Міністерство освіти і науки України

Національний університет «Одеська політехніка»

Інститут комп’ютерних систем

Кафедра інформаційних систем

Іщенко Олександр Геннадійович,

студент групи АІ-212

ДИСЦИПЛІНА

Об’єктно-орієнтоване програмування

КУРСОВА РОБОТА

Розробка веб-додатку для ведення обліку особистих витрат

Спеціальність:

122 Комп’ютерні науки

Освітня програма:

Комп’ютерні науки

Керівник:

Годовиченко Микола Анатолійович,

кандидат технічних наук, доцент

Одеса – 2023

ЗМІСТ

Анотація 3

Вступ 4

[1 Проєктування веб-додатку для ведення обліку особистих витрат](#_Toc137028414) 5

1.1 [Мета та задачі веб-додатку 16](#_Toc137028415)

[1.2 Визначення функціональних вимог до веб-додатку 17](#_Toc137028416)

[1.3 Формування користувацьких історій веб-додатку 19](#_Toc137028417)

[2 Програмна реалізація веб-додатку для ведення заміток 27](#_Toc137028423)

[2.1 Структура серверного програмного проєкту 27](#_Toc137028424)

[2.2 Інструкція користувача веб-ресурсу 34](#_Toc137028428)

[2.3 Висновки до третього розділу 41](#_Toc137028430)

[ВИСНОВКИ 43](#_Toc137028431)

[ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 44](#_Toc137028432)

АНОТАЦІЯ

Курсова робота присвячена розробці веб-додатку для ведення обліку особистих витрат. Додаток призначений для ведення щоденних витрат, щоб забезпечити користувачам простий, зручний і надійний інструмент для контролю своїх фінансових витрат. Він має допомогти користувачам визначити основні джерела витрат та ефективно розподіляти ці витрати. Це дозволить їм краще розуміти свої фінансові витрати та здійснювати розумніші фінансові рішення, що допоможе їм досягти своїх фінансових цілей. Основною метою цього проекту є створення простого, ефективного та інтуїтивно зрозумілого інструменту, який дозволяє користувачам легко створювати, редагувати, видаляти та відслідковувати свої витрати. В ході розробки додатку будуть використані сучасні технології та практики розробки веб-додатків, зокрема мова програмування Java з використанням фреймворку Spring Framework та його розширенням Spring Boot, Spring Data JPA, шаблонизатор Thymeleaf та архітектура MVС.

ABSTRACT

The course work is devoted to the development of a web application for keeping records of personal expenses. The application is designed to keep track of daily expenses to provide users with a simple, convenient and reliable tool for controlling their financial expenditures. It should help users to identify the main sources of expenses and allocate them efficiently. This will allow them to better understand their financial expenses and make smarter financial decisions, which will help them achieve their financial goals. The main goal of this project is to create a simple, efficient and intuitive tool that allows users to easily create, edit, delete and track their expenses. The application will be developed using modern web application development technologies and practices, including the Java programming language using the Spring Framework and its Spring Boot extension, Spring Data JPA, Thymeleaf templating engine and MVC architecture.

ВСТУП

Зважаючи на зростаючу важливість фінансової грамотності та управління особистими фінансами, веб-додатки для ведення обліку особистих витрат стають дедалі популярнішими. Такі додатки надають користувачам зручні інструменти для відстеження та аналізу своїх фінансових витрат.

Управління витратами є важливою складовою фінансового благополуччя. Веб-додаток для ведення щоденних витрат дозволить користувачам точно відстежувати, куди йдуть їх гроші, та аналізувати свої витрати. Це допоможе їм більш свідомо планувати свої фінанси, знаходити можливості для економії та приймати розумні фінансові рішення.

Веб-додаток для ведення витрат має надавати простий та інтуїтивно зрозумілий інтерфейс, що дозволяє користувачам з легкістю вносити та категоризувати свої витрати. Додаток повинен також надавати зручні функції аналізу та статистики, які допоможуть користувачам отримати цінну інформацію про свої фінанси.

Використання сучасних технологій та розробка веб-додатку на базі стабільних та ефективних фреймворків дозволять забезпечити надійну та швидку роботу додатку. Програмування на веб-платформі надає можливість забезпечити доступ до даних з будь-якого пристрою з підключенням до Інтернету, що робить додаток зручним та доступним для користувачів в будь-який час і в будь-якому місці.

Розробка веб-додатка для ведення щоденних витрат вимагає уважного планування, дизайну та програмування. Налагодження зв'язку з базою даних, реалізація функцій керування витратами та статистики, а також забезпечення безпеки даних - це важливі етапи процесу розробки.

Зрозуміння основних потреб користувачів і розробка зручного веб-додатку для ведення особистих витрат допоможуть користувачам краще контролювати свої фінанси, планувати витрати та досягати своїх фінансових цілей.

Для досягнення цієї мети необхідно виконати наступні задачі:

* здійснити огляд інших додатків для створення користувацьких заміток;
* проаналізувати та обрати сучасні технології для створення веб додатку;
* провести проєктування веб додатку своїх фінансових витрат;
* виконати програмну реалізацію спроєктованого веб додатку.

# проєктування ВЕБ-додатку для ведення обліку особистих витрат

## Мета та задачі веб-додатку

Мета веб-додатку для ведення обліку особистих витрат полягає в створенні зручного, ефективного і надійного інструменту для користувачів, що дозволить їм легко відстежувати та контролювати свої фінансові витрати. Основною метою додатку є надання користувачам зрозумілого та зручного інтерфейсу, що спрощує процес ведення обліку та аналізу особистих витрат. Додаток повинен також надавати зручний та інтуїтивно зрозумілий інтерфейс для введення даних про витрати.

Задачі веб-додатку можуть включати:

* Реєстрація та авторизація користувачів: додаток повинен забезпечувати можливість реєстрації нових користувачів та авторизацію вже існуючих користувачів для доступу до особистого обліку витрат.
* Додавання та відстеження витрат: користувачі повинні мати можливість додавати нові записи про свої витрати, вказуючи категорію, суму та дату. Додаток повинен зберігати ці записи та надавати зручний спосіб їх подальшого перегляду та аналізу.
* Категоризація витрат: додаток може надавати можливість створення різних категорій витрат (наприклад, їжа, транспорт, розваги) для більш зручного організування та фільтрації записів про витрати.
* Аналіз та статистика витрат: веб-додаток може надавати функціонал для аналізу та візуалізації витрат, дозволяючи користувачам переглядати статистику за певний період часу, визначати основні джерела витрат та встановлювати фінансові цілі.
* Нагадування та планування: додаток може надавати можливість встановлення нагадувань про платежі, регулярні витрати або фінансові цілі, що допомагатимуть користувачам вчасно планувати та керувати своїми фінансами.
* Забезпечення безпеки даних: веб-додаток повинен забезпечувати захист особистих фінансових даних користувачів, використовуючи відповідні методи шифрування та заходи безпеки.

В цілому, мета веб-додатку для ведення обліку особистих витрат полягає в наданні користувачам зручного та ефективного інструменту для контролю своїх фінансових витрат, а задачі додатку включають створення зручного інтерфейсу, збереження та аналіз записів про витрати, категоризацію, аналіз та планування фінансів, а також забезпечення безпеки даних користувачів.

## Визначення функціональних вимог до мобільного додатку

Додаток повинен бути доступним для використання всім, незалежно від рівня їхньої фінансової грамотності. Кожен користувач має мати свій окремий обліковий запис зі списком його витрат. Додаток має забезпечувати можливість додавання, редагування та видалення усіх категорій витрат для всіх користувачів. Користувачі повинні мати змогу переглядати статистику своїх витрат та доходів за певний період часу, яка повинна бути представлена у зручному та зрозумілому вигляді.

Акторами веб-додатку для ведення обліку особистих витрат є «Користувач» та «Гість»:

* Користувач: може додавати нові категорії витрат, додавати нові витрати, переглядати свої витрати та доходи за певний період, редагувати та видаляти свої записи, переглядати статистику тощо, редагувати налаштування.
* Гість: має доступ лише до деяких функцій додатку, таких, наприклад, як ознайомлення з можливостями додатку та можливість реєстрації як користувача в додатку.

## Формування користувацьких історій веб-додатку

Проектування діаграми прецедентів дозволяє визначити такі користувацькі історії до веб-додатку для ведення обліку особистих витрат:

US1 Як гість, я хочу мати можливість ознайомитись з можливостями додатку, щоб зрозуміти, чи цей додаток відповідає моїм потребам.

Критерії прийнятності:

* На вітальній сторінці присутній перелік можливостей додатку з коротким описом кожної з них.
* Користувач може детальніше ознайомитись з кожною можливістю, натиснувши на неї.
* Після ознайомлення з можливостями додатку користувач може перейти до реєстрації або авторизації.

Деталі історії:

* Гість, відкриває веб-додаток та потрапляє на вітальну сторінку.
* Гість бачить перелік можливостей додатку та короткий опис кожної з них.
* Гість може детальніше ознайомитись з кожною з можливостей, натиснувши на неї.
* Після ознайомлення з можливостями додатку, гість може перейти до реєстрації або авторизації для доступу до повного функціоналу додатку.

US2 Я, як гість, хочу мати можливість зареєструвати свій аккаунт в додатку.

Критерії прийнятності:

* Користувач може ввести свої особисті дані (ім'я, електронну пошту та пароль);
* Система перевіряє правильність введених даних;
* Користувач отримує підтвердження про успішну реєстрацію та може увійти до свого аккаунту.

Деталі історії:

* Користувач запускає додаток;
* Користувач вибирає опцію реєстрації;
* Користувач вводить свої особисті дані (ім'я, електронну пошту та пароль);
* Система перевіряє правильність введених даних;
* Якщо дані введені правильно, система створює новий аккаунт;
* Користувач отримує підтвердження про успішну реєстрацію та може увійти до свого аккаунту.

US3 Я, як користувач, хочу мати можливість корегувати свій аккаунт в додатку.

Критерії прийнятності:

* Користувач може змінювати свої особисті дані (ім'я, електронну пошту та пароль);
* Система перевіряє правильність введених даних;
* Користувач отримує підтвердження про успішне оновлення даних.

Деталі історії:

* Користувач входить у свій аккаунт;
* Користувач обирає опцію "Змінити дані";
* Користувач вводить нові дані;
* Система перевіряє валідність введених даних та їх унікальність;
* Після успішної зміни даних користувача система перенаправляє його на домашню сторінку.

US4 Я, як користувач, хочу мати можливість додавати нові витрати до свого списку.

Критерії прийнятності:

* Користувач може ввести назву витрати та її вартість;
* Введені дані зберігаються в базі даних додатку;

Деталі історії:

* Користувач входить у свій обліковий запис у додатку;
* Користувач вибирає опцію "Додати витрати";
* Користувач вибирає дату.
* Користувач вводить назву витрати та її вартість;
* Після введення даних, користувач натискає кнопку "Додати";
* Система зберігає дані у базі даних;

US5 Я, як користувач, хочу мати можливість редагувати існуючі витрати у своєму списку.

Критерії прийнятності:

* Користувач може редагувати будь-яку витрату у своєму списку.
* Користувач може змінити назву, дату, суму та опис витрати.
* Зміни повинні бути збережені у базі даних і відображені в списку витрат.

Деталі історії:

* Користувач входить у свій обліковий запис у додатку та переходить до списку витрат.
* Користувач обирає витрату, яку бажає відредагувати та натискає на неї.
* Користувач змінює необхідні поля та натискає на кнопку "Зберегти зміни".
* Система перевіряє коректність введених даних та зберігає зміни у базі даних.
* Список витрат оновлюється та показує нові дані про відредаговану витрату.

US6 Я, як користувач, хочу мати можливість видаляти витрати зі свого списку.

Критерії прийнятності:

* Користувач може видалити будь-яку витрату зі свого списку
* Підтвердження видалення відображається користувачеві перед остаточним видаленням витрати
* Після видалення витрати, вона більше не відображається у списку витрат користувача.

Деталі історії:

* Користувач відкриває список своїх витрат у додатку.
* Користувач виділяє витрату, яку бажає видалити зі списку.
* Додаток показує повідомлення для підтвердження видалення витрати.
* Користувач натискає на кнопку "Підтвердити видалення".
* Витрата видаляється зі списку витрат користувача і більше не відображається у додатку.

US7 Я, як користувач, хочу мати можливість переглядати свій список витрат за певний період.

Критерії прийнятності:

* Користувач може вибрати період для перегляду витрат (наприклад, за останній місяць, тиждень, день тощо);
* Список витрат відображається вірно і зрозуміло, включаючи назву, суму витрат, дату тощо;
* Користувач може переглядати свій список.

Деталі історії:

* Користувач входить до свого облікового запису в додатку;
* Користувач вибирає опцію "Переглянути список витрат" або подібну опцію на головному екрані додатку;
* Додаток показує список всіх витрат, які були занесені користувачем;
* Користувач може використовувати фільтри, щоб переглядати витрати за конкретний період (наприклад, за тиждень, за місяць, за рік тощо);
* Користувач може сортувати витрати за різними параметрами, наприклад, за датою або за сумою;
* Користувач може переглядати деталі кожної витрати, такі як назва, дата, сума тощо;
* Користувач може закрити список витрат та повернутися на головний екран додатку.

US8 Я, як користувач, хочу мати можливість налаштовувати бюджетні обмеження та ліміти.

Критерії прийнятності:

* Користувач може встановлювати бюджетні обмеження та ліміти на період.
* Встановлені бюджетні обмеження та ліміти відображаються на дашборді користувача.
* Користувач може змінювати встановлені бюджетні обмеження та ліміти.

Деталі історії:

* Користувач входить в додаток і переходить до налаштувань.
* В налаштуваннях користувач вибирає опцію "Ліміти витрат".
* Користувач вводить максимальну суму, яку він може витрачати за день, тиждень, місяць або інший встановлений період.
* Користувач вибирає, яким чином він отримуватиме сповіщення, якщо його витрати перевищують встановлені ліміти (повідомлення в додатку, електронна пошта, повідомлення на телефоні тощо).
* Встановлені бюджетні обмеження та ліміти відображаються на дашборді користувача.
* Користувач може змінювати встановлені бюджетні обмеження та ліміти на сторінці "Налаштування".

US9 Я, як користувач, хочу мати можливість отримувати сповіщення, якщо я перевищую ліміти.

Критерії прийнятності:

* Користувач може налаштувати ліміти для своїх витрат.
* Користувач може вказати, яким чином отримувати сповіщення про перевищення лімітів (повідомлення в додатку, електронна пошта, повідомлення на телефоні тощо).
* Користувач отримує сповіщення, якщо його витрати перевищують встановлені ліміти.
* Оповіщення повинні бути достатньо частими, щоб користувач міг вчасно реагувати на перевищення ліміту.

Деталі історії:

* Якщо користувач витратив більше встановленого ліміту, він отримує сповіщення відповідно до обраного способу повідомлення.
* Користувач може переглянути статистику своїх витрат за певний період, щоб визначити, чому він перевищив ліміт і як може заощадити гроші у майбутньому.
* Може відображати кольором наближення витрат до лимітів

# ПРОГРАМНА РЕАЛІЗАЦІЯ ВЕБ-додатку для ведення щоденних витрат

## Структура серверного програмного проекту

У процесі розробки додатку використані такі сучасні технології та практики розробки веб-додатків, як мова програмування Java, фреймворк Spring Framework з використанням розширення Spring Boot, Spring Data JPA, шаблонізатор Thymeleaf та архітектурний шаблон MVС. Використання стеку Spring Boot дозволить нам швидко розробляти та підтримувати веб-додаток. Ми будемо використовувати Spring Data JPA для зберігання та отримання даних з бази даних, а шаблонізатор Thymeleaf дозволить нам легко створювати динамічні HTML сторінки. Застосування архітектурного шаблону MVС дозволить нам розділити логіку додатку на моделі, представлення та контролери для забезпечення кращої організації коду та розширюваності.

Такий стек технологій є популярним та добре підтримується, тому я впевнений, що наприкінці вийде потужний та надійний додаток для наших користувачів.

Структура проекту DailyExpenses має наступний вид:

\dailyexpenses

\controllers

CategoryController.java

ExpensesController.java

UserController.java

\models

Category.java

Expense.java

User.java

\repositories

CategoryRepository.java

ExpenseRepository.java

UserRepository.java

\services

CategoryService.java

CategoryServiceImpl.java

ExpenseService.java

ExpenseServiceImpl.java

UserService.java

UserServiceImpl.java

DailyExpensesApplication.java

Компоненти додатку включають контролери, які відповідають за обробку запитів користувачів та повернення відповідей у форматі HTML сторінок. У проекті вже присутні такі контролери:

* ExpensesController.java, який відповідає за роботу з витратами, їх додавання, видалення та редагування, а також за відображення домашньої сторінки веб-сайту;
* UserController.java, який відповідає за роботу з користувачами, тобто відображення сторінок входу та реєстрації, редагування профілю користувача та інші функції, що пов'язані з аутентифікацією та авторизацією користувачів.
* CategoryController.java, який відповідає за роботу з категоріями витрат.

Моделі включають об'єкти, які представляють дані, що використовуються в системі. В проекті присутні такі моделі:

* Category.java, який представляє категорію витрат.
* Expense.java, який представляє витрату.
* User.java, який представляє користувача.

Репозиторії використовуються для зберігання та отримання даних з бази даних. У проекті вже присутні такі репозиторії:

* CategoryRepository.java, який відповідає за збереження та отримання даних про категорії витрат.
* ExpenseRepository.java, який відповідає за збереження та отримання даних про витрати.
* UserRepository.java, який відповідає за збереження та отримання даних про користувачів.

Сервіси використовуються для виконання бізнес-логіки та обробки даних. У проекті вже присутні такі сервіси:

* CategoryService.java та CategoryServiceImpl.java, які відповідають за логіку категорій витрат.
* ExpenseService.java та ExpenseServiceImpl.java, які відповідають за логіку витрат.
* UserService.java та UserServiceImpl.java, які відповідають за логіку користувачів.

Основним файлом додатку є DailyExpensesApplication.java, який містить точку входу до додатку.

## Інструкція користувача веб-ресурсу

На рисунку 3.1 зображена екранна форма початкової сторінки веб-додатку. Вона містить ознайомлювальну інформацію щодо додатка.

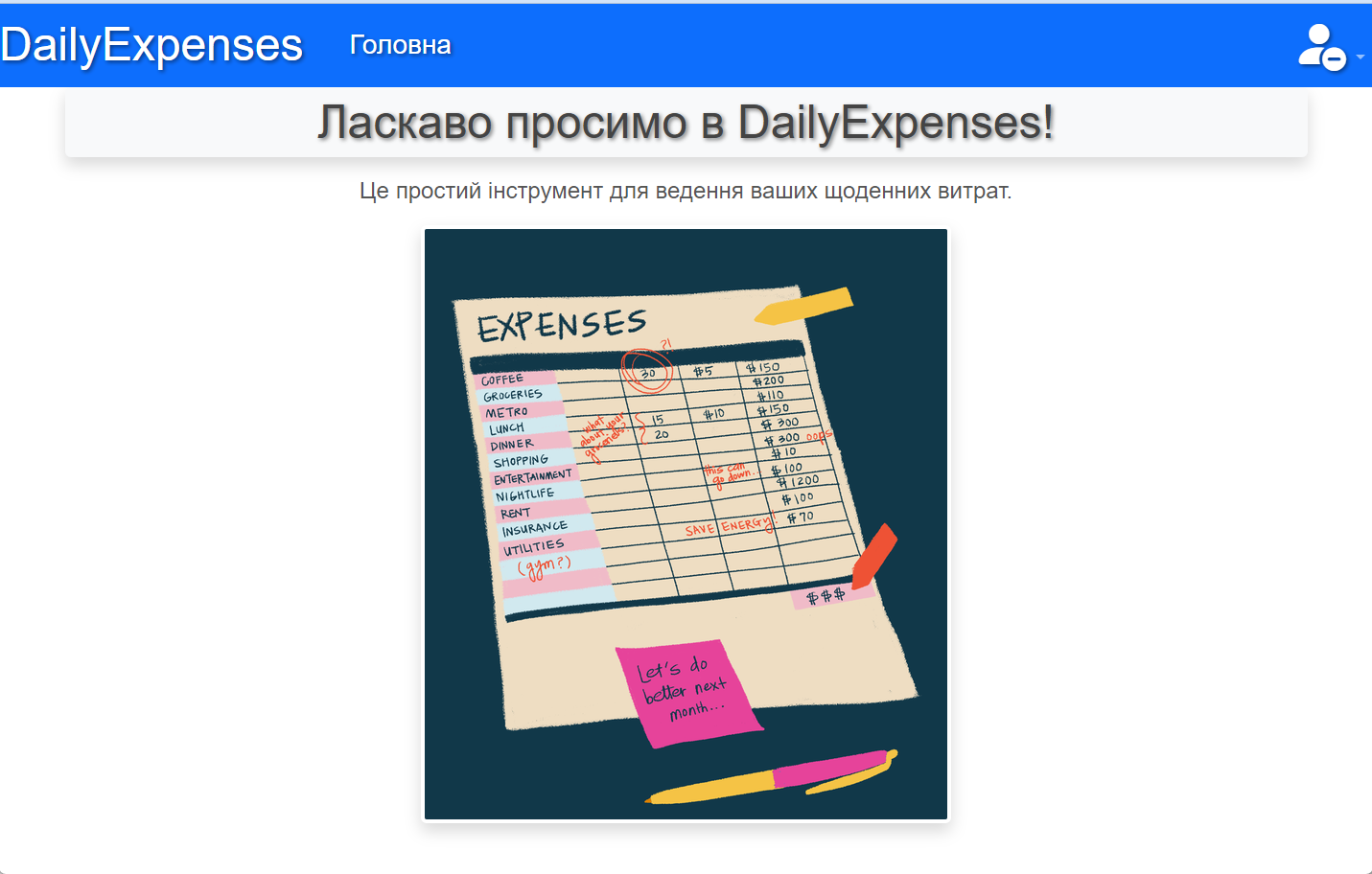
Шапка сайту містить логотип додатку та початкове навігаційне меню, яке дозволить користовачу або увійти до додатку, або зареєструватись.

Рисунок 3.1 – Екранна форма початкового вікна веб-додатку

На рисунку 3.2 зображено меню неавторизованого користувача, яке дозволяє перейти на сторінки авторізації або реєстрації користувача.

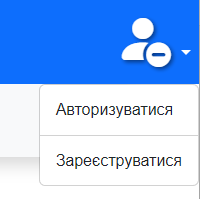


Рисунок 3.2 – меню неавторизованого користувача

На рисунку 3.3 зображена сторінка з формою авторизації користувача, яка дозволяє увійти до додатку під своїм логіном.

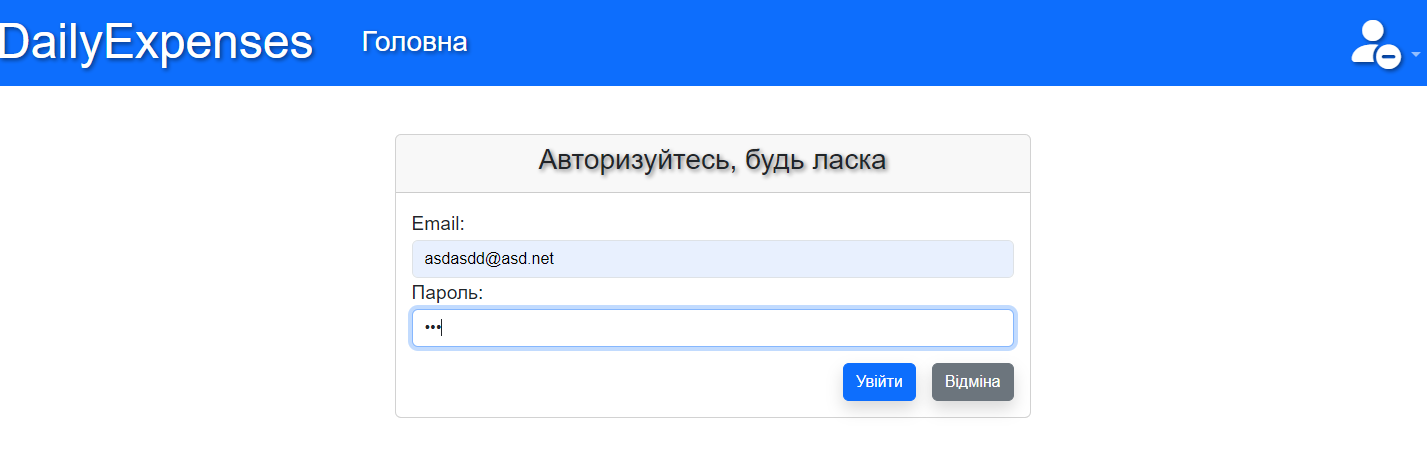


Рисунок 3.3 – сторінка з формою авторизації користувача

На рисунку 3.4 зображена сторінка з формою реєстрації користувача, яка дозволяє зареєструватися в додатку як новий користувач.

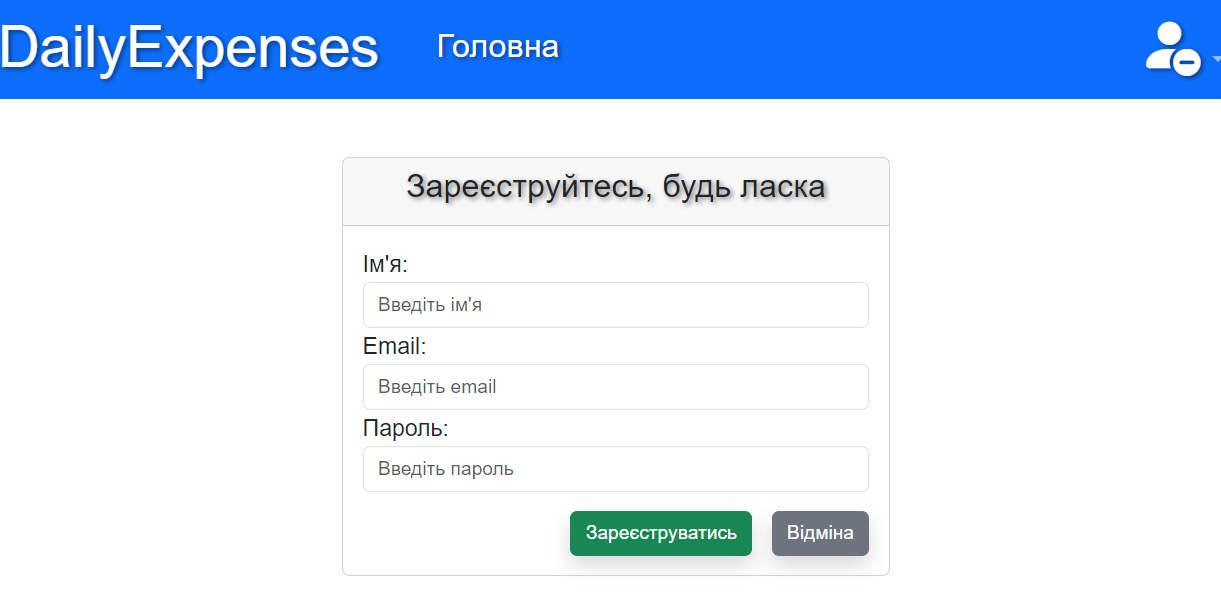


Рисунок 3.4 – меню неавторизованого користувача

На рисунку 3.5 зображена екранна форма початкової сторінки вже авторизованого користувача. Вона має повноцінне меню роботи з додатком, що дозволяє перейти на сторінки роботи з витратами, налаштуваннями додатку та користувача.

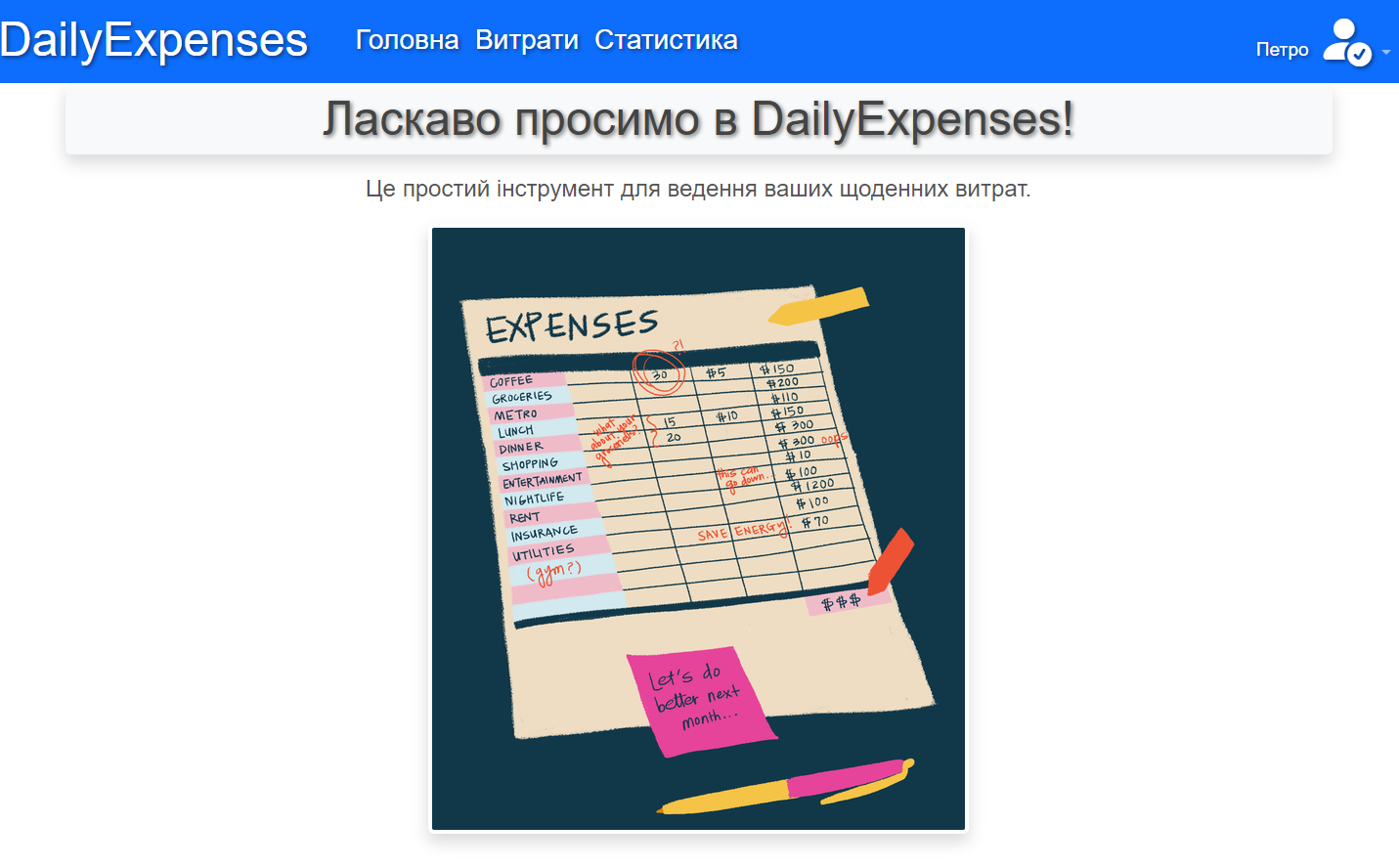


Рисунок 3.5 –початкова сторінка авторизованого користувача

На рисунку 3.6 зображено меню авторизованого користувача, яке дозволяє перейти на сторінки корегування даних користувача та налаштування додатку.

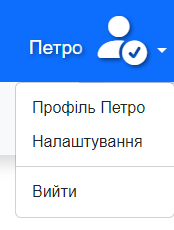


Рисунок 3.6 – меню авторизованого користувача

На рисунку 3.7 зображена сторінка з формою редагування користувача, яка дозволяє зареєструватися в додатку як новий користувач.

вікно створення нового акаунту. Дизайн сторінки виконаний в світлих тонах з використанням контрастних елементів, що робить її зручною для використання.

На сторінці присутня форма реєстрації, яка містить поля для введення основної інформації, такої як ім'я, прізвище, адреса електронної пошти та пароль. Також є кнопка "Зареєструватися", яка відправляє заповнену форму на сервер для обробки. Кнопка входу знаходиться під формою реєстрації та дозволяє користувачам, які вже зареєструвалися, увійти на сайт.

Ця сторінка є важливою для користувачів, оскільки дозволяє їм зареєструватися на сайті та отримати доступ до послуг. Дизайн сторінки є зручним та легким для використання, що сприяє покращенню користувацького досвіду.

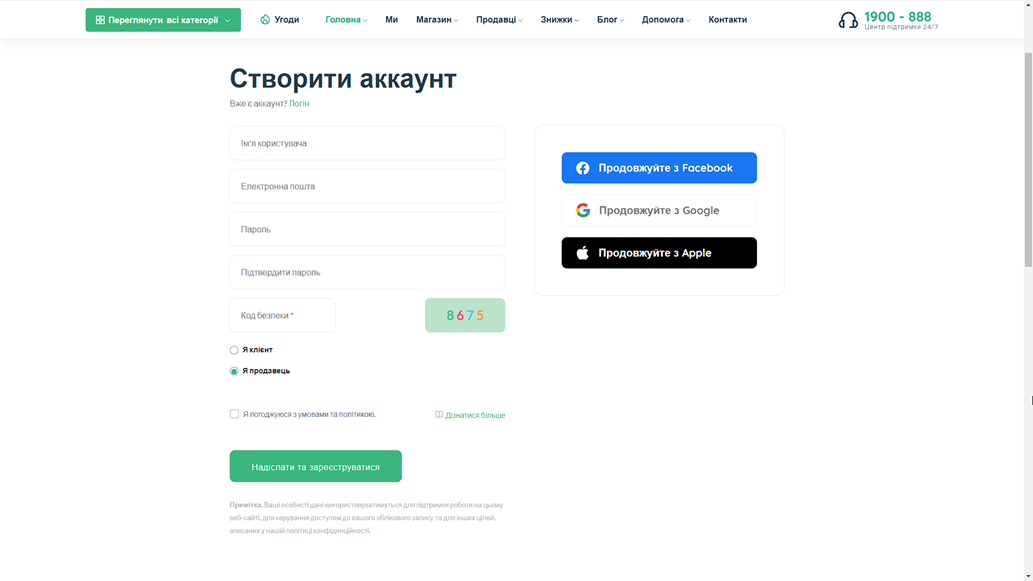


Рисунок 3.2 – Екранна форма створення нового акаунту

На рисунку 3.3 зображене вікно логіну веб-ресурсу. Вона містить форму для введення інформації користувача та кнопку "Увійти", щоб здійснити вхід на сайт.

На сторінці присутня форма входу, яка містить поля для введення ім'я користувача та пароля. Крім того, є кнопка "Увійти", яка дозволяє користувачам здійснити вхід на сайт.

Посилання на сторінку "Забули пароль?" дозволяє користувачам відновити свій пароль, якщо вони його забули. Посилання на сторінку реєстрації дозволяє користувачам зареєструватися на сайті та отримати доступ до послуг.

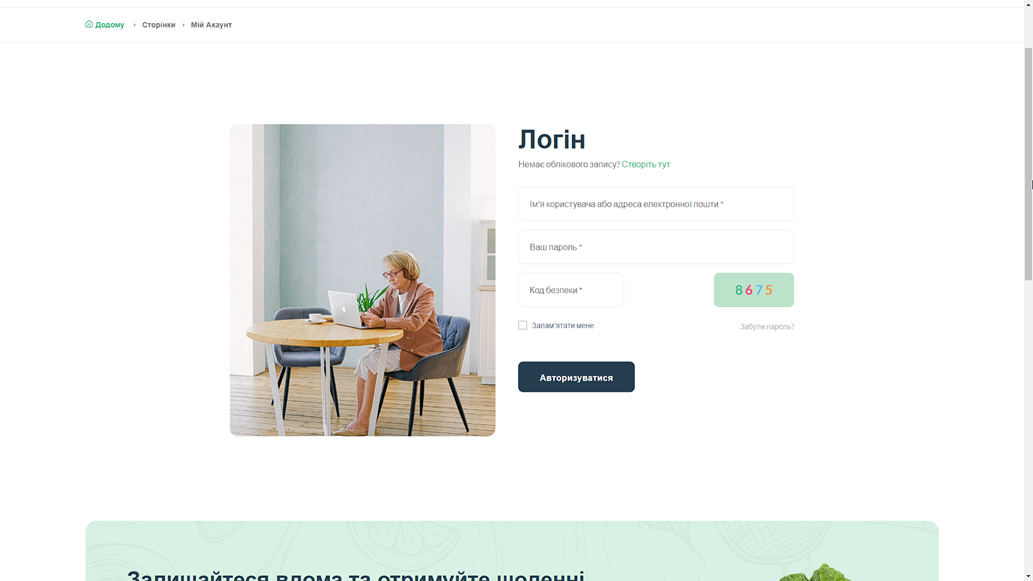


Рисунок 3.3 – Екранна форма логіну

На рисунку 3.4 зображена екранна форма головного вікна веб-ресурсу. Вона містить інформацію про продукти та надає користувачам можливість знайти потрібну інформацію та здійснити необхідні дії.

Шапка сайту містить логотип компанії, навігаційне меню та кнопку "Увійти", щоб здійснити вхід на сайт. Банер містить промо-контент компанії та посилання на послуги. Секції продуктів містять короткий опис та зображення продуктів, що продає компанія.

Секція "Про нас" містить інформацію про компанію та її місію. Секція "Наші клієнти" містить відгуки та рекомендації клієнтів компанії. Футер сайту містить навігаційне меню, контактну інформацію та посилання на соціальні мережі. Секція "Наші клієнти" містить відгуки та рекомендації клієнтів компанії. Секція "Останні новини" містить оновлення та новини компанії.

Ця сторінка є важливою для користувачів, оскільки вона надає інформацію про продукти компанії та дозволяє здійснити необхідні дії, такі як вхід на сайт, пошук інформації про продукти, зв'язок з компанією та інше.

Дизайн сторінки є зручним та легким для використання, що сприяє покращенню користувацького досвіду.

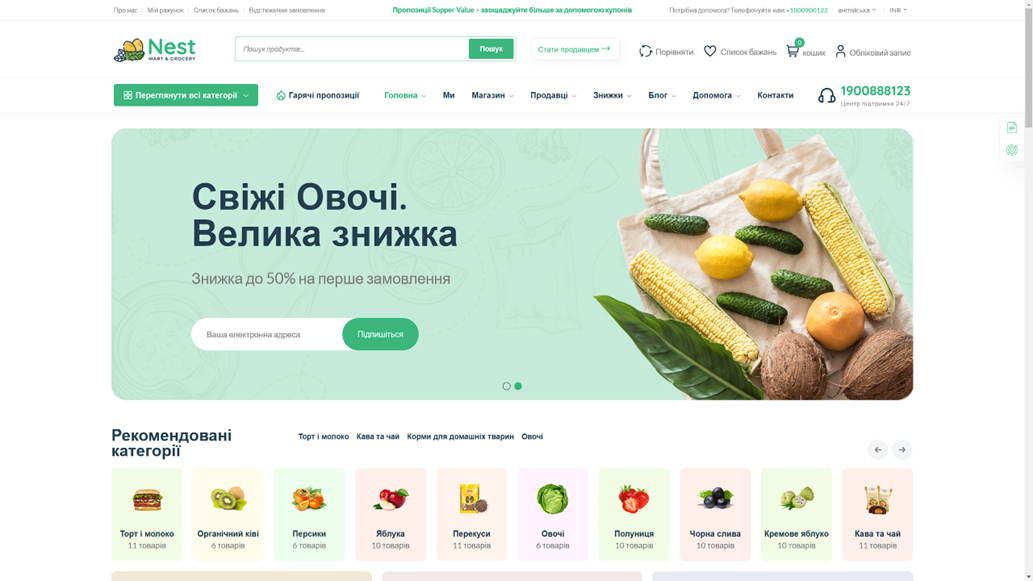


Рисунок 3.4 – Екранна форма головного вікна веб-ресурсу

На рисунку 3.5 зображена екранна форма переліку товарів певної категорії. На сторінці розміщена таблиця товарів, що дозволяє користувачам переглядати інформацію про товари, такі як назва товару, зображення, ціна та опис. Дизайн сторінки привабливий та легкий для сприйняття. На сторінці присутні наступні елементи:

* шапка сайту, яка містить логотип компанії, навігаційне меню та кнопку "Кошик", щоб здійснити перехід до кошика покупок;
* список категорій товарів, який містить список категорій товарів, які є доступними на сайті;
* фільтри товарів, які дозволяють користувачам фільтрувати товари за різними параметрами, такими як ціна, розмір, колір та інші;
* список товарів, який містить зображення, назву та ціну товарів.

Механізм пагінації дозволяє користувачам переходити до наступних сторінок, якщо на сайті доступний великий асортимент товарів. Футер сайту містить навігаційне меню, контактну інформацію та посилання на соціальні мережі. Крім того, на сторінці присутні фільтри, що дозволяють користувачам знайти товари, які відповідають їхнім потребам. Дизайн сторінки є зручним та легким для використання, що сприяє покращенню користувацького досвіду.

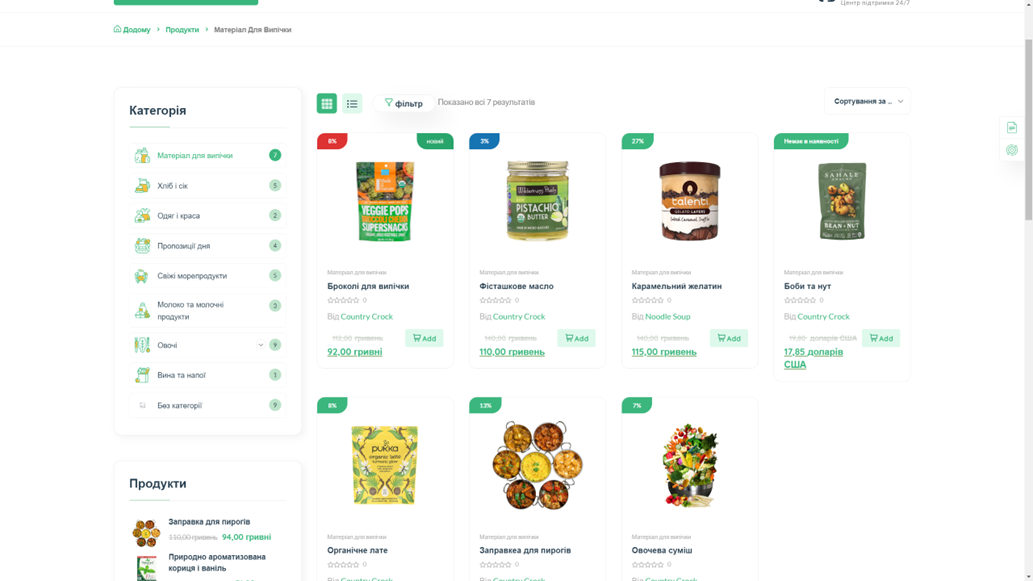


Рисунок 3.5 – Екранна форма переліку продуктів певної категорії

На рисунку 3.6 зображена екранна форма вікна продукту. На сторінці розміщена детальна інформація про обраний товар, така як назва, зображення, ціна, опис, характеристики та відгуки користувачів.

На сторінці присутні картка товару, яка містить зображення, назву, ціну, опис та характеристики товару. Крім того, на сторінці присутня можливість перегляду зображення товару в різних ракурсах за допомогою каруселі зображень. Це дозволяє користувачам отримати більш детальний огляд товару перед покупкою.

Кнопка "Додати у кошик" дозволяє користувачам додавати товар до кошика покупок. Секція "Відгуки користувачів" містить відгуки користувачів про товар, які допомагають іншим користувачам прийняти рішення про покупку. Футер сайту містить навігаційне меню, контактну інформацію та посилання на соціальні мережі.

Структура сторінки досить логічна та зрозуміла, що дозволяє користувачам легко зорієнтуватися та знайти необхідну інформацію. Дизайн сторінки є досить мінімалістичним, що сприяє увазі на основну інформацію та забезпечує більш зручний користувацький досвід.

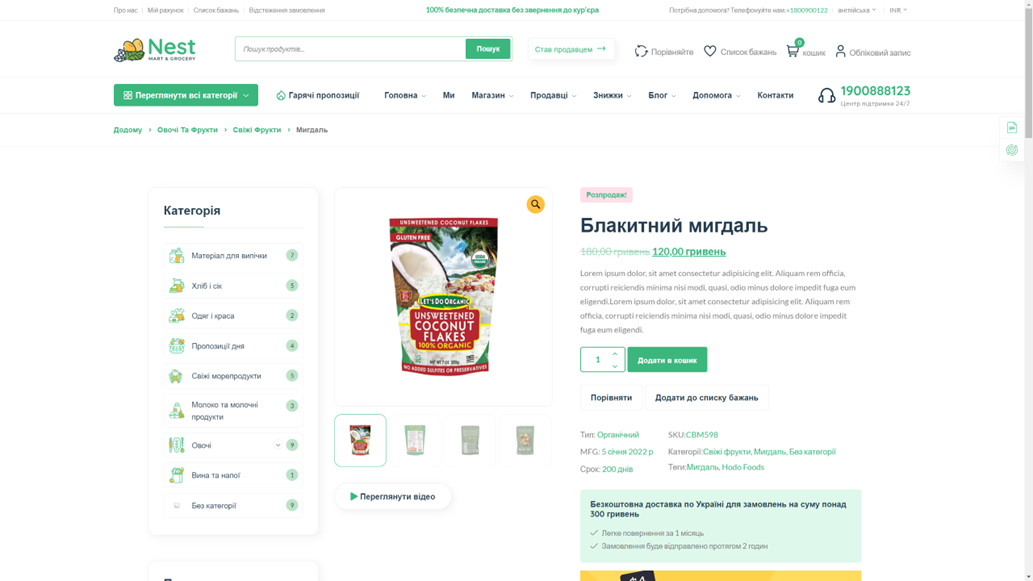


Рисунок 3.6 – Екранна форма вікна продукту

На рисунку 3.7 зображена екранна форма вікна кошику товарів. На цій сторінці користувач може переглянути всі товари, які він додав до кошика, зокрема їх зображення, назву, ціну, кількість та загальну вартість замовлення.

Крім того, на сторінці присутня можливість редагування кількості товару, видалення товару з кошика, а також використання промокодів та отримання знижок.

На сторінці присутні важливі елементи для успішного оформлення замовлення, такі як кнопка "Оформити замовлення", яка перенаправляє користувача на сторінку оформлення замовлення, а також інформаційний блок про доставку та оплату замовлення.

Сторінка має адаптивний дизайн, що дозволяє користувачам зручно переглядати її на різних пристроях та розмірах екранів.

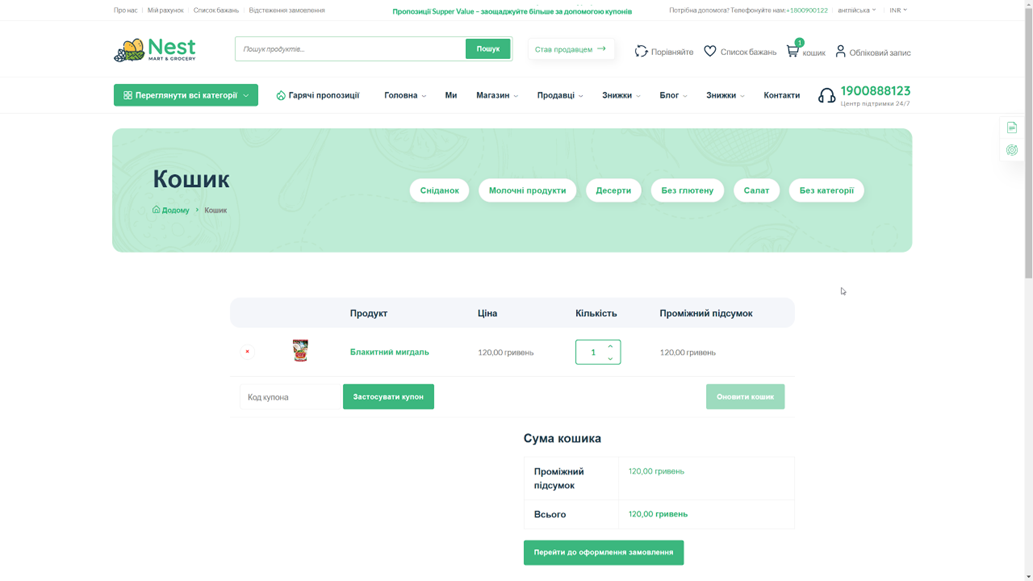


Рисунок 3.7 – Екранна форма вікна кошику товарів

На рисунку 3.8 зображена екранна форма вікна оформлення замовлення. На цій сторінці користувачі можуть оформити своє замовлення, заповнивши необхідну інформацію про себе та свої покупки.

Сторінка містить різні елементи, необхідні для успішного оформлення замовлення, такі як форма для введення інформації про замовлення, включаючи адресу доставки та спосіб оплати.

Крім того, сторінка містить блок з інформацією про вартість замовлення, включаючи вартість товарів, податки та доставку.

Користувачам також доступні різноманітні опції, такі як використання купонів та збереження адрес доставки для майбутніх покупок.

На сторінці є кнопка "Підтвердити замовлення", яка дозволяє користувачам підтвердити та оформити своє замовлення. Дизайн сторінки є простим та зрозумілим, зроблений в мінімалістичному стилі.

Після того, як користувач створив замовлення, він може відстежувати статус замовлення у відповідному вікні профілю користувача, до якого можна перейти з головного меню сайту, яке знаходиться у шапці кожної сторінки веб-ресурсу.

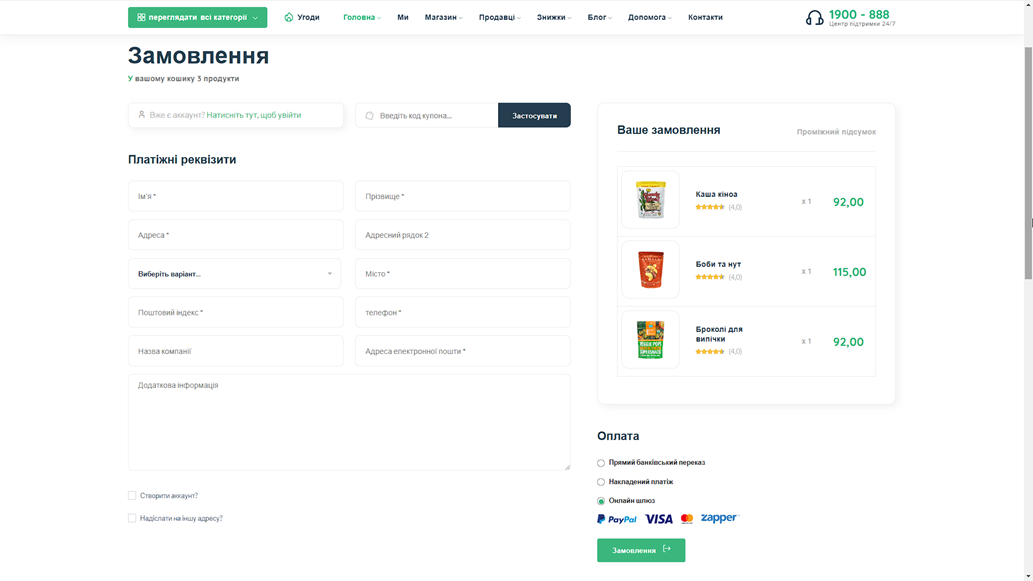


Рисунок 3.8 – Екранна форма оформлення замовлення

## Вихідний код веб-ресурсу

Вихідний код програмних класів розробленого веб-ресурсу наведений в Додатку А даної роботи.

## Висновки до другого розділу

В даному розділі курсової роботи було проведено програмну реалізацію веб-ресурсу продуктової крамниці на базі фреймворку Angular. Для того, щоб провести програмну реалізацію, був використаний проєкт веб-ресурсу, який був розроблений в другому розділі даної роботи.

Перш за все, була розглянута структура серверного програмного проєкту. Були визначені основні програмні компоненти, з яких складається програмний проєкт веб-ресурсу, а також наданий перелік основних класів, які відносяться до цих компонентів.

Далі була розроблена концептуальна діаграма класів, яка дозволяє визначити основні класи веб-ресурсу та визначити взаємовідносини між цими класами.

Окремо було розглянуто питання використання системи контроля версій для упорядкування розробки веб-ресурсу. Були визначені основні репозиторії проєкту та надані основні показники метрик цих репозиторіїв.

Далі було проведено функціональне тестування розробленого веб-ресурсу. Був розроблений протокол тестування у вигляді множини тест-кейсів та проведено тестування. Було визначено, що всі тест-кейси були пройдені, а отже фактична поведінка веб-ресурсу співпадає з очікуваною поведінкою, визначеною у функціональних вимогах.

Також була розроблена інструкція користувача у вигляді множини знімків екрану та пояснювального тексту, який визначає функціонал додатку у даному вікна та можливі дії користувача. Крім того, наданий вихідний код розробленого веб-ресурсу.

# ВИСНОВКИ

У даній курсовій роботі був розроблений веб-ресурс продуктової крамниці на базі фреймворку Angular.

Такий веб-ресурс дозволяє користувачам переглядати каталог товарів в магазині, робити онлайн-замовлення товарів, а також залишати відгуки. Таким чином, мета, поставлена перед даною курсовою роботою досягнута в повному обсязі. Для досягнення мети були вирішені наступні задачі.

У першому розділі даної курсової роботи була детально розглянута предметна область створення веб-ресурсу для продуктової крамниці з використанням фреймворку Angular. Були визначені основні завдання, які повинні бути вирішені в процесі розробки веб-ресурсу продуктової крамниці. Був проведений аналіз існуючих аналогів та визначені ключові вимоги до створення власного веб-ресурсу. У якості технологій для розробки клієнтської частини був обраний Angular, для розробки серверної частини – Java Spring. Також, було вирішено використовувати PostgreSQL в якості СУБД для даного веб-ресурсу.

У другому розділі роботи було проведено проектування веб-ресурсу. Була визначена мета сайту, його потенційна аудиторія та основні можливості. Визначені основні вимоги до веб-ресурсу, включаючи функціональні та нефункціональні. Також була розроблена діаграма сценаріїв, діаграма інформаційних потоків, діаграма станів. Крім того, була розроблена діаграма логічного уявлення системи, а також діаграма розгортання веб-ресурсу. Також була визначена схема даних системи та наданий опис ключовим технологіям.

В третьому розділі роботи було проведено програмну реалізацію веб-ресурсу. Розглянута структура серверного програмного проєкту, визначені основні програмні компоненти, розроблена діаграма класів. Розглянуто питання використання системи контроля версій для упорядкування розробки веб-ресурсу. Проведено функціональне тестування, розроблена інструкція користувача у вигляді множини знімків екрану та пояснювального тексту та наданий вихідний код розробленого веб-ресурсу.

# ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Уоллс К. «Spring в действии.» 6-е изд./ пер. с англ.А. Н. Киселева. – М.: ДМКПресс, 2022.
2. Гутьеррес Фелипе «Spring Boot 2: лучшие практики для профессионалов» СПб.: Питер, 2020 - 464 с.: ил. — (Серия «Библиотека программиста»).
3. Хеклер Марк «Spring Boot по-быстрому» СПб.: Питер, 2022. — 352 с.: ил. — (Серия «Бестселлеры O’Reilly»).
4. Spring Framework guides. URL: <https://spring.io/guides> (дата звернення 25.04.2023).
5. Tutorial: Thymeleaf + Spring. URL: https://www.thymeleaf.org/doc/tutorials/3.0/thymeleafspring.html (дата звернення 25.04.2023).
6. Учебник: Использование Thymeleaf. URL: <https://habr.com/ru/articles/350862/> (дата звернення 26.04.2023)
7. Spring Data JPA. URL: <https://habr.com/ru/articles/435114/> (дата звернення 28.04.2023)
8. Lombok. Полное руководство. URL: <https://habr.com/ru/companies/piter/articles/676394/> (дата звернення 28.04.2023)
9. Thymeleaf Page Layouts. URL: <https://www.thymeleaf.org/doc/articles/layouts.html> (дата звернення 30.04.2023)
10. Современный базовый стек приложений Java Spring. URL: <https://coderlessons.com/articles/java/sovremennyi-bazovyi-stek-prilozhenii-java-spring> (дата звернення 25.04.2023)
11. Spring MVC — основные понятия и архитектура. <https://highload.today/spring-mvc/#1> (дата звернення 28.04.2023)