в заданий період часу.

Модель `Sidetype` є класом, який визначає типи додаткових продуктів, які продаються разом з піцою в піцерії.

Поле `sidetype_name` має тип `CharField` та задає назву додаткового продукту, який продається разом з піцою в піцерії.

Модель `Sidecategory` є класом, який визначає категорії додаткових продуктів.

Поле `sidetype` має тип `ForeignKey` на основі класу `Sidetype` та задає додатковий продукт, який продається разом з піцою в піцерії.

Поле `scateg_name` має тип `CharField` та задає назву категорії додаткових продуктів до піц у меню піцерії.

Поле `sidemenus` має тип `ManyToManyField` на основі класу `Sidemenu` та задає позиції додаткових продуктів до піц у меню піцерії, що належать до цієї категорії.

Модель `Sidemenu` є класом, який визначає конкретну позицію додаткових продуктів до піц у меню піцерії.

Поле `fullsidename` має тип `CharField` та задає повну назву конкретної позиції додаткових продуктів до піц у меню піцерії.

Поле `shortsidename` має тип `CharField` та задає коротку назву конкретної позиції додаткових продуктів до піц у меню піцерії.

Поле `description` має тип `TextField` та задає опис конкретної позиції додаткових продуктів до піц у меню піцерії.

Поле `photo` має тип `ImageField` та задає фотографію конкретної позиції додаткових продуктів до піц у меню піцерії.

Модель `Volumemarker` є класом, який визначає можливий варіант обсягу додаткового продукту до піц у меню піцерії.

Поле `volumemarker_name` має тип `CharField` та задає назву можливого варіанта обсягу додаткового продукту до піц у меню піцерії.

Модель `Sidevolume` є класом, який визначає конкретний варіант обсягу конкретного додаткового продукту в меню піцерії.

Поле `sidemenu` має тип `ForeignKey` на основі класу `Sidemenu` та задає конкретну позицію додаткових продуктів до піц у меню піцерії, для якої визначено доступність цього обсягу.

Поле `volumemarker` має тип `ForeignKey` на основі класу `Volumemarker` та задає можливий варіант обсягу, визначений і фактично доступний для конкретного додаткового продукту до піц у меню піцерії.

Поле `volume_type` має тип `CharField` та задає тип обсягу конкретного додаткового продукту в піцерії.

Поле `volume_size` має тип `DecimalField` та задає значення обсягу конкретного додаткового продукту в піцерії.

Модель `Sideprice` є класом, який визначає вартість конкретного варіанта додаткового продукту в піцерії в конкретний період часу.

Поле `sidevolume` має тип `ForeignKey` на основі класу `Sidevolume` та задає конкретний варіант обсягу конкретного додаткового продукту в меню піцерії, для якого встановлена вартість.

Поле `price` має тип `DecimalField` та задає вартість цього конкретного варіанта додаткового продукту в піцерії.

Поле `activefrom` має тип `DateTimeField` та задає час, з якого ця пропозиція вартості конкретного додаткового продукту в піцерії доступна.

Поле `activeto` має тип `DateTimeField` та задає час, до якого ця пропозиція вартості конкретного додаткового продукту в піцерії доступна.

Модель `Pizzeriaorder` є класом, який визначає замовлення піц у піцерії.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який сформував замовлення піц у піцерії.

Поле `order_saved` має тип `DateTimeField` та задає час оформлення замовлення піц у піцерії.

Поле `order_planned` має тип `DateTimeField` та задає час, коли має бути виконано замовлення піц у піцерії.

Поле `price` має тип `DecimalField` та задає сумарну вартість замовлення піц у піцерії.

Поле `notation` має тип `CharField` та задає примітку до замовлення піц у піцерії.

Поле `delivery_address` має тип `CharField` та задає адресу доставки замовлення піц у піцерії, якщо така доставка здійснюється.

Поле `phone` має тип `CharField` та задає номер телефону для зв'язку за замовленням піц у піцерії.

Поле `toconnect` має тип `BooleanField` та задає необхідність зв'язку з клієнтом за замовленням піц у піцерії для уточнення інформації або відсутність такої необхідності.

Модель `Pizzeriaorder_pizza` є класом, який визначає вміст піц у замовленні піц у піцерії.

Поле `pizzeriaorder` має тип `ForeignKey` на основі класу `Pizzeriaorder` та задає замовлення піц у піцерії, до якого входять ці піци.

Поле `pizzavariant` має тип `ForeignKey` на основі класу `Pizzavariant` та задає варіант піци в піцерії, який внесено до замовлення піц у піцерії.

Поле `quantity` має тип `IntegerField` та задає кількість цього варіанта піц у піцерії, яку внесено до замовлення.

Модель `Pizzeriaorder_side` є класом, який визначає вміст додаткових продуктів у замовленні піц у піцерії.

Поле `pizzeriaorder` має тип `ForeignKey` на основі класу `Pizzeriaorder` та задає замовлення піц у піцерії, до якого входять ці додаткові продукти.

Поле `sidevolume` має тип `ForeignKey` на основі класу `Sidevolume` та задає конкретний варіант обсягу конкретного додаткового продукту в меню піцерії, який входить до замовлення піц у піцерії.

Поле `quantity` має тип `IntegerField` та задає кількість конкретного варіанта обсягу конкретного додаткового продукту в меню піцерії в замовленні піц у піцерії.

Модель `Pizzeriaorder_status` є класом, який визначає стани, в яких знаходиться або знаходилося замовлення піц у піцерії.

Поле `pizzeriaorder` має тип `ForeignKey` на основі класу `Pizzeriaorder` та задає замовлення піц у піцерії, до якого належить цей стан.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який створив замовлення піц у піцерії.

Поле `pizzeriaorder_status` має тип `PositiveSmallIntegerField` та задає стан, який було збережено для замовлення піц у піцерії.

Поле `pizzeriaorder_time` має тип `DateTimeField` та задає час внесення змін про цей стан для замовлення піц у піцерії.

Модель `Pizzamenu_like` є класом, який визначає вподобання піци у піцерії користувачем вебзастосунку піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу в меню піцерії, яку вподобав користувач вебзастосунку піцерії.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який вподобав піцу в меню піцерії.

Поле `like_time` має тип `DateTimeField` та задає час вподобання піци у піцерії користувачем вебзастосунку піцерії.

Модель `Sidemenu_like` є класом, який визначає вподобання додаткового продукту в піцерії користувачем вебзастосунку піцерії.

Поле `sidemenu` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає додатковий продукт у піцерії, який було вподобано користувачем вебзастосунку піцерії.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який вподобав додатковий продукт у піцерії.

Поле `like_time` має тип `DateTimeField` та задає час вподобання додаткового продукту в піцерії користувачем вебзастосунку піцерії.

Модель `Pizzamenu_choice` є класом, який визначає оцінювання піц у піцерії користувачем вебзастосунку піцерії.

Поле `pizzamenu` має тип `ForeignKey` на основі класу `Pizzamenu` та задає піцу у піцерії, яку було оцінено користувачем вебзастосунку піцерії.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який виконав оцінювання цієї піци у піцерії.

Поле `choice_mark` має тип `PositiveSmallIntegerField` та задає виставлену оцінку піци у піцерії користувачем вебзастосунку піцерії.

Поле `choice_text` має тип `CharField` та задає текст оцінки піци у піцерії користувачем вебзастосунку піцерії.

Поле `choice_time` має тип `DateTimeField` та задає час виставлення оцінки піци у піцерії користувачем вебзастосунку піцерії.

Модель `Sidemenu_choice` є класом, який визначає оцінювання додаткових продуктів у піцерії користувачем вебзастосунку піцерії.

Поле `sidemenu` має тип `ForeignKey` на основі класу `Sidemenu` та задає додатковий продукт у піцерії, який було оцінено користувачем вебзастосунку піцерії.

Поле `pizzeriauser` має тип `ForeignKey` на основі класу `Pizzeriauser` та задає користувача вебзастосунку піцерії, який виконав оцінювання цього додаткового продукту в піцерії.

Поле `choice_mark` має тип `PositiveSmallIntegerField` та задає виставлену оцінку додатковому продукту в піцерії користувачем вебзастосунку піцерії.

Поле `choice_text` має тип `CharField` та задає текст оцінки додаткового продукту в піцерії користувачем вебзастосунку піцерії.

Поле `choice_time` має тип `DateTimeField` та задає час виставлення оцінки додатковому продукту в піцерії користувачем вебзастосунку піцерії.

Модель `Pizzeriauser` є класом, який визначає користувача вебзастосунку піцерії.

Поле `user` має тип `OneToOneField` на основі класу `User` та задає обліковий запис користувача Django, який використовує користувач вебзастосунку піцерії.

Поле `base_delivery_address` має тип `CharField` та задає базову адресу доставки, яка використовується користувачем вебзастосунку піцерії.

Поле `base_phone` має тип `CharField` та задає базовий номер телефону, який використовується користувачем вебзастосунку піцерії.

Поле `photo` має тип `ImageField` та задає фотографію користувача вебзастосунку піцерії.

Модель `Pizzaaddition_type` є класом, який визначає типи додатків до піци.

Поле `addtion_name` має тип `CharField` та задає назву типу додатків до піци.

Модель `Pizzaaddition` є класом, який визначає існуючі додатки до піци.

Поле `pizzaaddition_type` має тип `ForeignKey` на основі класу `Pizzaaddition_type` та задає тип додатка до піци, до якого належить цей додаток.

Поле `addition_name` має тип `CharField` та задає назву додатка до піци.

Поле `price` має тип `DecimalField` та задає вартість додатка до піци.

Поле `volume` має тип `DecimalField` та задає обсяг додатка до піци.

Модель `Pizzeriaorder_addition` є класом, який визначає замовлені додатки до піци.

Поле `pizzeriaorder` має тип `ForeignKey` на основі класу `Pizzeriaorder` та задає замовлення піц у піцерії, до якого входить цей додаток до піци.

Поле `pizzamenu_addition` має тип `ForeignKey` на основі класу `Pizzamenu_addition` та задає додаток до піци в меню піцерії, який доданий до замовлення.

Поле `quantity` має тип `SmallIntegerField` та задає кількість одиниць додатку до піци, яку було замовлено.

Модель `Pizzamenuingredientstat` є класом, який визначає статистику замовлень піц за днями у піцерії за інгредієнтами.

Поле `pizzaingredient` має тип `ForeignKey` на основі класу `Pizzaingredient` та задає інгредієнт при приготуванні піци, за яким визначено цю кількість замовлень.

Поле `orderdate` має тип `DateTimeField` та задає дату, за якою визначено кількість замовлень. Поле `ordernumber` має тип `PositiveBigIntegerField` та задає фактичну кількість замовлень піц у піцерії на цей день.

Поле `prediction` має тип `PositiveBigIntegerField` та задає прогнозовану кількість замовлень піц у піцерії на цей день.

4.3 Висновки за розділом 4

У розділі представлено послідовність роботи з замовленнями піци в піцерії за допомогою вираження цієї послідовності у вигляді діаграми станів, описано реалізовані класи складових вебзастосунку піцерії.

5 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

5.1 Основне призначення програми

Програма призначена для забезпечення замовлення піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами.

5.2 Виконання програми

Виконання програми можливе за початкового запуску застосунку на сервері, після чого з клієнтських комп'ютерів або інших пристроїв можна звертатися до сервера з відповідними адресами вебсторінок застосунку піцерії.

Після звернення до сторінок вебзастосунку піцерії відкривається одна зі сторінок, після чого надається можливість виконати авторизацію. Якщо авторизація не виконана, то реалізується робота в режимі клієнта. Якщо авторизація виконана, то все залежить від типу облікового запису.

Якщо користувач вебзастосунку піцерії є клієнтом, то він працює після авторизації (як і до авторизації) з меню, що включає розділи:

- розділ «Піци» для відображення спершу категорій піц, а далі за вибраною категорією набору піц (рис. 5.1), які до неї належать, окрім того надається посилання на сторінці для переходу до пошуку за піцями, якщо такий засіб необхідний;

- розділ «Смаколики» для відображення типів додаткових продуктів у піцерії, а далі можна обрати потрібний тип, за ним категорію, а за категорією відобразити всі відповідні додаткові продукти;

- розділ «Замовлення» для відображення всіх власних замовлень клієнта, при цьому для створення замовлення необхідно зі сторінки відповідної піци, додаткового продукту додати їх до замовлення, відповідно у позиції кошику у верхній правій частині сторінки відображається сумарна

вартість замовлення (або тільки напис «Кошик», коли жодні піци або додаткові продукти до замовлення не додані);

- розділ «Лайки» містить уподобані клієнтом піци та додаткові продукти в піцерії;

- розділ «Оцінювання» містить оцінені клієнтом піци та додаткові продукти в піцерії;

- розділ «Профіль» містить можливості роботи з даними власного профілю.

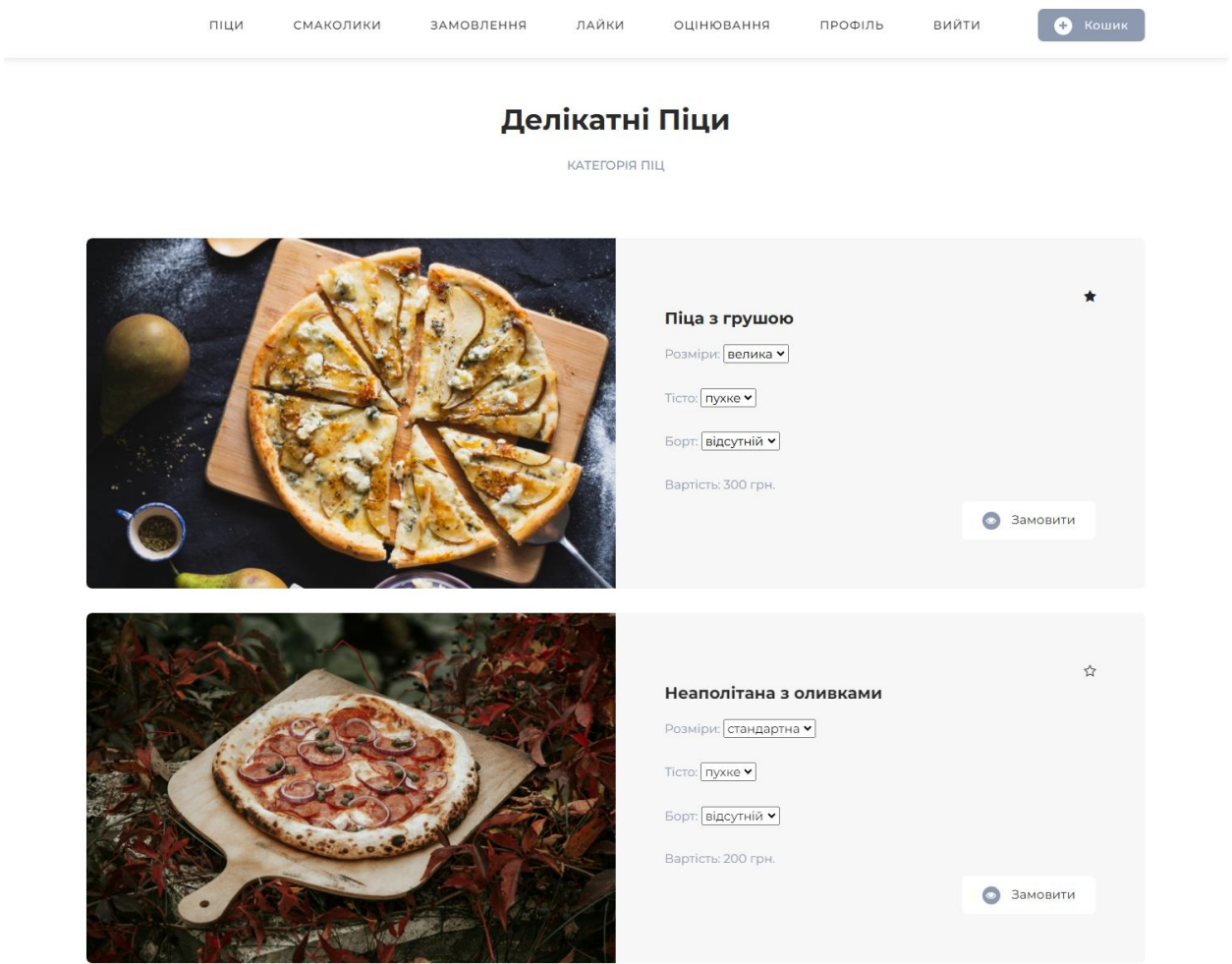


Рисунок 5.1 – Сторінка піц за категорією

При виборі категорії піц (рис. 5.1) відображаються всі піци за обраною категорією. За кожною піцою при цьому демонструється фотографія, назва піци, доступні розміри, варіанти тіста та варіанти бортів піци. В залежності від

обраного варіанта відображається його вартість. Окрім того за кожною піцою є кнопка замовлення обраного варіанта піци, після чого цей варіант піци додається в кошик. При цьому одразу кожну піцу можна вподобати або скасувати це вподобання.

Окрім того кожну таку піцу можна розглянути детальніше, відкривши відповідну сторінку піци (рис. 5.2).

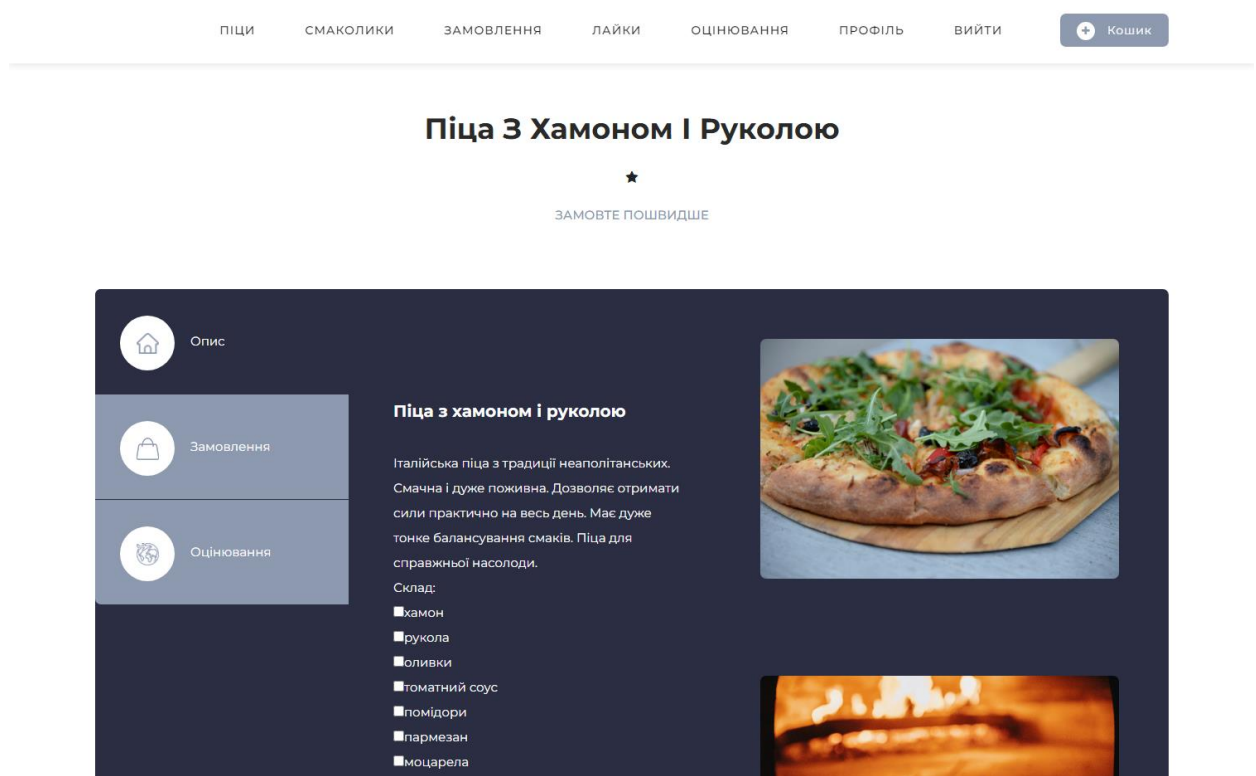


Рисунок 5.2 – Сторінка піци

Сторінка піци складається з трьох розділів, що включають опис, замовлення та оцінювання. Розділ опису містить загальний опис, фотографії, склад інгредієнтів. Розділ замовлень дозволяє обрати потрібний варіант піци щодо розміру, варіанту тіста та бортів. Окрім того можна побачити вартість такого варіанта, обрати додатки для конкретно цієї піци, вносячи все це до замовлення. У свою чергу для оператора та менеджера надається можливість переходу до варіантів вартості піци за різні періоди часу. Розділ оцінювання передбачає перегляд виконаних клієнтами оцінювань піци та можливість

переходу до власного оцінювання піци клієнтом, якщо клієнт раніше замовив цю піцу, але не створив після цього оцінювання.

Сторінка оператора вебзастосунку піцерії не містить розділи вподобання та оцінювання. У свою чергу для менеджера вебзастосунку піцерії доступні додаткові розділи меню, що включають планування та операторів. Розділ операторів дозволяє додати нового оператора вебзастосунку піцерії менеджеру. Розділ планування дозволяє переглянути для обраного інгредієнту піци та заданого періоду дат кількість піц, які було продано за цей час у піцерії (рис. 5.3).

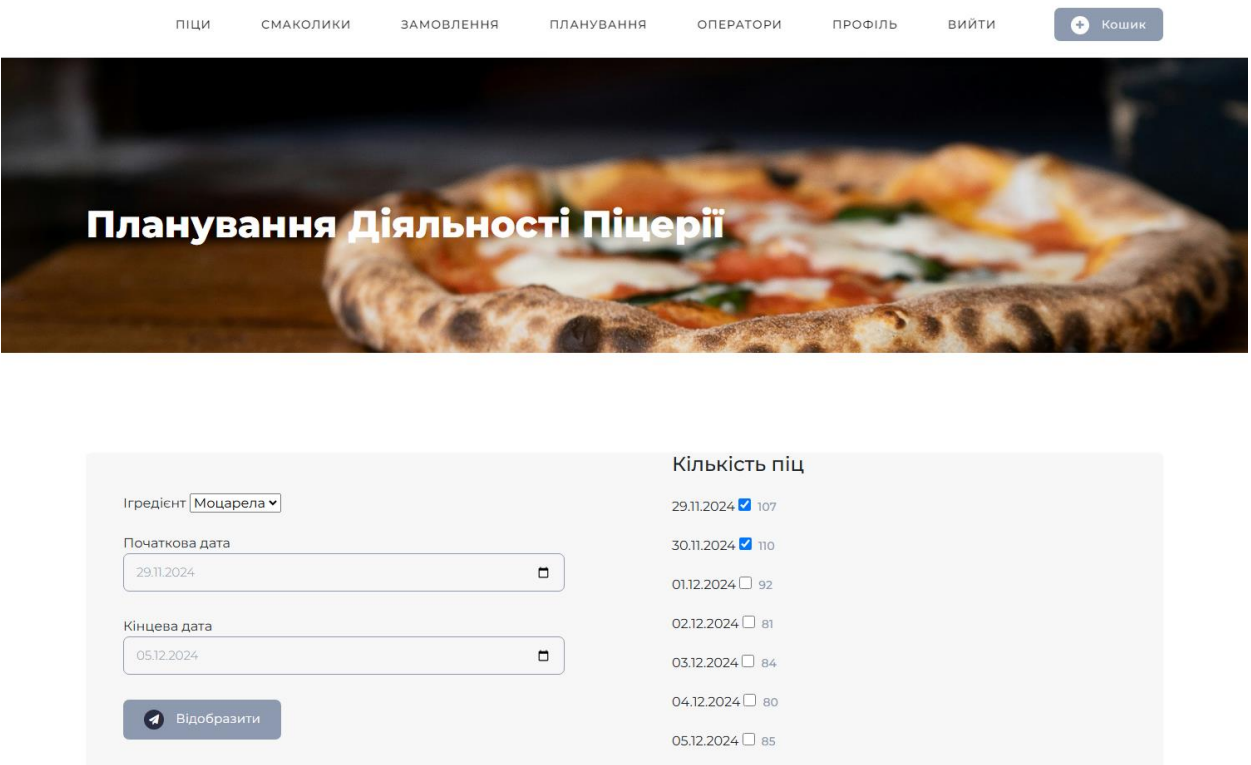


Рисунок 5.3 – Сторінка перегляду кількості замовлень піц у піцерії за період

Для кожної дати з заданого проміжку відображається кількість піц, проданих з цим інгредієнтом. Якщо біля кількості відображається галочка, то це означає, що це кількість піц, яка була зафіксована у самому вебзастосунку. Тобто це фактичні замовлення. Якщо галочка не встановлена, то це означає, що відповідне значення відображає результат прогнозування.

5.3 Експериментальне дослідження прогнозування кількості проданих піц за групами інгредієнтів

Експериментальне дослідження прогнозування кількості проданих піц за групами інгредієнтів виконано за допомогою відкритої вибірки даних [9] з оцінюванням отриманих результатів за показниками Mean Absolute Error (MAE), Mean Squared Error (MSE) та Root Mean Squared Error (RMSE).

Спочатку було виконано прогнозування кількості проданих піц з вмістом нарізаної шинки (табл. 5.1) та ананасу (табл. 5.2).

Таблиця 5.1 – Результати прогнозування кількості проданих піц з вмістом нарізаної шинки

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,18733839	0,25579459	0,06543087
Дерево рішень	0,27398721	0,36785637	0,13531831
Метод опорних векторів	0,18840699	0,26141713	0,06833892

Таблиця 5.2 – Результати прогнозування кількості проданих піц з вмістом ананасу

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14705455	0,21381892	0,04571853
Дерево рішень	0,19044776	0,2524625	0,06373731
Метод опорних векторів	0,14883056	0,21697285	0,04707722

Далі було виконано прогнозування кількості проданих піц з вмістом моцарели (табл. 5.3), пепероні (табл. 5.4) та грибів (табл. 5.5).

Таблиця 5.3 – Результати прогнозування кількості проданих піц з вмістом моцарели

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,11964441	0,16174376	0,02616104
Дерево рішень	0,18742354	0,26444252	0,06992985
Метод опорних векторів	0,11817650	0,16042699	0,02573682

Таблиця 5.4 – Результати прогнозування кількості проданих піц з вмістом пепероні

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,10748197	0,14998583	0,02249575
Дерево рішень	0,15243455	0,20572145	0,04232131
Метод опорних векторів	0,11462365	0,15843903	0,02510293

Таблиця 5.5 – Результати прогнозування кількості проданих піц з вмістом грибів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,11268758	0,16422791	0,02697081
Дерево рішень	0,15223881	0,19271294	0,03713828
Метод опорних векторів	0,11024943	0,16423002	0,0269715

Далі було виконано прогнозування кількості проданих піц з вмістом червоної цибулі (табл. 5.6), червоного перця (табл. 5.7) та бекону (табл. 5.8).

Таблиця 5.6 – Результати прогнозування кількості проданих піц з вмістом червоної цибулі

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1202379	0,18314009	0,03354029
Дерево рішень	0,17057569	0,25873421	0,06694339
Метод опорних векторів	0,1164482	0,17775939	0,0315984

Таблиця 5.7 – Результати прогнозування кількості проданих піц з вмістом червоного перця

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12593487	0,18163767	0,03299224
Дерево рішень	0,15760483	0,23145046	0,05356931
Метод опорних векторів	0,12084039	0,17192656	0,02955874

Таблиця 5.8 – Результати прогнозування кількості проданих піц з вмістом бекону

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12137203	0,16419526	0,02696008
Дерево рішень	0,16364606	0,23075483	0,05324779
Метод опорних векторів	0,12072999	0,16220018	0,0263089

Далі було виконано прогнозування кількості проданих піц з вмістом копченого сиру Гауда (табл. 5.9), сиру Романо (табл. 5.10) та блакитного сиру (табл. 5.11).

Таблиця 5.9 – Результати прогнозування кількості проданих піц з вмістом копченого сиру Гауда

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1624265	0,222729	0,04960821
Дерево рішень	0,3	0,37833098	0,14313433
Метод опорних векторів	0,17225018	0,23122236	0,05346378

Таблиця 5.10 – Результати прогнозування кількості проданих піц з вмістом сиру Романо

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1624265	0,222729	0,04960821
Дерево рішень	0,25074627	0,33768306	0,11402985
Метод опорних векторів	0,17225018	0,23122236	0,05346378

Таблиця 5.11 – Результати прогнозування кількості проданих піц з вмістом блакитного сиру

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1624265	0,222729	0,04960821
Дерево рішень	0,28208955	0,36096885	0,13029851
Метод опорних векторів	0,17225018	0,23122236	0,05346378

Далі було виконано прогнозування кількості проданих піц з вмістом калабрезької салями (табл. 5.12), капоколло (табл. 5.13) та помідорів (табл. 5.14).

Таблиця 5.12 – Результати прогнозування кількості проданих піц з вмістом калабрезької салямі

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,11914456	0,15651928	0,02449829
Дерево рішень	0,18706468	0,24215725	0,05864013
Метод опорних векторів	0,13494916	0,17374576	0,03018759

Таблиця 5.13 – Результати прогнозування кількості проданих піц з вмістом капоколло

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,15326062	0,20621909	0,04252631
Дерево рішень	0,23554104	0,30444373	0,09268598
Метод опорних векторів	0,1406497	0,19255463	0,03707729

Таблиця 5.14 – Результати прогнозування кількості проданих піц з вмістом помідорів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,11816775	0,18046561	0,03256784
Дерево рішень	0,14464264	0,21067152	0,04438249
Метод опорних векторів	0,11232453	0,16644356	0,02770346

Далі було виконано прогнозування кількості проданих піц з вмістом зелених оливков (табл. 5.15), перця Халапеньо (табл. 5.16) та кінзи (табл. 5.17).

Таблиця 5.15 – Результати прогнозування кількості проданих піц з вмістом зелених оливок

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14253885	0,1875646	0,03518048
Дерево рішень	0,22182192	0,28488213	0,08115783
Метод опорних векторів	0,14676638	0,18980452	0,03602576

Таблиця 5.16 – Результати прогнозування кількості проданих піц з вмістом перця Халапеньо

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,15176381	0,20924941	0,04378532
Дерево рішень	0,21569571	0,28836337	0,08315343
Метод опорних векторів	0,15384151	0,21112995	0,04457586

Таблиця 5.17 – Результати прогнозування кількості проданих піц з вмістом кінзи

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16797192	0,22579915	0,05098526
Дерево рішень	0,22537313	0,30062125	0,09037313
Метод опорних векторів	0,17067531	0,23466077	0,05506568

Далі було виконано прогнозування кількості проданих піц з вмістом кукурудзи (табл. 5.18), соусу Чипотле (табл. 5.19) та курки (табл. 5.20).

Таблиця 5.18 – Результати прогнозування кількості проданих піц з вмістом кукурудзи

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16797193	0,22579915	0,05098526
Дерево рішень	0,22238806	0,29422298	0,08656716
Метод опорних векторів	0,17067531	0,23466077	0,05506568

Таблиця 5.19 – Результати прогнозування кількості проданих піц з вмістом соусу Чипотле

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16797192	0,22579915	0,05098526
Дерево рішень	0,22910448	0,31021177	0,09623134
Метод опорних векторів	0,17067531	0,23466077	0,05506568

Таблиця 5.20 – Результати прогнозування кількості проданих піц з вмістом курки

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14913777	0,20566227	0,04229697
Дерево рішень	0,2250287	0,29029798	0,08427292
Метод опорних векторів	0,15412563	0,21123404	0,04461982

Далі було виконано прогнозування кількості проданих піц з вмістом тайського солодкого соусу чилі (табл. 5.21), прошутто ді Сан Даніеле (табл. 5.22) та руколи (табл. 5.23).

Таблиця 5.21 – Результати прогнозування кількості проданих піц з вмістом тайського солодкого соусу чилі

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,17514733	0,22292569	0,04969587
Дерево рішень	0,23995408	0,28889176	0,08345845
Метод опорних векторів	0,17887792	0,23070791	0,05322614

Таблиця 5.22 – Результати прогнозування кількості проданих піц з вмістом прошутто ді Сан Даніеле

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12866811	0,15783656	0,02491238
Дерево рішень	0,18588874	0,24358046	0,05933144
Метод опорних векторів	0,13838002	0,17102856	0,02925077

Таблиця 5.23 – Результати прогнозування кількості проданих піц з вмістом руколи

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12866811	0,15783656	0,02491238
Дерево рішень	0,19131615	0,24408634	0,05957814
Метод опорних векторів	0,13838002	0,17102856	0,02925077

Далі було виконано прогнозування кількості проданих піц з вмістом курки барбекю (табл. 5.24), зеленого перцю (табл. 5.25) та соусу барбекю (табл. 5.26).

Таблиця 5.24 – Результати прогнозування кількості проданих піц з вмістом курки барбекю

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16722069	0,21870553	0,04783211
Дерево рішень	0,24776119	0,30895963	0,09545605
Метод опорних векторів	0,1718916	0,2189033	0,04791866

Таблиця 5.25 – Результати прогнозування кількості проданих піц з вмістом зеленого перцю

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14667881	0,18818032	0,03541183
Дерево рішень	0,20149254	0,25007462	0,06253731
Метод опорних векторів	0,14744337	0,19119667	0,03655617

Таблиця 5.26 – Результати прогнозування кількості проданих піц з вмістом соусу барбекю

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16722069	0,21870553	0,04783211
Дерево рішень	0,23283582	0,2884165	0,08318408
Метод опорних векторів	0,1718916	0,2189033	0,04791866

Далі було виконано прогнозування кількості проданих піц з вмістом оливков Каламата (табл. 5.27), печені з яловичини (табл. 5.28) та шпинату (табл. 5.29).

Таблиця 5.27 – Результати прогнозування кількості проданих піц з вмістом оливков Каламата

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14870161	0,19448113	0,03782291
Дерево рішень	0,24925373	0,33812477	0,11432836
Метод опорних векторів	0,14904809	0,1915918	0,03670742

Таблиця 5.28 – Результати прогнозування кількості проданих піц з вмістом печені з яловичини

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16317365	0,19977359	0,03990949
Дерево рішень	0,22116689	0,28467714	0,08104108
Метод опорних векторів	0,17298243	0,21607575	0,04668873

Таблиця 5.29 – Результати прогнозування кількості проданих піц з вмістом шпинату

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12016457	0,16792032	0,02819723
Дерево рішень	0,18960752	0,24665317	0,06083779
Метод опорних векторів	0,11687563	0,15481727	0,02396839

Далі було виконано прогнозування кількості проданих піц з вмістом артишоку (табл. 5.30), сиру Азіаго (табл. 5.31) та козячого сиру (табл. 5.32).

Таблиця 5.30 – Результати прогнозування кількості проданих піц з вмістом артишоку

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14753398	0,18992981	0,03607333
Дерево рішень	0,24747232	0,34908007	0,1218569
Метод опорних векторів	0,14624243	0,19201598	0,03687014

Таблиця 5.31 – Результати прогнозування кількості проданих піц з вмістом сиру Азіаго

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14853971	0,19394755	0,03761565
Дерево рішень	0,25820896	0,32195763	0,10365672
Метод опорних векторів	0,14684109	0,19769092	0,0390817

Таблиця 5.32 – Результати прогнозування кількості проданих піц з вмістом козячого сиру

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14320041	0,19426232	0,03773785
Дерево рішень	0,22238806	0,28651274	0,08208955
Метод опорних векторів	0,15882235	0,20928271	0,04379925

Далі було виконано прогнозування кількості проданих піц з вмістом орегано (табл. 5.33), пеперончіні Верді (табл. 5.34) та в'ялених помідорів (табл. 5.35).

Таблиця 5.33 – Результати прогнозування кількості проданих піц з вмістом орегано

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14284413	0,19605598	0,03843795
Дерево рішень	0,20149254	0,25634567	0,0657131
Метод опорних векторів	0,15383901	0,20461054	0,04186547

Таблиця 5.34 – Результати прогнозування кількості проданих піц з вмістом пеперончіні Верді

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16045637	0,20118388	0,04047495
Дерево рішень	0,21469575	0,28083095	0,07886602
Метод опорних векторів	0,17498828	0,21654407	0,04689133

Таблиця 5.35 – Результати прогнозування кількості проданих піц з вмістом в'ялених помідорів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14252663	0,18599931	0,03459574
Дерево рішень	0,20042644	0,25945419	0,06731648
Метод опорних векторів	0,16058314	0,20126821	0,04050889

Далі було виконано прогнозування кількості проданих піц з вмістом соусу песто (табл. 5.36), цукіні (табл. 5.37) та сиру Фонтіна (табл. 5.38).

Таблиця 5.36 – Результати прогнозування кількості проданих піц з вмістом соусу песто

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,15633552	0,196852	0,03875071
Дерево рішень	0,24116261	0,30990858	0,09604333
Метод опорних векторів	0,14480268	0,1888248	0,0356548

Таблиця 5.37 – Результати прогнозування кількості проданих піц з вмістом цукіні

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1810669	0,22745222	0,05173451
Дерево рішень	0,25186567	0,296316	0,08780317
Метод опорних векторів	0,17211051	0,21332616	0,04550805

Таблиця 5.38 – Результати прогнозування кількості проданих піц з вмістом сиру Фонтіна

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,15609291	0,20144284	0,04057922
Дерево рішень	0,23566379	0,28683724	0,0822756
Метод опорних векторів	0,15285056	0,19924656	0,03969919

Далі було виконано прогнозування кількості проданих піц з вмістом сиру Гауда (табл. 5.39), італійської ковбаси (табл. 5.40) та ковбаси чорізо (табл. 5.41).

Таблиця 5.39 – Результати прогнозування кількості проданих піц з вмістом сиру Гауда

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,13160226	0,17354745	0,03011872
Дерево рішень	0,22914838	0,28393251	0,08061767
Метод опорних векторів	0,14476905	0,18383096	0,03379382

Таблиця 5.40 – Результати прогнозування кількості проданих піц з вмістом італійської ковбаси

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12472562	0,15744445	0,02478875
Дерево рішень	0,18308458	0,22095826	0,04882255
Метод опорних векторів	0,1403031	0,16774122	0,02813712

Таблиця 5.41 – Результати прогнозування кількості проданих піц з вмістом ковбаси чорізо

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,12472562	0,15744445	0,02478876
Дерево рішень	0,1880597	0,22790414	0,0519403
Метод опорних векторів	0,1403031	0,16774122	0,02813712

Далі було виконано прогнозування кількості проданих піц з вмістом салями Сопресата (табл. 5.42), сиру Рікотта (табл. 5.43) та сиру Горгонзола Піканте (табл. 5.44).

Таблиця 5.42 – Результати прогнозування кількості проданих піц з вмістом салямі Сопресата

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,14179582	0,17862817	0,03190802
Дерево рішень	0,2371476	0,31162005	0,09710706
Метод опорних векторів	0,15361941	0,19224739	0,03695906

Таблиця 5.43 – Результати прогнозування кількості проданих піц з вмістом сиру Рікотта

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1627658	0,20558244	0,04226414
Дерево рішень	0,2238806	0,28060463	0,07873896
Метод опорних векторів	0,17040463	0,21554671	0,04646039

Таблиця 5.44 – Результати прогнозування кількості проданих піц з вмістом сиру Горгонзола Піканте

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1627658	0,20558244	0,04226414
Дерево рішень	0,22921109	0,28823515	0,0830795
Метод опорних векторів	0,17040463	0,21554671	0,04646039

Далі було виконано прогнозування кількості проданих піц з вмістом Парміджано Реджано (табл. 5.45), анчоусів (табл. 5.46) та панчетти (табл. 5.47).

Таблиця 5.45 – Результати прогнозування кількості проданих піц з вмістом сиру Парміджано Реджано

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1627658	0,20558244	0,04226414
Дерево рішень	0,20895522	0,2726211	0,07432227
Метод опорних векторів	0,17040463	0,21554671	0,04646039

Таблиця 5.46 – Результати прогнозування кількості проданих піц з вмістом анчоусів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16069983	0,21050826	0,04431373
Дерево рішень	0,30348259	0,35475656	0,12585222
Метод опорних векторів	0,18722669	0,24688785	0,06095361

Таблиця 5.47 – Результати прогнозування кількості проданих піц з вмістом панчетти

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19037762	0,23784795	0,05657165
Дерево рішень	0,2750533	0,35036207	0,12275358
Метод опорних векторів	0,20006565	0,2503872	0,06269375

Далі було виконано прогнозування кількості проданих піц з вмістом перцю Фріджітелло (табл. 5.48), баклажанів (табл. 5.49) та сливових помідорів (табл. 5.50).

Таблиця 5.48 – Результати прогнозування кількості проданих піц з вмістом перцю Фріджітелло

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19037762	0,23784795	0,05657165
Дерево рішень	0,26652452	0,33616424	0,1130064
Метод опорних векторів	0,20006565	0,2503872	0,06269375

Таблиця 5.49 – Результати прогнозування кількості проданих піц з вмістом баклажанів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,17841411	0,23387971	0,05469972
Дерево рішень	0,24440299	0,31482345	0,09911381
Метод опорних векторів	0,18207928	0,23504601	0,05524663

Таблиця 5.50 – Результати прогнозування кількості проданих піц з вмістом сливових помідорів

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,18955322	0,23679145	0,05607019
Дерево рішень	0,26119403	0,32965948	0,10867537
Метод опорних векторів	0,18117374	0,22971379	0,05276842

Далі було виконано прогнозування кількості проданих піц з вмістом генуезького салямі (табл. 5.51), ковбасок Luganega (табл. 5.52) та цибулі (табл. 5.53).

Таблиця 5.51 – Результати прогнозування кількості проданих піц з вмістом генуезького салямї

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,17601598	0,21829171	0,04765127
Дерево рішень	0,23880597	0,30048684	0,09029234
Метод опорних векторів	0,18063258	0,21884264	0,0478921

Таблиця 5.52 – Результати прогнозування кількості проданих піц з вмістом ковбасок Luganega

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16994545	0,2152594	0,04633661
Дерево рішень	0,24339839	0,28504482	0,08125055
Метод опорних векторів	0,17453914	0,21298932	0,04536445

Таблиця 5.53 – Результати прогнозування кількості проданих піц з вмістом цибулі

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16994545	0,2152594	0,04633661
Дерево рішень	0,25258324	0,30883828	0,09538108
Метод опорних векторів	0,17453914	0,21298932	0,04536445

Далі було виконано прогнозування кількості проданих піц з вмістом соусу Альфредо (табл. 5.54), сиру Брі Карре (табл. 5.55) та прошутто (табл. 5.56).

Таблиця 5.54 – Результати прогнозування кількості проданих піц з вмістом соусом Альфредо

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,16061389	0,19407793	0,03766624
Дерево рішень	0,20232172	0,25898272	0,06707205
Метод опорних векторів	0,171284	0,20266723	0,04107401

Таблиця 5.55 – Результати прогнозування кількості проданих піц з вмістом сиру Брі Карре

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19762712	0,24963222	0,06231624
Дерево рішень	0,27761194	0,34812933	0,12119403
Метод опорних векторів	0,19670164	0,25125411	0,06312863

Таблиця 5.56 – Результати прогнозування кількості проданих піц з вмістом прошутто

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19762712	0,24963222	0,06231624
Дерево рішень	0,24477612	0,33143739	0,10985075
Метод опорних векторів	0,19670164	0,25125411	0,06312863

Далі було виконано прогнозування кількості проданих піц з вмістом карамелізованої цибулі (табл. 5.57), груші (табл. 5.58) та чебреця (табл. 5.59).

Таблиця 5.57 – Результати прогнозування кількості проданих піц з вмістом карамелізованої цибулі

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19762712	0,24963222	0,06231624
Дерево рішень	0,26567164	0,34294597	0,11761194
Метод опорних векторів	0,19670164	0,25125411	0,06312863

Таблиця 5.58 – Результати прогнозування кількості проданих піц з вмістом груші

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19762712	0,24963222	0,06231624
Дерево рішень	0,25970149	0,34294597	0,11761194
Метод опорних векторів	0,19670164	0,25125411	0,06312863

Таблиця 5.59 – Результати прогнозування кількості проданих піц з вмістом чебреця

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,19762712	0,24963222	0,06231624
Дерево рішень	0,26567164	0,34641016	0,12
Метод опорних векторів	0,19670164	0,25125411	0,06312863

Далі було виконано прогнозування кількості проданих піц з вмістом сиру проволоне (табл. 5.60), часнока (табл. 5.61).

Таблиця 5.60 – Результати прогнозування кількості проданих піц з вмістом сиру проволоне

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1624265	0,222729	0,04960821
Дерево рішень	0,26865672	0,34641016	0,12
Метод опорних векторів	0,17225018	0,23122236	0,05346378

Таблиця 5.61 – Результати прогнозування кількості проданих піц з вмістом часнока

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,1257666	0,190925	0,036452
Дерево рішень	0,19114107	0,26212501	0,06870952
Метод опорних векторів	0,125511	0,18113719	0,03281068

Окрім того були обчислені середні результати прогнозування кількості проданих піц за групами (табл. 5.62).

Таблиця 5.62 – Середні результати прогнозування кількості проданих піц за групами

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,155157689	0,203683452	0,042257167
Дерево рішень	0,225917901	0,291690097	0,086975576
Метод опорних векторів	0,159502846	0,208028502	0,044101856

Загалом використання моделі ARIMA в складі методу прогнозування кількості проданих піц за групами інгредієнтів дозволило отримати кращі результати з меншим рівнем похибки, оціненим одразу 3 окремими показниками. Це стосується і окремих випадків, і усереднених результатів за всіма групами. При цьому в середньому використання моделі ARIMA на 2,72 % покращило рівень MAE, на 2,09 % покращило рівень RMSE та на 4,19 % покращило рівень MSE порівняно з результатами методу опорних векторів. У середньому використання моделі ARIMA на 31,32 % покращило рівень MAE, на 30,17 % покращило рівень RMSE та на 51,41 % покращило рівень MSE порівняно з результатами дерева рішень.

5.4 Висновки за розділом 5

Розроблене програмне забезпечення піцерії призначене для забезпечення замовлення піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами, а відповідні основні сценарії роботи користувачів для цього представлені з відповідними інтерфейсами у цьому розділі. Окрім того проведено експериментальне дослідження прогнозування кількості проданих піц за групами інгредієнтів. Використання моделі ARIMA у складі методу прогнозування кількості проданих піц за групами інгредієнтів призвело до кращих результатів порівняно з методом опорних векторів, деревами рішень.

ВИСНОВКИ

У роботі здійснена розробка програмного забезпечення підтримки діяльності піцерії для забезпечення планування діяльності піцерії на основі прогнозування кількості проданих піц за певними групами. Програмне забезпечення може використовуватися для підтримки діяльності окремих піцерій стосовно підтримки роботи з замовленнями та планування подальших замовлень.

Для виконання розробки було виділено основні функції, виходячи з результатів огляду, встановлено відмінні можливості програми, що розроблялась, включаючи можливості прогнозування кількості проданих піц за групами інгредієнтів, оцінювання піц, уподобання піц, що відсутні у всіх інших програм.

Розроблене програмне забезпечення призначене для забезпечення замовлення піц, додатків до піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами. Для цього надаються засоби категоризації піц та додаткових продуктів, пошуку піц, вподобання та оцінювання піц та додаткових продуктів.

У програмне забезпечення, створене як вебзастосунок піцерії, було інтегровано модель прогнозування кількості проданих піц за групами інгредієнтів. Для розробки використано мову програмування Python, фреймворк Django, систему керування базами даних MySQL як основні засоби.

Наукова новизна роботи полягає в методі прогнозування кількості проданих піц за групами інгредієнтів, що характеризується розподілом піц на групи за конкретними інгредієнтами, застосуванням моделі ARIMA, що дозволяє визначати кількість піц з вмістом відповідного інгредієнта, яка буде продана в піцерії на певну дату.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Піца Celentano – твоя перша піцерія [Електрон. ресурс]. – Режим доступу : <https://pizzacelentano.com/>.
2. Доставка піци Львів до 29 хв. LA П'ЄЦ нормальна доставка їжі нормальна доставка їжі Львів [Електрон. ресурс]. – Режим доступу : <https://la.ua/>.
3. Domino's Pizza – Піцерія №1 в Києві. Доставка їжі (Піци та Напоїв) [Електрон. ресурс]. – Режим доступу : <https://dominos.ua/uk/kyiv/>.
4. Keste [Electronic resource] : Authentic Neapolitan Pizzeria in NYC – keste – Nationwide Shipping. – Access mode : <https://kestepizzeria.com/>.
5. Hayes, A. Autoregressive Integrated Moving Average (ARIMA) Prediction Model [Electronic resource] / A. Hayes. – Access mode : <https://www.investopedia.com/terms/a/autoregressive-integrated-moving-average-arima.asp>.
6. Django Tutorial [Electronic resource]. – Access mode : <https://www.w3schools.com/django/>.
7. Node.js v23.5.0 documentation [Electronic resource]. – Access mode : <https://nodejs.org/docs/latest/api/>
8. Free Template 564 Plot Listing [Electronic resource]. – Access mode : <https://templatemo.com/tm-564-plot-listing>.
9. Pizza Restaurant Sales [Electronic resource]. – Access mode : <https://www.kaggle.com/datasets/shilongzhuang/pizza-sales>.

ДОДАТОК А
Технічне завдання

Вступ

Формування замовлень піц у піцерії через програмні засоби є фактично обов'язковою умовою існування піцерії для досягнення успішних бізнес-результатів. Деякі піцерії можуть використовувати соціальні мережі для представлення меню, можливо формування замовлень, однак для контролю замовлень піц піцерією це достатньо незручно, відповідно можуть виникати проблеми з обслуговуванням, особливо коли таких замовлень піц стає багато. Тому створення вебзастосунку піцерії є важливим. Проте і розвиток такого програмного засобу з врахуванням тих можливостей, які б надали можливість опрацювання наявних даних, є також важливим.

Наявні дані у такій програмі звісно дозволяють проаналізувати, скільки яких саме піц було продано за кожен день роботи піцерії. Також відповідно можна і проаналізувати кількість піц з певними інгредієнтами, які було продано. Однак окрім аналізу статистики за цими параметрами на основі цих даних можна також і виконати прогнозування кількості проданих піц за певними групами. Наявність цього засобу дозволяє виконати планування діяльності піцерії. Тому важливо під час розробки вебзастосунку піцерії надати можливості і замовлення піц у піцерії, і планування діяльності піцерії на основі прогнозування кількості проданих піц за певними групами.

А.1 Підстави для розробки

Наказом № 398 від 7 жовтня 2024 року за Національним університетом «Запорізька політехніка» визначена тема дипломної кваліфікаційної роботи магістра «Дослідження та програмна реалізація методів планування діяльності піцерії», що обумовило отримання завдання на роботу.

A.2 Призначення розробки

Програма має бути призначена для забезпечення замовлення піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами.

A.3 Основні вимоги до програми

A.3.1 Вимоги до функціональних характеристик

У роботі шляхом реалізації програмного забезпечення необхідно надати наступні функціональні можливості:

- перегляд категорій піц;
- перегляд піц за категорією;
- перегляд інформації про піцу в меню піцерії;
- оформлення замовлення піц у піцерії;
- додавання піци до меню піцерії;
- редагування даних піци в меню піцерії;
- архівування піци з меню піцерії;
- авторизація і реєстрація клієнтів та менеджерів піцерії;
- додавання варіанта піци за існуючим пунктом меню;
- вилучення варіанта піци за існуючим пунктом меню;
- встановлення вартості піци на період;
- вилучення пропозиції вартості піци на період;
- перегляд всіх варіантів вартості піци, які існували або використовуються;
- перегляд категорій додаткових продуктів у піцерії за типом;
- перегляд додаткових продуктів у піцерії за категорією;
- перегляд інформації про додатковий продукт в меню піцерії;
- перегляд типів додаткових продуктів у піцерії;
- додавання додаткового продукту до меню піцерії;

- редагування додаткового продукту в меню піцерії;
- архівування додаткового продукту з меню піцерії;
- додавання варіанта обсягу додаткового продукту за існуючим пунктом меню;
- вилучення варіанта обсягу додаткового продукту за існуючим пунктом меню;
- встановлення вартості додаткового продукту на період;
- вилучення пропозиції вартості додаткового продукту на період;
- перегляд всіх варіантів вартості додаткового продукту, які існували або використовуються;
- перегляд замовлень піц у піцерії за період та за станами;
- перегляд кількості замовлень піц у піцерії за період за піцою в меню;
- перегляд власних замовлень піц у піцерії;
- підтвердження замовлення піц у піцерії;
- скасування замовлення піц у піцерії;
- визначення початку виготовлення піц у піцерії за замовленням;
- визначення готовності замовлення піц у піцерії;
- визначення початку доставки за замовленням піц у піцерії;
- визначення успішного або неуспішного завершення доставки за замовленням піц у піцерії;
- визначення видачі замовлення піц у піцерії;
- перегляд станів за замовленням піц у піцерії;
- уподобання піци;
- скасування вподобання піци;
- уподобання додаткового продукту в піцерії;
- скасування вподобання додаткового продукту в піцерії;
- перегляд вподобаних піц та додаткових продуктів користувачем;
- перегляд профілю користувача вебзастосунку піцерії;
- редагування даних користувача вебзастосунку піцерії;
- оцінювання піци в піцерії;

- оцінювання додаткового продукту в піцерії;
- пошук піци;
- перегляд замовлення піц у піцерії;
- перегляд оцінених піц та додаткових продуктів користувачем.

A.3.2 Умови експлуатації

Забезпечення експлуатації програмного забезпечення підтримки діяльності піцерії вимагає наявності адміністратора системи або делегування відповідних функцій іншому працівнику.

Адміністратор має налагодити роботу сервера та розгорнути на ньому вебзастосунок, підтримувати роботу з налаштуваннями у подальшому, а також вносити за необхідності базові дані каталогу в базу даних через адміністративну панель.

A.3.3 Вимоги до складу та параметрів технічних засобів

Враховуючи, що програмне забезпечення підтримки діяльності піцерії має бути створено у вигляді вебзастосунку, то вимоги до сервера є основними вимогами до технічних засобів, а відповідно мінімально включають двоядерний процесор, 4 Гб оперативної пам'яті, 100 Гб вільного обсягу жорсткого диску, щоби зберігати дані, які всі окрім файлів кодів вебзастосунку мають бути представлені базою даних, де велику частину обсягу потенційно мають займати фотографії.

A.4 Порядок контролю та приймання

Завдання на роботу визначає календарний план виконання роботи, включаючи періоди виконання, очікувані результати та консультантів, які забезпечують контроль виконання відповідних етапів.

У підсумку відбувається захист дипломної кваліфікаційної роботи у відведений для цього випусковою кафедрою програмних засобів період, де представляються остаточні результати з підтвердженням проходження всіх попередніх етапів.

ДОДАТОК Б
Текст програми

```

from django.shortcuts import render
from .models import Pizzacategory
from .models import Pizzamenu
from .models import Pizzaaddition_type
from .models import Pizzaaddition
from .models import Pizzaingredient
from .models import Pizzamenuphoto
from .models import Pizzasize
from .models import Pizzamenu_size
from .models import Pizzadought
from .models import Pizzaboard
from .models import Pizzavariant
from .models import Pizzaprice
from .models import Sidetype
from .models import Sidecategory
from .models import Sidemenu
from .models import Volumemarker
from .models import Sidevolume
from .models import Sideprice
from .models import Pizzeriaorder
from .models import Pizzeriaorder_pizza
from .models import Pizzeriaorder_side
from .models import Pizzeriaorder_addition
from .models import Pizzeriaorder_status
from .models import Pizzamenu_like
from .models import Sidemenu_like
from .models import Pizzamenu_choice
from .models import Sidemenu_choice
from .models import Pizzeriauser
from django.core.paginator import Paginator

```

```

#перегляд категорій піц
def pizza_categories_view(request):
    pizzacategories = Pizzacategory.objects.all()
    pizzacategories = pizzacategories.order_by("position")
    return render(request, "pizzaactorer/pizza_categories.html",
{"categories": pizzacategories })

```

```

#перегляд піц за категорією
def pizza_category_menus_view(request, pizzacategory_id):
    try:
        pizza_category =
Pizzacategory.objects.prefetch_related("pizzamenus").get(id =
pizzacategory_id)
    except Pizzacategory.DoesNotExist:

```

```

        pizza_category = None
    if not pizza_category:
        return render(request,
            "pizzaactorer/pizza_category_menus.html", {"category": None})
    else:
        if request.user.is_authenticated:
            user_authed = request.user
            pizzeria_user_authed = Pizzeriauser.objects.get(user
= user_authed)
            pizzamenu = Paginator(pizza_category.pizzamenu, 10)
            pizzamenu_page = request.GET.get('page')
            pizza_category.pizzamenu =
pizzamenu.get_page(pizzamenu_page)
            for pizza_menu in pizza_category.pizzamenu:
                pizza_menu.doughs =
Pizzamenu_dough.objects.filter(pizzamenu = pizza_menu)
                pizza_menu.boards =
Pizzamenu_board.objects.filter(pizzamenu = pizza_menu)
                pizza_menu.sizes =
Pizzamenu_size.objects.select_related("pizzasize").filter(pizzamenu =
pizza_menu)
                pizza_menu.variants =
Pizzavariant.objects.filter(pizzamenu_size__ pizzamenu = pizza_menu)
                pizza_menu.prices =
Pizzaprice.objects.select_related("pizzavariant").all()
                pizza_menu.photo =
Pizzamenu_photo.objects.filter(pizzamenu = pizza_menu)[0]
                pizza_menu.like = False
                if request.user.is_authenticated:
                    pizza_like =
Pizzamenu_like.objects.filter(pizzamenu = pizza_menu, pizzeriauser =
pizzeria_user_authed)
                    if pizza_like:
                        pizza_menu.like = True
                    else:
                        pizza_menu.like = False
            return render(request,
                "pizzaactorer/pizza_category_menus.html", {"category":
pizza_category})

```

```

#перегляд інформації про піцу в меню піцерії
def pizza_menu_view(request, pizzamenu_id):
    try:
        pizza_menu =
Pizzamenu.objects.prefetch_related("pizzaingredients", "pizzadoughs",

```

```

    "pizzaboards", "pizzaadditions",
    "pizzaadditions__pizzaaddition_type").get(id = pizzamenu_id)
    except Pizzamenu.DoesNotExist:
        pizza_menu = None
    if not pizza_menu:
        return render(request, "pizzaactorer/pizza_menu.html",
{"pizzamenu": None })
    else:
        pizza_menu.ingredients =
Pizzamenuingredient.objects.filter(pizzamenu = pizza_menu)
        pizza_menu.doughts =
Pizzamenudought.objects.filter(pizzamenu = pizza_menu)
        pizza_menu.boards =
Pizzamenuboard.objects.filter(pizzamenu = pizza_menu)
        pizza_menu.additions =
Pizzamenu_addition.objects.filter(pizzamenu = pizza_menu)
        pizza_menu.photos =
Pizzamenuphoto.objects.filter(pizzamenu = pizza_menu)
        pizza_additions = pizza_menu.pizzaadditions
        pizza_addition_types = []
        pizza_addition_types_additions = {}
        for pizza_addition in pizza_additions:
            if pizza_addition.pizzaaddition_type not in
pizza_addition_types:

pizza_addition_types.append(pizza_addition.pizzaaddition_type)

pizza_addition_types_additions[pizza_addition.pizzaaddition_type] =
[pizza_addition]
        else:

pizza_addition_types_additions[pizza_addition.pizzaaddition_type].appe
nd(pizza_addition)
        pizza_menu.pizza_addition_types = pizza_addition_types
        pizza_menu.pizza_addition_types_additions =
pizza_addition_types_additions
        pizza_menu.sizes =
Pizzamenu_size.objects.select_related("pizzasize").filter(pizzamenu =
pizza_menu)
        pizza_menu.variants =
Pizzavariant.objects.filter(pizzamenu_size__ pizzamenu = pizza_menu)
        pizza_menu.prices =
Pizzaprice.objects.select_related("pizzavariant").all()
        from datetime import datetime
        request_time = datetime.now()

```

```

        pizza_menu.prices =
pizza_menu.prices.filter(activefrom__lte = request_time, activeto__gte
= request_time)
        pizza_menu.like = False
        pizza_menu.choice = False
        if request.user.is_authenticated:
            user_authed = request.user
            pizzeria_user_authed = Pizzeriauser.objects.get(user
= user_authed)
            pizza_like = Pizzamenu_like.objects.filter(pizzamenu
= pizza_menu, pizzeriauser = pizzeria_user_authed)
            if pizza_like:
                pizza_menu.like = True
            else:
                pizza_menu.like = False
            pizza_choices =
Pizzamenu_choice.objects.filter(pizzamenu = pizza_menu, pizzeriauser =
pizzeria_user_authed)
            pizza_choice = pizza_choices.order_by("-
choice_time")[0]
            pizzeria_orders =
Pizzeriaorder_pizza.objects.filter(pizzavariant__pizzamenu__size__pizzam
enu = pizza_menu)
            pizzeria_order = pizzeria_orders.order_by("-
pizzeriaorder__order_planned")[0]
            if pizzeria_order.order_planned >
pizza_choice.choice_time:
                pizza_menu.choice = True
            return render(request, "pizzaactorer/pizza_menu.html",
{"pizzamenu": pizza_menu })

```

```

        #перегляд всіх варіантів вартості піци, які існували або
використовуються
        def pizza_prices_view(request, pizzamenu_id):
            try:
                pizza_menu = Pizzamenu.objects.prefetch_related.get(id =
pizzamenu_id)
            except Pizzamenu.DoesNotExist:
                pizza_menu = None
            if not pizza_menu:
                return render(request, "pizzaactorer/pizza_prices.html",
{"pizzamenu": None })
            else:
                pizza_menu.variants =
Pizzavariant.objects.filter(pizzamenu__size__pizzamenu = pizza_menu)
                pizzamenus = Paginator(pizza_menu.variants, 10)

```

```

        pizzamenu.page = request.GET.get('page')
        pizza_menu.variants =
pizzamenu.get_page(pizzamenu.page)
        for pizza_menu_variant in pizza_menu.variants:
            pizza_menu_variant.prices =
Pizzaprice.objects.filter(pizzavariant = pizza_menu_variant)
            pizza_menu_variant.prices =
pizza_menu_variant.prices.order_by("-activeto")
        return render(request, "pizzaactorer/pizza_prices.html",
{"pizzamenu": pizza_menu })

```

```

#перегляд категорій додаткових продуктів у піцерії за типом та
перегляд типів додаткових продуктів у піцерії
def side_type_categories_view(request, sidetype_id = None):
    side_types = Sidetype.objects.all()
    side_types = side_types.order_by("sidetype_name")
    if sidetype_id:
        try:
            side_type = Sidetype.objects.get(id = sidetype_id)
        except Sidetype.DoesNotExist:
            side_type = None
        if not side_type:
            return render(request,
"pizzaactorer/side_type.html", {"sidetypes": side_types, "categories":
[]})
        else:
            sidecategories =
Sidecategory.objects.filter(sidetype = side_type)
            sidecategories =
sidecategories.order_by("scateg_name")
            return render(request,
"pizzaactorer/side_type.html", {"sidetypes": side_types, "categories":
sidecategories})

```

```

#перегляд додаткових продуктів у піцерії за категорією
def sidecategory_sides_view(request, sidecategory_id):
    try:
        side_category =
Sidecategory.objects.prefetch_related("sidemenu").select_related("sid
etype").get(id = sidecategory_id)
    except Sidecategory.DoesNotExist:
        side_category = None
    if not side_category:
        return render(request,
"pizzaactorer/side_category.html", {"sidecategory": None})

```

```

    else:
        if request.user.is_authenticated:
            user_authed = request.user
            pizzeria_user_authed = Pizzeriauser.objects.get(user
= user_authed)
            pizzamenu = Paginator(side_category.sidemenus, 10)
            pizzamenu_page = request.GET.get('page')
            side_category.sidemenus =
pizzamenu.get_page(pizzamenu_page)
            for side_menu in side_category.sidemenus:
                side_menu.like = False
                if request.user.is_authenticated:
                    side_like =
Sidemenu_like.objects.filter(sidemenu = side_menu, pizzeriauser =
pizzeria_user_authed)
                    if side_like:
                        side_menu.like = True
                    else:
                        side_menu.like = False
            return render(request,
"pizzaactorer/side_category.html", {"sidecategory": side_category })

```

```

#перегляд інформації про додатковий продукт в меню піцерії
def side_menu_view(request, sidemenu_id):
    try:
        side_menu = Sidemenu.objects.get(id = sidemenu_id)
    except Sidemenu.DoesNotExist:
        side_menu = None
    if not side_menu:
        return render(request, "pizzaactorer/side_menu.html",
{"sidemenu": None })
    else:
        side_menu.volumes = Sidevolume.objects.filter(sidemenu =
side_menu)
        side_menu.prices =
Sideprice.objects.select_related("sidevolume").all()
        from datetime import datetime
        request_time = datetime.now()
        side_menu.prices =
side_menu.prices.filter(activefrom__lte = request_time, activeto__gte
= request_time)
        side_menu.like = False
        side_menu.choice = False
        if request.user.is_authenticated:
            user_authed = request.user

```

```

        pizzeria_user_authed = Pizzeriauser.objects.get(user
= user_authed)
        side_like = Sidemenu_like.objects.filter(sidemenu =
side_menu, pizzeriauser = pizzeria_user_authed)
        if side_like:
            side_like.like = True
        else:
            side_like.like = False
        side_choices =
Sidemenu_choice.objects.filter(sidemenu = side_menu, pizzeriauser =
pizzeria_user_authed)
        side_choice = side_choices.order_by("-
choice_time")[0]
        pizzeria_orders =
Pizzeriaorder_side.objects.filter(sidevolume__sidemenu = side_menu)
        pizzeria_order = pizzeria_orders.order_by("-
pizzeriaorder__order_planned")[0]
        if pizzeria_order.order_planned >
side_choice.choice_time:
            side_menu.choice = True
        return render(request, "pizzaactorer/side_menu.html",
{"sidemenu": side_menu })

```

```

#перегляд всіх варіантів вартості додаткового продукту, які
існували або використовуються
def side_prices_view(request, sidemenu_id):
    try:
        side_menu = Sidemenu.objects.get(id = sidemenu_id)
    except Sidemenu.DoesNotExist:
        side_menu = None
    if not side_menu:
        return render(request, "pizzaactorer/side_prices.html",
{"sidemenu": None })
    else:
        side_menu.volumes = Sidevolume.objects.filter(sidemenu =
side_menu)
        pizzamenus = Paginator(side_menu.volumes, 10)
        pizzamenuspage = request.GET.get('page')
        side_menu.volumes = pizzamenus.get_page(pizzamenuspage)
        for side_menu_volume in side_menu.volumes:
            side_menu_volume.prices =
Sideprice.objects.filter(sidevolume = side_menu_volume)
            side_menu_volume.prices =
side_menu_volume.prices.order_by("-activeto")
        return render(request, "pizzaactorer/side_prices.html",
{"sidemenu": side_menu })

```

```

#перегляд замовлень піц у піцерії за період та за станами
def pizzeria_orders_view(request, startperiod, endperiod, state
= -1):
    from datetime import datetime
    pizzeria_orders_startperiod = datetime.strptime(startperiod,
'%d.%m.%y %H:%M')
    pizzeria_orders_endperiod = datetime.strptime(endperiod,
'%d.%m.%y %H:%M')
    if request.user.is_authenticated and request.user.is_staff:
        pizzeria_orders =
Pizzeriaorder.objects.select_related("pizzeriauser").all()
        pizzeria_orders =
pizzeria_orders.filter(order_planned__gte =
pizzeria_orders_startperiod, order_planned__lte =
pizzeria_orders_endperiod)
        pizzamenu = Paginator(pizzeria_orders, 10)
        pizzamenu.page = request.GET.get('page')
        pizzeria_orders = pizzamenu.get_page(pizzamenu.page)
        if state != -1:
            pizzeria_status_orders = []
            for pizzeria_order in pizzeria_orders:
                pizzeria_order_last_status =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
                pizzeria_order_last_status =
pizzeria_order_last_status.order_by("-pizzeriaorder_time")
                pizzeria_order_last_status =
pizzeria_order_last_status[0]
                if
pizzeria_order_last_status.pizzeriaorder_status == state:

pizzeria_status_orders.append(pizzeria_order)
            return render(request,
"pizzaactorer/pizzeria_orders.html", {"orders": pizzeria_status_orders
})
        else:
            return render(request,
"pizzaactorer/pizzeria_orders.html", {"orders": pizzeria_orders })

#перегляд власних замовлень піц у піцерії
def own_pizzeria_orders_view(request, startperiod, endperiod,
state = -1):
    from datetime import datetime
    pizzeria_orders_startperiod = datetime.strptime(startperiod,
'%d.%m.%y %H:%M')
    pizzeria_orders_endperiod = datetime.strptime(endperiod,
'%d.%m.%y %H:%M')

```

```

        if request.user.is_authenticated:
            user_authed = request.user
            pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
            pizzeria_orders = Pizzeriaorder.objects.all()
            pizzeria_orders = pizzeria_orders.filter(pizzeriauser =
pizzeria_user_authed)
            pizzeria_orders =
pizzeria_orders.filter(order_planned__gte =
pizzeria_orders_startperiod, order_planned__lte =
pizzeria_orders_endperiod)
            pizzamenu = Paginator(pizzeria_orders, 10)
            pizzamenu_page = request.GET.get('page')
            pizzeria_orders = pizzamenu.get_page(pizzamenu_page)
            if state != -1:
                pizzeria_status_orders = []
                for pizzeria_order in pizzeria_orders:
                    pizzeria_order_last_status =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
                    pizzeria_order_last_status =
pizzeria_order_last_status.order_by("-pizzeriaorder_time")
                    pizzeria_order_last_status =
pizzeria_order_last_status[0]
                    if
pizzeria_order_last_status.pizzeriaorder_status == state:

pizzeria_status_orders.append(pizzeria_order)
                return render(request,
"pizzaactorer/opizzeria_orders.html", {"orders":
pizzeria_status_orders })
            else:
                return render(request,
"pizzaactorer/opizzeria_orders.html", {"orders": pizzeria_orders })

```

```

#перегляд станів за замовленням піц у піцерії
def pizzeriaorder_statuses_view(request, pizzeriaorder_id):
    try:
        pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
    except Pizzeriaorder.DoesNotExist:
        pizzeria_order = None
    if not pizzeria_order:
        return render(request,
"pizzaactorer/pizzeria_order_statuses.html", {"order": None,
"statuses": []})

```

```

        elif request.user.is_authenticated and
request.user.is_staff:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")
            return render(request,
"pizzaactorer/pizzeria_order_statuses.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses })
        else:
            return render(request,
"pizzaactorer/pizzeria_order_statuses.html", {"order": None,
"statuses": []})

```

```

#перегляд замовлення піц у піцерії
def pizzeriaorder_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        elif request.user.is_authenticated and
(request.user.is_staff or pizzeria_user_authed ==
pizzeria_order.pizzeriauser):
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
            pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu_size", "pizzavariant__pizzamenu_dough",
"pizzavariant__pizzamenu_board",
"pizzavariant__pizzamenu_size__pizzamenu",
"pizzavariant__pizzamenu_size__pizzasize",
"pizzavariant__pizzamenu_dough__pizzadough",
"pizzavariant__pizzamenu_board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)

```

```

        pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
        pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
        return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })
    else:
        return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})

#підтвердження замовлення піц у піцерії
def pizzeriaorder_approve_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        else:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
            pizzeria_order_last_status = pizzeria_order_statuses[0]
            if pizzeria_order_last_status.pizzeriaorder_status == 0
and request.user.is_authenticated and request.user.is_staff:
                pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 1)
                pizza_status.save()

```

```

        pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
        pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu__size", "pizzavariant__pizzamenu__dough",
"pizzavariant__pizzamenu__board",
"pizzavariant__pizzamenu__size__pizzamenu",
"pizzavariant__pizzamenu__size__pizzasize",
"pizzavariant__pizzamenu__dough__pizzadough",
"pizzavariant__pizzamenu__board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
        pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
        pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
        return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })

#скасування замовлення піц у піцерії
def pizzeriaorder_cancel_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        else:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]

```

```

        pizzeria_order_last_status = pizzeria_order_statuses[0]
        if (pizzeria_order_last_status.pizzeriaorder_status == 0
or pizzeria_order_last_status.pizzeriaorder_status == 1 or
pizzeria_order_last_status.pizzeriaorder_status == 3) and
((request.user.is_authenticated and request.user.is_staff) or
(pizzeria_user_authed == pizzeria_order.pizzeriauser):
            pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 2)
            pizza_status.save()
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu_size", "pizzavariant__pizzamenu_dough",
"pizzavariant__pizzamenu_board",
"pizzavariant__pizzamenu_size__pizzamenu",
"pizzavariant__pizzamenu_size__pizzasize",
"pizzavariant__pizzamenu_dough__pizzadough",
"pizzavariant__pizzamenu_board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
            pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
            pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })

```

```

#визначення початку виготовлення піц у піцерії за замовленням
def pizzeriaorder_start_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:

```

```

        pizzeria_order = None
    if not pizzeria_order:
        return render(request,
            "pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
            "pizzas": [], "sides": [], "additions": []})
    else:
        pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
        pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
        pizzeria_order_last_status = pizzeria_order_statuses[0]
        if pizzeria_order_last_status.pizzeriaorder_status == 1
and request.user.is_authenticated and request.user.is_staff:
            pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 3)
            pizza_status.save()
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu_size", "pizzavariant__pizzamenu_dough",
"pizzavariant__pizzamenu_board",
"pizzavariant__pizzamenu_size__pizzamenu",
"pizzavariant__pizzamenu_size__pizzasize",
"pizzavariant__pizzamenu_dough__pizzadough",
"pizzavariant__pizzamenu_board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
            pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
            pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
            return render(request,
            "pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
            "statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
            "sides": pizzeria_sides, "additions": pizzeria_additions })

#Визначення готовності замовлення піц у піцерії
def pizzeriaorder_done_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user

```

```

        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
            except Pizzeriaorder.DoesNotExist:
                pizzeria_order = None
            if not pizzeria_order:
                return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
            else:
                pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
                pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
                pizzeria_order_last_status = pizzeria_order_statuses[0]
                if pizzeria_order_last_status.pizzeriaorder_status == 3
and request.user.is_authenticated and request.user.is_staff:
                    pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 4)
                    pizza_status.save()
                    pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
                    pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu_size", "pizzavariant__pizzamenu_dough",
"pizzavariant__pizzamenu_board",
"pizzavariant__pizzamenu_size__pizzamenu",
"pizzavariant__pizzamenu_size__pizzasize",
"pizzavariant__pizzamenu_dough__pizzadough",
"pizzavariant__pizzamenu_board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
                    pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
                    pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
                    return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,

```

```
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })
```

```
#Визначення початку доставки за замовленням піц у піцерії
def pizzeriaorder_delivery_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        else:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
            pizzeria_order_last_status = pizzeria_order_statuses[0]
            if pizzeria_order_last_status.pizzeriaorder_status == 4
and request.user.is_authenticated and request.user.is_staff:
                pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 5)
                pizza_status.save()
                pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
                pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu", "pizzavariant__pizzamenu__pizzamenu",
"pizzavariant__pizzamenu__pizzamenu__pizzasize",
"pizzavariant__pizzamenu__pizzamenu__pizzadough",
"pizzavariant__pizzamenu__pizzamenu__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
                pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
```

```

        pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
        return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })

#визначення успішного або неуспішного завершення доставки за
замовленням піц у піцерії
def pizzeriaorder_end_view(request, pizzeriaorder_id, endvariant
= 0):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        else:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
            pizzeria_order_last_status = pizzeria_order_statuses[0]
            if (pizzeria_order_last_status.pizzeriaorder_status == 5
or (pizzeria_order_last_status.pizzeriaorder_status == 4 and
endvariant == 1)) and request.user.is_authenticated and
request.user.is_staff:
                if not endvariant:
                    pizza_status =
Pizzeriaorder_status(pizzeriaorder = pizzeria_order, pizzeriauser =
pizzeria_user_authed, pizzeriaorder_status = 6)
                else:
                    pizza_status =
Pizzeriaorder_status(pizzeriaorder = pizzeria_order, pizzeriauser =
pizzeria_user_authed, pizzeriaorder_status = 7)

```

```

        pizza_status.save()
        pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
        pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu__size", "pizzavariant__pizzamenu__dough",
"pizzavariant__pizzamenu__board",
"pizzavariant__pizzamenu__size__pizzamenu",
"pizzavariant__pizzamenu__size__pizzasize",
"pizzavariant__pizzamenu__dough__pizzadough",
"pizzavariant__pizzamenu__board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
        pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
        pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
        return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })

```

```

#визначення видачі замовлення піц у піцерії
def pizzeriaorder_get_view(request, pizzeriaorder_id):
    if request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
        try:
            pizzeria_order =
Pizzeriaorder.objects.select_related("pizzeriauser").get(id =
pizzeriaorder_id)
        except Pizzeriaorder.DoesNotExist:
            pizzeria_order = None
        if not pizzeria_order:
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": None, "statuses": [],
"pizzas": [], "sides": [], "additions": []})
        else:
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)

```

```

        pizzeria_order_statuses =
pizzeria_order_statuses.order_by("-pizzeriaorder_time")[:3]
        pizzeria_order_last_status = pizzeria_order_statuses[0]
        if pizzeria_order_last_status.pizzeriaorder_status == 4
and request.user.is_authenticated and request.user.is_staff:
            pizza_status = Pizzeriaorder_status(pizzeriaorder =
pizzeria_order, pizzeriauser = pizzeria_user_authed,
pizzeriaorder_status = 8)
            pizza_status.save()
            pizzeria_order_statuses =
Pizzeriaorder_status.objects.filter(pizzeriaorder = pizzeria_order)
            pizzeria_pizzas =
Pizzeriaorder_pizza.objects.select_related("pizzavariant",
"pizzavariant__pizzamenu_size", "pizzavariant__pizzamenu_dough",
"pizzavariant__pizzamenu_board",
"pizzavariant__pizzamenu_size__pizzamenu",
"pizzavariant__pizzamenu_size__pizzasize",
"pizzavariant__pizzamenu_dough__pizzadough",
"pizzavariant__pizzamenu_board__pizzaboard").filter(pizzeriaorder =
pizzeria_order)
            pizzeria_sides =
Pizzeriaorder_side.objects.select_related("sidevolume",
"sidevolume__sidemenu",
"sidevolume__volumemarker").filter(pizzeriaorder = pizzeria_order)
            pizzeria_additions =
Pizzeriaorder_addition.objects.select_related("pizzamenu_addition",
"pizzamenu_addition__pizzamenu", "pizzamenu_addition__pizzaaddition",
"pizzamenu_addition__pizzaaddition__pizzaaddition_type").filter(pizzer
iaorder = pizzeria_order)
            return render(request,
"pizzaactorer/pizzeria_order.html", {"order": pizzeria_order,
"statuses": pizzeria_order_statuses, "pizzas": pizzeria_pizzas,
"sides": pizzeria_sides, "additions": pizzeria_additions })

```

```

#перегляд оцінених піц та додаткових продуктів користувачем
def pizza_and_sides_likes_view(request, pizzeria_user = None):
    user_authed = None
    if not pizzeria_user and request.user.is_authenticated:
        user_authed = request.user
        pizzeria_user_authed = Pizzeriauser.objects.get(user =
user_authed)
    elif request.user.is_authenticated and
request.user.is_staff:
        user_authed = request.user
        pizzeria_user_authed =
Pizzeriauser.objects.get(user__username = pizzeria_user)

```

```

        if not user_authed:
            return render(request,
                "pizzaactorer/liked_pizzas_and_sides.html", {"pizzas": [], "sides":
                [], "user": None})
        else:
            pizza_choices =
Pizzamenu_choice.objects.select_related("pizzamenu").filter(pizzeriauser = user_authed)
            pizza_choices = pizza_choices.order_by("-choice_time")
            pizzamenus = Paginator(pizza_choices, 10)
            pizzamenuspage = request.GET.get('page')
            pizza_choices = pizzamenus.get_page(pizzamenuspage)
            for pizza_choice in pizza_choices:
                pizza_choice.photo =
Pizzamenuphoto.objects.filter(pizzamenu = pizza_choice.pizzamenu)[0]
                pizza_choice.like = False
                if not pizzeria_user:
                    pizza_like =
Pizzamenu_like.objects.filter(pizzamenu = pizza_choice.pizzamenu,
pizzeriauser = user_authed)
                    if pizza_like:
                        pizza_choice.like = True
            side_choices =
Sidemenu_choice.objects.select_related("sidemenu").filter(pizzeriauser
= user_authed)
            side_choices = side_choices.order_by("-choice_time")
            pizzamenus = Paginator(side_choices, 10)
            pizzamenuspage = request.GET.get('page')
            side_choices = pizzamenus.get_page(pizzamenuspage)
            if not pizzeria_user:
                for side_choice in side_choices:
                    side_choice.like = False
                    side_like =
Sidemenu_like.objects.filter(sidemenu = side_choice.pizzamenu,
pizzeriauser = user_authed)
                    if side_like:
                        side_choice.like = True
            return render(request,
                "pizzaactorer/pizza_category_menus.html", {"pizzas": pizza_choices,
                "sides": side_choices, "user": user_authed })

```

ДОДАТОК В
Слайди презентації

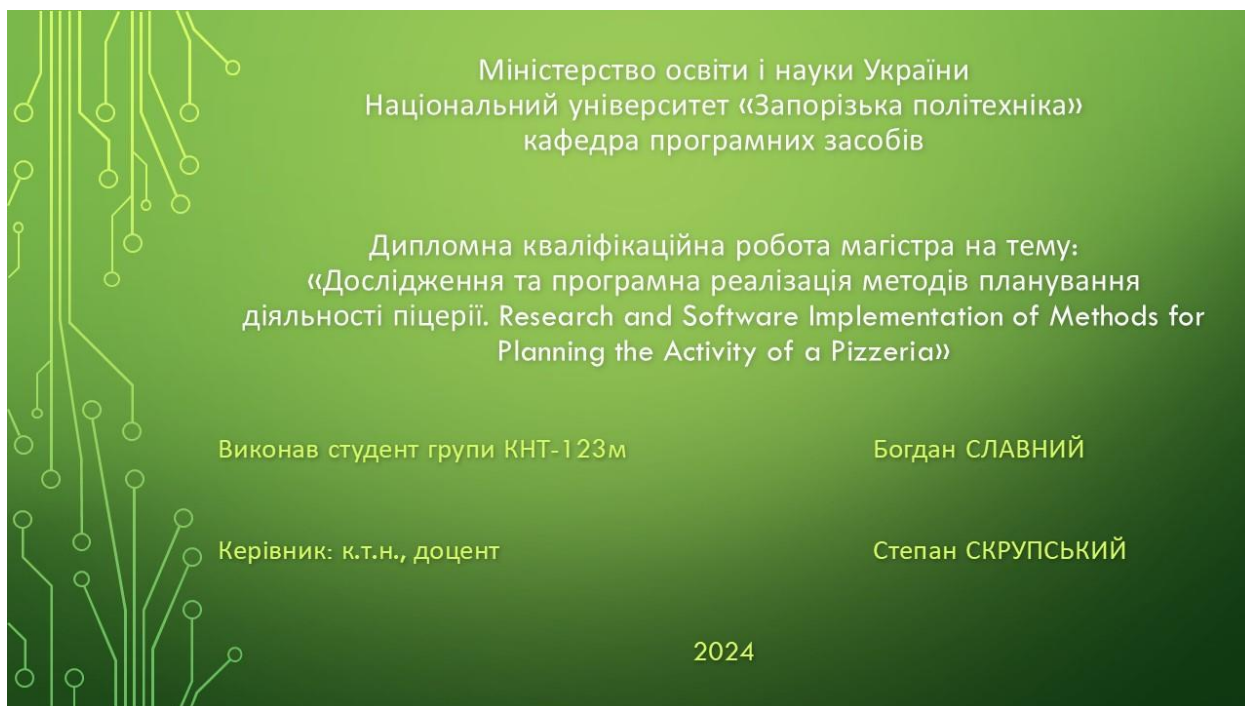


Рисунок В.1 – Початковий слайд

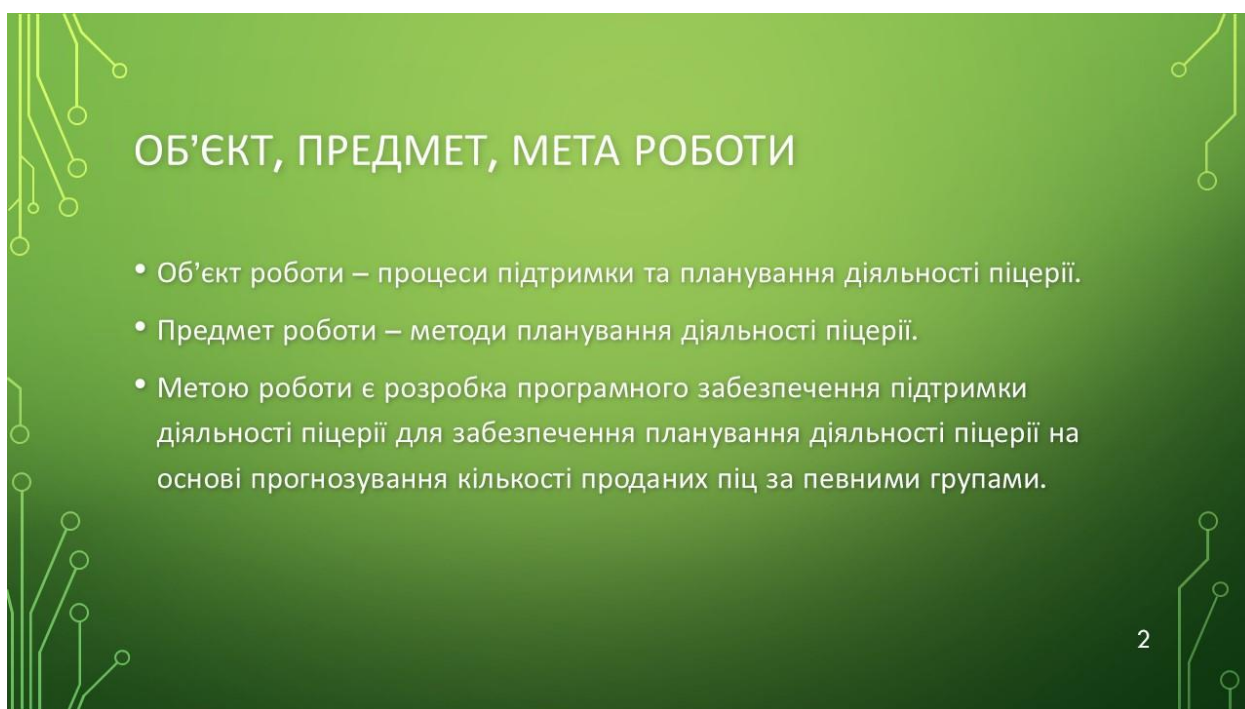


Рисунок В.2 – Об'єкт, предмет, мета роботи

ЗАВДАННЯ РОБОТИ

- виконати аналіз програмних застосунків забезпечення діяльності піцерій;
- розробити метод прогнозування кількості проданих піц за групами;
- виконати проектування вебзастосунку піцерії;
- реалізувати вебзастосунок піцерії;
- виконати експериментальне дослідження прогнозування кількості проданих піц за групами.

3

Рисунок В.3 – Завдання роботи

ПОРІВНЯННЯ МОЖЛИВОСТЕЙ АНАЛОГІВ РОЗРОБКИ ВЕБЗАСТОСУНКУ ПІЦЕРІЇ

Показник	Celentano	La P'ezz	Domino's	Розробка
Замовлення піц	+	+	+	+
Вибір доступних варіантів піц	-	-	+	+
Вибір додатків до піц	-	+	+	+
Уподобання піц	-	-	-	+
Оцінювання піц	-	-	-	+
Прогнозування кількості проданих піц за групами	-	-	-	+

4

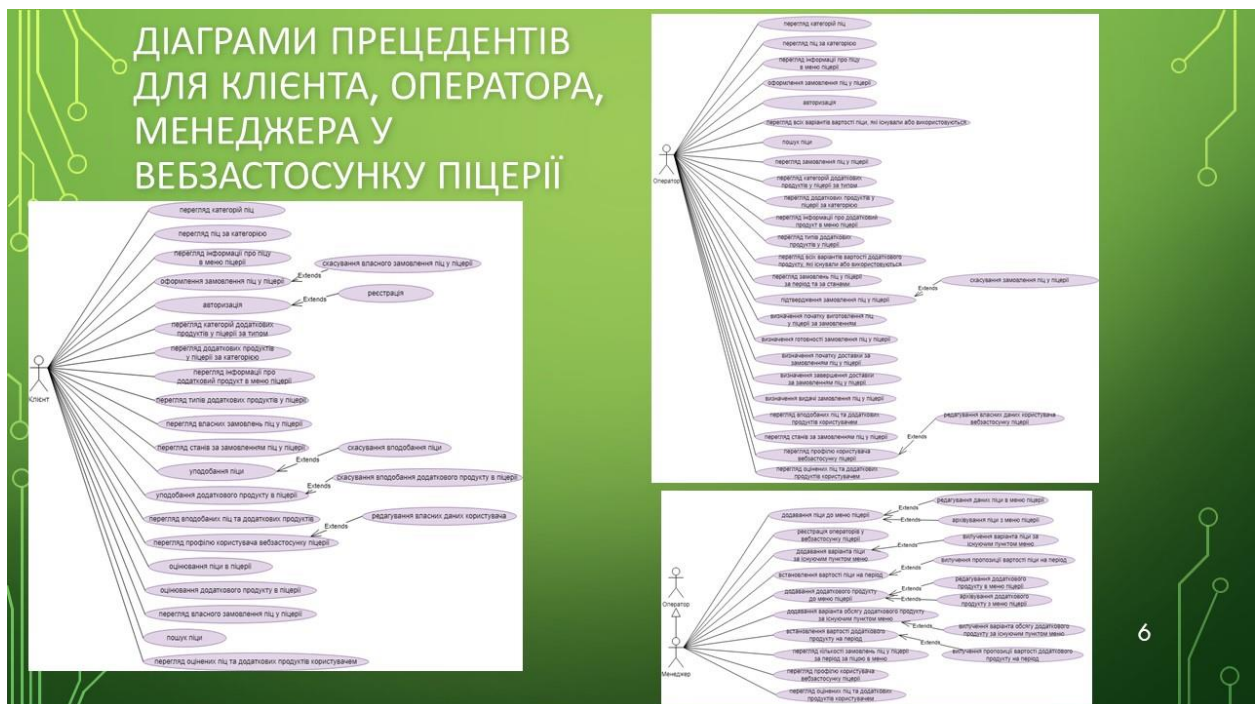
Рисунок В.4 – Порівняння можливостей аналогів розробки вебзастосунку піцерії

МЕТОД ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПРОДАНИХ ПІЦ ЗА ГРУПАМИ ІНГРЕДІЄНТІВ

- встановлення переліку інгредієнтів, за якими необхідно згрупувати піци, та встановлення для переліку інгредієнтів для кожного замовлення кількості входжень;
- формування часових рядів за кожним інгредієнтом за доступними датами замовлень з виконанням нормування та акумуляцією значень за всіма замовленнями за датами;
- створення моделей ARIMA за інгредієнтами та навчання кожної такої моделі на основі відповідної навчальної вибірки даних, сформованої з відповідних часових рядів;
- формування екземплярів даних, які характеризують новий день за кожним інгредієнтом, та прогнозування на основі навчених моделей.

5

Рисунок В.5 – Метод прогнозування кількості проданих піц за групами інгредієнтів



6

Рисунок В.6 – Діаграми прецедентів для клієнта, оператора, менеджера у вебзастосунку піцерії

8

	Django	Node.js
Базова мова програмування	Python	JavaScript
Продуктивність	Висока для практичного використання	Вища
Час, що витрачається на розробку	Менше	Більше
Рівень безпеки	Вищий	Дещо нижчий
Рівень засобів машинного навчання, роботи з даними	Вищий	Нижчий

Рисунок В.7 – Порівняння можливих рішень для реалізації вебзастосунку
піцерії

8

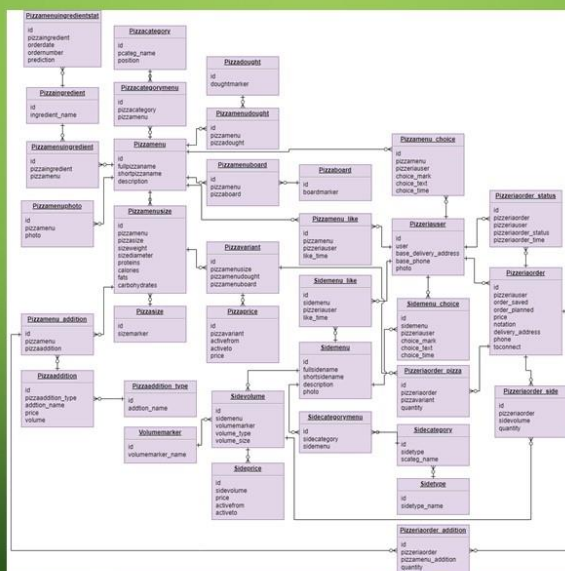
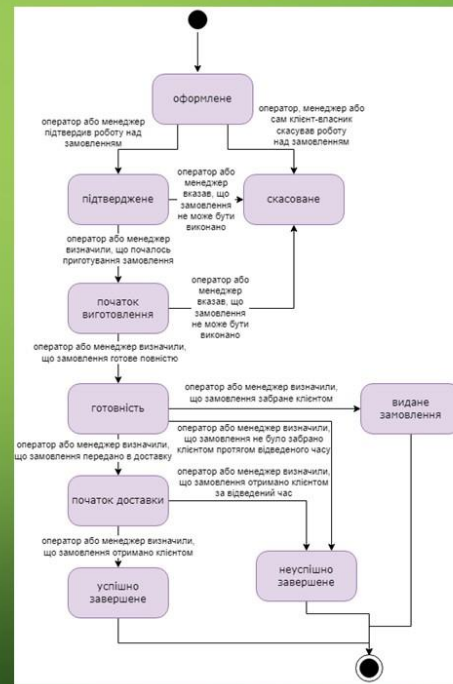


Рисунок В.8 Схема бази даних вебзастосунку піцерії

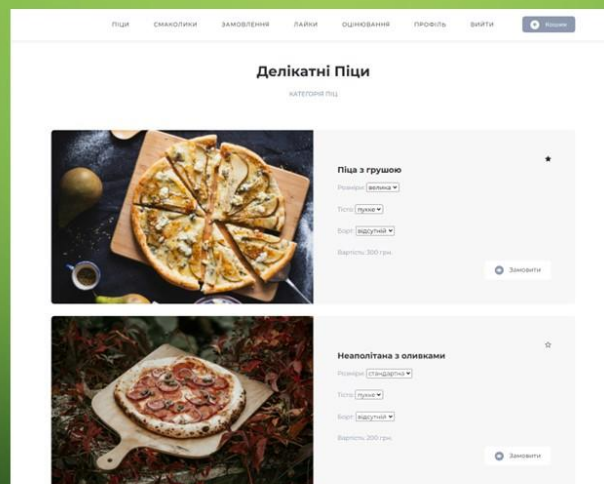
ДІАГРАМА СТАНІВ ЗАМОВЛЕННЯ ПІЦ У ПІЦЕРІЇ



9

Рисунок В.9 – Діаграма станів замовлення піц у піцерії

СТОРІНКА ПІЦ ЗА КАТЕГОРІЄЮ



10

Рисунок В.10 – Сторінка піц за категорією

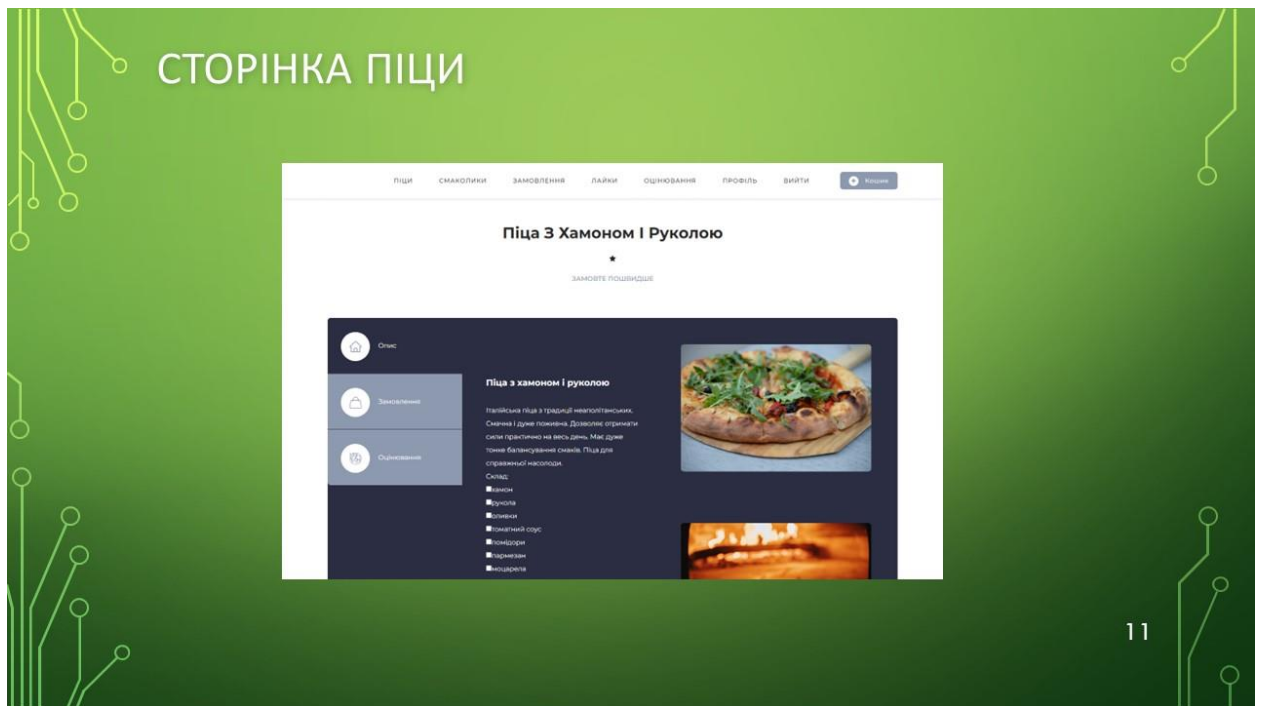


Рисунок В.11 – Сторінка піци

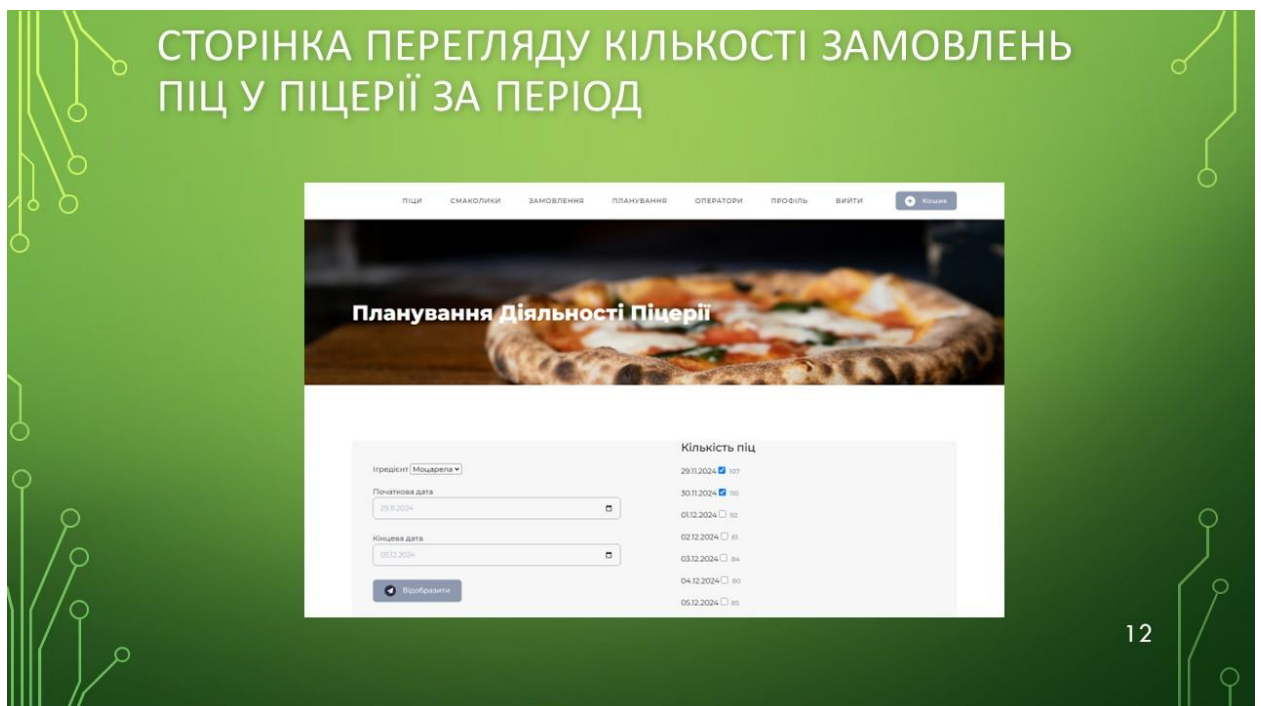


Рисунок В.12 – Сторінка перегляду кількості замовлень піц у піцерії за період

ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГНОЗУВАННЯ КІЛЬКОСТІ ПРОДАНИХ ПІЦ ЗА ГРУПАМИ ІНГРЕДІЄНТІВ

Модель прогнозування	MAE	RMSE	MSE
ARIMA	0,155157689	0,203683452	0,042257167
Дерево рішень	0,225917901	0,291690097	0,086975576
Метод опорних векторів	0,159502846	0,208028502	0,044101856

13

Рисунок В.13 – Експериментальне дослідження прогнозування кількості проданих піц за групами інгредієнтів

ВИСНОВКИ

- Розроблене програмне забезпечення призначене для забезпечення замовлення піц, додатків до піц і додаткових товарів у піцерії, планування діяльності цієї піцерії через прогнозування кількості проданих піц за певними групами. Для цього надаються засоби категоризації піц та додаткових продуктів, пошуку піц, вподобання та оцінювання піц та додаткових продуктів.
- У програмне забезпечення, створене як вебзастосунок піцерії, було інтегровано модель прогнозування кількості проданих піц за групами інгредієнтів. Для розробки використано мову програмування Python, фреймворк Django, систему керування базами даних MySQL як основні засоби.
- Наукова новизна роботи полягає в методі прогнозування кількості проданих піц за групами інгредієнтів, що характеризується розподілом піц на групи за конкретними інгредієнтами, застосуванням моделі ARIMA, що дозволяє визначати кількість піц з вмістом відповідного інгредієнта, яка буде продана в піцерії на певну дату.

14

Рисунок В.14 – Висновки