

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)

бакалавра

(ступінь вищої освіти)

на тему РОЗРОБКА ВЕБЗАСТОСУНКУ ДЛЯ ФІЗИЧНОЇ ОСОБИ

(назва теми)

ПІДПРИЄМЦЯ. DEVELOPMENT OF A WEB APPLICATION FOR

INDIVIDUAL ENTREPRENEURS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-120

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

ХОХЛОВ В.О.

(ПРІЗВИЩЕ та ініціали)

Керівник ЗЕЛЕНЬОВА І.Я.

(ПРІЗВИЩЕ та ініціали)

Рецензент ПОЛЯКОВ М.О.

(ПРІЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет _____ КНТ _____
Кафедра _____ програмних засобів _____
Ступінь вищої освіти _____ бакалавр _____
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)

ХОХЛОВА Всеволода Олександровича

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка вебзастосунку для фізичної особи підприємця
Development of a Web Application for Individual Entrepreneurs

керівник проєкту (роботи) к.т.н., доцент ЗЕЛЕНЬОВА Ірина Яківна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від « 24 » квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 10 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне
завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програми. 3. Розробка програми. 4. Експлуатація та тестування програми.

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРІЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЗЕЛЕНЬОВА І.Я., доцент		
Нормоконтроль	КАМІНСЬКА Ж.К., асистент		

7. Дата видачі завдання « 15 » квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка)

_____ Всеволод ХОХЛОВ
(підпис) (Ім'я ПРІЗВИЩЕ)

Керівник проєкту (роботи)

_____ Ірина ЗЕЛЕНЬОВА
(підпис) (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 93 с., 25 рис., 2 табл., 3 дод., 10 джерел.

ФІЗИЧНА ОСОБА ПІДПРИЄМЕЦЬ, ПРОДАЖ ТОВАРІВ, ВЕБЗАСТОСУНОК, КОНСУЛЬТУВАННЯ, БАЗА ДАНИХ.

Об'єкт роботи – процес роздрібного продажу товарів фізичними особами підприємцями. Предмет роботи – вебзастосунок для фізичної особи підприємця. Мета роботи – розробити вебзастосунок фізичної особи підприємця для поліпшення можливості продажу товарів через Інтернет.

Матеріали, методи та технічні засоби: мультипарадигмальне програмування, мова Python, фреймворк Django, система MySQL для підтримки баз даних.

Результати. Виконана розробка вебзастосунку для фізичної особи підприємця призначена для забезпечення фізичним особам підприємцям можливості виставляти на продаж власні товари, пошуку потрібних товарів, оформлення замовлень на придбання товарів у фізичних осіб підприємців клієнтами, керування замовленнями, керування доступними товарами.

Висновки. Розроблена програма є результатом аналізу аналогів, який дозволив визначити функції, які мають бути обов'язково реалізовані, а також встановити відсутню функцію, яка є потенційно дуже корисною для забезпечення довіри до фізичної особи підприємця особливо. Така функція передбачає консультування в тих клієнтів, які вже замовляли товари. За цією логікою було забезпечено подачу запитів на консультування для клієнтів у тих клієнтів, які створили відгуки про певну фізичну особу підприємця, а створити відгук можна тільки в тому випадку, коли було реалізовано замовлення.

Галузь використання. Роздрібний продаж товарів фізичними особами підприємцями.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 93 pages, 25 figures, 2 tables, 3 appendixes, 10 sources.

INDIVIDUAL ENTREPRENEUR, SALE OF GOODS, WEB APPLICATION, CONSULTING, DATABASE.

The object of the work is the process of retail sale of goods by individual entrepreneurs. The subject of the work is a web application for an individual entrepreneur. The purpose of the work is to develop a web application for an individual entrepreneur to improve the ability to sell goods via the Internet.

Materials, methods and technical means: multi-paradigm programming, Python language, Django framework, MySQL system for database support.

The results. The completed development of the web application for an individual entrepreneur is intended to provide individual entrepreneurs with the opportunity to put their own goods on sale, search for the necessary goods, place orders for the purchase of goods from individual entrepreneurs by customers, manage orders, manage available goods.

Conclusions. The developed program is the result of the analysis of analogues, which made it possible to determine the functions that must be implemented, as well as to establish the missing function, which is potentially very useful for ensuring trust in the individual entrepreneur in particular. This function involves consulting those customers who have already ordered goods. According to this logic, it was ensured that requests for consulting for customers were submitted to those customers who created reviews about a certain individual entrepreneur. At the same time it is possible to create a review only in the case when the order was implemented.

Field of use. Retail sale of goods by individual entrepreneurs.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	7
Вступ.....	8
1 Аналіз предметної області.....	9
1.1 Огляд програмних засобів для фізичної особи підприємця	9
1.2 Особливості розробки.....	12
1.3 Висновки за розділом 1	14
2 Проєктування програми.....	15
2.1 Вибір засобів проєктування і розробки	15
2.2 Визначення функціональних вимог до програмного забезпечення	16
2.3 Проєктування структури бази даних.....	22
2.4 Висновки за розділом 2	36
3 Розробка програми	37
3.1 Порядок роботи з програмою над замовленнями	37
3.2 Реалізація функцій програми	38
3.3 Висновки за розділом 3	54
4 Експлуатація та тестування програми	55
4.1 Призначення програми	55
4.2 Умови виконання програми	55
4.3 Виконання і тестування програми.....	55
4.4 Висновки за розділом 4	60
Висновки	61
Перелік джерел посилання	62
Додаток А Технічне завдання	63
Додаток Б Текст програми	68
Додаток В Слайди презентації.....	87

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

БД – база даних;

ДВВ – діаграма варіантів використання.

ВСТУП

Майданчики для підтримки роботи фізичних осіб підприємців є достатньо перспективними і за правильної організації роботи корисними для всіх сторін:

- самих фізичних осіб підприємців;
- клієнтів;
- самого майданчика.

Такі програмні засоби дозволяють організувати роздрібний продаж, а за рахунок розвиненої інфраструктури доставки в нашій країні це є перспективним напрямком подальшої роботи. Однак при цьому потрібно надати адекватні засоби керування, що зроблять процес взаємодії між фізичними особами підприємцями та клієнтами послідовним та контрольованим.

У 2023 році більше ніж 304000 нових фізичних осіб підприємців було зареєстровано в Україні, а за статистикою це всього лише на 7 % більше ніж у 2021 році, що вказує на достатньо великі темпи розвитку [1]. Однак необхідно розуміти, що такі фізичні особи підприємці часом самі продають товари, з іншого боку потребують товарів від інших осіб. Відповідно це визначає потребу в активному розвитку та наявності майданчиків для взаємодії між сторонами, а також відповідного ступеня довіри для такої взаємодії.

Подібні існуючі майданчики зазвичай мають засоби для створення відгуків про продавців, але з них можна отримати тільки ту інформацію, яку побажав розкрити клієнт. З іншого боку можна отримати детальні консультації, але від самого продавця. Уточнити інформацію від клієнта, який скористався послугами продавця, за умов, що клієнт готовий таку інформацію надати, неможливо. Це варто було б врахувати в програмі, а також під час аналізу аналогів перевірити, чи така функція присутня у комусь з цих аналогів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд програмних засобів для фізичної особи підприємця

Prom.ua (рис. 1.1) є одним з найвідоміших в Україні майданчиків для фізичних осіб підприємців, де вони можуть організувати роздрібну торгівлю [2].

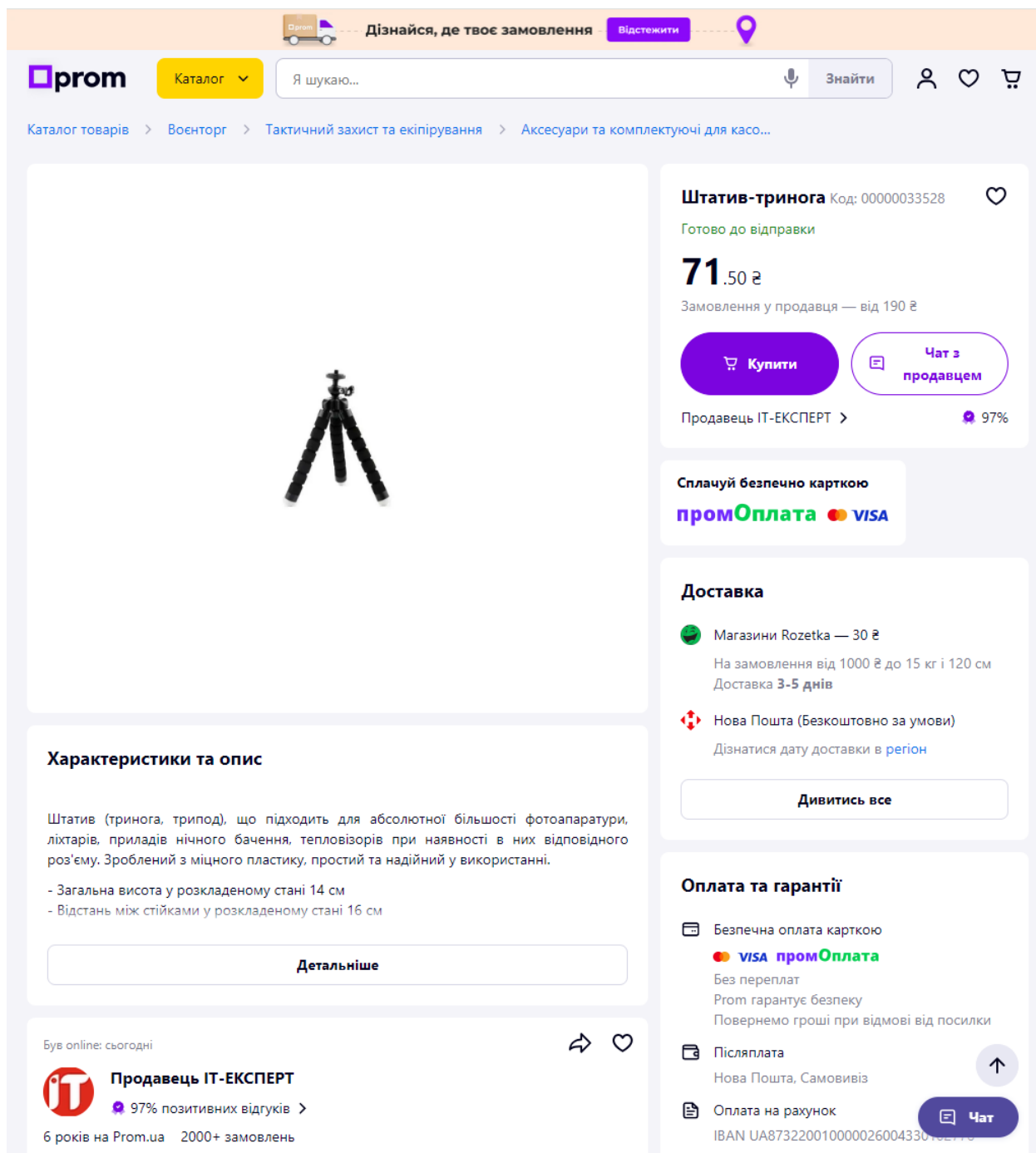


Рисунок 1.1 – Сторінка товару у вебзастосунку Prom.ua

Кожен підприємець має свою сторінку, що представляє фактично його магазин, за яким можна замовляти товари, що наявні в продажу, а за переліком всіх товарів магазину можна виконувати фільтрування, щоб вивести потрібні користувачу. При відфільтровуванні задається категорія товарів, обмежується ціна, можна визначити назву товару, задати область пошуку, встановити значення різних характеристик. Проте тут необхідно зазначити, що нерідко продавці не виділяють значення характеристик окремо для товарів, а приводять значення в опису товару текстом, що ускладнює далі пошук. Тобто фактично відсутня система введення товарів, яка б приводила до певної структурованості для щонайменше типових товарів. У підсумку можна виділити потрібну категорію, встановити ціну, а ось обмежити характеристики непросто, зважаючи на те, що ці характеристики виділені в тексті опису і не впливають на фільтрування в такому випадку.

За самим магазином покупець може залишити відгук, які потім можна разом переглядати. У відгуку вказується товар, який був придбаний, а за кожним відгуком можна позначити, на скільки він був корисним або навпаки неінформативним. До відгуку можна залишити коментар. Проте варто зазначити, що цей інструмент важко використовувати для консультування між клієнтами, адже для реалізації такої роботи потрібно, щоб не тільки певні коментарі до відгуку або інші відгуки, а саме адресні запитання окремо відображались, а до того ж необхідно, щоб подібно до позначення повідомлень в системах обміну повідомленнями відображались нові запитання, адже тоді користувач зверне увагу на те, що є запит, і якщо в нього є можливість, відповідь на нього. У Prom.ua такої можливості немає.

Проте в застосунку є розгалужена система підтримки замовлень товарів зокрема стосовно використання служб доставки, що є прогресом цього майданчика за останні роки, але потребує окремої роботи щодо підтримки цієї діяльності.

У той же час консультування з продавцем клієнта організовано як чат і може оперативно використовуватися. Товари можна додавати до обраних, щоб

потім повертатися до них.

Ria.com (рис. 1.2) надає можливість продавати широку групу товарів, серед яких окремо можна виділити автомобілі, нерухомість, але є і звичайна роздрібна торгівля фізичних осіб підприємців, включаючи вживані та нові запчастини, канцтовари, промислові товари, комп'ютери та телефони, літературу, книги, журнали, одяг і взуття, товари для дому та саду, спорядження для військових, товари для краси і здоров'я, дитячі товари, побутову техніку, продукти харчування, товари для хобі та спорту [3]. Тобто загалом фізичні особи підприємці можуть розташовувати велику кількість товарів з різних категорій, проте зробити це може будь-який користувач, не обов'язково фізична особа підприємець. Для цього достатньо відповідним чином зареєструватись.

Принцип організації роботи з товарами в застосунку подібний до Prom.ua. З відмінностей можна визначити систему відгуків. У даному випадку за продавцем можна подавати інформацію про неточності, формуючи в підсумку рейтинг продавця. Рейтинг відображається зокрема на сторінці пошуку товарів, що достатньо зручно. Проте не відображаються конкретні відгуки за продавцями або товарами. Тобто в підсумку єдиною метрикою визначення довіри до продавця є встановлений рейтинг. У той же час з відгуків нерідко можна отримати додаткові факти, які дозволяють покупцю зрозуміти поведінку, робочі процеси продавця, отримати інформацію про якість товару. В застосунку Ria.com це унеможливлено. Тоді можна сконтактувати з продавцем, але сподіватись на його надійність можна тільки на основі рейтингу. Звісно, що запитати іншого клієнта про досвід роботи з продавцем у такій ситуації також неможливо.

Таким чином, проведений огляд демонструє, що існують програмні аналоги розробки, що ведеться в роботі. Вони мають стандартний набір функцій, які мають бути реалізовані й у розробки, проте присутні й ті моменти, які потребують покращення і можуть визначити відмінність розробки від аналогів.

RIA.com
Автомобілі
Нерухомість
Автотовари
Запчастини
Збір на авто для ЗСУ
Увійти в кабінет

Перевірені Продавці
RIA.com
Додати оголошення

RIA.com > Промислові товари > Устаткування для сфери послуг > Устаткування для автосервісу > Інструмент для автосервісу

< Інструменти для автосервісу в Україні

VAS6154 оригінальний діагностичний прилад

80 077 грн.
В наявності

Іван

Тернопіль

5+ років працює з RIA.com

(067) xxx xx xx

Написати в чат

Знайшли недостовірну інформацію?
Залишити коментар

Додано: 02.06.2024
Переглядів: 2
ID: 180153342
Підняти оголошення в ТОП

Рубрика

Інструмент для автосервісу

Продається оригінальний діагностичний адаптер VAS6154. Серійник "чистий", можна працювати онлайн!!!
В комплекті сам адаптер, юсб модуль, віфі модуль, та док станція (для налаштувань адаптера)
Планшет для діагностики Panasonic cf-d1
16гб озу
256 ссд
i5-6 generation
Встановлені програми ODIS Service та ODIS Engineering

Читати далі ↓

Рисунок 1.2 – Сторінка товару у вебзастосунку RIA.com

1.2 Особливості розробки

З базових можливостей Prom.ua та RIA.com вирішено, що розробка потребує обов'язкової реалізації пошуку товарів, перегляду товарів фізичної особи підприємця за категорією, відкладання товарів у продаж фізичної особи підприємця, консультування продавцями клієнтів. Створення

повноцінних відгуків клієнтами про фізичних осіб підприємців відсутнє у Ria.com, але є у Prom.ua і має бути реалізоване в розробці для подальшої підтримки консультування клієнтів іншими клієнтами щодо фізичних осіб підприємців (табл. 1.1).

Таблиця 1.1 – Порівняння розробки з проаналізованими аналогами

Критерій порівняння	Prom.ua	Ria.com	Розробка
Пошук товарів	+	+	+
Створення сторінок фізичної особи підприємця	Самостійне будь-яким користувачем	Самостійне будь-яким користувачем	Керуючим за наданими даними
Перегляд товарів фізичної особи підприємця за категорією	+	+	+
Відкладання товарів	+	+	+
Створення базових карток товарів	—	—	+
Консультування продавцями клієнтів	+	+	+
Створення повноцінних відгуків клієнтами про фізичних осіб підприємців	+	—	+
Консультування клієнтів іншими клієнтами щодо фізичних осіб підприємців	—	—	+

У той же час розробка має відмінності від аналогів, виражені наявністю консультування клієнтів іншими клієнтами щодо фізичних осіб підприємців, створенням базових карток товарів, що дозволяє далі фізичним особам підприємцям за потреби розширювати або уточнювати дані, але не вводити

кожен товар повністю самостійно, впливаючи на якість внесених даних, а відповідно і на придатність товарів до пошуку.

У розробці прийнято рішення створювати сторінки фізичних осіб підприємців через керуючого, який буде мати можливість перевірити надані користувачем дані і таким чином ідентифікувати особу, щоби у випадку некоректного обслуговування було зрозуміло, до кого можна адресувати роботу щодо реалізації наслідків. Це має сприяти захисту покупців таким чином. За моделі, реалізованої в застосунках Prom.ua, Ria.com потрібні додаткові засоби підтримки, коли сплата проводиться тільки через платформу (а це фактично потім не завжди відбувається таким чином), створення різноманітних рейтингів продавців, які звичайно що не завжди відповідають дійсності щонайменше в певні моменти. Оскільки програма створюватиметься саме для фізичних осіб підприємців, тобто осіб, які мають певний статус, то такий механізм є прийнятним.

1.3 Висновки за розділом 1

У роботі для розробки вебзастосунку для фізичної особи підприємця потрібно розв'язати наступні завдання:

- аналіз проблеми розробки вебзастосунку для фізичної особи підприємця;
- визначення вимог до вебзастосунку для фізичної особи підприємця;
- проєктування вебзастосунку для фізичної особи підприємця;
- програмна реалізація вебзастосунку для фізичної особи підприємця.

У даному розділі виконано аналіз проблеми розробки вебзастосунку для фізичної особи підприємця, для чого проаналізовано можливості вебзастосунків Prom.ua, Ria.com, виділено головні, визначено відмінності розробки, до яких належать наявність базових карток товарів, можливість консультування клієнтів клієнтами щодо фізичних осіб підприємців тощо.

2 ПРОЄКТУВАННЯ ПРОГРАМИ

2.1 Вибір засобів проєктування і розробки

Оскільки у відповідності з темою та завданням на роботу розробка має вестися як веброзробка, то логічним виглядає в сучасних умовах розглянути серед можливих засобів JavaScript [4]. З іншого боку на даний момент достатньо якісні і широковідомі рішення на Python [5]. Також відомо, що зараз для ефективної веброзробки необхідне використання вебфреймворку. Для JavaScript рішенням відповідного рівня є Express JS [6]-[7], для Python – Django [8]-[9] (табл. 2.1).

Таблиця 2.1 – Порівняння фреймворків для розробки вебзастосунку для фізичної особи підприємця

Критерій порівняння	Express JS	Django
Мова розробки	JavaScript	Python
Концепція розробки	Мінімалістичність, гнучкість	Включеність найкращих практик за замовчуванням
Підтримка баз даних	Можна обрати систему на основі підключення модулів	Робота з базами даних типова, вбудована підтримка об'єктно- реляційного відображення
Вимоги до вебсервера	Не обмежені конкретним	Не обмежені конкретним
Безпека	Необхідні спеціальні бібліотеки, які потрібно вибирати	Інструменти включені за багатьма проблемами
Тип підтримуваних проєктів	Малі та середні	Складні, масштабні

Тому порівняння відбувалось комплексно, не тільки за мовами, але за набором технологій. Результатом такого вибору стало використання Django та мови Python в основі. Це обумовлено тим, що Django зокрема за рахунок концепції розробки на основі включеності найкращих практик за замовчуванням підтримує складні масштабні проекти, а використання вебзастосунку в якості платформи для фізичних осіб підприємців як комерційна платформа саме є масштабним проектом. До того ж саме Django для комерційних проектів такого типу застосовується, а для Express JS це не характерно. Також слід додати, що Django має потужні стандартні засоби перевірки безпеки, що важливо.

2.2 Визначення функціональних вимог до програмного забезпечення

Ролі користувачів:

- неавторизований користувач, який працює без облікового запису;
- клієнт, який є авторизованим користувачем, але фактично виконує завдання з купівлі потрібних йому товарів;
- представник фізичної особи підприємця, який є авторизованим користувачем і представляє фізичну особу підприємця;
- керуючий, який є авторизованим користувачем і реалізує нагляд за програмою.

Також у програмі має бути адміністратор, якому доступне додавання, змінювання будь-яких даних, але в Django для цього реалізується підтримка через адміністративну панель, тому окремо ці стандартні сценарії не визначались.

Діаграма варіантів використання (ДВВ) програми для неавторизованого користувача побудована на основі наступного основного потоку сценаріїв:

- перегляд сторінки фізичної особи підприємця;

- пошук фізичних осіб підприємців;
- пошук товарів у продажі фізичних осіб підприємців;
- перегляд товарів у фізичної особи підприємця за підкатегорією;
- перегляд відгуків про фізичну особу підприємця;
- перегляд сторінки товару фізичної особи підприємця;
- перегляд сторінки клієнта (рис. 2.1).

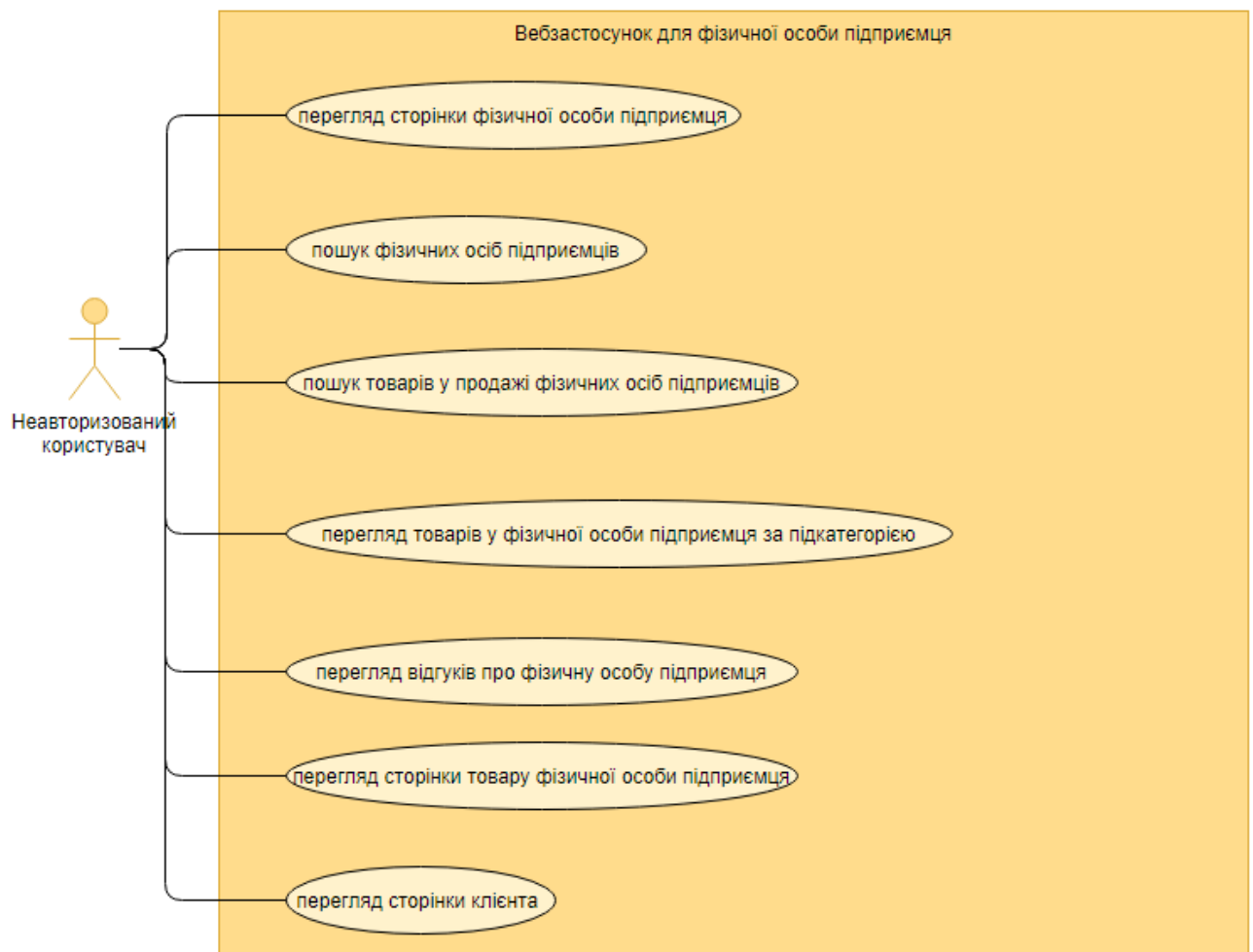


Рисунок 2.1 – ДВВ програми для неавторизованого користувача

ДВВ програми для клієнта побудована на основі наступного основного потоку сценаріїв додатково до неавторизованого користувача:

- авторизація клієнта;
- відкладання сторінки фізичної особи підприємця;
- перегляд відкладених сторінок фізичних осіб підприємців;

– надання власного відгуку про фізичну особу підприємця після замовлення;

- запитування консультації фізичної особи підприємця;
- перегляд окремої консультації;
- запитання консультації у клієнта, який є автором відгуку;
- перегляд запитаних клієнтами консультацій у клієнта;
- перегляд запитаних консультацій клієнтом;
- замовлення у фізичної особи підприємця товарів;
- перегляд клієнтом власних замовлень;
- встановлення отримання клієнтом замовлення;
- відкладання сторінки товару фізичної особи підприємця;
- перегляд відкладених сторінок товарів фізичних осіб підприємців;
- перегляд відгуків про фізичних осіб підприємців клієнта;
- оновлення даних клієнта (рис. 2.2).

Альтернативні сценарії клієнта:

– скасування відкладання сторінки фізичної особи підприємця є альтернативним для відкладання сторінки фізичної особи підприємця, коли відкладена сторінка більше не цікавить клієнта;

– скасування замовлення є альтернативною для встановлення отримання клієнтом замовлення, коли покупець створив замовлення, але вирішив, що це помилково, поки фізична особа підприємця не почала працювати над замовленням;

– скасування відкладання сторінки товару фізичної особи підприємця є альтернативною для відкладання сторінки товару фізичної особи підприємця, коли відкладена сторінка більше не цікавить клієнта;

– реєстрація клієнта є альтернативною для авторизації, коли немає облікового запису.

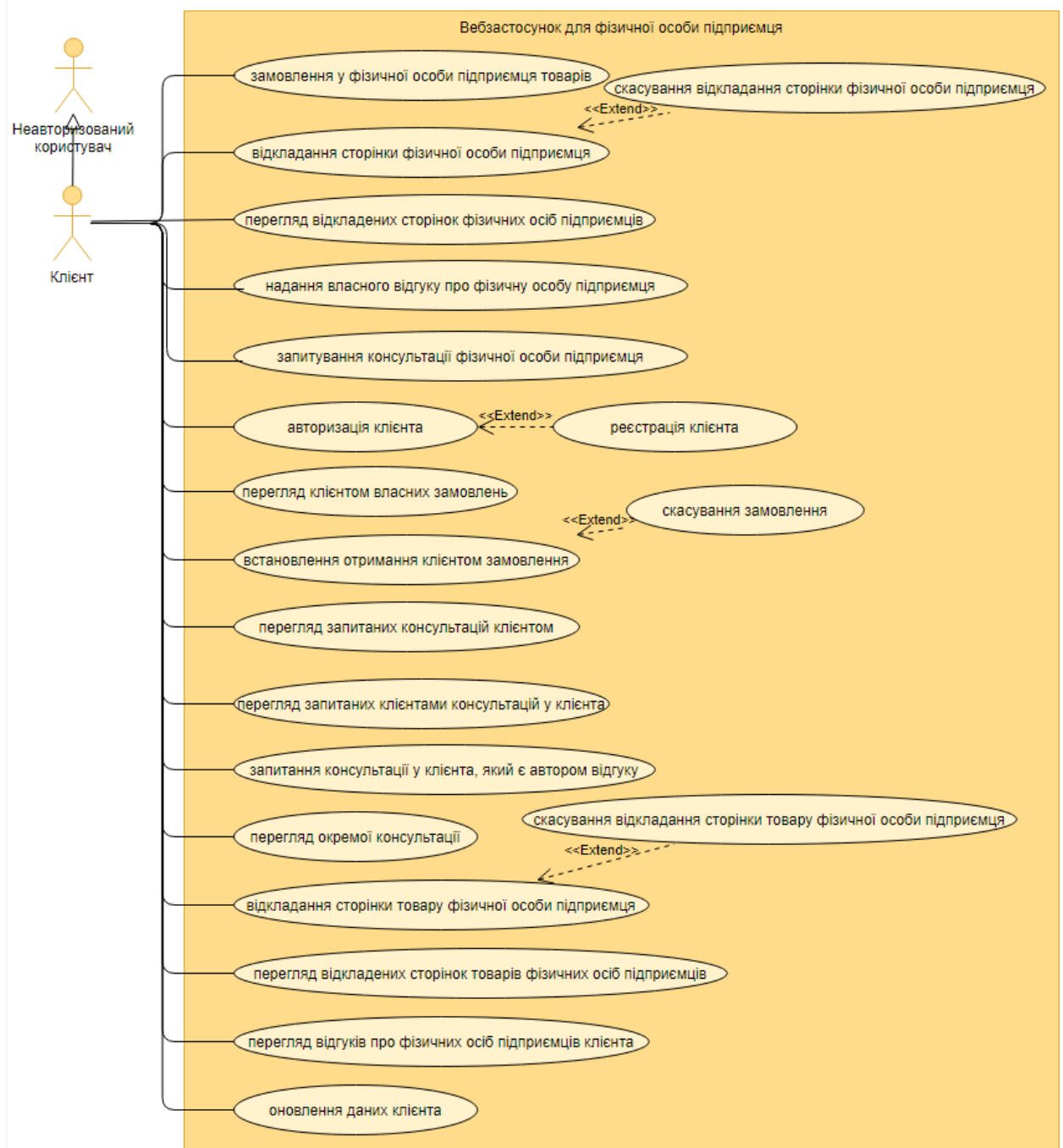


Рисунок 2.2 – ДВВ програми для клієнта

ДВВ програми для представника фізичної особи підприємця побудована на основі наступного основного потоку сценаріїв додатково до неавторизованого користувача:

- перегляд запитаних клієнтами консультацій у фізичної особи підприємця;
- формування відповіді на запитану клієнтом консультацію;

- оновлення даних фізичної особи підприємця;
- додавання працівника фізичної особи підприємця;
- авторизація працівника фізичної особи підприємця;
- перегляд кількості продаж товарів у фізичної особи підприємця;
- визначення наявних для продажу обсягів товарів у фізичної особи підприємця;
- перегляд клієнтських замовлень фізичною особою підприємцем з визначенням стану;
- встановлення прийняття замовлення фізичною особою підприємцем до виконання;
- встановлення відправлення замовлення фізичною особою підприємцем;
- додавання товару до продажу фізичною особою підприємцем;
- визначення вартості товару в продажу фізичної особи підприємця;
- перегляд історії зміни вартості товару в продажу фізичної особи підприємця;
- визначення додаткових атрибутів товару фізичної особи підприємця (рис. 2.3).

Альтернативні сценарії представника фізичної особи підприємця:

- оновлення даних товару в продажі фізичної особи підприємця є альтернативним для додавання товару до продажу фізичною особою підприємцем, коли товар вже доданий, але потрібні зміни даних;
- заборона роботи працівника за фізичною особою підприємцем є альтернативною для додавання працівника фізичної особи підприємця, коли працівник вже внесений до фізичної особи підприємця, але його доступ має бути заблокований;
- реєстрація працівника фізичної особи підприємця є альтернативною для авторизації працівника фізичної особи підприємця;
- скасування замовлення є альтернативним для встановлення прийняття замовлення фізичною особою підприємцем до виконання, коли

замовлення не може бути виконано.

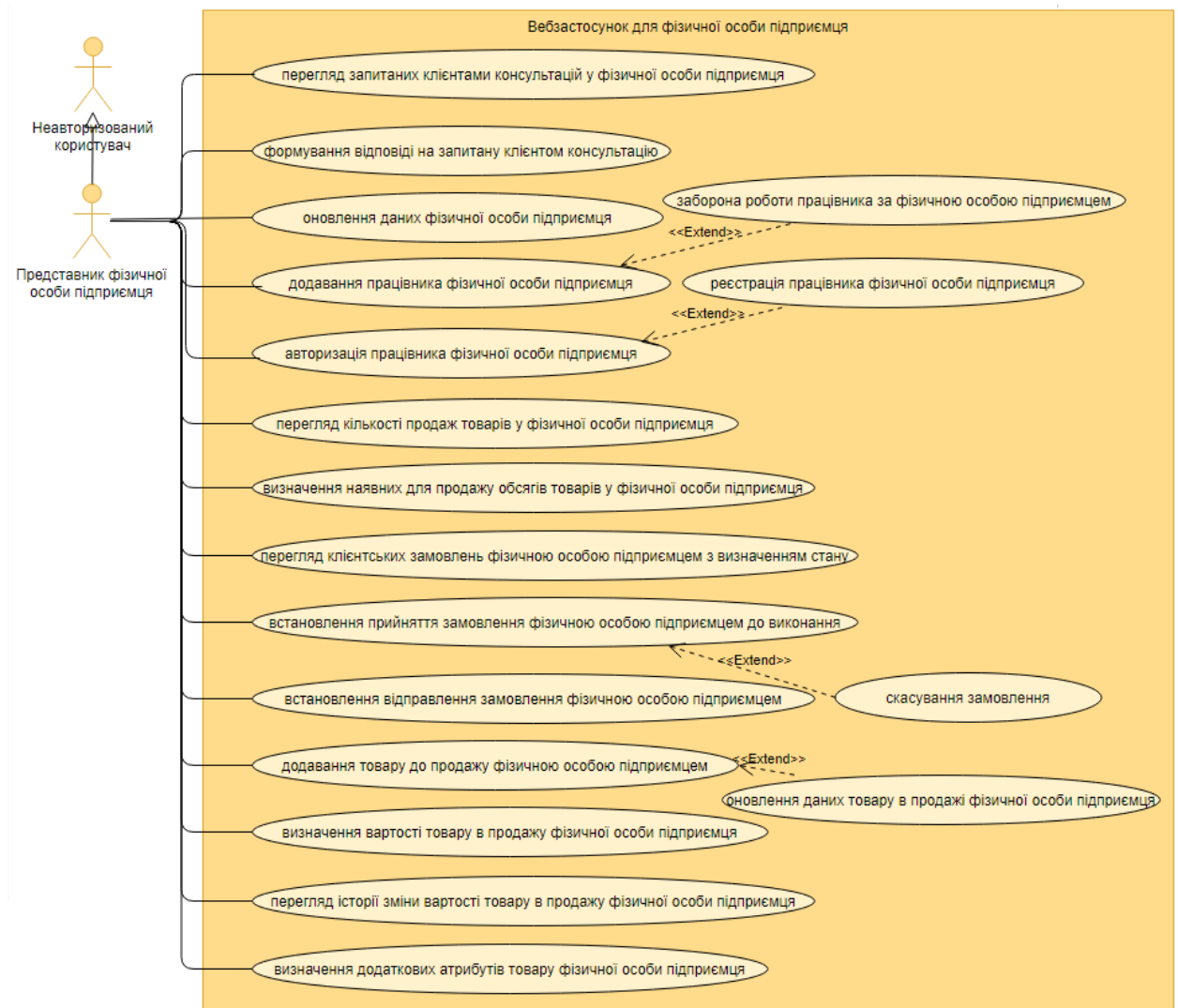


Рисунок 2.3 – ДБВ програми для представника фізичної особи підприємця

ДБВ програми для керуючого побудована на основі наступного основного потоку сценаріїв додатково до неавторизованого користувача:

- реєстрація керуючим роботи фізичної особи підприємця;
- перегляд клієнтських замовлень у всіх фізичних осіб підприємців з визначенням стану;
- додавання базової картки товару;
- пошук базових карток товарів;
- блокування роботи фізичної особи підприємця (рис. 2.4).

Альтернативним для блокування роботи фізичної особи підприємця є скасування блокування роботи фізичної особи підприємця, коли ці зміни були виконані некоректно.

Альтернативним для додавання базової картки товару є оновлення базової картки товару, коли при внесенні були некоректності.

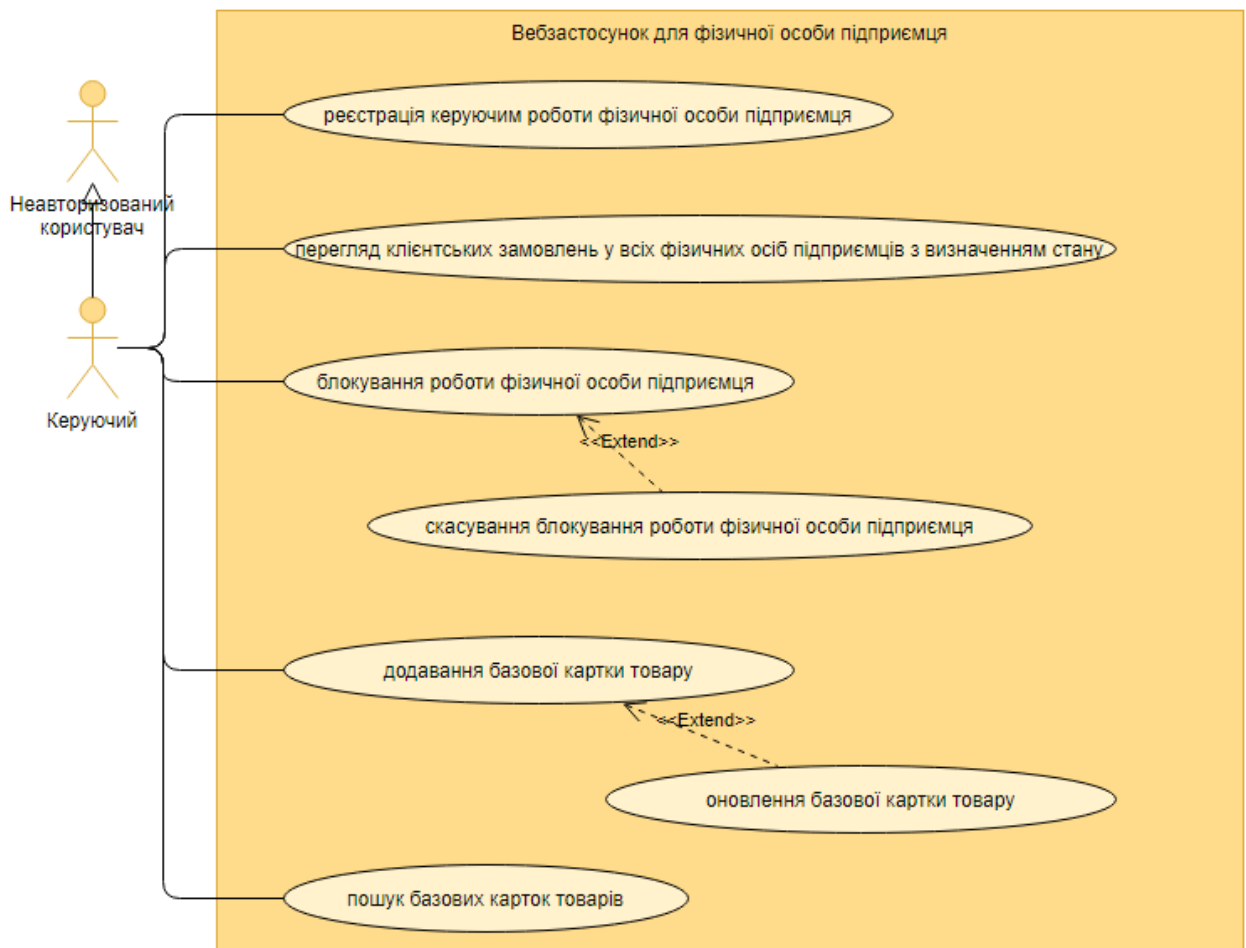


Рисунок 2.4 – ДВВ програми для керуючого

2.3 Проєктування структури бази даних

База даних (БД) спроектована як реляційна, тому передбачає визначення таблиць, зв'язків між цими таблицями, що і зроблено нижче. Системою керування, на основі якої робота з БД забезпечується, є MySQL, що окрім того, що є широковідомою, є безкоштовною, що є критичним для

розробки в межах кваліфікаційної роботи, потужною, використовуваною на практиці для реальних застосунків.

Дані фізичної особи підприємця зберігаються в таблиці БД Indiventrepreneur, у якій структурно виділені поля:

- id – ідентифікатор фізичної особи підприємця;
- entrepreneur_name – ім'я фізичної особи підприємця;
- entrepreneur_phone – номер телефону для контактування з фізичною особою підприємцем;
- entrepreneur_email – електронна поштова адреса фізичної особи підприємця;
- entrepreneur_city – місто, де фізично знаходиться фізична особа підприємець: зв'язок визначається з таблицею Businesscity через це поле і поле id;
- entrepreneur_aboutshop – опис представництва фізичної особи підприємця в програмі;
- entrepreneur_score – чисельне вираження оцінки, що всередньому фізичній особі підприємцю виставили клієнти;
- entrepreneur_logo – логотип, яким представлено діяльність фізичної особи підприємця;
- physical_address – адреса, за якою фізично знаходиться фізична особа підприємець;
- physical_address_display – дозвіл відображати адресу, за якою фізично знаходиться фізична особа підприємець, клієнту;
- blocked – фізична особа не може працювати з вебзастосунком, доступ заблокований (або ні).

Встановлений робочий час фізичної особи підприємця зберігається в таблиці БД Indiventrepreneurworktime, у якій структурно виділені поля:

- id – ідентифікатор запису встановленого робочого часу фізичної особи підприємця;

– `indiventrepreneur` – фізична особа підприємець, для якої встановлено цей робочий час: зв'язок визначається з таблицею `Indiventrepreneur` через це поле і поле `id`;

– `day_of_week` – номер дня тижня, в який діє цей графік: восьме значення відповідає святковому дню;

– `worktime_start` – година і хвилина початку роботи в цей робочий день цієї фізичної особи підприємця;

– `worktime_end` – година і хвилина закінчення роботи в цей робочий день цієї фізичної особи підприємця.

Фізична особа підприємця може бути представлена не обов'язково одним працівником, але і декількома. У випадку єдиного облікового запису в програмі у всіх таких працівників можуть виникати проблеми. Тоді або тільки одна людина працює в програмі, а якщо вона не на роботі, то і всі інші не можуть отримати доступ, або дані облікового запису мають передаватися між працівниками, що не є безпечним. Тому було вирішено ввести працівників фізичної особи підприємця, які його репрезентують у програмі.

Працівники фізичної особи підприємця, які його репрезентують у програмі, зберігаються в таблиці БД `Indiventrepreneurperson`, у якій структурно виділені поля:

– `id` – ідентифікатор працівника фізичної особи підприємця, який його репрезентує в програмі;

– `indiventrepreneur` – фізична особа підприємця, яку репрезентує цей працівник у програмі: зв'язок визначається з таблицею `Indiventrepreneur` через це поле і поле `id`;

– `user_account` – обліковий запис працівника, який репрезентує фізичну особу підприємця: зв'язок визначається з таблицею `User`, яка автоматично створюється в Django, через це поле і поле `id`;

– `personal_phone` – номер телефону, за яким можна зв'язатися з працівником, але ці дані використовуються тільки для роботи всередині фізичної особи підприємця та не передаються назовні;

– allowed – наявний дозвіл на представлення цьому працівнику фізичної особи підприємця або дозвіл відсутній.

Дані клієнтських користувачів програми зберігаються в таблиці БД Appclient, у якій структурно виділені поля:

- id – ідентифікатор клієнтського користувача програми;
- user_account – обліковий запис клієнтського користувача програми: зв'язок визначається з таблицею User, яка автоматично створюється в Django, через це поле і поле id;
- personal_phone – номер телефону для контактування з клієнтом фізичної особи підприємця під час роботи над замовленням;
- base_delivery_address – базова адреса, за якою має виконуватися доставка для даного клієнта;
- photo – фотографія.

Дані відкладання сторінок фізичних осіб підприємців для подальшої роботи зберігаються в таблиці БД Indiventrepreneursave, у якій структурно виділені поля:

- id – ідентифікатор запису про відкладання сторінки фізичної особи підприємця для подальшої роботи;
- appclient – користувач програми, який відклав для себе для подальшої роботи сторінку цієї фізичної особи підприємця: зв'язок визначається з таблицею Appclient через це поле і поле id;
- indiventrepreneur – фізична особа підприємець, сторінку якої відклав для подальшої власної роботи цей користувач програми: зв'язок визначається з таблицею Indiventrepreneur через це поле і поле id.

Дані відгуків про фізичних осіб підприємців зберігаються в таблиці БД Indiventrepreneurback, у якій структурно виділені поля:

- id – ідентифікатор відгуку про фізичну особу підприємця;
- appclient – користувач програми, що відгукнувся про фізичну особу підприємця, з яким працював: зв'язок визначається з таблицею Appclient через це поле і поле id;

– `indiventrepreneur` – фізична особа підприємець, про яку було створено відгук: зв'язок визначається з таблицею `Indiventrepreneur` через це поле і поле `id`;

– `final_score` – підсумкова цілочисельна оцінка, виставлена у відгуку фізичній особі підприємцю;

– `back_date` – дата публікації відгуку про фізичну особу підприємця;

– `feedback` – текст відгуку про фізичну особу підприємця.

Міста, в яких фізично можуть знаходитися фізичні особи підприємці та звідки вести свою роботу, зберігаються в таблиці БД `Businesscity`, у якій структурно виділені поля:

– `id` – ідентифікатор міста, в якому фізично може знаходитися фізична особа підприємця та звідки вести свою роботу;

– `region` – область, до якої належить місто, в якому фізично може знаходитися фізична особа підприємця та звідки вести свою роботу;

– `city_name` – назва міста, в якому фізично може знаходитися фізична особа підприємця та звідки вести свою роботу.

Консультування клієнта представниками фізичної особи підприємця зберігаються в таблиці БД `Indiventrepreneurconsult`, у якій структурно виділені поля:

– `id` – ідентифікатор консультування клієнта представниками фізичної особи підприємця;

– `indiventrepreneur` – фізична особа підприємця, якій було адресовано запит на консультування клієнтом: зв'язок визначається з таблицею `Indiventrepreneur` через це поле і поле `id`;

– `appclient` – користувач, який адресував запит на консультування цій фізичній особі підприємцю: зв'язок визначається з таблицею `Appclient` через це поле і поле `id`;

– `consult_question` – текст запиту на консультування клієнта до фізичної особи підприємця;

– consult_time – дата і час, коли запит на консультування клієнта до фізичної особи підприємця був створений;

– remove_mark – запит на консультування клієнта вже переглянуто фізичною особою підприємцем або ще ні.

Відповіді на запитані консультування клієнта до фізичної особи підприємця зберігаються в таблиці БД Indiventrepreneurconsultanswer, у якій структурно виділені поля:

– id – ідентифікатор відповіді на запитане консультування клієнта до фізичної особи підприємця;

– indiventrepreneurconsult – консультування клієнта представниками фізичної особи підприємця, за яким було надано відповідь: зв'язок визначається з таблицею Indiventrepreneurconsult через це поле і поле id;

– consult_answer – текст відповіді на запитане консультування клієнта до фізичної особи підприємця;

– consult_time – дата і час надання відповіді на запитане консультування клієнта до фізичної особи підприємця.

Консультування, враховуючи особливості програми, можуть реалізовуватися не тільки представниками фізичної особи підприємця, але і іншими клієнтами, які вже мали певний досвід взаємодії з певною фізичною особою підприємцем.

Дані консультувань клієнтів іншими досвідченими клієнтами зберігаються в таблиці БД Appclientconsult, у якій структурно виділені поля:

– id – ідентифікатор консультування клієнта іншим досвідченим клієнтом;

– indiventrepreneurback – відгук клієнтом про фізичну особу підприємця, на основі якого клієнт запитав у автора цього відгуку консультацію: зв'язок визначається з таблицею Indiventrepreneurback через це поле і поле id;

– `appclient` – клієнт, який запитав консультування в іншого досвідченого клієнта, автора відгуку: зв'язок визначається з таблицею `Appclient` через це поле і поле `id`;

– `consult_question` – текст запиту на консультування клієнта до іншого досвідченого клієнта;

– `consult_time` – дата і час запиту на консультування клієнта іншим досвідченим клієнтом;

– `remove_mark` – запит на консультування клієнта вже переглянуто клієнтом або ще ні.

Відповіді на консультування клієнта іншим досвідченим клієнтом зберігаються в таблиці БД `Appclientconsultanswer`, у якій структурно виділені поля:

– `id` – ідентифікатор відповіді на консультування клієнта іншим досвідченим клієнтом;

– `appclientconsult` – запит на консультування клієнта іншим досвідченим клієнтом, на який було надано відповідь: зв'язок визначається з таблицею `Appclientconsult` через це поле і поле `id`;

– `author_side` – сторона, яка сформувала цю відповідь (оскільки за такого консультування клієнт може щось перепитати додатково або прокоментувати відповідь іншого більш досвідченого в цьому питанні клієнта, то необхідно в подальшому зазначати сторону відповіді);

– `consult_answer` – текст відповіді на консультування клієнта іншим досвідченим клієнтом;

– `consult_time` – дата і час формування відповіді на консультування клієнта іншим досвідченим клієнтом.

Категорії товарів, які можуть продаватися фізичними особами підприємцями в програмі, зберігаються в таблиці БД `Commodcategory`, у якій структурно виділені поля:

– `id` – ідентифікатор категорії товарів, що може продаватися фізичною особою підприємцем у програмі;

– `commod_category_name` – назва категорії товарів, що може продаватися фізично особою підприємцем у програмі.

Підкатегорії товарів, які можуть продаватися фізичними особами підприємцями в програмі, зберігаються в таблиці БД `Commodsubcategory`, у якій структурно виділені поля:

– `id` – ідентифікатор підкатегорії товарів, що може продаватися фізичною особою підприємцем у програмі;

– `commodcategory` – категорія, до якої належить ця підкатегорія товарів, що може продаватися фізичною особою підприємцем у програмі: зв'язок визначається з таблицею `Commodcategory` через це поле і поле `id`;

– `commod_subcategory_name` – назва підкатегорії товарів, що може продаватися фізичною особою підприємцем у програмі.

Виробники товарів, відомі в програмі, зберігаються в таблиці БД `Commodityproducer`, у якій структурно виділені поля:

– `id` – ідентифікатор виробника, товари якого продаються в програмі;

– `producer_caption` – назва виробника, товари якого продаються в програмі;

– `producer_site` – офіційний вебсайт виробника, товари якого продаються в програмі.

Базові картки товарів, що продаються в програмі різними фізичними особами підприємцями, зберігаються в таблиці БД `Commodtemplate`, у якій структурно виділені поля:

– `id` – ідентифікатор базової картки товару, що продається в програмі різними фізичними особами підприємцями;

– `commodsubcategory` – підкатегорія товарів, до якої належить цей товар, що продається різними особами підприємцями: зв'язок визначається з таблицею `Commodsubcategory` через це поле і поле `id`;

– `commodityproducer` – виробник цього товару, що продається в програмі різними фізичними особами підприємцями: зв'язок визначається з таблицею `Commodityproducer` через це поле і поле `id`;

- commodity_caption – назва товару, що продається в програмі різними фізичними особами підприємцями;

- commodity_photo – основна фотографія товару, що продається в програмі різними фізичними особами підприємцями.

Конкретні товари в продажу фізичних осіб підприємців базуються на тих базових картках, що описані вище, та зберігаються в таблиці БД Commodindiventrepreneur, у якій структурно виділені поля:

- id – ідентифікатор конкретного товару в продажі фізичної особи підприємця;

- indiventrepreneur – фізична особа підприємець, що продає цей товар;

- commodtemplate – базова картка товару, на основі якої визначено цей конкретний товар у продажі фізичної особи підприємця: зв'язок визначається з таблицею Commodtemplate через це поле і поле id;

- additional_caption – додаткова назва товару, яку вже встановила фізична особа підприємець на додаток до стандартної частини з базової картки;

- commodity_about – опис товару фізичною особою підприємцем;

- main_page – товар має показуватися на сторінці фізичної особи підприємця (тобто є основним).

Фотографії конкретного товару в продажі фізичної особи підприємця зберігаються в таблиці БД Commodindiventrepreneurphoto, у якій структурно виділені поля:

- id – ідентифікатор фотографії конкретного товару в продажі фізичної особи підприємця;

- commodindiventrepreneur – конкретний товар в продажі фізичної особи підприємця, який показаний на фотографії: зв'язок визначається з таблицею Commodindiventrepreneur через це поле і поле id;

- photoimage – фотографія конкретного товару в продажі фізичної особи підприємця.

Дані вартості товару, за якою його продає фізична особа підприємця, зберігаються в таблиці БД *Commodprice*, у якій структурно виділені поля:

- *id* – ідентифікатор запису про вартість товару, за якою його продає фізична особа підприємця;
- *commodindiventrepreneur* – товар фізичної особи підприємця, який він продає за встановленою вартістю: зв'язок визначається з таблицею *Commodindiventrepreneur* через це поле і поле *id*;
- *min_units* – мінімальна кількість одиниць, за якою ця фізична особа підприємця готова продати цей товар за встановленою вартістю;
- *price_for_unit* – сума в гривнях, за яку ця фізична особа підприємця на вказаний період часу готова продати цей товар за покупки кількості одиниць його, що не менша ніж вказана в полі *min_units*;
- *start_price_time* – дата і час, починаючи з яких цей товар продається цією фізичною особою підприємцем за цю вартість одиниці за вказаних умов;
- *end_price_time* – дата і час, до яких цей товар продається цією фізичною особою підприємцем за цю вартість одиниці за вказаних умов.

Дані доступної для продажу на поточний момент кількості товару в конкретної фізичної особи підприємця зберігаються в таблиці БД *Commodnumber*, у якій структурно виділені поля:

- *id* – ідентифікатор запису доступної для продажу на поточний момент кількості товару в конкретної фізичної особи підприємця;
- *commodindiventrepreneur* – конкретний товар в продажі фізичної особи підприємця, за яким визначена доступна кількість одиниць: зв'язок визначається з таблицею *Commodindiventrepreneur* через це поле і поле *id*;
- *available_number* – кількість одиниць цього товару, що наявна у цієї фізичної особи підприємця.

Дані відкладання конкретного товару в продажі фізичної особи підприємця для подальшої роботи зберігаються в таблиці БД *Commodindiventrepreneursave*, у якій структурно виділені поля:

– id – ідентифікатор відкладання конкретного товару в продажі фізичної особи підприємця для подальшої роботи;

– commodindiventrepreneur – конкретний товар в продажі фізичної особи підприємця, яку відклав для подальшої роботи клієнт: зв'язок визначається з таблицею Commodindiventrepreneur через це поле і поле id;

– appclient – клієнт, який відклав для подальшої роботи конкретний товар в продажі фізичної особи підприємця: зв'язок визначається з таблицею Appclient через це поле і поле id.

Клієнтські замовлення у фізичних осіб підприємців зберігаються в таблиці БД Appclientorder, у якій структурно виділені поля:

– id – ідентифікатор клієнтського замовлення у фізичної особи підприємця;

– indiventrepreneur – фізична особа підприємець, яка створила клієнтське замовлення: зв'язок визначається з таблицею Indiventrepreneur через це поле і поле id;

– appclient – клієнт, який замовив у фізичної особи підприємця товари: зв'язок визначається з таблицею Appclient через це поле і поле id;

– order_state – стан клієнтського замовлення у фізичних осіб підприємців;

– order_time – дата і час, коли було сформоване клієнтське замовлення у фізичної особи підприємця;

– send_date – дата і час фактичного або очікуваного (за відповідного стану) відправлення клієнту замовлення фізичною особою підприємцем;

– delivery_date – дата і час фактичної або очікуваної (за відповідного стану) доставки клієнту замовлення фізичною особою підприємцем;

– notation – допис клієнтом щодо особливостей замовлення у фізичної особи підприємця;

– order_price – вартість в сумі клієнтського замовлення у фізичної особи підприємця;

– `delivery_details` – деталі виконання доставки клієнтського замовлення у фізичної особи підприємця.

Оскільки фізичні особи підприємці можуть продавати дуже різні товари, то категоризація цих товарів критично важлива для знаходження товарів, тому введено категорії та підкатегорії. Але в залежності від підкатегорії товари можуть мати різні атрибути, тобто по-різному характеризуватися: для настільної лампи і для велосипеда атрибути різні, хоча при цьому вони можуть мати спільний атрибут, наприклад, тип. Але значення цих типів для лампи та для велосипеда можуть бути зовсім різними.

Загальні атрибути, які можуть підв'язуватися під підкатегорії товарів або під товари, зберігаються в таблиці БД `Generalattribute`, у якій структурно виділені поля:

– `id` – ідентифікатор загального атрибуту, який далі може підв'язуватися під підкатегорії товарів або під товари;

– `attribute_name` – назва загального атрибуту, який далі може підв'язуватися під підкатегорії товарів або під товари.

Підв'язування атрибутів під підкатегорії товарів зберігаються в таблиці БД `Commodsubcategoryattribute`, у якій структурно виділені поля:

– `id` – ідентифікатор підв'язування атрибуту під підкатегорію товару;

– `commodsubcategory` – підкатегорія товару, під яку підв'язано атрибут: зв'язок визначається з таблицею `Commodsubcategory` через це поле і поле `id`;

– `generalattribute` – загальний атрибут, який підв'язаний під підкатегорію товару: зв'язок визначається з таблицею `Generalattribute` через це поле і поле `id`.

Значення підв'язаних атрибутів до підкатегорій товарів зберігаються в таблиці БД `Commodsubcategoryattributevalue`, у якій структурно виділені поля:

– `id` – ідентифікатор запису значення підв'язаного атрибуту до підкатегорії товару;

– commodsubcategoryattribute – підв'язування атрибуту під підкатегорію товару, за яким встановлено це значення: зв'язок визначається з таблицею Commodsubcategoryattribute через це поле і поле id;

– possible_int_value – можливе значення атрибуту, встановлене для підкатегорії товару як ціле число (якщо такий варіант потрібний);

– possible_str_value – можливе значення атрибуту, встановлене для підкатегорії товару як рядок (якщо такий варіант потрібний).

Конкретні значення атрибутів для базової картки товару, що продається в програмі різними фізичними особами підприємцями, на основі підв'язаного до підкатегорії товарів атрибуту зберігаються в таблиці БД Commodtemplateattribute, у якій структурно виділені поля:

– id – ідентифікатор значення атрибуту базової картки товару, що продається в програмі різними фізичними особами підприємцями;

– commodtemplate – базова картка товару, для якої встановлено це значення підв'язаного для підкатегорії атрибуту: зв'язок визначається з таблицею Commodtemplate через це поле і поле id;

– commodsubcategoryattributevalue – значення підв'язаного атрибуту до підкатегорії товару, яке має бути встановлено для цієї базової картки товару: зв'язок визначається з таблицею Commodsubcategoryattributevalue через це поле і поле id;

– value_detalization – конкретне значення (уточнене) підв'язаного атрибуту до підкатегорії товару за цією базовою картою товару: наприклад, у commodsubcategoryattributevalue може бути встановлено інтервал, а за цим полем конкретне значення, яке воно є у цього товару.

Окремі атрибути можуть бути особливими для товару в конкретної фізичної особи підприємця, а не підкатегорії. Це можуть бути певні особливості, які бажає задекларувати продавець. Такі атрибути товару фізичної особи підприємця зберігаються в таблиці БД Commodindiventrepreneurattribute, у якій структурно виділені поля:

- id – ідентифікатор атрибуту, прив'язаного конкретною фізичною особою підприємцем до свого товару;
- commodindiventrepreneur – конкретний товар фізичної особи підприємця, який вніс ці дані за своїм товаром: зв'язок визначається з таблицею Commodindiventrepreneur через це поле і поле id;
- generalattribute – загальний атрибут, який вирішила прив'язати до свого товару фізична особа підприємця: зв'язок визначається з таблицею Generalattribute через це поле і поле id;
- value_detalization – конкретне значення, яке за даними фізичної особи підприємця має її товар за цим атрибутом (рис. 2.5).

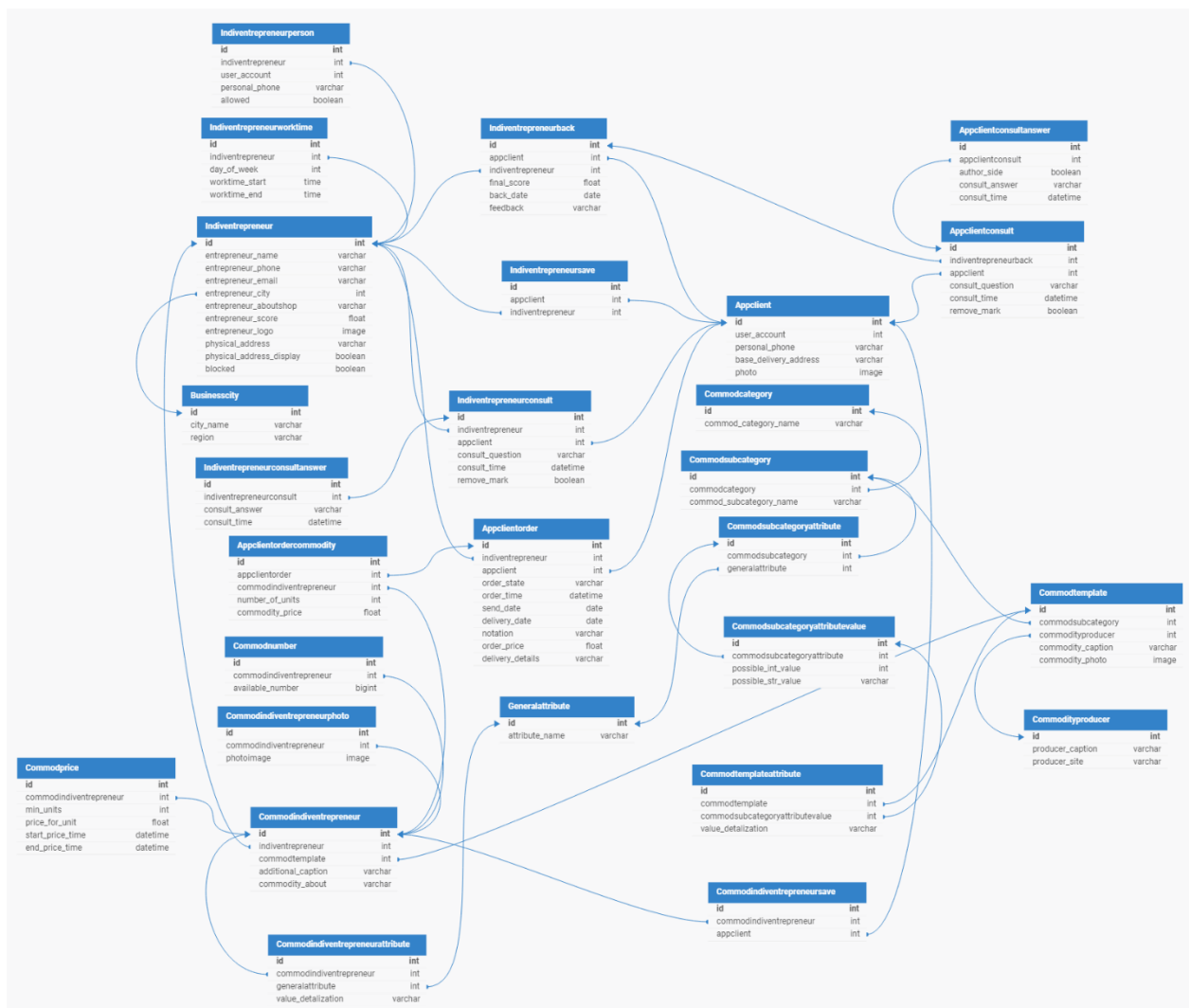


Рисунок 2.5 – Схема БД вебзастосунку для фізичної особи підприємця

Вміст клієнтського замовлення зберігається в таблиці БД `Appclientordercommodity`, у якій структурно виділені поля:

- `id` – ідентифікатор вмісту клієнтського замовлення;
- `appclientorder` – клієнтське замовлення, за яким визначено цей вміст: зв'язок визначається з таблицею `Appclientorder` через це поле і поле `id`;
- `commodindiventpreneur` – конкретний товар фізичної особи підприємця, який входить до складу клієнтського замовлення: зв'язок визначається з таблицею `Commodindiventpreneur` через це поле і поле `id`;
- `number_of_units` – кількість одиниць, яку за цим товаром замовив клієнт;
- `commodity_price` – вартість в сумі цього товару в клієнтському замовленні.

2.4 Висновки за розділом 2

Проектування вебзастосунку для фізичної особи підприємця успішно виконано, включаючи вибір Python-фреймворку Django для розробки вебзастосунку, моделювання сценаріїв на основі ДВВ для клієнта, фізичної особи підприємця, керуючого, розроблення структури БД.

3 РОЗРОБКА ПРОГРАМИ

3.1 Порядок роботи з програмою над замовленнями

Діаграма станів замовлення у фізичної особи підприємця (рис. 3.1) була побудована для визначення порядку роботи з замовленням, що є допустимим. Необхідно зазначити, що замовлення може бути не доставлено, втрачене під час доставки від фізичної особи підприємця клієнту. У такій ситуації самі клієнт або фізична особа підприємця не можуть прийняти рішення щодо зміни стану, тому на діаграмі такі переходи не продемонстровані. У таких випадках рішення приймаються адміністратором після надання інформації від керуючого. Адміністратор може змінювати стани потрібним чином.

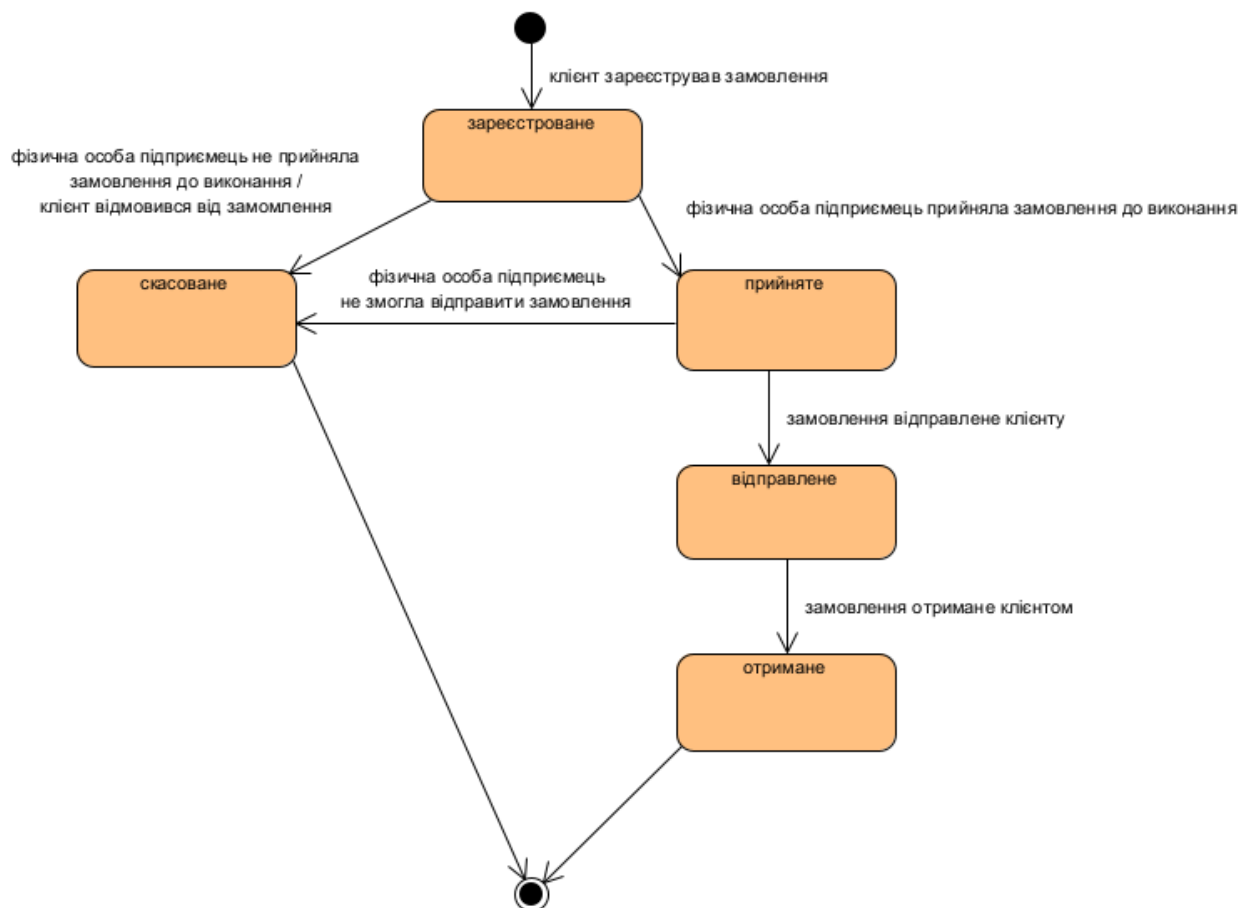


Рисунок 3.1 – Діаграма станів замовлення у фізичної особи підприємця

3.2 Реалізація функцій програми

Дотримуючись тих функціональних вимог, які визначені забезпечуваними сценаріями, використовуючи фреймворк Django, реалізовано всі функції програми. Реалізація функцій базується на моделях, класи яких описані далі, але за створення кожної конкретної функції відповідає кожна програмна функція програми, що уявляє собою представлення в архітектурі Model-View-Template, що характерна для Django. Кожне представлення забезпечується базовим шаблоном вебсторінки, а також зазвичай має шаблон вебсторінки помилки `error.html`, де в залежності від проблеми, що виникла, виводиться повідомлення про те, як такої проблеми уникнути. Реалізацію шаблонів кожної вебсторінки забезпечено використанням шаблону [10] для створення макетів цих сторінок.

Функція `indiventrepreneur_page_view` реалізує перегляд сторінки фізичної особи підприємця, маючи параметр `indiventrepreneur_id`, який визначає ідентифікатор фізичної особи підприємця. За цим ідентифікатором виділяється з БД об'єкт моделі `Indiventrepreneur`, а якщо такого об'єкта немає, то повертається сторінка з помилкою, в якій надається повідомлення «Задано невірний ідентифікатор фізичної особи підприємця. Будь-ласка, перевірте вказаний номер. Якщо посилання було згенеровано автоматично, то повідомте про це адміністратора». Подібні повідомлення в таких ситуаціях, у ситуаціях відсутності авторизації за потреби такої повертаються у всіх функціях, представлених далі, тому для скорочення опису така інформація далі не приводиться. У випадку успішності пошуку об'єкту фізичної особи підприємця шаблон вебсторінки заповнюється також робочим часом фізичної особи підприємця, інформацією про те, чи користувач, який працює зі сторінкою, є працівником цієї фізичної особи підприємця (а якщо це так і є, то також надається перелік інших працівників), кількість необроблених запитів на консультування від клієнтів. Якщо користувач, що авторизувався, є клієнтом, то перевіряється, чи не відклав цю сторінку собі для подальшої

роботи цей користувач, встановлюється кількість нових запитів на консультування, направлених цьому клієнту від інших клієнтів. Також до шаблону передаються дані всіх категорій, товари яких продаються цією фізичною особою підприємцем, товари, які призначені для відображення на основній сторінці з тих, що продаються цією фізичною особою підприємцем. За кожним таким товаром встановлюється, чи був він відкладений цим клієнтом.

Функція `indiventrepreneur_backs_page_view` реалізує перегляд відгуків про фізичну особу підприємця з номером `indiventrepreneur_id`, що задається параметром цієї функції. Окрім самих відгуків також встановлюється кількість необроблених запитів від клієнтів до фізичної особи підприємця або до клієнта в залежності від того, в якій ролі з програмою працює користувач. Ця інформація є стандартною і додається фактично до всіх сторінок (або тільки для відповідної однієї ролі, якщо функція використовується тільки певною роллю користувача).

Функція `indiventrepreneur_subcategory_page_view` реалізує перегляд товарів у фізичної особи підприємця з номером `indiventrepreneur_id` за підкатегорією з номером `commodsubcategory_id`, таким чином визначаючи два параметри для роботи. У функцію також передається перелік всіх категорій та підкатегорій за кожною категорією, що надає можливість далі обрати потрібну підкатегорію та оновити результати. Передаються в шаблон всі товари фізичної особи підприємця, які відповідають вказаним параметрам, для кожного товару визначається одна його фотографія для відображення, інформація про те, чи відкладено цей товар у цієї фізичної особи підприємця клієнтом.

Функція `indiventrepreneur_consults_page_view` реалізує перегляд запитаних клієнтами консультацій у фізичної особи підприємця. Вона не має параметрів, оскільки такі дані мають відображатися тільки працівнику фізичної особи підприємця. Відповідно за авторизацією визначається, який це працівник і до якої фізичної особи підприємця він має відношення. Далі за

цією фізичною особою витягаються всі запити на консультування від всіх клієнтів, впорядковуючи їх з найновіших. Тоді формується перелік зі всіх клієнтів, які такі запити на консультування створювали, а далі за кожним клієнтом визначається перелік консультувань.

Функція `appclient_consults_page_view` реалізує перегляд запитаних консультацій клієнтом. Консультації відбираються і з тих, які були запитані у фізичних осіб підприємців, групуючи їх за логікою, описаною в попередній функції за такими фізичними особами підприємцями, і з тих, які були запитані у клієнтів, знову ж таки групуючи їх за клієнтами.

Функція `appclient_to_consults_page_view` реалізує перегляд запитаних клієнтами консультацій у клієнта. Тобто у результуючу сторінку передаються ті консультування, які адресовані клієнту, який авторизувався, за тими відгуками про фізичних осіб підприємців, які ним створювались.

Функція `appclient_backs_page_view` реалізує перегляд відгуків про фізичних осіб підприємців клієнта. Клієнт може бути або визначений параметром за логіном `appclient_name`, або якщо такий параметр не визначено, то встановлюється за тим, хто авторизувався. Тобто в останньому випадку така сторінка формується персонально для користувача, який працює з програмою. За кожним відгуком перевіряється, чи фізична особа підприємець, стосовно якої було створено відгук, відкладена для подальшої роботи користувачем, який авторизований, чи ні.

Функція `appclient_orders_page_view` реалізує перегляд клієнтом власних замовлень з визначенням стану замовлень товарів фізичної особи підприємця з номером `order_state`. Нульове значення стану вказує, що обмеження на стан замовлень немає, тобто потрібно демонструвати всі замовлення. Замовлення відбираються тільки ті, які створив клієнт, який авторизувався. За кожним замовленням виділяється його склад з тих товарів фізичної особи підприємця, які в нього входять.

Функція `indiventrepreneur_orders_page_view` реалізує перегляд клієнтських замовлень фізичною особою підприємцем з визначенням стану з

номером `order_state`. Цей параметр аналогічно працює як і у попередній функції, але ця функція має також додаткові параметри. Параметр `order_sort` визначає впорядкування замовлень. Параметр `appclientorder_id` може не мати значення, але якщо має, то він задає конкретний номер замовлення. Це дозволяє виконати працівнику фізичної особи підприємця швидко пошук потрібного замовлення. Замовлення відбираються тільки ті, які були адресовані на покупку товарів у фізичної особи підприємця, якого представляє працівник, що авторизувався. Інша логіка роботи функції аналогічна попередній.

Функція `indentrepreneur_search_view` реалізує пошук фізичних осіб підприємців. Пошук виконується відфільтруванням за параметрами: `entrepreneur_name`, що задає назву фізичної особи підприємця або частину її, `entrepreneur_phone`, що задає номер телефону фізичної особи підприємця, `entrepreneur_city`, що задає місто, в якому має працювати фізична особа підприємця, `entrepreneur_region`, що задає область, в якій має працювати фізична особа підприємця, `entrepreneur_min_score`, що задає мінімальну оцінку, на яку в середньому мали б у відгуках оцінити клієнта потрібну фізичну особу підприємця.

Запитування консультації фізичної особи підприємця реалізує функція `indiventrepreneurconsult_add_view`, у якій параметр `indiventrepreneur` задає номер фізичної особи підприємця, а саме запитання для формування об'єкта моделі `Indiventrepreneurconsult` та збереження в БД передається параметром `consult_question`.

Функція `indiventrepreneur_commodities_statistics` реалізує перегляд кількості продаж товарів у фізичної особи підприємця. Ця інформація формується за фізичною особою підприємцем, яка пов'язана з працівником, який авторизувався. За цією фізичною особою підприємцем визначаються категорії, підкатегорії, наявні товари за цими підкатегоріями та для кожного обчислюється сумарна кількість фактично проданих товарів у завершених замовленнях. Історія підбивається за абсолютно весь час.

Функція `appclientconsult_add_view` реалізує запитання консультації у клієнта, який є автором відгуку, відповідно параметр `indiventrepreneurback_id` задає номер такого відгуку. Таке консультування можливо тільки тоді, коли клієнт вже залишив інформацію, публічно доступну, про те, що він скористався послугами цієї фізичної особи підприємця. Відповідно сама заявка на консультування підв'язується при збереженні під цей відгук. Параметр функції `consult_question` задає текст запиту на консультування, всі інші поля моделі `Appclientconsult` при збереженні в БД записуються в програмі.

Функція `commodity_price_history` реалізує перегляд історії зміни вартості товару в продажу фізичної особи підприємця з номером `entcommodity_id`. За таким товаром відбираються всі записи про вартість, впорядковуючи їх за датою, а результати заповнюють вебсторінку.

Функція `appclient_order_manage_view` здійснює загальне керування замовленням товарів у фізичної особи підприємця, реалізуючи таким чином варіанти використання програми і через встановлення відправлення замовлення фізичною особою підприємцем, і через встановлення отримання клієнтом замовлення, і через встановлення прийняття замовлення фізичною особою підприємцем до виконання, і через скасування замовлення. Функція працює повністю за діаграмою станів (рис. 3.1), перевіряючи можливість переходу в потрібний стан, який задається номером `order_state`. Окрім того визначається роль користувача, який працює з програмою. В залежності від поточного стану, потрібного стану, ролі користувача та можливості переходу з одного стану до іншого виконується або не виконується зміна стану замовлення. Якщо зміна не виконується, то надається повідомлення.

Функція `commodindiventrepreneur_saved_view` реалізує перегляд відкладених сторінок товарів фізичних осіб підприємців для клієнта, який працює як авторизований у програмі. За кожним товаром витягаються дані про нього, фотографія.

Подібно запрограмовані ті представлення, які реалізують функції, що не приведені в даному описі.

Модель фізичної особи підприємця має реалізацію класом `Indiventrepreneur`.

Поле моделі `entrepreneur_name` визначає ім'я фізичної особи підприємця за типом `CharField`.

Поле моделі `entrepreneur_phone` визначає номер телефону для контактування з фізичною особою підприємцем за типом `CharField`.

Поле моделі `entrepreneur_email` визначає електронну поштову адресу фізичної особи підприємця за типом `CharField`.

Поле моделі `entrepreneur_city` визначає місто, де фізично знаходиться фізична особа підприємець, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Businesscity`.

Поле моделі `entrepreneur_aboutshop` визначає опис представництва фізичної особи підприємця в програмі за типом `CharField`.

Поле моделі `entrepreneur_score` визначає чисельне вираження оцінки, що всередньому фізичній особі підприємцю виставили клієнти за типом `FloatField`.

Поле моделі `entrepreneur_logo` визначає логотип, яким представлено діяльність фізичної особи підприємця, за типом `ImageField`.

Поле моделі `physical_address` визначає адресу, за якою фізично знаходиться фізична особа підприємець, за типом `CharField`.

Поле моделі `physical_address_display` визначає дозвіл відображати адресу, за якою фізично знаходиться фізична особа підприємець, клієнту за типом `BooleanField`.

Поле `blocked` визначає те, що фізична особа не може працювати з вебзастосунком, доступ заблокований (або ні), типом `BooleanField`.

Встановлений робочий час фізичної особи підприємця має реалізацію класом `Indiventrepreneurworktime`.

Поле моделі `indiventrepreneur` визначає фізичну особу підприємця, для якої встановлено цей робочий час, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneur`.

Поле моделі `day_of_week` визначає номер дня тижня, в який діє цей графік, за типом `PositiveSmallIntegerField`.

Поле моделі `worktime_start` визначає годину і хвилину початку роботи в цей робочий день цієї фізичної особи підприємця за типом `TimeField`.

Поле моделі `worktime_end` визначає годину і хвилину закінчення роботи в цей робочий день цієї фізичної особи підприємця за типом `TimeField`.

Модель працівників фізичної особи підприємця, які його репрезентують у програмі, має реалізацію класом `Indiventrepreneurperson`.

Поле моделі `indiventrepreneur` визначає фізичну особу підприємця, яку репрезентує цей працівник у програмі, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneur`.

Поле моделі `user_account` визначає обліковий запис працівника, який репрезентує фізичну особу підприємця, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `User` (стандартною в Django).

Поле моделі `personal_phone` визначає номер телефону, за яким можна зв'язатися з працівником, за типом `CharField`.

Поле моделі `allowed` визначає наявний дозвіл на представлення цьому працівнику фізичної особи підприємця за типом `BooleanField`.

Модель клієнтських користувачів програми має реалізацію класом `Appclient`.

Поле моделі `user_account` визначає обліковий запис клієнтського користувача програми, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `User` (стандартною в Django).

Поле моделі `personal_phone` визначає номер телефону для контактування з клієнтом фізичної особи підприємця під час роботи над замовленням за типом `CharField`.

Поле моделі `base_delivery_address` визначає базову адресу, за якою має виконуватися доставка для даного клієнта, за типом `CharField`.

Поле моделі `photo` визначає фотографію за типом `ImageField`.

Поле моделі `indiventrepreneursaves` визначає фізичних осіб підприємців, сторінки яких відклав для подальшої власної роботи цей користувач програми, реалізовано полем `ManyToManyField` з моделлю `Indiventrepreneur`.

Поле моделі `commodindiventrepreneursaves` визначає конкретні товари в продажі фізичної особи підприємця, які відкладені для подальшої роботи клієнтом, реалізовано полем `ManyToManyField` з моделлю `Commodindiventrepreneur`.

Модель відгуків про фізичних осіб підприємців має реалізацію класом `Indiventrepreneurback`.

Поле моделі `appclient` визначає користувача програми, що відгукнувся про фізичну особу підприємця, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclient`.

Поле моделі `indiventrepreneur` визначає фізичну особу підприємця, про яку було створено відгук, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneur`.

Поле моделі `final_score` визначає підсумкову цілочисельну оцінку, виставлену у відгуку фізичній особі підприємцю, за типом `PositiveSmallIntegerField`.

Поле моделі `back_date` визначає дату публікації відгуку про фізичну особу підприємця за типом `DateField`.

Поле моделі `feedback` визначає текст відгуку про фізичну особу підприємця за типом `CharField`.

Модель міст, в яких фізично можуть знаходитися фізичні особи підприємці та звідки вести свою роботу, має реалізацію класом `Businesscity`.

Поле моделі `region` визначає область, до якої належить місто, в якому фізично може знаходитися фізична особа підприємця та звідки вести свою роботу, за типом `CharField`.

Поле моделі `city_name` визначає назву міста, в якому фізично може знаходитися фізична особа підприємця та звідки вести свою роботу, за типом `CharField`.

Модель консультування клієнта представниками фізичної особи підприємця має реалізацію класом `Indiventrepreneurconsult`.

Поле моделі `indiventrepreneur` визначає фізичну особу підприємця, якій було адресовано запит на консультування клієнтом, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneur`.

Поле моделі `appclient` визначає користувача, який адресував запит на консультування цій фізичній особі підприємцю, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclient`.

Поле моделі `consult_question` визначає текст запиту на консультування клієнта до фізичної особи підприємця за типом `CharField`.

Поле моделі `consult_time` визначає дату і час, коли запит на консультування клієнта до фізичної особи підприємця був створений, за типом `DateTimeField`.

Поле моделі `remove_mark` визначає те, чи запит на консультування клієнта вже переглянуто фізичною особою підприємцем, чи ще ні, типом `BooleanField`.

Модель відповіді на запитанні консультування клієнта до фізичної особи підприємця має реалізацію класом `Indiventrepreneurconsultanswer`.

Поле моделі `indiventrepreneurconsult` визначає консультування клієнта представниками фізичної особи підприємця, за яким було надано відповідь, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneurconsult`.

Поле моделі `consult_answer` визначає текст відповіді на запитане консультування клієнта до фізичної особи підприємця за типом `CharField`.

Поле моделі `consult_time` визначає дату і час надання відповіді на запитане консультування клієнта до фізичної особи підприємця за типом `DateTimeField`.

Модель консультувань клієнтів іншими досвідченими клієнтами має реалізацію класом `Appclientconsult`.

Поле моделі `indiventrepreneurback` визначає відгук клієнтом про фізичну особу підприємця, на основі якого клієнт запитав у автора цього відгуку консультацію, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneurback`.

Поле моделі `appclient` визначає клієнта, який запитав консультування в іншого досвідченого клієнта, автора відгуку, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclient`.

Поле моделі `consult_question` визначає текст запиту на консультування клієнта до іншого досвідченого клієнта за типом `CharField`.

Поле моделі `consult_time` визначає дату і час запиту на консультування клієнта іншим досвідченим клієнтом за типом `DateTimeField`.

Поле моделі `remove_mark` визначає те, чи запит на консультування клієнта вже переглянуто клієнтом, чи ще ні, типом `BooleanField`.

Модель відповіді на консультування клієнта іншим досвідченим клієнтом має реалізацію класом `Appclientconsultanswer`.

Поле моделі `appclientconsult` визначає запит на консультування клієнта іншим досвідченим клієнтом, на який було надано відповідь, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclientconsult`.

Поле моделі `author_side` визначає сторону, яка сформувала цю відповідь, за типом `BooleanField`.

Поле моделі `consult_answer` визначає текст відповіді на консультування клієнта іншим досвідченим клієнтом за типом `CharField`.

Поле моделі `consult_time` визначає дату і час формування відповіді на консультування клієнта іншим досвідченим клієнтом за типом `DateTimeField`.

Модель категорії товарів, які можуть продаватися фізичними особами підприємцями в програмі, має реалізацію класом `Commodcategory`.

Поле моделі `commod_category_name` визначає назву категорії товарів, що може продаватися фізично особою підприємцем у програмі, за типом `CharField`.

Модель підкатегорії товарів, які можуть продаватися фізичними особами підприємцями в програмі, має реалізацію класом `Commodsubcategory`.

Поле моделі `commodcategory` визначає категорію, до якої належить ця підкатегорія товарів, що може продаватися фізичною особою підприємцем у програмі, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodcategory`.

Поле моделі `commod_subcategory_name` визначає назву підкатегорії товарів, що може продаватися фізичною особою підприємцем у програмі, за типом `CharField`.

Поле моделі `generalattributes` визначає загальні атрибути, які підв'язані під цю підкатегорію товару, реалізовано полем `ManyToManyField` з моделлю `Generalattribute`.

Модель виробника товарів, відомого в програмі, має реалізацію класом `Commodityproducer`.

Поле моделі `producer_caption` визначає назву виробника, товари якого продаються в програмі, за типом `CharField`.

Поле моделі `producer_site` визначає офіційний вебсайт виробника, товари якого продаються в програмі, за типом `URLField`.

Модель базової картки товарів, що продаються в програмі різними фізичними особами підприємцями, має реалізацію класом `Commodtemplate`.

Поле моделі `commodsubcategory` визначає підкатегорію товарів, до якої належить цей товар, що продається різними особами підприємцями, а оскільки

це зовнішній ключ, то реалізовано воно як поле типу ForeignKey зі зв'язуванням з моделлю Commodsubcategory.

Поле моделі commodityproducer визначає виробника цього товару, що продається в програмі різними фізичними особами підприємцями, а оскільки це зовнішній ключ, то реалізовано воно як поле типу ForeignKey зі зв'язуванням з моделлю Commodityproducer.

Поле моделі commodity_caption визначає назву товару, що продається в програмі різними фізичними особами підприємцями, за типом CharField.

Поле моделі commodity_photo визначає основну фотографію товару, що продається в програмі різними фізичними особами підприємцями, за типом ImageField.

Модель конкретного товару в продажу фізичних осіб підприємців має реалізацію класом Commodindiventrepreneur.

Поле моделі indiventrepreneur визначає фізичну особу підприємця, що продає цей товар, а оскільки це зовнішній ключ, то реалізовано воно як поле типу ForeignKey зі зв'язуванням з моделлю Indiventrepreneur.

Поле моделі commodtemplate визначає базову картку товару, на основі якої визначено цей конкретний товар у продажі фізичної особи підприємця, а оскільки це зовнішній ключ, то реалізовано воно як поле типу ForeignKey зі зв'язуванням з моделлю Commodtemplate.

Поле моделі additional_caption визначає додаткову назву товару, яку вже встановила фізична особа підприємець на додаток до стандартної частини з базової картки, за типом CharField.

Поле моделі commodity_about визначає опис товару фізичною особою підприємцем за типом CharField.

Поле моделі main_page визначає, що товар має показуватися на сторінці фізичної особи підприємця (тобто є основним), типом BooleanField.

Модель фотографії конкретного товару в продажі фізичної особи підприємця має реалізацію класом Commodindiventrepreneurphoto.

Поле моделі `commodindiventrepreneur` визначає конкретний товар в продажі фізичної особи підприємця, який показаний на фотографії, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodindiventrepreneur`.

Поле моделі `photoimage` визначає фотографію конкретного товару в продажі фізичної особи підприємця за типом `ImageField`.

Модель вартості товару, за якою його продає фізична особа підприємець, має реалізацію класом `Commodprice`.

Поле моделі `commodindiventrepreneur` визначає товар фізичної особи підприємця, який він продає за встановленою вартістю, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodindiventrepreneur`.

Поле моделі `min_units` визначає мінімальну кількість одиниць, за якою ця фізична особа підприємець готова продати цей товар за встановленою вартістю, за типом `PositiveIntegerField`.

Поле моделі `price_for_unit` визначає суму в гривнях, за яку ця фізична особа підприємця на вказаний період часу готова продати цей товар за покупки кількості одиниць його, що не менша ніж вказана в полі `min_units`, має тип `FloatField`.

Поле моделі `start_price_time` визначає дату і час, починаючи з яких цей товар продається цією фізичною особою підприємцем за цю вартість одиниці за вказаних умов, за типом `DateTimeField`.

Поле моделі `end_price_time` визначає дату і час, до яких цей товар продається цією фізичною особою підприємцем за цю вартість одиниці за вказаних умов, за типом `DateTimeField`.

Модель доступної для продажу на поточний момент кількості товару в конкретної фізичної особи підприємця має реалізацію класом `Commodnumber`.

Поле моделі `commodindiventrepreneur` визначає конкретний товар в продажі фізичної особи підприємця, за яким визначена доступна кількість

одиниць, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodindiventrepreneur`.

Поле моделі `available_number` визначає кількість одиниць цього товару, що наявна у цієї фізичної особи підприємця, за типом `BigIntegerField`.

Моделі клієнтського замовлення у фізичних осіб підприємців має реалізацію класом `Appclientorder`.

Поле моделі `indiventrepreneur` визначає фізичну особу підприємця, яка створила клієнтське замовлення, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Indiventrepreneur`.

Поле моделі `appclient` визначає клієнта, який замовив у фізичної особи підприємця товари, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclient`.

Поле моделі `order_state` визначає стан клієнтського замовлення у фізичних осіб підприємців за типом `CharField`.

Поле моделі `order_time` визначає дату і час, коли було сформоване клієнтське замовлення у фізичної особи підприємця, за типом `DateTimeField`.

Поле моделі `send_date` визначає дату і час фактичного або очікуваного відправлення клієнту замовлення фізичною особою підприємцем за типом `DateField`.

Поле моделі `delivery_date` визначає дату і час фактичної або очікуваної доставки клієнту замовлення фізичною особою підприємцем за типом `DateField`.

Поле моделі `notation` визначає допис клієнтом щодо особливостей замовлення у фізичної особи підприємця за типом `CharField`.

Поле моделі `order_price` визначає вартість в сумі клієнтського замовлення у фізичної особи підприємця за типом `FloatField`.

Поле моделі `delivery_details` визначає деталі виконання доставки клієнтського замовлення у фізичної особи підприємця за типом `CharField`.

Моделі загального атрибута, який може підв'язуватися під підкатегорії товарів або під товари, має реалізацію класом `Generalattribute`.

Поле моделі `attribute_name` визначає назву загального атрибуту, який далі може підв'язуватися під підкатегорії товарів або під товари, за типом `CharField`.

Модель значення підв'язаних атрибутів до підкатегорій товарів має реалізацію класом `Commodsubcategoryattributevalue`.

Поле моделі `commodsubcategoryattribute` визначає підв'язування атрибуту під підкатегорію товару, за яким встановлено це значення, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodsubcategoryattribute`.

Поле моделі `possible_int_value` визначає можливе значення атрибуту, встановлене для підкатегорії товару як ціле число, за типом `IntegerField`.

Поле моделі `possible_str_value` визначає можливе значення атрибуту, встановлене для підкатегорії товару як рядок, за типом `CharField`.

Модель конкретного значення атрибутів для базової картки товару, що продається в програмі різними фізичними особами підприємцями, на основі підв'язаного до підкатегорії товарів атрибуту має реалізацію класом `Commodtemplateattribute`.

Поле моделі `commodtemplate` визначає базову картку товару, для якої встановлено це значення підв'язаного для підкатегорії атрибуту, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodtemplate`.

Поле моделі `commodsubcategoryattributevalue` визначає значення підв'язаного атрибуту до підкатегорії товару, яке має бути встановлено для цієї базової картки товару, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodsubcategoryattributevalue`.

Поле моделі `value_detalization` визначає конкретне значення (уточнене) підв'язаного атрибуту до підкатегорії товару за цією базовою карткою товару за типом `CharField`.

Модель додаткових атрибутів товару фізичної особи підприємця має реалізацію класом `Commodindiventrepreneurattribute`.

Поле моделі `commodindiventrepreneur` визначає конкретний товар фізичної особи підприємця, який вніс ці дані за своїм товаром, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodindiventrepreneur`.

Поле моделі `generalattribute` визначає загальний атрибут, який вирішила прив'язати до свого товару фізична особа підприємця, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Generalattribute`.

Поле моделі `value_detalization` визначає конкретне значення, яке за даними фізичної особи підприємця має її товар за цим атрибутом, за типом `CharField`.

Модель вмісту клієнтського замовлення має реалізацію класом `Appclientordercommodity`.

Поле моделі `appclientorder` визначає клієнтське замовлення, за яким визначено цей вміст, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Appclientorder` відповідним чином.

Поле моделі `commodindiventrepreneur` визначає конкретний товар фізичної особи підприємця, який входить до складу клієнтського замовлення, а оскільки це зовнішній ключ, то реалізовано воно як поле типу `ForeignKey` зі зв'язуванням з моделлю `Commodindiventrepreneur`.

Поле моделі `number_of_units` визначає кількість одиниць, яку за цим товаром замовив клієнт, за типом `IntegerField`.

Поле моделі `commodity_price` визначає вартість в сумі цього товару в клієнтському замовленні за типом `FloatField`.

3.3 Висновки за розділом 3

За розділом 3 представлено порядок зміни станів клієнтського замовлення на купівлю товарів у фізичної особи підприємця на основі діаграми станів, створені функції та класи програми.

4 ЕКСПЛУАТАЦІЯ ТА ТЕСТУВАННЯ ПРОГРАМИ

4.1 Призначення програми

Вебзастосунок призначений для забезпечення фізичним особам підприємцям можливості виставляти на продаж власні товари, пошуку потрібних товарів, оформлення замовлень на придбання товарів у фізичних осіб підприємців клієнтами, керування замовленнями, керування доступними товарами.

4.2 Умови виконання програми

Забезпечення програмної складової для підтримки роботи, включаючи інструменти фреймворку Django, інтерпретатора мови Python, системи для роботи з базою даних MySQL, є умовами виконання програми.

З огляду на потребу в створенні платформи, яка буде використовуватися різними фізичними особами підприємцями та потенційно великий попит в подальшому, розрахунок вимог до технічних засобів здійснювався з забезпечення 100 звернень на секунду (у інших випадках вимоги мають бути скориговані) і включав:

- процесор чотирьохядерний;
- оперативна пам'ять обсягом 8 Гб;
- жорсткий диск з 500 Гб вільного простору.

4.3 Виконання і тестування програми

У цьому підрозділі описано логіку використання наявних засобів у вебзастосунку під час його виконання. У процесі роботи було виконано тестування, яке дозволило перевірити, як забезпечується логіка роботи та виконання кожної функціональної вимоги, представлені в розділі 2. Для цього було розгорнуто на сервері вебзастосунок та з різних пристроїв у мережі

здійснювався доступ до сервера, включаючи мобільний пристрій та стаціонарний комп'ютер. Тестування продемонструвало успішне виконання всіх функціональних вимог.

Неавторизований користувач у меню має два розділи, кожен з яких містить по 1 функції для пошуку: відповідно пошуку серед фізичних осіб підприємців та пошуку серед товарів цих фізичних осіб підприємців (рис. 4.1). Через ці основні функції можна отримати доступ до інших. Наприклад, можна зі сторінки фізичної особи підприємця відобразити відгуки про нього. Якщо користувач є клієнтом, а не неавторизованим користувачем, то біля відгуків відображаються також кнопки «Проконсультуватись», які дозволяють поставити запитання автору відгуку щодо фізичної особи підприємця, про яку він склав відгук.

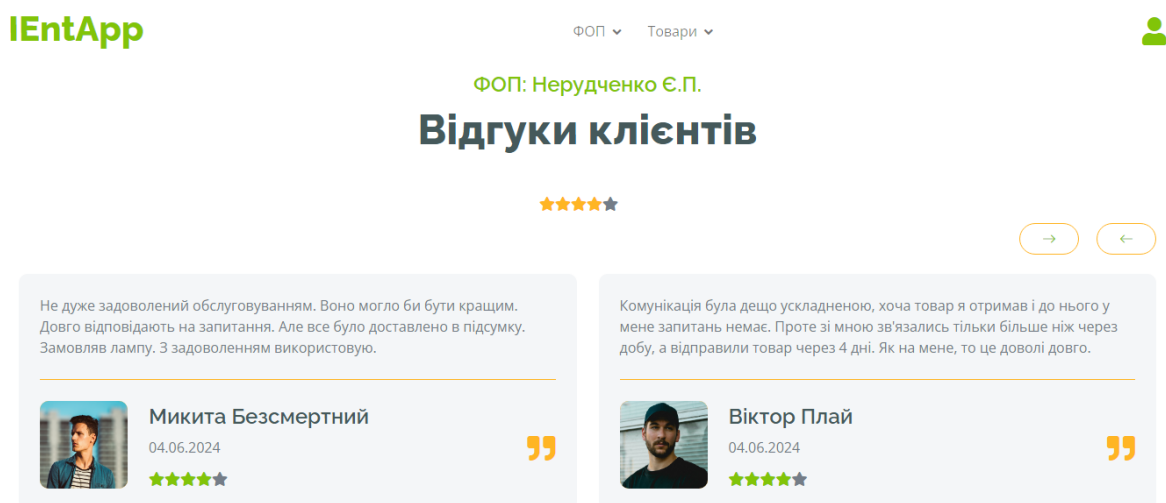


Рисунок 4.1 – Вебсторінка відгуків клієнтів про фізичну особу підприємця

Якщо вже користувач авторизувався, то якщо він є клієнтом, його меню розширюється. Додається розділ з консультаціями, який містить окремо доступ до тих консультацій, які клієнт сам запитував, та тих, які в нього запитували. Інформація про кількість нових запитаних консультацій весь час відображається клієнту біля знаку питання (рис. 4.2). Як тільки клієнт переглянув таке консультування, то кількість зменшується.

**Brille Egg**

ФОП: Нерудченко Є.П.



Підкатегорія: Настільні лампи

1800 ₪

Категорії

- ✓ Лампи (2)
- ✓ Побутова техніка (2)

Опис Фотографії

Настільна лампа Brille Egg має стильний дизайн, але одночасно він доволі простий і збалансований, що дозволяє лампі вписуватися в доволі різні інтер'єри. Лампу можна повертати під потрібним кутом, вона здатна прекрасно освітити робоче місце.

Стиль	Класичний
Колір	Бежевий
Висота, см	430
Джерело живлення	Від мережі
Тип лампи	Енергозберігаюча
Тип цоколя	E14

Рисунок 4.2 – Вебсторінка товару фізичної особи підприємця

Клієнт, перейшовши на сторінку товару фізичної особи підприємця, бачить його основну фотографію, опис, може переключитися на відображення послідовності фотографій замість опису, а також може одразу виконувати дії: відкладати товар для подальшої роботи (позначається зображенням серця), визначати кількість товару та додавати його в кошик відповідною кнопкою (після цього збільшується кількість товарів біля іконки згори). Відповідно можна переглянути відкладені товари фізичних осіб підприємців через додаткову функцію у клієнта з розділу товарів. Натиснувши на іконку з сумкою і кількістю товарів, можна відкрити склад замовлення (рис. 4.3), звідки і завершити його формування, додавши, якщо потрібно, певну текстову інформацію, особливості доставки.

Клієнт також має в меню замовлення, що відображають сторінку з його власними замовленнями у фізичних осіб підприємців, які він створив. Усі власні створені відгуки клієнт можна бачити через пункт меню з відгуками.

IEntApp ФОП ▾ Товари ▾ Консультації ▾ Замовлення Відгуки ? 1 2 👤

ФОП: Нерудченко Є.П.

Товари	Назва	Вартість одиниці	Кількість	Вартість товару
	Brille Egg	1800 ₴	1	1800 ₴
	Brille Moderna	1450 ₴	1	1450 ₴

Повідомлення продавцю

Доставка

☐ використати стандартну адресу

Доставка

Сумарно 3250 ₴

ЗАМОВИТИ

Рисунок 4.3 – Вебсторінка замовлення у фізичної особи підприємця

Зі сторінок фізичних осіб підприємців клієнт також може відкладати і їх сторінки, потім відображаючи їх з відповідного пункту розділу фізичних осіб підприємців.

При переході до консультувань за клієнтами, які в клієнта їх запитали, відображається перелік всіх таких клієнтів, а відкриваючи конкретного, можна побачити запит, відповісти на нього (рис. 4.4).

За іконкою людини можна перейти на дані про себе, звідки їх можна змінити. Також при натисканні на цю іконку випадає меню, яке дозволяє і вийти з облікового запису.

Представник фізичної особи підприємця має дещо змінене меню, яке містить не всі пункти клієнта. У розділі фізичних осіб підприємців він має окрему функцію для відображення сторінки його компанії, звідки можна перейти до оновлення даних.

ФОП: Нерудченко Є.П.

Користувач

Час

Запит



Ірина Мотренко

05.06.2024 19:06

Скажіть, будь ласка, а вони надають гарантію? Ви отримали гарантійний талон? І чи заповнений він?

Введіть текст

ВІДПОВІСТИ

Рисунок 4.4 – Вебсторінка консультування клієнта клієнтом

На сторінці товару фізичної особи підприємця (рис. 4.2) відображаються категорії товарів, доступні за конкретною фізичною особою. Якщо відкрити таку категорію, то далі можна обрати підкатегорію (рис. 4.5). За кожним товаром відображаються кнопки для виконання дій. При наведенні на кнопку вона підсвічується помаранчевим кольором, а сама картка підсвічується. Для представника фізичної особи підприємця такі кнопки включають можливість оновлення даних товару, переходу до роботи над наявною кількістю такого товару та вартістю. У свою чергу клієнт має кнопки для додавання кожного товару до кошику замовлення.

Представник фізичної особи підприємця також має у розділі товарів доступ до функцій керування обсягами доступних товарів, додавання нового товару, де він може скористатися базовою картою, яку додав керуючий, або додати взагалі новий товар, перегляду статистики за продажем товарів різних категорій, керування вартістю товарів, звідки він може переглядати за кожним товаром і зміну вартості в минулому, а також встановлювати різну вартість за придбання різної кількості товару. З меню замовлень представник фізичної особи підприємця може відображати всі замовлення у цієї фізичної особи підприємця, керувати ними, вибирати, в якому стані замовлення демонструвати.

Категорії

- ✓ Лампи
- ✓ Побутова техніка

(2)
(2)

Підкатегорія: Настільні лампи ▾

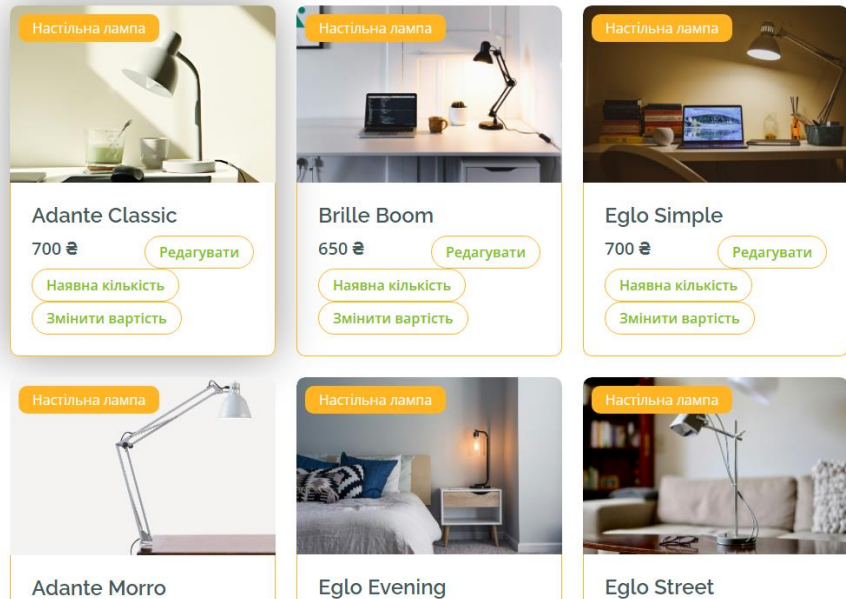


Рисунок 4.5 – Вебсторінка товарів фізичної особи підприємця за підкатегорією

Керуючий відповідно з розділу товарів має доступ до засобів додавання та пошуку базових карток товарів, які потім використовують представники фізичних осіб підприємців. Також керуючий має в розділі фізичних осіб підприємців доступ до реєстрації таких осіб, що розпочинає процес роботи.

4.4 Висновки за розділом 4

Проведене тестування функцій клієнта, представника фізичної особи підприємця, керуючого, неавторизованого користувача підтвердило успішність реалізації всіх функцій. Принципи роботи цих користувачів описано в розділі.

ВИСНОВКИ

У роботі виконано розробку вебзастосунку фізичної особи підприємця для поліпшення можливості продажу товарів через Інтернет. Для цього було проведено огляд і порівняння аналогів, визначено технології для розробки, що охопили Python-фреймворк для веброзробки Django, систему MySQL для підтримки БД, визначено структуру БД, варіанти використання програми клієнтом, фізичною особою підприємцем, керуючим, представлено порядок зміни станів клієнтського замовлення на купівлю товарів у фізичної особи підприємця на основі діаграми станів, створені функції та класи програми. Усі функції програми відповідно до варіантів використання реалізовані, а ця реалізація є успішною, що підтверджено під час тестування програми.

Вебзастосунок призначений для забезпечення фізичним особам підприємцям можливості виставляти на продаж власні товари, пошуку потрібних товарів, оформлення замовлень на придбання товарів у фізичних осіб підприємців клієнтами, керування замовленнями, керування доступними товарами. Програма дозволяє підтримувати роботу для цілого ряду фізичних осіб підприємців, таким чином сприяючи формуванню певного майданчика, що на практиці можливо пристосувати, наприклад, для продажу товарів за певною нішею.

Розроблена програма є результатом аналізу аналогів, який дозволив визначити функції, які мають бути обов'язково реалізовані, а також встановити відсутню функцію, яка є потенційно дуже корисною для забезпечення довіри до фізичної особи підприємця особливо. Така функція передбачає консультування в тих клієнтів, які вже замовляли товари. За цією логікою було забезпечено подачу запитів на консультування для клієнтів у тих клієнтів, які створили відгуки про певну фізичну особу підприємця, а створити відгук можна тільки в тому випадку, коли було реалізовано замовлення.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. 300,000 Ukrainians become solo entrepreneurs in 2023, primarily in retail, IT, personalized services [Electronic resource]. – Access mode : <https://english.nv.ua/business/over-300-000-ukrainians-become-individual-entrepreneurs-in-2023-50382285.html>.
2. Prom – найбільший маркетплейс України [Електрон. ресурс]. – Режим доступу : <https://prom.ua/ua/>.
3. RIA.com – дошка безкоштовних приватних оголошень України [Електрон. ресурс]. – Режим доступу : <https://www.ria.com/uk/>.
4. The Modern JavaScript Tutorial [Electronic resource]. – Access mode : <https://javascript.info/>.
5. Korsun, J. Python for Web Development [Electronic resource] : Pros and Cons / J. Korsun. – Access mode : <https://djangostars.com/blog/python-web-development/#:~:text=Python%20is%20predominantly%20used%20for,and%20serve%20front%2Dend%20content>.
6. Express – Node.js web application framework [Electronic resource]. – Access mode : <https://expressjs.com/>.
7. Express web framework (Node.js/JavaScript) [Electronic resource]. – Access mode : https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs.
8. Dinder, M. Becoming an Enterprise Django Developer [Text] : Discover best practices, tooling, and solutions for writing and organizing Django applications in production / M. Dinder. – Birmingham : Packt Publishing, 2022. – 526 p.
9. Django Web Development in Python [Electronic resource]. – Access mode : <https://www.datacamp.com/tutorial/web-development-django>.
10. Fruitables – Vegetable Website Template [Electronic resource]. – Access mode : <https://htmlcodex.com/vegetable-website-template/>.

ДОДАТОК А
Технічне завдання

Вступ

Майданчики для підтримки роботи фізичних осіб підприємців є достатньо перспективними і за правильної організації роботи корисними для всіх сторін: самих фізичних осіб підприємців, клієнтів та самого майданчика. Такі програмні засоби дозволяють організувати роздрібний продаж, а за рахунок розвиненої інфраструктури доставки в нашій країні це є перспективним напрямком подальшої роботи. Однак при цьому потрібно надати адекватні засоби керування, що зроблять процес взаємодії між фізичними особами підприємцями та клієнтами послідовним та контрольованим.

Подібні існуючі майданчики зазвичай мають засоби для створення відгуків про продавців, але з них можна отримати тільки ту інформацію, яку побажав розкрити продавець. З іншого боку можна отримати детальні консультації, але від самого продавця. Уточнити інформацію від клієнта, який скористався послугами продавця, за умов, що клієнт готовий таку інформацію надати, неможливо. Це варто було б врахувати в програмі.

A.1 Підстави для розробки

Підстави розробки за дипломною кваліфікаційною роботою визначені та затверджені наказом № 168 від 24 квітня 2024 р. за Національним університетом «Запорізька політехніка», у якому зафіксовано тему «Розробка вебзастосунку для фізичної особи підприємця».

A.2 Призначення розробки

Розробка програми призначена для забезпечення фізичним особам підприємцям можливості виставляти на продаж власних товарів, пошуку потрібних товарів, оформлення замовлень на придбання товарів у фізичних

осіб підприємців клієнтами, керування замовленнями, керування доступними товарами.

А.3 Основні вимоги до програми

А.3.1 Вимоги до функціональних характеристик

Наступні функції потрібно забезпечити при реалізації програми:

- реєстрація керуючим роботи фізичної особи підприємця;
- оновлення даних фізичної особи підприємця;
- додавання працівника фізичної особи підприємця;
- заборона роботи працівника за фізичною особою підприємцем;
- реєстрація і авторизація клієнта;
- реєстрація і авторизація працівника фізичної особи підприємця;
- відкладання сторінки фізичної особи підприємця та його скасування;
- перегляд відкладених сторінок фізичних осіб підприємців;
- перегляд сторінки фізичної особи підприємця;
- надання власного відгуку про фізичну особу підприємця після замовлення;
- пошук фізичних осіб підприємців;
- запитування консультації фізичної особи підприємця;
- перегляд запитаних клієнтами консультацій у фізичної особи підприємця;
- перегляд окремої консультації;
- формування відповіді на запитану клієнтом консультацію;
- пошук товарів у продажі фізичних осіб підприємців;
- запитання консультації у клієнта, який є автором відгуку;
- перегляд запитаних клієнтами консультацій у клієнта;
- перегляд запитаних консультацій клієнтом;
- перегляд товарів у фізичної особи підприємця за підкатегорією;
- перегляд кількості продаж товарів у фізичної особи підприємця;

- визначення наявних для продажу обсягів товарів у фізичної особи підприємця;
- замовлення у фізичної особи підприємця товарів;
- перегляд клієнтських замовлень фізичною особою підприємцем з визначенням стану;
- перегляд клієнтом власних замовлень;
- встановлення прийняття замовлення фізичною особою підприємцем до виконання;
- встановлення відправлення замовлення фізичною особою підприємцем;
- встановлення отримання клієнтом замовлення;
- скасування замовлення;
- додавання товару до продажу фізичною особою підприємцем;
- оновлення даних товару в продажі фізичної особи підприємця;
- визначення вартості товару в продажі фізичної особи підприємця;
- перегляд історії зміни вартості товару в продажі фізичної особи підприємця;
- відкладання сторінки товару фізичної особи підприємця та його скасування;
- перегляд відкладених сторінок товарів фізичних осіб підприємців;
- визначення додаткових атрибутів товару фізичної особи підприємця;
- перегляд відгуків про фізичних осіб підприємців клієнта;
- перегляд сторінки клієнта та оновлення її даних;
- перегляд відгуків про фізичну особу підприємця;
- перегляд клієнтських замовлень у всіх фізичних осіб підприємців з визначенням стану;
- перегляд сторінки товару фізичної особи підприємця;
- блокування роботи фізичної особи підприємця та скасування такого блокування.

A.3.2 Умови експлуатації

До умов експлуатації належить забезпечення програмної складової для підтримки роботи, включаючи інструменти фреймворку за його використання, інтерпретатора мови програмування, системи для роботи з базою даних.

A.3.3 Вимоги до складу та параметрів технічних засобів

З огляду на потребу в створенні платформи, яка буде використовуватися різними фізичними особами підприємцями та потенційно великий попит в подальшому, розрахунок вимог до технічних засобів здійснювався з забезпечення 100 звернень на секунду (у інших випадках вимоги мають бути скориговані):

- процесор чотирьохядерний;
- оперативна пам'ять обсягом 8 Гб;
- жорсткий диск з 500 Гб вільного простору.

A.4 Порядок контролю та приймання

Строки виконання та результати виконання за кожним етапом дипломної кваліфікаційної роботи визначаються календарним планом у завданні на роботу та контролюються керівником роботи, який закріплений наказом.

Захист роботи реалізується перед екзаменаційною комісією як відкрита процедура.

ДОДАТОК Б
Текст програми

```

from django.shortcuts import render
from .models import Indiventrepreneur
from .models import Indiventrepreneurworktime
from .models import Indiventrepreneurperson
from .models import Appclient
from .models import Indiventrepreneurback
from .models import Commodindiventrepreneur
from .models import Commodindiventrepreneurphoto
from .models import Appclientconsult
from .models import Indiventrepreneurconsult
from .models import Appclientconsultanswer
from .models import Indiventrepreneurconsultanswer
from .models import Appclientorder
from .models import Appclientordercommodity

def indiventrepreneur_page_view(request,
indiventrepreneur_id):
    try:
        indiventrepreneur_object =
Indiventrepreneur.objects.select_related("entrepreneur_city")
        indiventrepreneur_object =
indiventrepreneur_object.get(id = indiventrepreneur_id, blocked
= False)
    except Indiventrepreneur.DoesNotExist:
        return render(request, "ientapp/error.html",
{"error_text": "Задано невірний ідентифікатор фізичної особи
підприємця. Будь-ласка, перевірте вказаний номер. Якщо посилання
було згенеровано автоматично, то повідомте про це
адміністратора."})
        data_for_enterpreneur_page = {}
        data_for_enterpreneur_page["entrepreneur"] =
indiventrepreneur_object
        entrep_worktime =
Indiventrepreneurworktime.objects.filter(indiventrepreneur =
indiventrepreneur_object)
        entrep_worktime =
entrep_worktime.order_by("day_of_week")
        data_for_enterpreneur_page["worktime"] =
entrep_worktime
        data_for_enterpreneur_page["ownmode"] = False
        if request.user.is_active:
            entrepreneur_persons =
Indiventrepreneurperson.objects.filter(indiventrepreneur =

```

```

indiventrepreneur_object, user_account = request.user, allowed =
True)

    if entrepreneur_persons:
        data_for_enterpreneur_page["ownmode"] = True
        entrep_persons =
Indiventrepreneurperson.objects.select_related("user_account")
        data_for_enterpreneur_page["persons"] =
entrep_persons.filter(indiventrepreneur =
indiventrepreneur_object)
        admin_consults =
Indiventrepreneurconsult.objects.filter(indiventrepreneur =
indiventrepreneur_object, remove_mark = False)
        if admin_consults:
            admin_consults = admin_consults.count()

data_for_enterpreneur_page["newadminconsults"] = admin_consults
    else:

data_for_enterpreneur_page["newadminconsults"] = 0
        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]
            if indiventrepreneur_object in
appclient_object.indiventrepreneurs:
                data_for_enterpreneur_page["save"] = True
                client_consults =
Appclientconsult.objects.filter(indiventrepreneurback__appclient
= appclient_object, remove_mark = False)
                if client_consults:
                    client_consults = client_consults.count()

data_for_enterpreneur_page["newclientconsults"] =
client_consults
    else:

data_for_enterpreneur_page["newclientconsults"] = 0
        indentrep_backs =
Indiventrepreneurback.objects.select_related("appclient")
        indentrep_backs =
indentrep_backs.filter(indiventrepreneur =
indiventrepreneur_object)

```

```

        indentrep_backs = indentrep_backs.order_by("-
back_date")
        indentrep_backs = indentrep_backs[:3]
        data_for_entrepreneur_page["backs"] = indentrep_backs
        indentrep_categories =
Commodindiventrepreneur.objects.select_related("commodtemplate")
        indentrep_categories =
indentrep_categories.select_related("commodtemplate__commodsubca
tegory")
        indentrep_categories =
indentrep_categories.select_related("commodtemplate__commodsubca
tegory__commodcategory")
        indentrep_categories =
indentrep_categories.filter(indiventrepreneur =
indiventrepreneur_object)
        indentrep_categories =
indentrep_categories.values("commodtemplate__commodsubcategory__
commodcategory")
        indentrep_categories = indentrep_categories.distinct()
        data_for_entrepreneur_page["categories"] =
indentrep_categories
        indentrep_commodities =
Commodindiventrepreneur.objects.select_related("commodtemplate")
        indentrep_commodities =
indentrep_commodities.select_related("commodtemplate__commodityp
roducer")
        indentrep_commodities =
indentrep_commodities.filter(indiventrepreneur =
indiventrepreneur_object, main_page = True)
        for indentrep_commodity in indentrep_commodities:
            indentrep_commodity.photo =
Commodindiventrepreneurphoto.objects.filter(commodindiventrepren
eur = indentrep_commodity)
            if appclient_object:
                for indentrep_commodity in indentrep_commodities:
                    if indentrep_commodity in
appclient_object.commodindiventrepreneurs:
                        indentrep_commodity.saved = True
                    else:
                        indentrep_commodity.saved = False
                data_for_entrepreneur_page["commodities"] =
indentrep_commodities
            return render(request,
"ientapp/indiventrepreneur.html", data_for_entrepreneur_page)

```

```

def indiventrepreneur_backs_page_view(request,
indiventrepreneur_id):
    try:
        indiventrepreneur_object =
Indiventrepreneur.objects.select_related("entrepreneur_city")
        indiventrepreneur_object =
indiventrepreneur_object.get(id = indiventrepreneur_id)
    except Indiventrepreneur.DoesNotExist:
        return render(request, "ientapp/error.html",
{"error_text": "Задано невірний ідентифікатор фізичної особи
підприємця. Будь-ласка, перевірте вказаний номер. Якщо посилання
було згенеровано автоматично, то повідомте про це
адміністратора."})
        data_for_enterpreneur_page = {}
        data_for_enterpreneur_page["entrepreneur"] =
indiventrepreneur_object
        entrep_worktime =
Indiventrepreneurworktime.objects.filter(indiventrepreneur =
indiventrepreneur_object)
        entrep_worktime =
entrep_worktime.order_by("day_of_week")
        data_for_enterpreneur_page["worktime"] =
entrep_worktime
        if request.user.is_active:
            entrepreneur_persons =
Indiventrepreneurperson.objects.filter(indiventrepreneur =
indiventrepreneur_object, user_account = request.user, allowed =
True)

            if entrepreneur_persons:
                admin_consults =
Indiventrepreneurconsult.objects.filter(indiventrepreneur =
indiventrepreneur_object, remove_mark = False)
                if admin_consults:
                    admin_consults = admin_consults.count()

data_for_enterpreneur_page["newadminconsults"] = admin_consults
            else:

data_for_enterpreneur_page["newadminconsults"] = 0

        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)

```

```

        if appclient_object:
            appclient_object = appclient_object[0]
            if indiventrepreneur_object in
appclient_object.indiventrepreneurs:
                data_for_entrepreneur_page["save"] = True
                client_consults =
Appclientconsult.objects.filter(indiventrepreneurback__appclient
= appclient_object, remove_mark = False)
                if client_consults:
                    client_consults = client_consults.count()

data_for_entrepreneur_page["newclientconsults"] =
client_consults
            else:

data_for_entrepreneur_page["newclientconsults"] = 0
            indentrep_backs =
Indiventrepreneurback.objects.select_related("appclient")
            indentrep_backs =
indentrep_backs.filter(indiventrepreneur =
indiventrepreneur_object)
            indentrep_backs = indentrep_backs.order_by("-
back_date")
            data_for_entrepreneur_page["backs"] = indentrep_backs
            return render(request,
"ientapp/indiventrepreneurreviews.html",
data_for_entrepreneur_page)

def indiventrepreneur_subcategory_page_view(request,
indiventrepreneur_id, commodsubcategory_id):
    commodcategories = Commodcategory.objects.all()
    commodcategories =
commodcategories.order_by("commod_category_name")
    for commodcategory in commodcategories:
        subcategories =
Commodsubcategory.objects.filter(commodcategory =
commodcategory)
        subcategories =
subcategories.order_by("commod_subcategory_name")
        commodcategory.subcategories = subcategories
        data_for_entrepreneur_page = {}
        data_for_entrepreneur_page["categories"] =
commodcategories
    try:

```

```

        indiventrepreneur_object =
Indiventrepreneur.objects.select_related("entrepreneur_city")
        indiventrepreneur_object =
indiventrepreneur_object.get(id = indiventrepreneur_id, blocked
= False)
    except Indiventrepreneur.DoesNotExist:
        return render(request, "ientapp/error.html",
{"error_text": "Задано невірний ідентифікатор фізичної особи
підприємця. Будь-ласка, перевірте вказаний номер. Якщо посилання
було згенеровано автоматично, то повідомте про це
адміністратора."})
    try:
        commodsubcategory_object =
Commodsubcategory.objects.select_related("commodcategory")
        commodsubcategory_object =
commodsubcategory_object.get(id = commodsubcategory_id)
    except Commodsubcategory.DoesNotExist:
        return render(request, "ientapp/error.html",
{"error_text": "Задано невірний ідентифікатор підкатегорії
товарів. Будь-ласка, перевірте вказаний номер. Якщо посилання
було згенеровано автоматично, то повідомте про це
адміністратора."})
    data_for_entrepreneur_page["entrepreneur"] =
indiventrepreneur_object
    data_for_entrepreneur_page["subcategory"] =
commodsubcategory_object
    entrep_worktime =
Indiventrepreneurworktime.objects.filter(indiventrepreneur =
indiventrepreneur_object)
    entrep_worktime =
entrep_worktime.order_by("day_of_week")
    data_for_entrepreneur_page["worktime"] =
entrep_worktime
    data_for_entrepreneur_page["ownmode"] = False
    if request.user.is_active:
        entrepreneur_persons =
Indiventrepreneurperson.objects.filter(indiventrepreneur =
indiventrepreneur_object, user_account = request.user, allowed =
True)
        if entrepreneur_persons:
            data_for_entrepreneur_page["ownmode"] = True
            admin_consults =
Indiventrepreneurconsult.objects.filter(indiventrepreneur =
indiventrepreneur_object, remove_mark = False)

```

```

        if admin_consults:
            admin_consults = admin_consults.count()

data_for_entrepreneur_page["newadminconsults"] = admin_consults
    else:

data_for_entrepreneur_page["newadminconsults"] = 0

        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]
            if indiventrepreneur_object in
appclient_object.indiventrepreneurs:
                data_for_entrepreneur_page["save"] = True
                client_consults =
Appclientconsult.objects.filter(indiventrepreneurback__appclient
= appclient_object, remove_mark = False)
                if client_consults:
                    client_consults = client_consults.count()

data_for_entrepreneur_page["newclientconsults"] =
client_consults
    else:

data_for_entrepreneur_page["newclientconsults"] = 0
        indentrep_commodities =
Commodindiventrepreneur.objects.select_related("commodtemplate")
        indentrep_commodities =
indentrep_commodities.select_related("commodtemplate__commodityp
roducer")
        indentrep_commodities =
indentrep_commodities.filter(indiventrepreneur =
indiventrepreneur_object, commodtemplate__commodsubcategory =
commodsubcategory_object)
        indentrep_commodities =
indentrep_commodities.order_by("commodtemplate__commodity_captio
n")
        for indentrep_commodity in indentrep_commodities:
            indentrep_commodity.photo =
Commodindiventrepreneurphoto.objects.filter(commodindiventrepre
neur = indentrep_commodity)[0]

```

```

        if appclient_object:
            for indentrep_commodity in indentrep_commodities:
                if indentrep_commodity in
appclient_object.commodindiventreprenuers:
                    indentrep_commodity.saved = True
                else:
                    indentrep_commodity.saved = False
            data_for_entrepreneur_page["commodities"] =
indentrep_commodities
            return render(request,
"ientapp/indentreprenuersubcategory.html",
data_for_entrepreneur_page)

def indiventreprenuer_consults_page_view(request):
    data_for_entrepreneur_page = {}
    if request.user.is_active:
        try:
            entrepreneur_person =
Indiventreprenuerperson.objects.get(user_account = request.user,
allowed = True)
        except Indiventreprenuer.DoesNotExist:
            return render(request, "ientapp/error.html",
{"error_text": "У вас недостатньо прав для роботи з цією
функцією. Зверніться, будь ласка, до адміністратора."})
            indiventreprenuer_object =
Indiventreprenuer.objects.select_related("entrepreneur_city")
            indiventreprenuer_object =
indiventreprenuer_object.get(id = indiventreprenuer.id, blocked
= False)

            data_for_entrepreneur_page["entrepreneur"] =
indiventreprenuer_object
            if entrepreneur_person:
                admin_consults =
Indiventreprenuerconsult.objects.filter(indiventreprenuer =
indiventreprenuer_object, remove_mark = False)
                if admin_consults:
                    admin_consults = admin_consults.count()

data_for_entrepreneur_page["newadminconsults"] = admin_consults
            else:

data_for_entrepreneur_page["newadminconsults"] = 0
            admin_consults =
Indiventreprenuerconsult.objects.select_related("appclient")

```

```

        admin_consults =
admin_consults.filter(indiventrepneur =
indiventrepneur_object)
        admin_consults = admin_consults.order_by("-
consult_time")
        admin_consults_users = []
        for admin_consult in admin_consults:
            if admin_consult.appclient not in
admin_consults_users:

admin_consults_users.append(admin_consult.appclient)
        for admin_consults_user in
admin_consults_users:
            consults =
Indiventrepneurconsult.objects.filter(indiventrepneur =
indiventrepneur_object, appclient = admin_consults_user)
            for admin_consult in consults:
                admin_consult.answers =
Indiventrepneurconsultanswer.objects.filter(indiventrepneurc
onsult = admin_consult)
                admin_consult.answers =
admin_consult.answers.order_by("-consult_time")
                admin_consults_user.consults = consults
                data_for_enterpreneur_page["adminconsults"] =
admin_consults
            else:
                return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
                return render(request,
"ientapp/indentrepneurconsults.html",
data_for_enterpreneur_page)

def appclient_consults_page_view(request):
    data_for_appclient_page = {}
    if request.user.is_active:
        appclient_object =
Appclient.objects.prefetch_related("indiventrepneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]

```

```

        client_consults =
Appclientconsult.objects.filter(indiventreprenurback__appclient
= appclient_object, remove_mark = False)
        if client_consults:
            client_consults = client_consults.count()

data_for_appclient_page["newclientconsults"] = client_consults
        else:

data_for_appclient_page["newclientconsults"] = 0
            client_consults =
Appclientconsult.objects.select_related("indiventreprenurback")
            client_consults =
client_consults.select_related("indiventreprenurback__appclient
")

            client_consults =
client_consults.select_related("indiventreprenurback__indiventr
eprenur")

            client_consults =
client_consults.filter(appclient = appclient_object)
            appclient_addressants = []
            for client_consult in client_consults:
                if
client_consult.indiventreprenurback.appclient not in
appclient_addressants:

                    curr_consult =
client_consult.indiventreprenurback.appclient
                    curr_client_consults =
Appclientconsult.objects.select_related("indiventreprenurback")
                    curr_client_consults =
curr_client_consults.select_related("indiventreprenurback__appc
lient")

                    curr_client_consults =
curr_client_consults.select_related("indiventreprenurback__indi
ventreprenur")

                    curr_client_consults =
curr_client_consults.filter(indiventreprenurback__appclient =
curr_consult)

                    curr_client_consults =
curr_client_consults.order_by("-consult_time")
                    for curr_client_consult in
curr_client_consults:

```

```

curr_answers =
Appclientconsultanswer.objects.filter(appclientconsult =
curr_client_consult)

curr_answers =
curr_answers.order_by("-consult_time")
curr_client_consult.answers =
curr_answers

curr_consult.consults =
curr_client_consults

appclient_addressants.append(curr_consult)
data_for_appclient_page["clientconsults"] =
appclient_addressants

entrepreneur_consults =
Indiventrepreneurconsult.objects.select_related("indiventreprene
ur")

entrepreneur_consults =
entrepreneur_consults.filter(appclient = appclient_object)
entrepreneur_consults =
entrepreneur_consults.order_by("-consult_time")
entrepreneur_addressers = []
for entrepreneur_consult in
entrepreneur_consults:
    if entrepreneur_consult.indiventrepreneur
not in entrepreneur_addressers:
        curr_consult =
entrepreneur_consult.indiventrepreneur
curr_ent_consults =
Indiventrepreneurconsult.objects.select_related("indiventreprene
ur")

curr_ent_consults =
curr_ent_consults.filter(indiventrepreneur = curr_consult)
curr_ent_consults =
curr_ent_consults.order_by("-consult_time")
for curr_ent_consult in
curr_ent_consults:
    curr_answers =
Indiventrepreneurconsultanswer.objects.filter(indiventrepreneurc
onsult = curr_ent_consult)

curr_answers =
curr_answers.order_by("-consult_time")
curr_ent_consult.answers =
curr_answers

curr_consult.consults =
curr_client_consults

```

```

entrepreneur_addressers.append(curr_consult)

data_for_appclient_page["enterpconsultaddressers"] =
entrepreneur_addressers
    else:
        return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
        return render(request,
"ientapp/appclientconsults.html", data_for_appclient_page)

def appclient_to_consults_page_view(request):
    data_for_appclient_page = {}
    if request.user.is_active:
        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]
            client_consults =
Appclientconsult.objects.filter(indiventrepreneurback__appclient
= appclient_object, remove_mark = False)
            if client_consults:
                client_consults = client_consults.count()

data_for_appclient_page["newclientconsults"] = client_consults
    else:

data_for_appclient_page["newclientconsults"] = 0
        client_consults =
Appclientconsult.objects.select_related("indiventrepreneurback")
        client_consults =
client_consults.select_related("indiventrepreneurback__appclient
")
        client_consults =
client_consults.select_related("indiventrepreneurback__indiventr
epreneur")
        client_consults =
client_consults.filter(indiventrepreneurback__appclient =
appclient_object)
        client_consults = client_consults.order_by("-
consult_time")

```

```

        appclient_addressers = []
        for client_consult in client_consults:
            if client_consult.appclient not in
appclient_addressers:
                curr_consult = client_consult.appclient
                curr_client_consults =
Appclientconsult.objects.select_related("indiventrepreneurback")
                curr_client_consults =
curr_client_consults.select_related("indiventrepreneurback__appc
lient")
                curr_client_consults =
curr_client_consults.select_related("indiventrepreneurback__indi
ventrepreneur")
                curr_client_consults =
curr_client_consults.filter(indiventrepreneurback =
curr_consult)
                curr_client_consults =
curr_client_consults.order_by("-consult_time")
                for curr_client_consult in
curr_client_consults:
                    curr_answers =
Appclientconsultanswer.objects.filter(appclientconsult =
curr_client_consult)
                    curr_answers =
curr_answers.order_by("-consult_time")
                    curr_client_consult.answers =
curr_answers
                    curr_consult.consults =
curr_client_consults
                    appclient_addressers.append(curr_consult)

data_for_appclient_page["clientconsultaddressers"] =
appclient_addressers
    else:
        return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
        return render(request,
"ientapp/toappclientconsults.html", data_for_appclient_page)

def appclient_backs_page_view(request, appclient_name =
None):
    data_for_appclient_page = {}
    if not appclient_name and request.user.is_active:

```

```

        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]
            backs =
Indiventrepreneurback.objects.select_related("indiventrepreneur"
)

            backs = backs.order_by("-back_date")
            for back in backs:
                if back.indiventrepreneur in
appclient_object.indiventrepreneurs:
                    back.saved = True
                else:
                    back.saved = False
            data_for_appclient_page["entrepbacks"] = backs
            client_consults =
Appclientconsult.objects.filter(indiventrepreneurback__appclient
= appclient_object, remove_mark = False)
            if client_consults:
                client_consults = client_consults.count()

data_for_entrepreneur_page["newclientconsults"] =
client_consults

            else:

data_for_entrepreneur_page["newclientconsults"] = 0
            elif appclient_name:
                appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
                appclient_object =
appclient_object.filter(user_account__username = appclient_name)
                if appclient_object:
                    appclient_object = appclient_object[0]
                    backs =
Indiventrepreneurback.objects.select_related("indiventrepreneur"
)

                    backs = backs.order_by("-back_date")
                    for back in backs:
                        if back.indiventrepreneur in
appclient_object.indiventrepreneurs:
                            back.saved = True
                        else:

```

```

        back.saved = False
        data_for_appclient_page["entrepbacks"] = backs
        client_consults =
Appclientconsult.objects.filter(indiventreprenneurback__appclient
= appclient_object, remove_mark = False)
        if client_consults:
            client_consults = client_consults.count()

data_for_enterpreneur_page["newclientconsults"] =
client_consults
        else:

data_for_enterpreneur_page["newclientconsults"] = 0
        else:
            return render(request, "ientapp/error.html",
{"error_text": "Невірно заданий логін користувача. Такого не
знайдено."})
        else:
            return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
            return render(request, "ientapp/appclientbacks.html",
data_for_appclient_page)

def appclient_orders_page_view(request, order_state = 0):
    data_for_appclient_page = {}
    if request.user.is_active:
        appclient_object =
Appclient.objects.prefetch_related("indiventrepreneurs")
        appclient_object =
appclient_object.filter(user_account = request.user)
        if appclient_object:
            appclient_object = appclient_object[0]
            appclient_orders =
Appclientorder.objects.select_related("indiventreprenneur")
            appclient_orders =
appclient_orders.filter(appclient = appclient_object)
            if order_state != 0:
                appclient_orders =
appclient_orders.filter(order_state = order_state)
            appclient_orders = appclient_orders.order_by("-
order_time")

            for appclient_order in appclient_orders:

```

```

        apporder_commodities =
Appclientordercommodity.objects.filter(appclientorder =
appclient_order)
        for apporder_commodity in
apporder_commodities:
            if apporder_commodity in
appclient_object.commodindiventreneurs:
                apporder_commodity.saved = True
            else:
                apporder_commodity.saved = False
            appclient_order.commodities =
apporder_commodities
            if appclient_order.indiventreneur in
appclient_object.indiventreneurs:
                appclient_order.saved = True
            else:
                appclient_order.saved = False
            data_for_appclient_page["orders"] =
appclient_orders
                client_consults =
Appclientconsult.objects.filter(indiventreneurback__appclient
= appclient_object, remove_mark = False)
                if client_consults:
                    client_consults = client_consults.count()

data_for_enterpreneur_page["newclientconsults"] =
client_consults
                else:

data_for_enterpreneur_page["newclientconsults"] = 0
                else:
                    return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
                    return render(request, "ientapp/appclientorders.html",
data_for_appclient_page)

def indiventreneur_orders_page_view(request,
order_state, order_sort, appclientorder_id = None):
    data_for_enterpreneur_page = {}
    if request.user.is_active:
        try:

```

```

        entrepreneur_person =
Indiventpreneurperson.objects.get(user_account = request.user,
allowed = True)
        except Indiventpreneur.DoesNotExist:
            return render(request, "ientapp/error.html",
{"error_text": "У вас недостатньо прав для роботи з цією
функцією. Зверніться, будь ласка, до адміністратора."})
            indiventpreneur_object =
Indiventpreneur.objects.select_related("entrepreneur_city")
            indiventpreneur_object =
indiventpreneur_object.get(id = indiventpreneur.id, blocked
= False)

            data_for_enterpreneur_page["entrepreneur"] =
indiventpreneur_object
            if entrepreneur_person:
                admin_consults =
Indiventpreneurconsult.objects.filter(indiventpreneur =
indiventpreneur_object, remove_mark = False)
                if admin_consults:
                    admin_consults = admin_consults.count()

data_for_enterpreneur_page["newadminconsults"] = admin_consults
                else:

data_for_enterpreneur_page["newadminconsults"] = 0
                entrepreneur_orders =
Appclientorder.objects.select_related("appclient")
                entrepreneur_orders =
entrepreneur_orders.filter(indiventpreneur =
indiventpreneur_object)
                if appclientorder_id:
                    entrepreneur_orders =
entrepreneur_orders.filter(id = appclientorder_id)
                if order_state != 0:
                    entrepreneur_orders =
entrepreneur_orders.filter(order_state = order_state)
                if order_sort == 0:
                    entrepreneur_orders =
entrepreneur_orders.order_by("order_time")
                elif order_sort == 1:
                    entrepreneur_orders =
entrepreneur_orders.order_by("-order_time")
                elif order_sort == 2:

```

```

        entrepreneur_orders =
entrepreneur_orders.order_by("send_date")
        elif order_sort == 3:
            entrepreneur_orders =
entrepreneur_orders.order_by("-send_date")
        elif order_sort == 4:
            entrepreneur_orders =
entrepreneur_orders.order_by("delivery_date")
        elif order_sort == 5:
            entrepreneur_orders =
entrepreneur_orders.order_by("-delivery_date")
        elif order_sort == 6:
            entrepreneur_orders =
entrepreneur_orders.order_by("order_price")
        elif order_sort == 7:
            entrepreneur_orders =
entrepreneur_orders.order_by("-order_price")
        for appclient_order in appclient_orders:
            apporder_commodities =
Appclientordercommodity.objects.filter(appclientorder =
appclient_order)
            appclient_order.commodities =
apporder_commodities
            if appclient_order.indiventrepreneur in
appclient_object.indiventrepreneurs:
                appclient_order.saved = True
            else:
                appclient_order.saved = False
            data_for_appclient_page["orders"] =
appclient_orders
        else:
            return render(request, "ientapp/error.html",
{"error_text": "Без авторизації неможливо виконати цю функцію.
Спершу авторизуйтеся, а далі поверніться."})
            return render(request,
"ientapp/indentrepreneurorders.html",
data_for_enterpreneur_page)

```

ДОДАТОК В
Слайди презентації

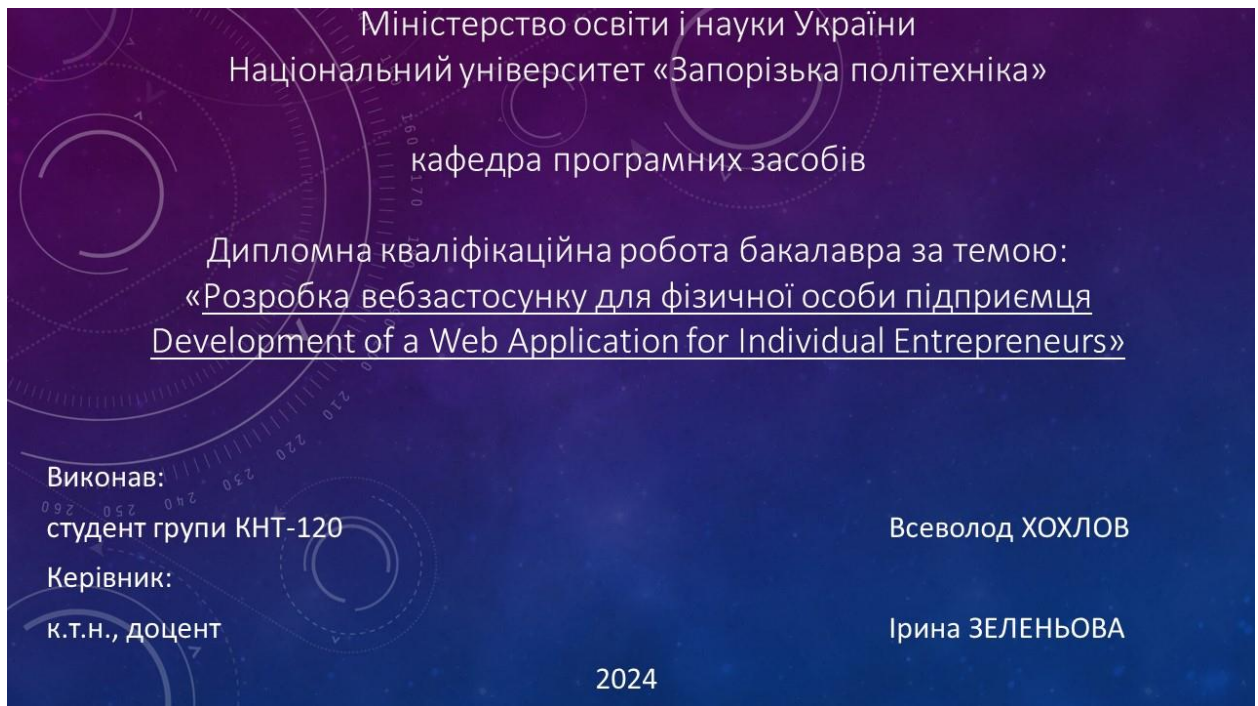


Рисунок В.1 – Перший слайд

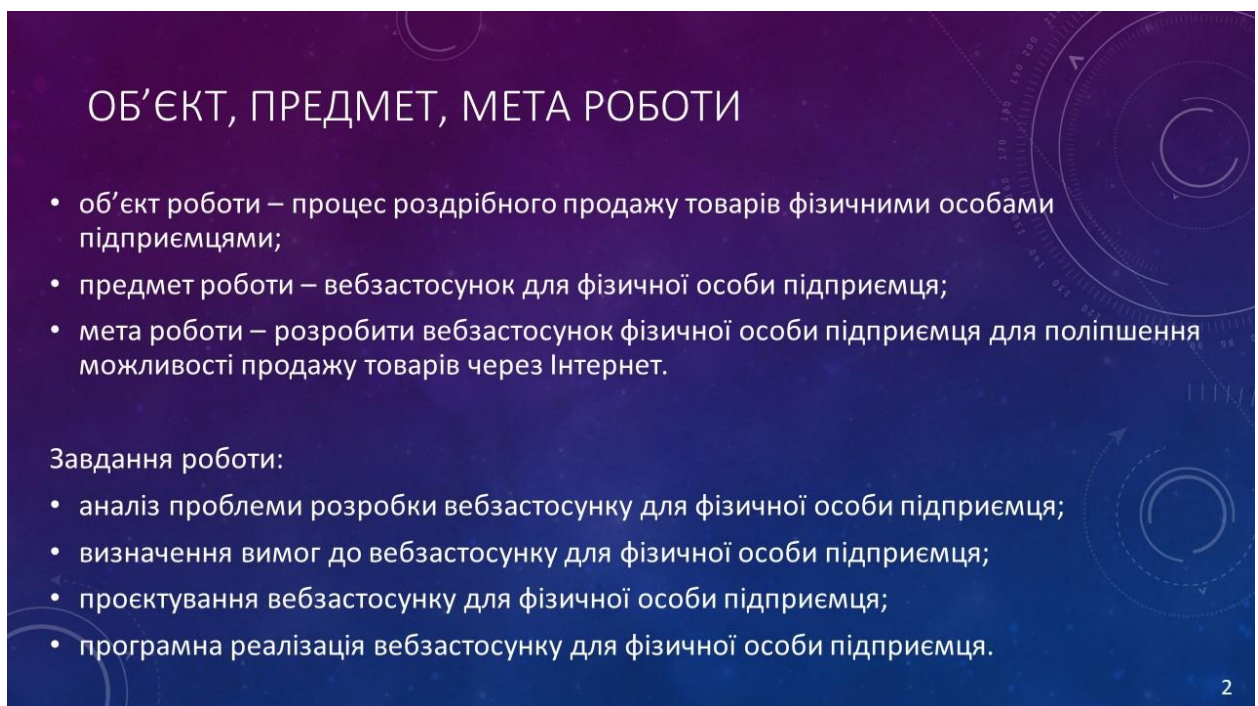


Рисунок В.2 – Об'єкт, предмет, мета роботи

ПОРІВНЯННЯ РОЗРОБКИ З ПРОАНАЛІЗОВАНИМИ АНАЛОГАМИ ВЕБЗАСТОСУНКІВ ДЛЯ ФІЗИЧНИХ ОСІБ ПІДПРИЄМЦІВ

Критерій порівняння	Prom.ua	Ria.com	Розробка
Пошук товарів	+	+	+
Створення сторінок фізичної особи підприємця	Самостійне будь-яким користувачем	Самостійне будь-яким користувачем	Керуючим за наданими даними
Перегляд товарів фізичної особи підприємця за категорією	+	+	+
Відкладання товарів	+	+	+
Створення базових карток товарів	—	—	+
Консультування продавцями клієнтів	+	+	+
Створення повноцінних відгуків клієнтами про фізичних осіб підприємців	+	—	+
Консультування клієнтів іншими клієнтами щодо фізичних осіб підприємців	—	—	+

3

Рисунок В.3 – Порівняння розробки з проаналізованими аналогами вебзастосунків для фізичних осіб підприємців

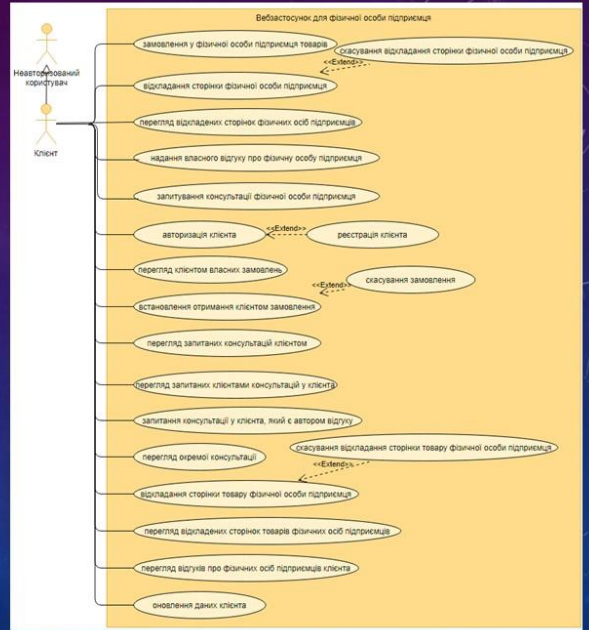
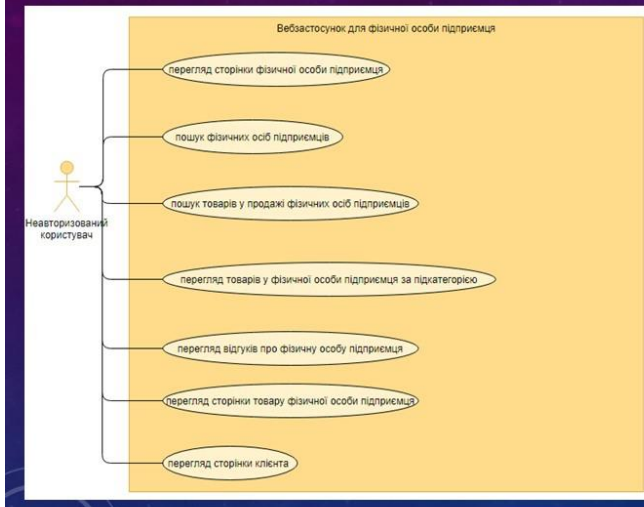
ПОРІВНЯННЯ ФРЕЙМВОРКІВ ДЛЯ РОЗРОБКИ ВЕБЗАСТОСУНКУ ДЛЯ ФІЗИЧНОЇ ОСОБИ ПІДПРИЄМЦЯ

Критерій порівняння	Express JS	Django
Мова розробки	JavaScript	Python
Концепція розробки	Мінімалістичність, гнучкість	Включеність найкращих практик за замовчуванням
Підтримка баз даних	Можна обрати систему на основі підключення модулів	Робота з базами даних типова, вбудована підтримка об'єктно-реляційного відображення
Вимоги до вебсервера	Не обмежені конкретним	Не обмежені конкретним
Безпека	Необхідні спеціальні бібліотеки, які потрібно вибирати	Інструменти включені за багатьма проблемами
Тип підтримуваних проєктів	Малі та середні	Складні, масштабні

4

Рисунок В.4 – Порівняння фреймворків для розробки вебзастосунку для фізичної особи підприємця

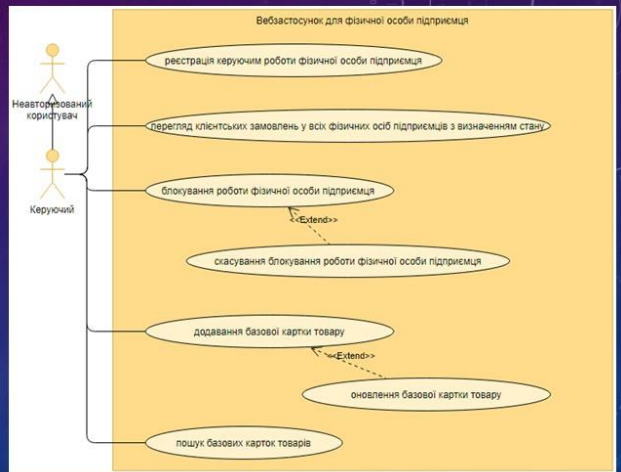
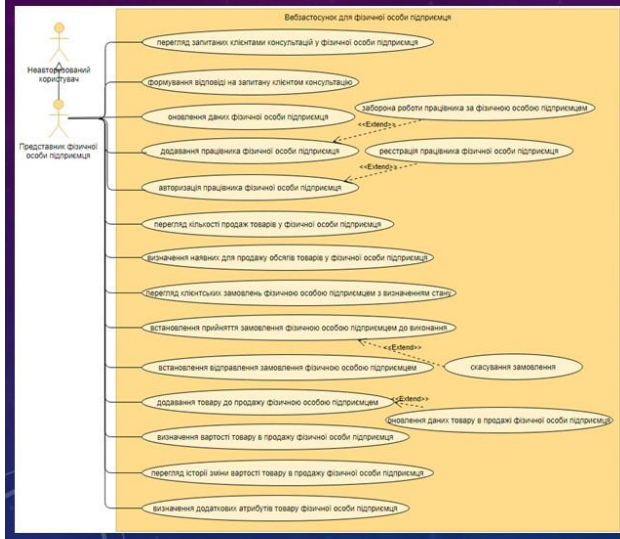
ДІАГРАМИ ВАРІАНТІВ ВИКОРИСТАННЯ ПРОГРАМИ



5

Рисунок В.5 – Діаграми варіантів використання програми

ДІАГРАМИ ВАРІАНТІВ ВИКОРИСТАННЯ ПРОГРАМИ



6

Рисунок В.6 – Діаграми варіантів використання програми



ДІАГРАМА СТАНІВ ЗАМОВЛЕННЯ У ФІЗИЧНОЇ ОСОБИ ПІДПРИЄМЦЯ



Рисунок В.8 – Діаграма станів замовлення у фізичної особи підприємця

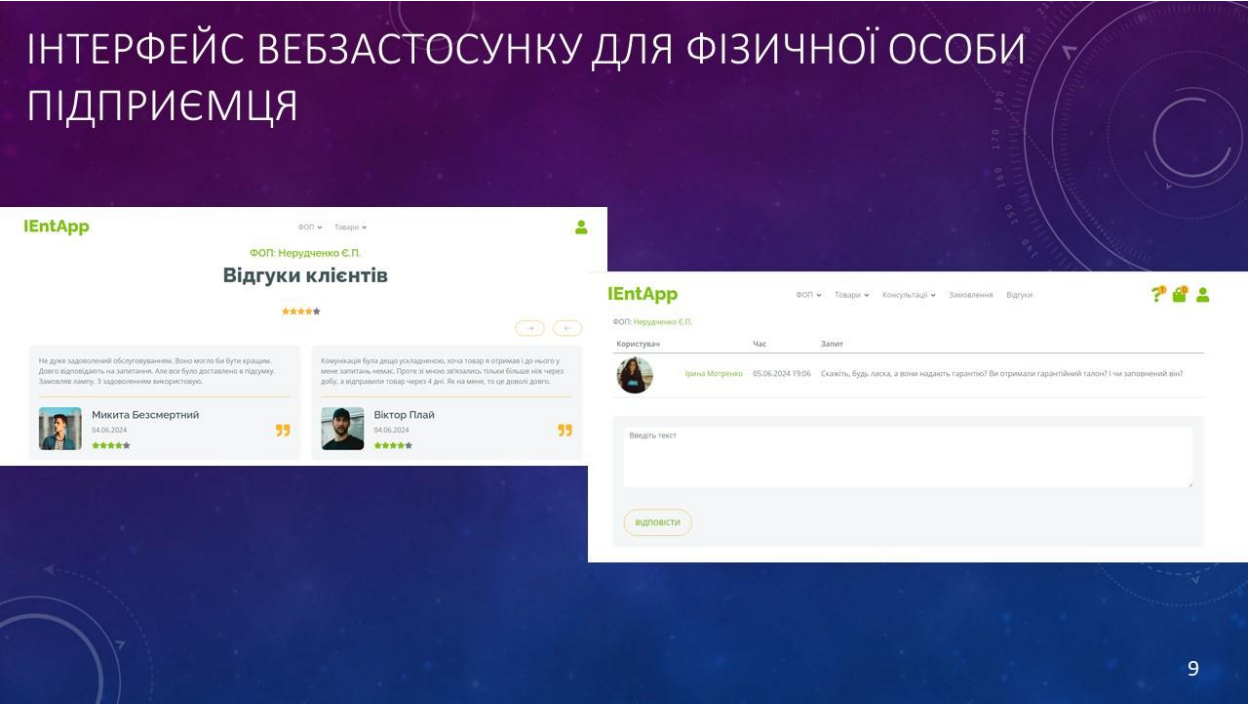


Рисунок В.9 – Інтерфейс вебзастосунку для фізичної особи підприємця

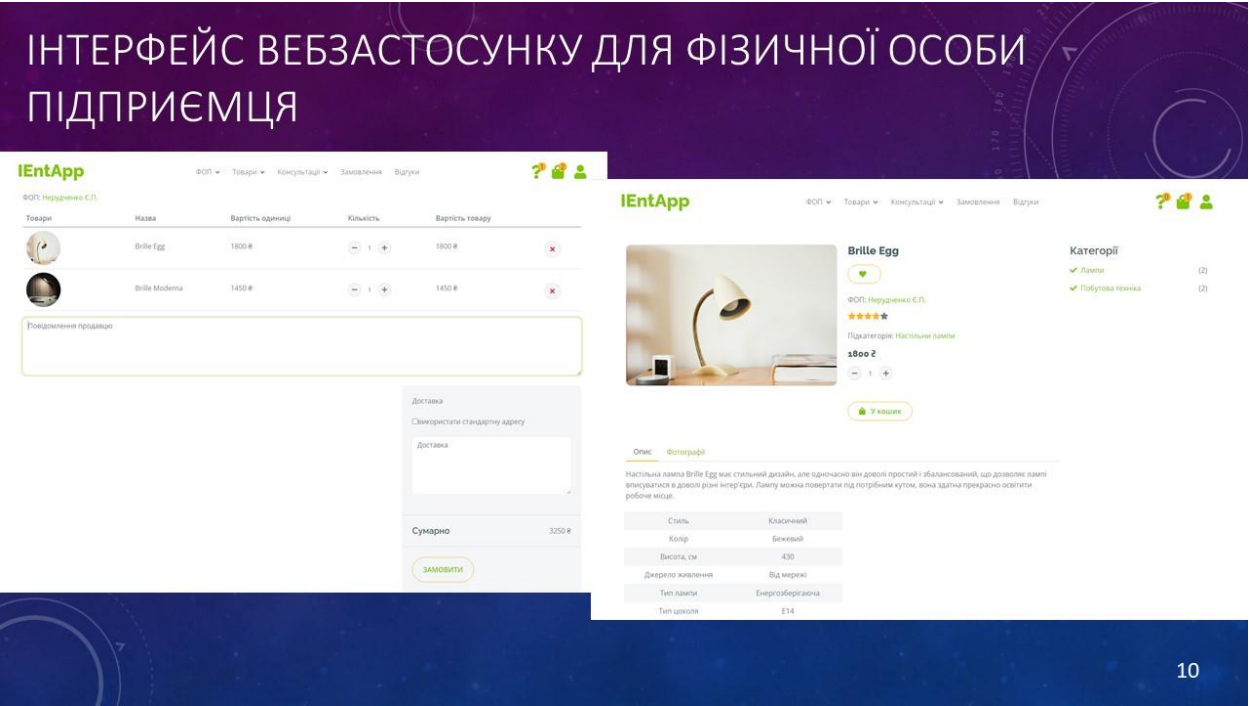
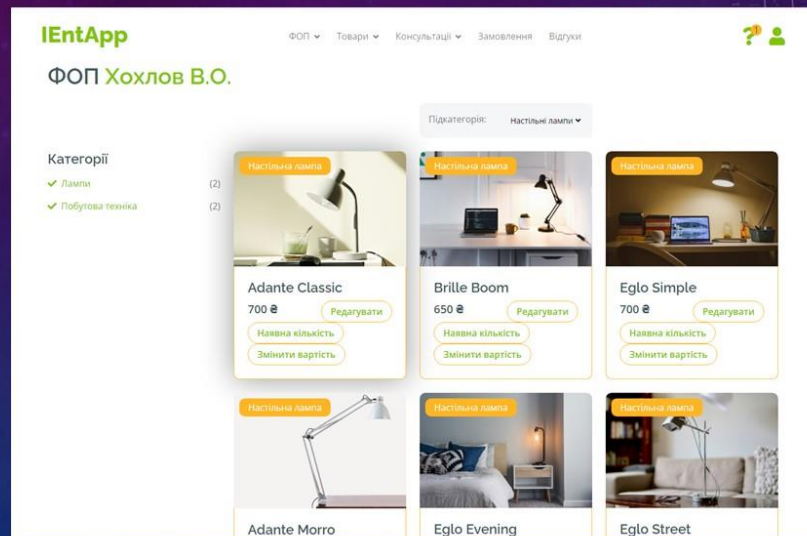


Рисунок В.10 – Інтерфейс вебзастосунку для фізичної особи підприємця

ІНТЕРФЕЙС ВЕБЗАСТОСУНКУ ДЛЯ ФІЗИЧНОЇ ОСОБИ ПІДПРИЄМЦЯ



11

Рисунок В.11 – Інтерфейс вебзастосунку для фізичної особи підприємця

ВИСНОВКИ

- У роботі виконано розробку вебзастосунку для фізичної особи підприємця. Для цього було проведено огляд і порівняння аналогів, визначено технології для розробки, що охопили Python-фреймворк для веброзробки Django, систему MySQL для підтримки БД, визначено структуру БД, варіанти використання програми клієнтом, фізичною особою підприємцем, керуючим, представлено порядок зміни станів клієнтського замовлення на купівлю товарів у фізичної особи підприємця на основі діаграми станів, створені функції та класи програми. Усі функції програми відповідно до варіантів використання реалізовані, а ця реалізація є успішною, що підтверджено під час тестування програми.
- Вебзастосунок призначений для забезпечення фізичним особам підприємцям можливості виставляти на продаж власні товари, пошуку потрібних товарів, оформлення замовлень на придбання товарів у фізичних осіб підприємців клієнтами, керування замовленнями, керування доступними товарами. Програма дозволяє підтримувати роботу для цілого ряду фізичних осіб підприємців, таким чином сприяючи формуванню певного майданчика, що на практиці можливо пристосувати, наприклад, для продажу товарів за певною нішею.

Рисунок В.12 – Висновки