

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 проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему ПРОГРАМНА РЕАЛІЗАЦІЯ
ЗАСТОСУНКУ ДЛЯ КОНДИТЕРІВ
SOFTWARE IMPLEMENTATION OF AN INTERACTIVE
APPLICATION FOR CONFECTIONERS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-121

Спеціальності 121 Інженерія програмного

(код і найменування спеціальності)

забезпечення

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

КРУПНИЦЬКА Т.А.

(ПРИЗВИЩЕ та ініціали)

Керівник ЛЕОЩЕНКО Д.С.

(ПРИЗВИЩЕ та ініціали)

Рецензент БАБЕНКО Н.В.

(ПРИЗВИЩЕ 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.
С.О. СУББОТІН
 « » 2025 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

КРУПНИЦЬКОЇ Тетяни Андріївни

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Програмна реалізація інтерактивного застосунку для кондитерів. Software Implementation of an Interactive Application for Confectioners.
 керівник проєкту (роботи) д-р. філос., доцент, ЛЕОЩЕНКО Сергій Дмитрович,

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “28” квітня 2025 року № 209

2. Строк подання студентом проєкту (роботи) 02 червня 2025 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. Матеріали і методи. 3. Основні рішення щодо реалізації компонентів системи 4. Експлуатація, тестування та експериментальне дослідження програми.

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ЛЕОЩЕНКО С.Д., доцент		
Нормоконтроль	КАЛІНІНА М.В., асистент		

7. Дата видачі завдання «14» квітня 2025 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка структури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділи 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

(підпис) Тетяна КРУПНИЦЬКА
(Імя ПРИЗВИЩЕ)

Керівник проєкту (роботи)

(підпис) Сергій ЛЕОЩЕНКО
(Імя ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра:
154 с., 66 табл., 48 рис., 5 дод., 69 джерел.

ІНТЕРАКТИВНИЙ ЗАСТОСУНОК, КОНДИТЕРСЬКІ РЕЦЕПТИ,
JAVA, БАЗА ДАНИХ, ПОШУК РЕЦЕПТІВ, ІНТЕРФЕЙС, ТЕСТУВАННЯ.

Об'єкт дослідження – процес розробки програмного забезпечення для інтерактивного застосунку для кондитерів з функціями реєстрації, управління рецептами, пошуку та взаємодії користувачів.

Предмет дослідження – методи та підходи до створення такого застосунку, зокрема використання сучасних технологій програмування, баз даних та інтерфейсів користувача.

Метою роботи є розробка застосунку, що забезпечує зручний доступ до рецептів, створення, редагування та поширення власних рецептів, а також ефективний пошук інформації.

Матеріали, методи та технічні засоби: використано Java, IntelliJ IDEA, JavaFX для інтерфейсу, SQLite для зберігання даних та JSON для обміну інформацією.

Результати: розроблено інтерактивний застосунок із функціями реєстрації, авторизації, зміни паролю, публікації та редагування рецептів, пошуку за ключовими словами й інгредієнтами, а також з можливістю залишення відгуків про рецепт.

Практична цінність роботи: застосунок є корисним інструментом для професійних кондитерів та аматорів, сприяючим покращенню навичок та обміну досвідом.

Висновки: реалізовано сучасний кондитерський застосунок, що забезпечує зручний доступ до рецептів та оптимізує пошук інформації.

Галузь використання – кулінарна сфера, зокрема для кондитерів і любителів десертів, як персональний інструмент та платформа для взаємодії.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 154 pages, 66 tables, 48 figures, 5 appendixes, 69 sources.

INTERACTIVE APPLICATION, PASTRY RECIPES, JAVA, DATABASE, RECIPE SEARCH, INTERFACE, TESTING.

The object of research is the process of developing software for an interactive application for confectioners with functions of registration, recipe management, search and user interaction.

The subject of the research is Methods and approaches to creating such an application, in particular the use of modern programming technologies, databases and user interfaces.

The goal of the work is to develop an application that provides convenient access to recipes, create, edit and distribute your own recipes, as well as effective search for information.

Materials, methods, and technical tools: Java, IntelliJ IDEA, JavaFX for the interface, SQLite for data storage, and JSON for information exchange were used.

Results: an interactive application has been developed with the functions of registration, authorization, password change, publishing and editing recipes, searching by keywords and ingredients, as well as with the ability to leave reviews about the recipe.

Practical value of the work: the app is a useful tool for professional confectioners and amateurs, contributing to the improvement of skills and exchange of experience.

Conclusions: a modern confectionery application has been implemented that provides convenient access to recipes and optimizes the search for information.

The field of use is the culinary sphere, in particular for confectioners and dessert lovers, as a personal tool and platform for interaction.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	9
Вступ.....	10
1 Аналіз предметної області	13
1.1 Огляд сучасних кондитерських застосунків та їх функціоналу	13
1.2 Аналіз конкурентів та існуючих рішень	17
1.3 Визначення потреб користувачів та специфікацій	26
1.3.1 Функціональні вимоги до застосунку	26
1.3.2 Нефункціональні вимоги.....	30
1.4 Постановка завдання	35
1.4.1 Опис основних функцій програми	35
1.4.2 Вимоги до інтерфейсу	37
1.4.3 Вимоги до безпеки	41
1.4.4 Нефункціональні вимоги.....	42
1.5 Висновки за розділом 1	43
2 Матеріали і методи.....	45
2.1 Вибір мови програмування Java.....	45
2.2 Вибір середовища розробки	47
2.2.1 Критерії вибору середовища розробки	47
2.2.2 Аналіз доступних середовищ розробки	48
2.2.3 Обґрунтування вибору IntelliJ IDEA	51
2.2.4 Використання додаткових інструментів.....	52
2.3 Вибір технологій для реалізації інтерфейсу	53
2.3.1 JavaFX як основа графічного інтерфейсу	53
2.3.2 Використання ObservableList	54
2.3.3 Використання CSS для стилізації інтерфейсу.....	55
2.3.4 Використання компонентів JavaFX.....	55
2.3.5 Використання JSON для обміну даними	56
2.3.6 Використання HTTP-запитів для взаємодії з сервером	57

2.3.7 Використання ObjectMapper для роботи з JSON	57
2.3.8 Використання потоків для асинхронної роботи	58
2.3.9 Використання Alert для відображення повідомлень	58
2.3.10 Використання Scene та Stage для керування екранами.....	58
2.4 Структурна схема розробки застосунку.....	59
3 Основні рішення щодо реалізації компонентів системи.....	63
3.1 Проєктування моделей даних для рецепта	63
3.1.1 Реалізація реєстрації користувача та авторизації	67
3.2 Реалізація механізму публікації рецептів	70
3.3 Розробка функцій пошуку та коментування рецептів	72
3.3.1. Функція пошуку рецептів.....	72
3.3.2. Функція коментування рецептів.....	72
3.3.3. Інтеграція функцій пошуку та коментування	73
4 Експлуатація, тестування та експериментальне дослідження програми	74
4.1 Призначення та умови застосування програми.....	74
4.2 Вимоги до інтерфейсу застосунку	77
4.3 Умови виконання програми.....	81
4.4 Методика тестування основних функцій: пошук, коментарі.....	87
4.5 Результати роботи алгоритму пошуку та фільтрації	91
4.6 Висновки за отриманими результатами тестування	93
Висновки	95
Перелік джерел посилання	97
Додаток А Технічне завдання	104
Додаток Б Покроковий опис роботи програми	111
Додаток В Фрагменти тексту програми.....	119
Додаток Г UML-діаграми	141
Додаток Д Слайди презентації.....	147

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

AWT – Abstract Window Toolkit;
CORS – Cross-Origin Resource Sharing;
CSS – Cascading Style Sheets;
GUI – Graphical User Interface;
HTML – Hyper Text Markup Language;
HTTP – Hypertext Transfer Protocol;
IDE – Integrated Development Environment;
JDBC - Java Database Connectivity;
JDK – Java Development Kit;
JIT – Just-In-Time;
JRE – Java Runtime Environment;
JSON – JavaScript Object Notation;
JVM – Java Virtual Machine;
SQL – Structured Query Language;
UI – User Interface;
UML – Unified Modeling Language;
UX – User experience;
XML – eXtensible Markup Language.

ВСТУП

Сучасний світ характеризується стрімким розвитком інформаційних технологій, які проникають у всі сфери життєдіяльності людини. Однією з таких сфер є кулінарія, зокрема кондитерська справа, яка набуває все більшої популярності серед професіоналів та аматорів. Створення інтерактивного застосунку для кондитерів, що дозволяє користувачам ділитися рецептами, знаходити нові ідеї та оптимізувати процес приготування десертів, є актуальним напрямком у розвитку програмного забезпечення. Такий застосунок не лише спрощує доступ до інформації, але й створює спільноту для обміну досвідом та натхненням.

Актуальність розробки інтерактивного застосунку для кондитерів обумовлена декількома факторами. По-перше, зростання інтересу до домашнього приготування їжі, зокрема десертів, що пов'язано з популяризацією здорового способу життя та бажанням контролювати склад продуктів. По-друге, збільшення кількості онлайн-ресурсів, які надають доступ до рецептів, але часто не мають зручного інтерфейсу та функціональності для ефективного пошуку та зберігання інформації. По-третє, необхідність створення платформи, яка об'єднує користувачів з різним досвідом, дозволяючи їм ділитися своїми знаннями та навичками. Таким чином, розробка спеціалізованого застосунку для кондитерів є важливим кроком у задоволенні потреб сучасного суспільства.

Метою дослідження є розробка інтерактивного застосунку для кондитерів, який забезпечує зручний доступ до рецептів, дозволяє користувачам створювати, редагувати та поширювати власні рецепти, а також оптимізує процес пошуку інформації за допомогою інтелектуальних механізмів. Застосунок має стати інструментом, що сприяє розвитку кулінарних навичок та створенню спільноти однодумців.

Об'єктом дослідження є процес розробки програмного забезпечення для створення інтерактивного застосунку для кондитерів, який включає в себе

функції реєстрації користувачів, управління рецептами, пошуку за ключовими словами та інгредієнтами, а також взаємодії між користувачами.

Для досягнення поставленої мети необхідно виконати наступні завдання:

- провести аналіз існуючих рішень у сфері кулінарних застосунків та визначити їхні переваги та недоліки;
- розробити архітектуру застосунку, що включає базу даних для зберігання рецептів та інформації про користувачів;
- реалізувати функціональність застосунку, зокрема реєстрацію та автентифікацію користувачів, створення та редагування рецептів, пошук за ключовими словами та інгредієнтами;
- забезпечити зручний та інтуїтивно зрозумілий інтерфейс для користувачів;
- протестувати застосунок на предмет коректності роботи та зручності використання;
- оптимізувати роботу застосунку для забезпечення швидкості та стабільності його роботи.

Предметом дослідження є методи та підходи до розробки програмного забезпечення для створення інтерактивного застосунку для кондитерів, зокрема використання сучасних технологій програмування, баз даних та інтерфейсів користувача.

Інформаційну базу дослідження складають наукові праці з розробки програмного забезпечення, документація до мови програмування Java, а також аналіз існуючих кулінарних застосунків та платформ. Використовуються також дані з відкритих джерел, що стосуються тенденцій у сфері кулінарії та харчової індустрії.

У процесі дослідження використовуються такі методи:

- аналіз існуючих рішень та їх порівняння;
- проектування архітектури застосунку з використанням UML-діаграм;

- програмна реалізація на мові Java з використанням сучасних бібліотек та фреймворків;
- тестування продукту на різних етапах розробки для забезпечення якості та стабільності;
- оптимізація коду та баз даних для підвищення продуктивності застосунку.

Практичне значення дослідження полягає у створенні функціонального інструменту, який може бути використаний як професійними кондитерами, так і аматорами для покращення їхніх навичок та обміну досвідом. Застосунок також може стати основою для подальшого розширення функціональності, наприклад, додавання можливості онлайн-замовлення інгредієнтів або проведення майстер-класів.

Структура та обсяг кваліфікаційної роботи: кваліфікаційна робота присвячена розробці інтерактивного застосунку для кондитерів, спрямованого на полегшення доступу до рецептів, їх публікацію, а також реалізацію функцій пошуку та коментування. Робота складається з вступу, чотирьох основних розділів, висновків, переліку джерел посилань та п'яти додатків. Загальний обсяг роботи становить 154 сторінки, що включають 66 таблиць 48 рисунків та 5 додатків, які займають 50 сторінок. Перелік джерел містить 69 посилань на наукові праці, технічну документацію, сучасні дослідження та інші матеріали, які були використані для розробки застосунку.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Огляд сучасних кондитерських застосунків та їх функціоналу

У сучасному світі кондитерська індустрія активно інтегрується з цифровими технологіями, пропонуючи споживачам інноваційні рішення для задоволення їхніх смакових уподобань та кулінарних потреб. Інтерактивні кондитерські застосунки стають невід'ємною частиною цього процесу, надаючи користувачам можливість не лише замовляти солодощі, але й брати участь у їхньому створенні, дізнаватися про нові рецепти та тренди.

Tasty (рисунок 1.1, рисунок 1.2) – це популярний кулінарний застосунок від BuzzFeed, який пропонує користувачам тисячі рецептів, включаючи кондитерські вироби [1]-[2]. Основні функції застосунка включають покрокові відеоінструкції, можливість зберігати улюблені рецепти та функцію "Що у вас в холодильнику?", яка пропонує рецепти на основі наявних інгредієнтів. Користувачі можуть налаштовувати пошук за дієтичними вподобаннями, кухнями світу та рівнем складності. Інтерактивність забезпечується через можливість залишати відгуки, ділитися рецептами та брати участь у кулінарних викликах.

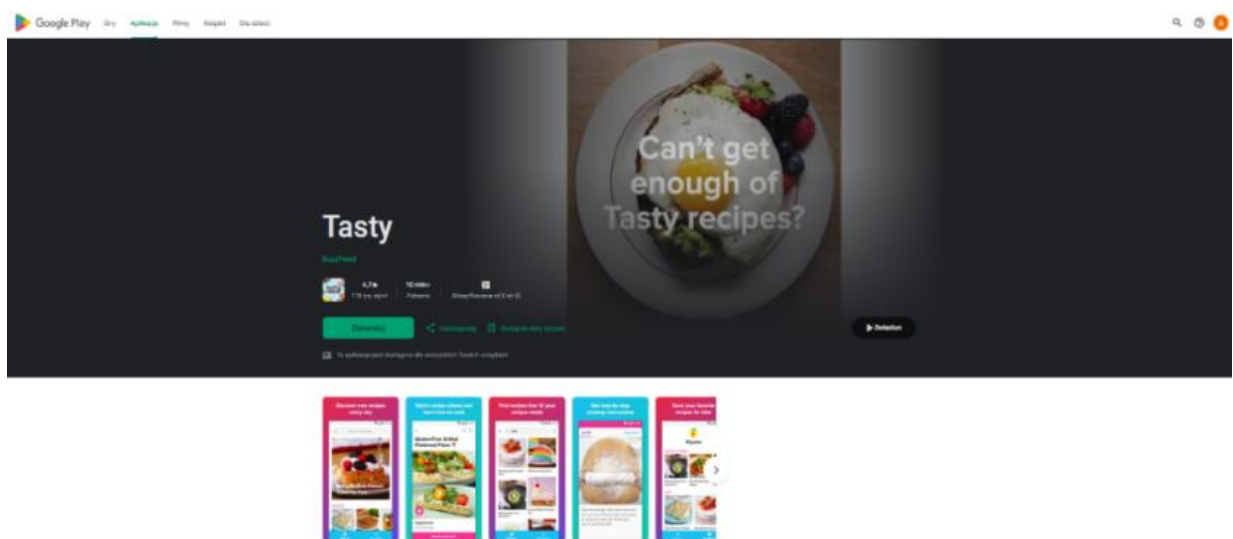


Рисунок 1.1 – Головна сторінка Tasty (Google Play) [1]



Рисунок 1.2 – Головна сторінка сайту Tasty [2]

Kitchen Stories (рисунок 1.3) – це кулінарний застосунок, який надає високоякісні рецепти з детальними фото- та відеоінструкціями [3]. Особливістю є інтеграція з функцією списку покупок та можливість автоматичного перетворення одиниць виміру. Кондитерські рецепти представлені в окремих категоріях, що дозволяє користувачам легко знаходити необхідні десерти. Застосунок також пропонує кулінарні курси та статті з порадами, що робить його корисним як для новачків, так і для досвідчених кондитерів.

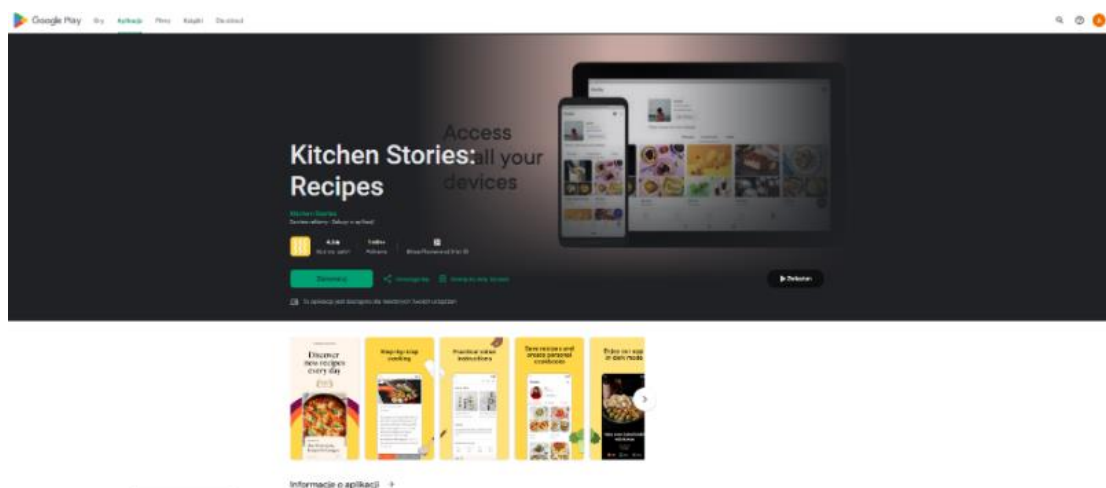


Рисунок 1.3 – Головна сторінка Kitchen Stories (Google Play) [3]

Dessert Recipes (рисунок 1.4) – це застосунок, присвячений виключно десертам. Він містить велику колекцію рецептів з усього світу, включаючи

традиційні та сучасні кондитерські вироби [4]. Користувачі можуть шукати рецепти за інгредієнтами, типом десерту або національною кухнею. Застосунок також пропонує функцію "Випадковий рецепт" для тих, хто хоче спробувати щось нове. Інтерактивні елементи включають можливість додавати власні рецепти та обговорювати їх з іншими користувачами.

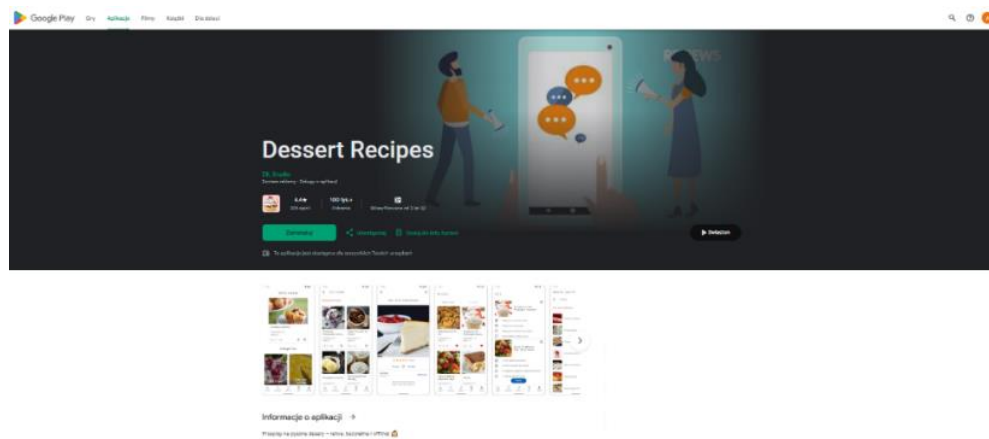


Рисунок 1.4 – Головна сторінка Dessert Recipes (Google Play) [4]

Great British Chefs (рисунок 1.5) – це застосунок, який об'єднує рецепти від провідних британських шеф-кухарів [5]. Хоча він охоплює різні страви, кондитерські рецепти займають значну частину контенту. Кожен рецепт супроводжується професійними фотографіями та детальними інструкціями. Застосунок також пропонує відеоуроки, поради щодо технік та інформацію про інгредієнти. Користувачі можуть створювати власні колекції рецептів та ділитися ними з друзями.



Рисунок 1.5 – Головна сторінка Great British Chefs [5]

Cake Master (рисунок 1.6) – орієнтований на тих, хто прагне вдосконалити свої навички у випіканні тортів [6]. Застосунок пропонує інтерактивні майстер-класи, 3D-моделі для декорування та можливість віртуально "створювати" дизайн торта перед його реальним втіленням. Це дозволяє кондитерам-аматорам експериментувати з дизайном та отримувати професійні поради.



Рисунок 1.6 – Головна сторінка Cake Master [6]

Dr. Oetker Rezeptideen (рисунок 1.7) розроблений німецькою компанією Dr. Oetker, цей застосунок пропонує безліч рецептів випічки та десертів [7]. Окрім традиційних рецептів, застосунок надає інноваційні ідеї та сезонні рекомендації. Користувачі можуть створювати списки покупок, зберігати улюблені рецепти та ділитися ними з друзями. Це робить процес приготування десертів зручним та інтерактивним.

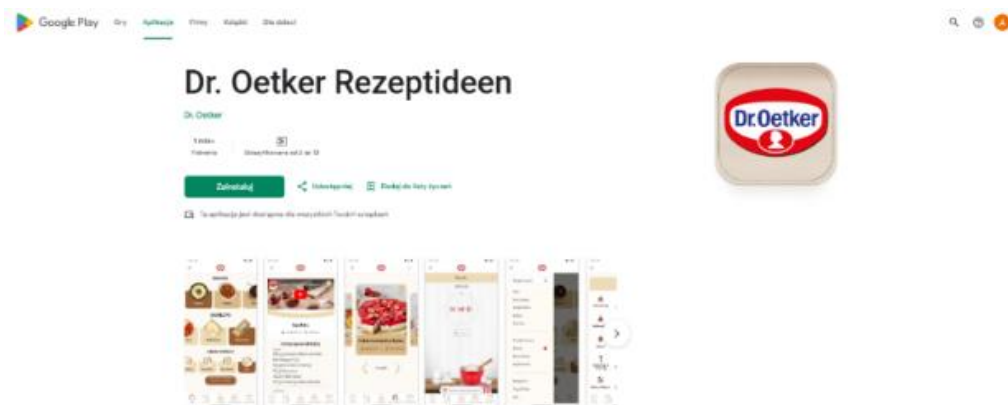


Рисунок 1.7 – Головна сторінка Dr. Oetker Rezeptideen (Google Play) [7]

Betty Crocker Cookbook (рисунок 1.8) — це цифрова версія відомої кулінарної книги, адаптована для мобільних пристроїв [8]. Застосунок містить тисячі рецептів, включаючи кондитерські вироби, з детальними інструкціями та фотографіями. Функція "Кулінарний помічник" допомагає користувачам планувати меню, створювати списки покупок та знаходити рецепти відповідно до дієтичних потреб.



Рисунок 1.8 – Головна сторінка Betty Crocker [8]

Інтерактивні кондитерські застосунки значно розширюють можливості як для аматорів, так і для професіоналів у сфері кондитерського мистецтва. Вони сприяють обміну знаннями, натхненню та вдосконаленню навичок, роблячи процес приготування солодощів більш доступним та захоплюючим. З розвитком технологій очікується поява ще більш інтерактивних та персоналізованих рішень у цій галузі.

1.2 Аналіз конкурентів та існуючих рішень

Аналіз предметної області інтерактивних кондитерських застосунків є важливим етапом у розробці нового програмного забезпечення для кондитерів та кулінарних ентузіастів. Цей аналіз включає вивчення існуючих рішень на

ринку, їхніх переваг та недоліків, що дозволяє визначити можливості для покращення та інновацій.

Tasty — це офіційний мобільний застосунок від BuzzFeed, який надає користувачам доступ до великої колекції відеорецептів та кулінарних інструкцій. Застосунок пропонує покрокові відеоінструкції з приготування різноманітних страв, включаючи десерти та кондитерські вироби [1].

Переваги представлені в таблиці 1.1.

Таблиця 1.1 – Переваги Tasty

№	Переваги	Опис
1	Велика база рецептів	Tasty пропонує понад 4000 рецептів, що охоплюють різні кухні світу та рівні складності, що дозволяє користувачам знайти страви на будь-який смак
2	Покрокові відеоінструкції	Кожен рецепт супроводжується детальним відео, що демонструє процес приготування, що особливо корисно для візуального сприйняття та повторення
3	Персоналізовані рекомендації	Застосунок надає рекомендації щодо наступного прийому їжі залежно від часу доби, дня тижня та основних свят, що спрощує вибір страви
4	Збереження улюблених рецептів	Користувачі можуть зберігати улюблені рецепти для швидкого доступу в майбутньому, що підвищує зручність використання
5	Пошук за інгредієнтами	Можливість пошуку рецептів за наявними інгредієнтами дозволяє ефективно використовувати продукти та мінімізувати відходи

Недоліки наведені в таблиці 1.2.

Таблиця 1.2 – Недоліки Tasty

№	Недоліки	Опис
1	Відсутність інтерактивних елементів	Хоча "Tasty" пропонує відеоінструкції, застосунок не надає можливості інтерактивної взаємодії, такої як віртуальне декорування страв або симуляція процесу приготування
2	Обмежена локалізація	Деякі рецепти можуть містити інгредієнти, які важко знайти в певних регіонах, що може ускладнити процес приготування для користувачів з різних країн

Продовження таблиці 1.2

№	Недоліки	Опис
3	Реклама та внутрішньоігрові покупки	Безкоштовна версія застосунка може містити рекламу, а деякі функції можуть вимагати додаткових витрат, що може бути незручним для користувачів
4	Великі обсяги даних	Відео контент може споживати значний обсяг інтернет-трафіку, що може бути проблематичним для користувачів з обмеженим доступом до мережі або з обмеженим тарифним планом
5	Відсутність можливості додавання власних рецептів	Користувачі не можуть додавати або ділитися власними рецептами через застосунок, що обмежує спільноту кулінарів

Kitchen Stories — це популярний мобільний застосунок, який орієнтований на кулінарні рецепти та навчання. Користувачі можуть знаходити рецепти для приготування різних страв, включаючи десерти, та покроково слідувати інструкціям. Окрім рецептів, застосунок містить численні відеоуроки та можливість збереження улюблених рецептів [3].

Переваги описані в таблиці 1.3.

Таблиця 1.3 – Переваги Kitchen Stories

№	Переваги	Опис
1	Велика кількість рецептів	Kitchen Stories пропонує величезну базу рецептів, включаючи безліч варіантів для приготування десертів. Користувачі можуть знаходити рецепти для різних рівнів складності та стилів кухні
2	Інтерактивні відеоуроки	Застосунок включає покрокові відеоуроки, що полегшують навчання та допомагають користувачам краще засвоїти процес приготування
3	Можливість збереження рецептів	Користувачі можуть зберігати рецепти та створювати власну колекцію улюблених страв, що є зручним для майбутнього використання
4	Персоналізовані рекомендації	Завдяки алгоритмам, застосунок може пропонувати рецепти, що відповідають інтересам та уподобанням користувачів

Недоліки показані в таблиці 1.4.

Таблиця 1.4 – Недоліки Kitchen Stories

№	Недоліки	Опис
1	Обмежена взаємодія	На відміну від ігрових застосунків, таких як Cake Master, Kitchen Stories не пропонує інтерактивної можливості для створення власних десертів чи їх віртуального декорування
2	Не завжди точність інгредієнтів	Інколи користувачі скаржаться на те, що деякі рецепти не мають точних пропорцій або інструкцій щодо інгредієнтів, що може ускладнити процес приготування
3	Мовна бар'єрність	Хоча застосунок підтримує багато мов, деякі регіони все ще не мають локалізованих версій, що обмежує доступність для ширшої аудиторії
4	Безкоштовний доступ до всіх функцій обмежений	Для доступу до деяких преміум-функцій, таких як додаткові рецепти чи відео, користувачам може знадобитися підписка

Dessert Recipes — це безкоштовний мобільний застосунок, доступний на платформі Android, який пропонує користувачам широкий вибір рецептів десертів. Застосунок містить рецепти для класичних десертів, таких як амішський пиріг, імбирні гарбузові кекси, класичний торт на день народження, гранита з кавуна та червоний оксамитовий пломбір [4].

Переваги представлені в таблиці 1.5.

Таблиця 1.5 – Переваги Dessert Recipes

№	Переваги	Опис
1	Різноманітність рецептів	Застосунок пропонує широкий спектр десертів, від простих до більш складних, що задовольняють різні смаки та рівні майстерності
2	Безкоштовний доступ	Користувачі можуть безкоштовно завантажити та використовувати застосунок, що робить його доступним для широкої аудиторії
3	Оновлення контенту	Розробники регулярно додають нові рецепти, забезпечуючи постійне оновлення контенту

Недоліки надані в таблиці 1.6.

Таблиця 1.6 – Недоліки Dessert Recipes

№	Недоліки	Опис
1	Обмежені можливості персоналізації	Застосунок не надає можливості зберігати улюблені рецепти або створювати списки покупок, що може бути незручним для користувачів
2	Відсутність інтерактивних функцій	Немає можливості взаємодіяти з іншими користувачами, залишати відгуки або ділитися власними рецептами
3	Обмежена інформація про авторів рецептів	Користувачі не отримують детальної інформації про авторів рецептів, що може знижувати довіру до їхньої якості

Great British Chefs — це британський кулінарний вебсайт, який містить тисячі рецептів, інструкцій, відео та профілів шеф-кухарів і ресторанів Великобританії. Він був заснований групою підприємців, включаючи Вімала та Гілл Хосла, Оллі Ллойда та Аластера Крімер. Перший застосунок "Great British Chefs" для iOS був випущений у липні 2011 року та включав 12 шеф-кухарів, кожен з яких надав по 5 рецептів [9].

Переваги прописані в таблиці 1.7 [10]. Недоліки описані в таблиці 1.8.

Таблиця 1.7 – Переваги Great British Chefs

№	Переваги	Опис
1	Висока якість контенту	Рецепти надаються відомими британськими шеф-кухарями, що гарантує їхню якість та автентичність
2	Інтерактивність	Користувачі можуть зберігати рецепти, створювати персональні колекції та отримувати рекомендації на основі своїх вподобань
3	Різноманітність форматів	Окрім текстових рецептів, доступні відеоінструкції та статті, що поглиблюють кулінарні знання користувачів
4	Підтримка декількох платформ	Окрім вебсайту, Great British Chefs пропонує застосунки для iOS та Windows 8, що забезпечує зручність доступу

Таблиця 1.8 – Недоліки Great British Chefs

№	Недоліки	Опис
1	Фокус на британській кухні	Основна увага приділяється британській кухні, що може обмежувати інтерес користувачів з інших культур
2	Платний контент	Деякі розділи або функції можуть вимагати оплати, що може бути недоліком для користувачів, які шукають безкоштовні ресурси
3	Потреба в інтернет-з'єднанні	Для доступу до більшості функцій потрібне постійне підключення до інтернету, що може бути незручним у деяких ситуаціях

Cake Master — це інтерфейсний застосунок для створення та оформлення тортів та інших десертів. Окрім рецептів, застосунок пропонує інструменти для планування приготування, вибору інгредієнтів та надає покрокові інструкції для випікання. Основний акцент у ньому робиться на те, щоб допомогти користувачам покроково створювати приголомшливі десерти навіть без попереднього досвіду у випічці [6].

Переваги наведені в таблиці 1.9.

Таблиця 1.9 – Переваги Cake Master

№	Переваги	Опис
1	Різноманіття рецептів	Cake Master пропонує велику кількість рецептів, які підходять для всіх рівнів складності — від початківців до досвідчених кулінарів
2	Інтерактивні функції	Користувач може зберігати улюблені рецепти, експериментувати з інгредієнтами та отримувати рекомендації, залежно від вибору продуктів
3	Покрокові інструкції	Кожен рецепт супроводжується чіткими, зрозумілими інструкціями з фото та відео, що робить процес випічки легким і доступним для всіх
4	Планувальник покупок	Застосунок дозволяє створювати списки покупок на основі вибраних рецептів, що значно економить час та зменшує ймовірність помилок
5	Інтеграція з соцмережами	Можливість ділитися готовими тортами та десертами у соціальних мережах, що сприяє популяризації та обміну досвідом серед користувачів

Недоліки описані в таблиці 1.10.

Таблиця 1.10 – Недоліки Cake Master

№	Недоліки	Опис
1	Регіональна орієнтованість	Деякі рецепти орієнтовані на певні країни, що може ускладнити їх використання для користувачів з інших регіонів через специфічні інгредієнти
2	Обмежена доступність на інших платформах	Хоча застосунок доступний на Android та iOS, його функціональність може бути обмежена на деяких пристроях, таких як планшети або старіші версії мобільних телефонів
3	Не всі функції безкоштовні	Для доступу до всіх рецептів та функцій необхідно підписатися на преміум-версію застосунку, що може бути недоступно для деяких користувачів

Dr. Oetker — це німецька компанія з понад столітньою історією, відома своїми продуктами для випічки та десертів. Їхній застосунок "Rezeptideen" пропонує користувачам широкий спектр рецептів для випічки, десертів та інших страв [7].

Переваги представлені в таблиці 1.11 а недоліки описані у таблиці 1.12.

Таблиця 1.11 – Переваги Dr. Oetker

№	Переваги	Опис
1	Різноманітність рецептів	Застосунок містить велику кількість перевірених рецептів, які підходять для різних рівнів складності та смакових уподобань
2	Інтерактивність	Користувачі можуть зберігати улюблені рецепти, створювати списки покупок та отримувати персоналізовані рекомендації
3	Покрокові інструкції	Кожен рецепт супроводжується детальними інструкціями та фотографіями, що полегшує процес приготування

Таблиця 1.12 – Недоліки Dr. Oetker

№	Недоліки	Опис
1	Обмежена доступність мов	Основна версія застосунку доступна німецькою мовою, що може бути незручним для користувачів, які не володіють нею
2	Регіональна спрямованість	Деякі інгредієнти або рецепти можуть бути специфічними для німецької кухні, що може ускладнити їх використання в інших країнах

Betty Crocker — це бренд, створений компанією General Mills у США, відомий своїми кулінарними книгами та продуктами для випічки. Застосунок Betty Crocker Cookbook надає користувачам доступ до великої бази рецептів та кулінарних порад [8].

Переваги таблиці 1.13.

Таблиця 1.13 – Переваги Betty Crocker

№	Переваги	Опис
1	Широкий асортимент рецептів	Застосунок пропонує рецепти для різних страв, включаючи десерти, закуски та основні страви
2	Зручний інтерфейс	Інтуїтивно зрозумілий дизайн дозволяє легко знаходити потрібні рецепти та додавати їх до списку улюблених
3	Додаткові функції	Користувачі можуть створювати списки покупок, переглядати відеоінструкції та отримувати поради щодо приготування

Недоліки до даного застосунку були детально описані в наступній таблиці 1.14.

Таблиця 1.14 – Недоліки Betty Crocker

№	Недоліки	Опис
1	Реклама	У безкоштовній версії застосунку може бути присутня реклама, яка відволікає від користування
2	Необхідність інтернет-з'єднання	Деякі функції вимагають постійного підключення до інтернету, що може бути незручним для користувачів з обмеженим доступом

Вивчення конкурентів є важливою частиною при створенні нового програмного продукту, оскільки воно дає змогу виявити сильні та слабкі сторони поточних рішень на ринку, а також визначити, які інноваційні функції можна додати, щоб привернути увагу користувачів. Аналіз конкурентів наведено в таблиці 1.15.

Таблиця 1.15 – Аналіз функцій конкурентів та існуючих рішень

Функція	Tasty	Kitchen Stories	Dessert Recipes	Great British Chefs	Cake Master	Dr. Oetker Rezeptideens	Betty Crocker Cookbook
Покрокові інструкції	Так	Так	Так	Так	Так	Так	Так
Відеоуроки	Так	Так	Ні	Так	Так	Так	Так
Створення власної кулінарної книги	Ні	Так	Ні	Ні	Ні	Ні	Так
Інтерактивні майстер-класи	Так	Так	Ні	Ні	Так	Так	Ні
3D-моделі для декорування	Ні	Ні	Ні	Ні	Так	Ні	Ні
Віртуальний дизайн	Ні	Ні	Ні	Ні	Так	Ні	Ні
Спільнота	Так	Так	Так	Так	Так	Так	Так
Професійні поради	Так	Так	Так	Так	Так	Так	Так

Вивчення конкурентів показало, що на ринку існує безліч рішень для кондитерських застосунків, кожен з яких має свої переваги та обмеження. Найбільш універсальними є застосунки, які пропонують покрокові інструкції та професійні поради, а також функції для створення власної кулінарної книги та активні спільноти користувачів. Однак для застосунків, орієнтованих на

творчість у кондитерському мистецтві, важливою є наявність таких інструментів, як 3D-моделі для декорування та інтерактивні майстер-класи.

Для розробників, які бажають створити конкурентоспроможний продукт, важливо звернути увагу на ці функції та намагатися інкорпорувати їх у нові рішення для задоволення потреб користувачів.

1.3 Визначення потреб користувачів та специфікацій

1.3.1 Функціональні вимоги до застосунку

Функціональні вимоги до інтерактивного застосунку для кондитерів визначають основні можливості, які повинен надавати застосунок для задоволення потреб користувачів. Вони охоплюють широкий спектр функцій, спрямованих на полегшення процесу створення, пошуку, зберігання та обміну рецептами кондитерських виробів. Функціональні вимоги, які реалізовані та які можуть бути додані в майбутніх версіях в застосунку, показані на рисунку 1.9.

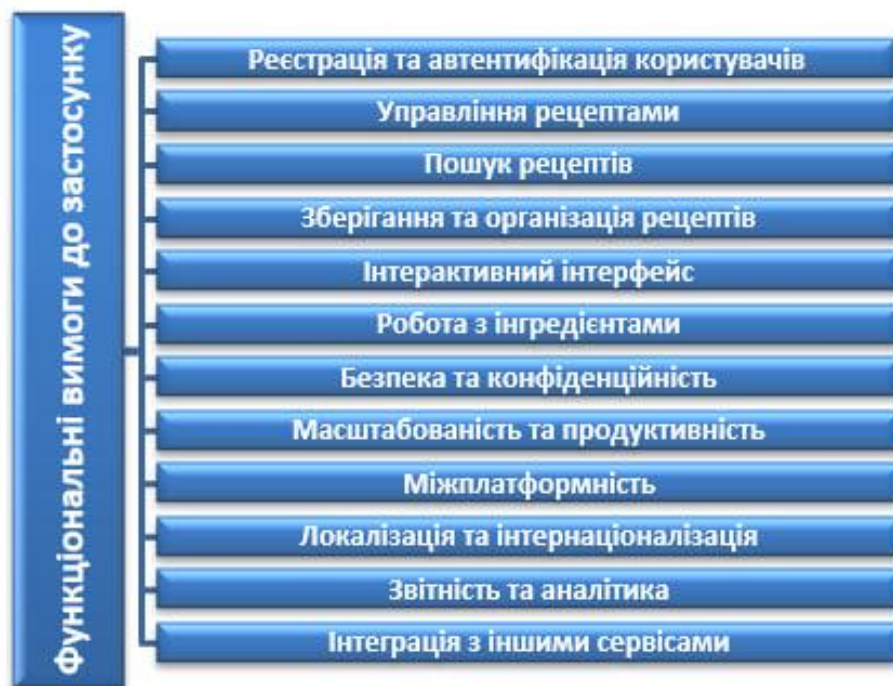


Рисунок 1.9 – Функціональні вимоги до застосунку

Детальний опис функціональних вимог до застосунку показаний в таблиці 1.16.

Таблиця 1.16 – Функціональні вимоги до застосунку

№	Функції	Опис
1	Реєстрація та автентифікація користувачів	
1.1	Реєстрація нового користувача	Застосунок повинен надавати можливість новим користувачам створювати облікові записи. Під час реєстрації користувачі вказують унікальне ім'я користувача, пароль та отримують роль «REGISTERED». Це дозволяє їм отримати доступ до повного функціоналу застосунку, включаючи створення та редагування рецептів [11]
1.2	Автентифікація	Зареєстровані користувачі можуть увійти в систему, ввівши свої облікові дані. Після успішної автентифікації користувач отримує токен, який використовується для ідентифікації під час подальшої роботи з застосунком [12]
1.3	Гостьовий доступ	Застосунок надає можливість використовувати базові функції (перегляд рецептів, пошук) без реєстрації. Гості не мають доступу до функцій створення, редагування або видалення рецептів [13]
2	Управління рецептами	
2.1	Створення нового рецепту	Зареєстровані користувачі можуть створювати нові рецепти. Для цього вони вказують назву рецепту, список інгредієнтів (кожен інгредієнт з нового рядка) та детальні інструкції з приготування. Застосунок автоматично зберігає рецепт у базі даних та пов'язує його з користувачем, який його створив
2.2	Редагування рецептів	Автори рецептів можуть редагувати свої рецепти. Це включає зміну назви, оновлення списку інгредієнтів або інструкцій. Редагування доступне лише для користувачів, які є авторами конкретного рецепту
2.3	Видалення рецептів	Користувачі можуть видаляти свої рецепти. Після видалення рецепт повністю видаляється з бази даних, і його більше неможливо відновити
2.4	Перегляд рецептів	Усі користувачі, незалежно від їхнього статусу (зареєстровані або гості), можуть переглядати рецепти. Для кожного рецепту відображається назва, список інгредієнтів, інструкції та ім'я автора

Продовження таблиці 1.16

№	Функції	Опис
3	Пошук рецептів	
3.1	Пошук за назвою	Користувачі можуть шукати рецепти за ключовими словами, які входять до назви рецепту. Наприклад, пошук за словом «торт» поверне всі рецепти, у назві яких є це слово
3.2	Пошук за інгредієнтами	Застосунок дозволяє шукати рецепти за інгредієнтами. Користувачі можуть вводити назви інгредієнтів, і система поверне всі рецепти, які містять ці інгредієнти
4	Зберігання та організація рецептів	
4.1	Збереження рецептів у базі даних	Усі рецепти зберігаються в централізованій базі даних, що забезпечує їхню доступність для всіх користувачів. Кожен рецепт містить унікальний ідентифікатор, назву тощо
5	Інтерактивний інтерфейс	
5.1	Зручний інтерфейс для створення рецептів	Застосунок надає інтуїтивно зрозумілий інтерфейс для введення даних про рецепти. Користувачі можуть легко додавати інгредієнти та інструкції, використовуючи текстові поля та кнопки
5.2	Візуалізація рецептів	Кожен рецепт відображається у зручному форматі з чітким розділенням на інгредієнти та інструкції. Це полегшує процес приготування для користувачів
5.3	Підтримка мультимедіа	У майбутніх версіях застосунок може підтримувати завантаження зображень або відео для ілюстрації рецептів [14]
6	Робота з інгредієнтами	
6.1	Автоматичне розпізнавання інгредієнтів	Під час створення рецепту застосунок автоматично розпізнає назви інгредієнтів у тексті та зберігає їх у базі даних. Це дозволяє оптимізувати пошук рецептів за інгредієнтами
6.2	Унікальні інгредієнти	Кожен інгредієнт зберігається в базі даних як унікальний запис, що дозволяє уникнути дублювання та полегшує пошук
7	Безпека та конфіденційність	
7.1	Захист даних користувачів	Застосунок забезпечує збереження паролів у звичайному вигляді. Токени автентифікації використовуються для захисту доступу до особистих даних користувачів [15]. Також користувачі повинні мати змогу змінити пароль
7.2	Обмежений доступ до функцій	Лише зареєстровані користувачі можуть створювати, редагувати та видаляти рецепти. Гості мають доступ лише до перегляду та пошуку [16]

Завершення таблиці 1.16

№	Функції	Опис
8	Масштабованість та продуктивність	
8.1	Оптимізація запитів	Для підвищення продуктивності використовуються оптимізовані SQL-запити, які мінімізують час відгуку системи [17]
8.2	Підтримка великої кількості рецептів	Застосунок розроблений для ефективної роботи з великими обсягами даних. Використання індексів у базі даних забезпечує швидкий пошук рецептів навіть за наявності сотні записів
9	Міжплатформність	
9.1	Підтримка різних пристроїв	Застосунок може працювати на різних платформах, включаючи настільні комп'ютери та ноутбуки [18]
10	Локалізація та інтернаціоналізація	
10.1	Підтримка різних мов	Застосунок може бути адаптований для різних мовних аудиторій. Наразі інтерфейс підтримує українську мову, але в майбутньому можна додати підтримку інших мов [19]
11	Звітність та аналітика	
11.1	Статистика використання	Адміністратори застосунку можуть отримувати звіти про активність користувачів, включаючи кількість створених рецептів, популярність певних рецептів та інші метрики [20]
11.2	Аналітика пошукових запитів	Застосунок може збирати дані про пошукові запити, що дозволяє визначати популярні інгредієнти [21]
12	Інтеграція з іншими сервісами	
12.1	Інтеграція з соціальними мережами	У майбутніх версіях можна реалізувати функцію обміну рецептами через соціальні мережі [22]

Ці функціональні вимоги забезпечують повноцінну роботу застосунку, спрямованого на задоволення потреб користувачів у створенні, пошуку та обміні кондитерськими рецептами. Вони враховують як базові потреби користувачів, так і можливості для подальшого розширення функціоналу.

1.3.2 Нефункціональні вимоги

Нефункціональні вимоги (рисунок 1.10) визначають якісні характеристики системи, які не стосуються безпосередньо функціональності, але впливають на загальну ефективність, зручність використання, безпеку та інші критерії, що забезпечують стабільну роботу програми (таблиця 1.17). У контексті інтерактивного застосунку для кондитерів, нефункціональні вимоги можна розділити на кілька ключових категорій: продуктивність, масштабованість, зручність використання, безпека, надійність, підтримка та сумісність. Кожна з цих категорій має важливе значення для забезпечення якісної роботи застосунку та задоволення потреб користувачів.



Рисунок 1.10 – Нефункціональні вимоги

Таблиця 1.17 – Нефункціональні вимоги до застосунку

№	Функції	Опис
1	Продуктивність	Продуктивність є однією з ключових характеристик, яка визначає, наскільки швидко та ефективно система обробляє запити користувачів. У застосунку для кондитерів продуктивність має бути забезпечена на всіх етапах взаємодії з системою, включаючи автентифікацію, пошук рецептів, створення нових рецептів, оновлення та видалення існуючих [23]
1.1	Час відгуку	Система повинна забезпечувати час відгуку не більше 2 секунд для більшості операцій, таких як пошук рецептів, відображення списку рецептів та автентифікація користувачів. Для операцій, що вимагають обробки великих обсягів даних (наприклад, пошук за ключовими словами), час відгуку може збільшуватися до 5 секунд, але не більше [24]
1.2	Обробка запитів	Система повинна підтримувати одночасну обробку до 1000 запитів від користувачів без значного зниження продуктивності. Це забезпечить стабільну роботу застосунку під час пікового навантаження, наприклад, під час проведення кулінарних майстер-класів або акцій у майбутніх версіях [25]
1.3	Оптимізація бази даних	Для забезпечення швидкості пошуку та обробки даних необхідно використовувати індексацію ключових полів у базі даних, таких як назви рецептів, інгредієнти та імена користувачів. Це дозволить зменшити час виконання запитів та підвищити загальну продуктивність системи [26]
2	Масштабованість	Масштабованість визначає здатність системи обробляти зростаюче навантаження без втрати продуктивності. У контексті кондитерського застосунку масштабованість є критично важливою, оскільки кількість користувачів та рецептів може зростати з часом [27]
2.1	Горизонтальне масштабування	Система повинна підтримувати можливість додавання нових серверів для розподілу навантаження. Це дозволить забезпечити стабільну роботу застосунку навіть при значному збільшенні кількості користувачів [28]
2.2	Вертикальне масштабування	У разі необхідності система повинна дозволяти підвищувати продуктивність окремих компонентів, наприклад, збільшення обсягу оперативної пам'яті або кількості ядер процесора на сервері [28]

Продовження таблиці 1.17

№	Функції	Опис
2.3	Розподілене зберігання даних	Для забезпечення масштабованості необхідно використовувати розподілені системи зберігання даних, такі як NoSQL бази даних або хмарні сховища. Це дозволить ефективно керувати великими обсягами даних та забезпечити їх доступність у разі виходу з ладу окремих вузлів [29]
3	Зручність використання	Зручність використання є ключовим фактором, який впливає на задоволеність користувачів від взаємодії з застосунком. Інтерфейс повинен бути інтуїтивно зрозумілим, а функціональність доступною для користувачів з різним рівнем технічних знань [30]
3.1	Інтуїтивний інтерфейс	Інтерфейс застосунку повинен бути простим та зрозумілим, з чіткою навігацією та мінімальною кількістю кроків для виконання основних операцій. Наприклад, пошук рецептів має бути доступним з головного екрану, а створення нового рецепту – у кілька кліків [31]
3.2	Підтримка різних пристроїв	Наразі застосунок коректно працює на різних пристроях, включаючи ноутбуки та комп'ютери. У майбутніх версіях можна додати можливість працювати із застосунком на планшетах та смартфонах. Для цього необхідно використовувати адаптивний дизайн [32]
3.3	Локалізація	Наразі застосунок підтримує лише одну мову – українську. Для забезпечення зручності використання застосунок може підтримувати кілька мов, включаючи українську, англійську та інші популярні мови. Це дозволить залучити більшу аудиторію користувачів [33]
3.4	Доступність	Застосунок повинен бути доступним для користувачів з обмеженими можливостями, наприклад, мати високу контрастність для забезпечення зручного використання для людей з порушенням зору [34]
4	Безпека	Безпека є критично важливим питанням будь-якого застосунку, особливо коли мова йде про зберігання особистих даних користувачів та їхніх рецептів. У кондитерському застосунку необхідно забезпечити захист від несанкціонованого доступу, втрати даних та інших загроз [35]
4.1	Автентифікація та авторизація	Доступ до певних функцій (наприклад, редагування або видалення рецептів) повинен бути обмежений, лише для авторизованих користувачів [36]

Продовження таблиці 1.17

№	Функції	Опис
4.2	Зміна паролю	У зареєстрованих користувачів має бути можливість змінити пароль за потреби. Унікальність та непередбачувана послідовність символів у паролі обумовлюють його складність [37]. Така можливість викликає довіру користувачів до застосунку
4.3	Резервне копіювання	У майбутніх версіях застосунку для запобігання втрати даних у разі збоїв або атак необхідно реалізувати регулярне резервне копіювання даних. Резервні копії повинні зберігатися в безпечному місці та бути доступними для відновлення у разі необхідності [38]
5	Надійність	Надійність визначає здатність системи працювати без збоїв та забезпечувати доступність даних у будь-який час [39]. У кондитерському застосунку надійність є критично важливою, оскільки користувачі покладаються на систему для зберігання та пошуку рецептів
5.1	Стабільність роботи	Система повинна забезпечувати безперебійну роботу 24/7 з мінімальним часом простою [40]. Для цього необхідно використовувати надійне обладнання та програмне забезпечення
5.2	Моніторинг та логування	Для оперативного виявлення та усунення проблем необхідно реалізувати систему моніторингу, яка відстежує стан серверів, бази даних та інших критичних компонентів. Логування подій дозволить аналізувати причини збоїв та вдосконалювати систему [41]
5.3	Обробка помилок	Система повинна коректно обробляти помилки, що виникають під час роботи, та надавати користувачам зрозумілі повідомлення про причини проблем [42]. Наприклад, у разі недоступності сервера користувач повинен отримати повідомлення про тимчасові технічні проблеми
6	Підтримка	Підтримка включає забезпечення технічної допомоги користувачам, оновлення системи та виправлення помилок [43]. У контексті кондитерського застосунку підтримка є важливим елементом, який забезпечує довгострокову роботу системи

Завершення таблиці 1.17

№	Функції	Опис
6.1	Технічна підтримка	Користувачі повинні мати можливість звертатися за допомогою у разі виникнення проблем. Для цього необхідно надати доступ до служби підтримки через електронну пошту або телефон [44]
6.2	Оновлення системи	Система повинна підтримувати регулярні оновлення, які включають виправлення помилок, покращення продуктивності та додавання нових функцій. Оновлення повинні бути безпечними та не порушувати роботу існуючих функцій [45]
6.3	Документація	Для забезпечення зручності використання застосунку необхідно надати детальну документацію, яка включає інструкції з використання [46]
7	Сумісність	Сумісність визначає здатність системи працювати з різними операційними системами та пристроями [47]. У контексті кондитерського застосунку сумісність є важливою для забезпечення широкого охоплення аудиторії
7.1	Підтримка операційних систем	Застосунок повинен коректно працювати на популярних операційних системах, таких як Windows, macOS, Linux [48]
7.2	Адаптація до нових технологій	Система повинна бути готова до інтеграції у майбутньому з новими технологіями, такими як віртуальна реальність або штучний інтелект, для забезпечення конкурентоспроможності на ринку [49]

Нефункціональні вимоги є важливим елементом розробки інтерактивного кондитерського застосунку, оскільки вони забезпечують стабільну, безпечну та зручну роботу системи. Врахування цих вимог дозволяє створити продукт, який відповідає очікуванням користувачів та забезпечує високу якість обслуговування. Реалізація цих вимог вимагає ретельного планування, тестування та постійного вдосконалення системи для забезпечення її ефективної роботи у довгостроковій перспективі.

1.4 Постановка завдання

1.4.1 Опис основних функцій програми

Сучасний світ диктує нові вимоги до інформаційних технологій, зокрема в галузі кулінарії. Інтерактивний застосунок для кондитерів, розроблений на мові програмування Java, є прикладом інноваційного підходу до організації процесу створення, зберігання та поширення рецептів кондитерських виробів. Основна мета застосунку полягає в забезпеченні зручного та ефективного інструменту для користувачів, які цікавляться кулінарією, зокрема кондитерським мистецтвом. Застосунок надає можливість реєстрації, авторизації, зміни паролю, публікації рецептів та коментування рецептів, що робить його незамінним помічником як для професійних кулінарів, так і для аматорів (рисунок 1.11).

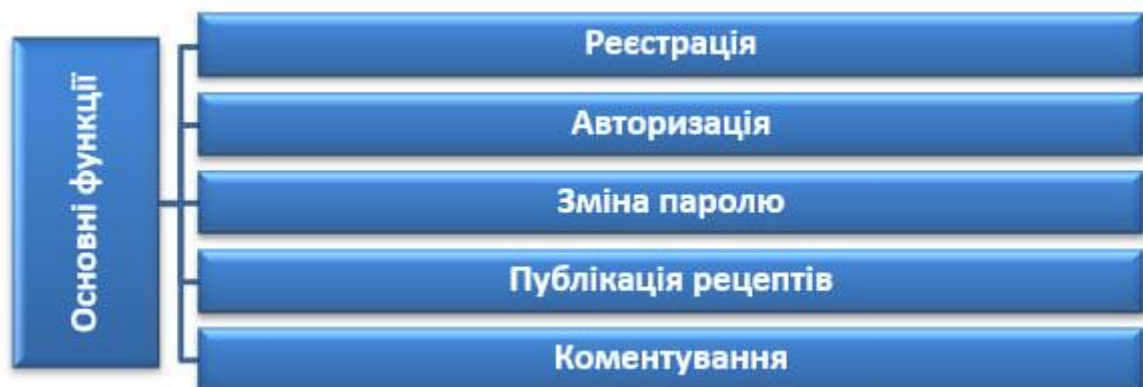


Рисунок 1.11 - Основні функції

Реєстрація є першим кроком для користувача, який бажає скористатися всіма можливостями застосунка. Вона передбачає створення особистого облікового запису, де зберігаються ім'я та пароль. Під час реєстрації дані перевіряються на унікальність та коректність. У разі успішної реєстрації користувач отримує доступ до особистого кабінету та роль «REGISTERED», що дозволяє публікувати рецепти, редагувати їх та видаляти за потреби.

Авторизація забезпечує вхід користувача в систему через введення логіну та пароллю. Після успішної авторизації користувач отримує токен для ідентифікації під час роботи із застосунком. Цей механізм забезпечує доступ до особистого кабінету та функцій, доступних для зареєстрованих користувачів. Гості мають обмежений доступ, що підтримує порядок у застосунку.

Можливість змінити пароль як додатковий захист даних користувача. Якщо пройшла певна кількість часу після реєстрації або зміни пароллю, то бажано оновити пароль знов. Форма зміни пароллю включає такі поля для введення даних користувача: поточний пароль, новий та підтвердження нового пароллю.

Публікація рецептів дозволяє користувачам ділитися своїми рецептами зі спільнотою. Форма публікації включає поля для назви, інгредієнтів та інструкцій. Після збереження рецепт стає доступним для інших користувачів. Кожен рецепт зв'язується з обліковим записом автора, що забезпечує прозорість. Користувачі також можуть редагувати або видаляти свої рецепти.

Функція коментування дозволяє зареєстрованим користувачам додавати коментарі до рецептів, переглядати їх та асоціювати з конкретним рецептом і автором. Користувач може ввести текст коментаря, який після відправки зберігається в базі даних із зазначенням часу створення та імені автора. Ця функція сприяє взаємодії між користувачами, дозволяючи обмінюватися думками та відгуками про рецепти. Коментарі відображаються в інтерфейсі застосунка, що робить його більш інтерактивним і соціально орієнтованим.

Таким чином, застосунок забезпечує зручність та функціональність для користувачів, які цікавляться кулінарією. Реєстрація та авторизація забезпечують безпеку, публікація рецептів дозволяє ділитися знаннями, а коментування рецептів робить застосунок більш корисним у використанні.

1.4.2 Вимоги до інтерфейсу

Інтерфейс інтерактивного застосунку для кондитерів є ключовим елементом, який забезпечує зручну взаємодію користувача з функціональними можливостями програми. Він повинен бути інтуїтивно зрозумілим, естетично привабливим та відповідати сучасним стандартам UX/UI дизайну. Основна мета інтерфейсу – забезпечити ефективну навігацію, швидкий доступ до інформації та зручність використання для користувачів з різним рівнем технічної підготовки.

Інтерфейс застосунку розроблений з урахуванням логічної структури, яка дозволяє користувачам легко орієнтуватися в програмі. Він складається з основних елементів, що описані в таблиці 1.18.

Таблиця 1.18 – Структура інтерфейсу

№	Елементи	Опис
1	Головне меню	містить посилання на основні розділи застосунку, такі як рецепти, пошук, створення нового рецепту, контакт із підтримкою, зміна пароля та вихід (деякі пункти лише для авторизованого користувача)
2	Панель навігації	забезпечує швидкий доступ до часто використовуваних функцій, таких як оновлення списку рецептів, пошук за ключовими словами або інгредієнтами
3	Робоча область	відображає основну інформацію, наприклад, список рецептів, деталі конкретного рецепту або форму для створення/редагування рецепту
4	Панель стану	відображає поточний статус користувача (авторизований/неавторизований), ім'я користувача та роль (гість, зареєстрований користувач)

Кольорова схема інтерфейсу обрана з урахуванням тематики застосунку. Використовуються теплі та природні кольори, які асоціюються з кондитерськими виробами:

- основний фон: світлий бежевий колір (#DDC5A2), який нагадує крем або глазур, що створює затишну атмосферу;

- акцентні кольори: темно-коричневий (#301B28) для заголовків та тексту, який асоціюється з шоколадом, та червонувато-коричневий (#B6452C) для кнопок та інтерактивних елементів, що нагадує карамель;

- додаткові кольори: використовуються для підкреслення важливих елементів, таких як повідомлення про помилки або успішні дії.

Типографіка – для забезпечення читабельності тексту використовуються сучасні шрифти з чіткими лініями:

- заголовки: шрифт із напівжирним накресленням (FontWeight.BOLD) та розміром 20-24 пікселів;

- основний текст: стандартний шрифт із розміром 14-16 пікселів;

- підказки та додатковий текст: шрифт із розміром 12 пікселів та світлішим кольором для візуального розділення інформації.

Інтерфейс містить різноманітні інтерактивні елементи, які забезпечують зручну взаємодію (таблиця 1.19).

Таблиця 1.19 – Інтерактивні елементи інтерфейсу

№	Елементи	Опис
1	Кнопки	Мають контрастні кольори, відповідні кольоровій схемі
2	Текстові поля	Використовуються для введення даних, таких як назва, інгредієнти та інструкції для створення рецепту або таких як ім'я користувача та пароль для авторизації чи реєстрації. Вони мають підказки, які зникають після початку введення

Інтерфейс забезпечує доступність для користувачів з обмеженими можливостями:

- висока контрастність: кольори підібрані таким чином, щоб забезпечити чітку видимість тексту та елементів;

- підказки та описи: всі інтерактивні елементи мають текстові підказки, які допомагають користувачам зрозуміти їх призначення.

Інтерфейс забезпечує зворотний зв'язок з користувачем:

– повідомлення про помилки відображаються у випадку некоректного введення даних або виникнення проблем під час виконання операцій. Вони мають червоний колір та чіткий текст.

Зареєстровані користувачі мають доступ до додаткових функцій, які відображаються в інтерфейсі (таблиця 1.20).

Таблиця 1.20 – Особливості інтерфейсу для зареєстрованих користувачів

№	Особливості	Опис
1	Кнопка «Створити рецепт»	Дозволяє додавати нові рецепти до бази даних
2	Редагування та видалення рецептів	Користувачі можуть редагувати або видаляти рецепти, які вони створили
3	Оновлення паролю	Користувачі мають можливість за потреби змінити поточний пароль на новий

Гості мають обмежений доступ до функцій застосунку:

- перегляд списку рецептів;
- пошук рецептів за ключовими словами або інгредієнтами;
- перегляд деталей рецептів без можливості редагування або видалення.

Інтерфейс забезпечує зручний пошук та фільтрацію рецептів (таблиця 1.21).

Таблиця 1.21 – Особливості пошуку

№	Особливості	Опис
1	Пошук за назвою	Користувачі можуть вводити ключові слова для пошуку рецептів
2	Фільтрація за інгредієнтами	Дозволяє вибирати рецепти, які містять певні інгредієнти
3	Сортування	Рецепти автоматично сортуються за датою додавання

Інтерфейс відображення деталей рецепту включає:

- назву рецепту;
- ім'я автора;

- список інгредієнтів;
- детальні інструкції з приготування;
- кнопки для редагування або видалення (для автора рецепту);
- рейтинг рецепту та коментарі.

Форма для створення або редагування рецепту містить:

- поле для введення назви рецепту;
- текстове поле для введення інгредієнтів;
- текстове поле для введення інструкцій;
- кнопки для збереження або скасування змін.

Інтерфейс авторизації та реєстрації включає:

- поля для введення логіну та пароля;
- кнопки для входу, реєстрації або входу як гість;
- повідомлення про помилки при некоректному введенні даних.

Форма для зміни паролю включає:

- поле для введення поточного паролю;
- поля для введення та підтвердження нового паролю.

Інтерфейс застосунку розроблений з урахуванням сучасних тенденцій дизайну:

- використання мінімалістичного стилю з акцентом на контент.

Інтерфейс пройшов кілька етапів тестування для забезпечення стабільності та зручності:

- оптимізація швидкості завантаження та відгуку інтерфейсу;
- виправлення помилок, виявлених під час тестування.

Інтерфейс інтерактивного кондитерського застосунку розроблений з урахуванням сучасних вимог до UX/UI дизайну. Він забезпечує зручну навігацію, швидкий доступ до інформації та ефективну взаємодію з користувачем. Використання теплих кольорів та інтуїтивно зрозумілих елементів робить застосунок привабливим та зручним для користувачів з різним рівнем технічної підготовки.

1.4.3 Вимоги до безпеки

Безпека є одним із ключових питань розробки будь-якого програмного забезпечення, особливо коли мова йде про застосунки, які працюють з конфіденційними даними користувачів, такими як особиста інформація, паролі та інші відомості, що потребують захисту. У контексті інтерактивного застосунку для кондитерів, який надає можливість реєстрації, автентифікації, створення та зберігання рецептів, вимоги до безпеки мають бути ретельно продумані та реалізовані (рисунк 1.12). Це забезпечить не лише захист даних користувачів, але й довіру до застосунку в цілому.

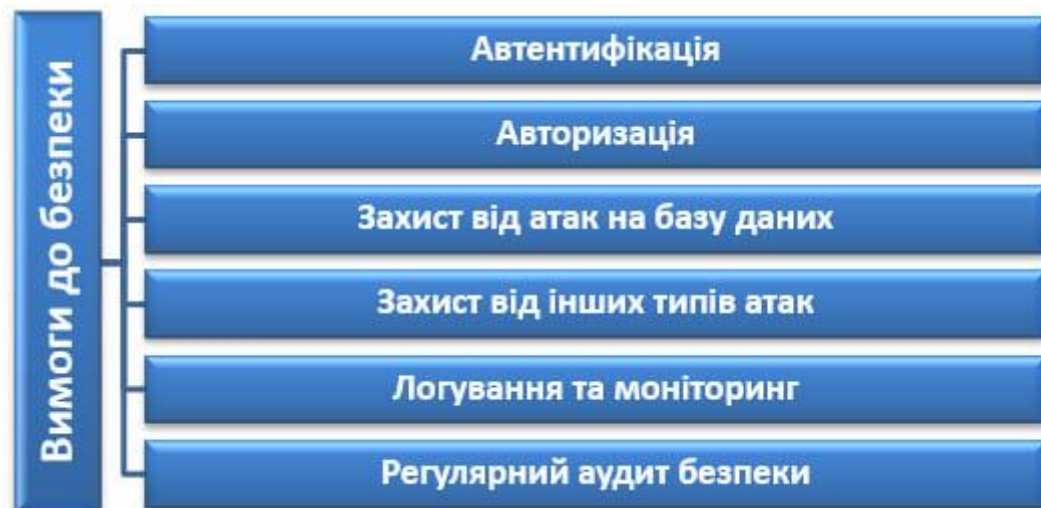


Рисунок 1.12 – Вимоги до безпеки

Автентифікація реалізується через токени, які містять інформацію про користувача та термін дії.

Авторизація забезпечує доступ до функцій застосунку лише для авторизованих користувачів. Наприклад, створення, редагування та видалення рецептів доступні лише для зареєстрованих користувачів, а перегляд — для всіх. Ролі (наприклад, "GUEST", "REGISTERED") контролюють рівень доступу.

Для захисту від SQL-ін'єкцій використовуються параметризовані запити або підготовлені оператори.

Для захисту від інших атак (ін'єкція команд, атаки на файлову систему) ретельно перевіряються всі вхідні дані, обмежується доступ до файлової системи та регулярно оновлюються сторонні бібліотеки.

Система логування та моніторингу виявляє підозрілу активність. Логи містять інформацію про спроби входу, зміни даних та інші важливі події.

Для підтримки високого рівня безпеки проводяться регулярні аудити системи, включаючи перевірку коду, конфігурацій серверів та баз даних.

Вимоги до безпеки включають використання ретельної перевірки вхідних даних, регулярний аудит та постійне оновлення знань розробників. Це забезпечує довіру користувачів та стабільну роботу застосунку.

1.4.4 Нефункціональні вимоги

Нефункціональні вимоги до інтерактивного кондитерського застосунку охоплюють продуктивність, безпеку, зручність, масштабованість, підтримку та інші фактори, які забезпечують стабільну та ефективну роботу системи (таблиця 1.22).

Таблиця 1.22 – Нефункціональні вимоги

№	Елементи	Опис
1	Продуктивність	Застосунок повинен забезпечувати швидкий час відгуку під час автентифікації, пошуку, створення, оновлення та видалення рецептів
2	Масштабованість	Система повинна підтримувати зростання кількості користувачів та рецептів без втрати продуктивності
3	Надійність	Застосунок має бути відмовостійким, з постійним моніторингом для уникнення збоїв
4	Безпека	Важливо забезпечити захист особистих даних користувачів та рецептів через автентифікацію та інші механізми безпеки
5	Зручність інтерфейсу	Інтерфейс повинен бути інтуїтивно зрозумілим для користувачів з різним рівнем технічної підготовки

Продовження таблиці 1.22

№	Елементи	Опис
6	Підтримка та обслуговування	Система має бути легко підтримуваною, з чіткою документацією та можливістю швидкого виправлення помилок
7	Сумісність	Застосунок повинен працювати на різних операційних системах
8	Тестування	Система повинна пройти ретельне тестування для забезпечення якості та стабільності
9	Ліцензійні вимоги	Застосунок має відповідати вимогам щодо використання сторонніх бібліотек та компонентів

Ці вимоги забезпечують стабільну, безпечну та ефективну роботу застосунку, що є ключовим для задоволення потреб користувачів та довгострокового успіху системи.

1.5 Висновки за розділом 1

Сучасний ринок застосунків для кондитерів пропонує широкий спектр функціональних можливостей, що відображає зростаючий інтерес до кулінарії. Більшість платформ надають доступ до рецептів, їх збереження та пошук за ключовими параметрами. Однак багато застосунків мають обмежену функціональність, що не задовольняє потреби користувачів, які шукають інтерактивний та персоналізований досвід. Висока конкуренція на ринку зумовлює необхідність розробки унікальних функцій, які забезпечують зручність та швидкість використання.

Аналіз конкурентів показав, що популярні застосунки мають зручний інтерфейс, можливість швидкого пошуку рецептів за інгредієнтами або категоріями, а також функції збереження улюблених рецептів. Однак багато платформ не враховують потреб користувачів у соціальній взаємодії, що створює нішу для розробки застосунку, який поєднає функціональність соціальної мережі з інструментами для створення та зберігання рецептів. Основні потреби користувачів включають зручність, простоту використання та

персоналізацію. Користувачі очікують можливості швидкого пошуку рецептів, обміну власними кулінарними ідеями.

Функціональні вимоги до застосунку включають реєстрацію та авторизацію, можливість змінити пароль, публікацію рецептів, пошук за ключовими словами або інгредієнтами, редагування публікацій та коментування рецептів. Кожна функція повинна бути реалізована з урахуванням принципів зручності та інтуїтивності. Наприклад, процес реєстрації має бути простим, а публікація рецептів – з можливістю додавання детальних описів. Нефункціональні вимоги включають високу продуктивність, стабільність роботи та забезпечення безпеки даних. Інтерфейс повинен бути інтуїтивно зрозумілим та зручним для користувачів. Безпека даних є критично важливою, оскільки застосунок працює з особистою інформацією користувачів, включаючи логіни, паролі та дані про створені рецепти. Для цього необхідно використовувати сучасні методи захисту від кібератак.

Таким чином, розробка інтерактивного застосунку для кондитерів передбачає створення платформи, яка поєднує функціональність соціальної мережі з інструментами для створення та зберігання рецептів. Основними функціями є реєстрація, авторизація, зміна паролю, публікація рецептів, пошук, редагування створених рецептів та коментування. Застосунок повинен забезпечувати високу продуктивність, стабільність роботи та захист даних користувачів, а інтерфейс – бути інтуїтивно зрозумілим та зручним.

2 МАТЕРІАЛИ І МЕТОДИ

2.1 Вибір мови програмування Java

Вибір мови програмування є ключовим етапом у розробці будь-якого програмного забезпечення, оскільки він визначає не лише ефективність реалізації проєкту, а й його масштабованість, підтримку та можливості для подальшого розвитку. Для створення інтерактивного кондитерського застосунку була обрана мова програмування Java, що обумовлено низкою переваг, які вона надає для розробки програмних продуктів, орієнтованих на користувача.

На сьогоднішній день Java залишається однією з основних мов для розробки корпоративних застосунків, веб-сервісів, мобільних застосунків (Android) та навіть вбудованих систем [50]. Її популярність підтверджується великою кількістю активних розробників, які використовують Java для створення сучасних програмних рішень. Це робить Java ідеальним вибором для проєкту, що вимагає стабільності, масштабованості та широкої підтримки спільноти (рисунок 2.1).



Рисунок 2.1 – Причини вибору мови програмування Java

Java є кросплатформенною мовою завдяки JVM, що дозволяє запускати код на різних ОС без змін [51]. Це важливо для кондитерського застосунку, який має бути доступним на різних пристроях.

Об'єктно-орієнтований підхід Java дозволяє структурувати код у класи та об'єкти, що спрощує розробку складних систем [52]. У кондитерському застосунку це допомогло чітко відокремити сутності, такі як користувачі, рецепти та інгредієнти.

Java надає вбудовані механізми безпеки, такі як Security Manager та шифрування, що захищає конфіденційні дані користувачів, зокрема паролі.

Стандартні бібліотеки Java включають тисячі класів для роботи з мережею, базами даних, XML, JSON та іншим. Наприклад, JDBC дозволив легко інтегрувати SQLite у застосунок для ефективного зберігання даних.

Підтримка багатопоточності в Java дозволяє обробляти кілька завдань одночасно [53]. У застосунку це використовується для обробки HTTP-запитів, що забезпечує ефективну роботу з кількома користувачами.

Java постійно розвивається, додаючи нові функції, такі як лямбда-вирази, Stream API (Java 8) та модульна система (Java 11). Це робить мову сучасною та відповідною до вимог сучасних застосунків.

Велика спільнота Java надає підтримку через форуми, блоги та документацію, що спрощує вирішення проблем під час розробки.

Java легко інтегрується з іншими технологіями, такими як JavaFX для створення GUI, що робить її універсальним інструментом для розробки.

Висока продуктивність Java забезпечується оптимізацією JVM та JIT-компіляцією, що важливо для застосунків з великим навантаженням, таких як кондитерський застосунок.

Вибір Java для розробки кондитерського застосунку є обґрунтованим через її кросплатформенність, безпеку, продуктивність та підтримку сучасних технологій. Це забезпечує стабільність, масштабованість та можливість подальшого вдосконалення застосунку.

2.2 Вибір середовища розробки

Розробка програмного забезпечення, особливо такого, що передбачає створення інтерактивного застосунку, вимагає ретельного підходу до вибору інструментів та середовищ розробки. Вибір середовища розробки є критично важливим етапом, оскільки він безпосередньо впливає на продуктивність команди, якість коду, швидкість розробки та можливість подальшого підтримання проєкту. У контексті створення інтерактивного кондитерського застосунку на мові програмування Java, було проведено аналіз доступних інструментів та середовищ, що дозволило обрати оптимальні рішення для реалізації проєкту.

2.2.1 Критерії вибору середовища розробки

Для вибору середовища розробки було визначено низку ключових критеріїв, які враховують специфіку проєкту (таблиця 2.1).

Таблиця 2.1 – Ключові критерії вибору середовища розробки

№	Критерії	Опис
1	Підтримка мови програмування Java	Оскільки основним інструментом розробки є Java, середовище має забезпечувати повну підтримку цієї мови, включаючи сучасні версії JDK, інтеграцію з бібліотеками та фреймворками
2	Інтегроване середовище розробки (IDE)	IDE повинно надавати зручний інтерфейс для написання, тестування та налагодження коду, а також інтеграцію з системами контролю версій
3	Підтримка графічного інтерфейсу користувача (GUI)	Оскільки застосунок має інтерактивний інтерфейс, важливо, щоб середовище розробки підтримувало створення GUI, зокрема за допомогою таких інструментів, як JavaFX
4	Продуктивність та стабільність	Середовище розробки має бути стабільним і не сповільнювати роботу навіть при роботі з великими проєктами

Продовження таблиці 2.1

№	Критерії	Опис
5	Інструменти для роботи з базами даних	Застосунок використовує SQLite для зберігання даних, тому середовище розробки має забезпечувати зручну роботу з базами даних, включаючи підтримку SQL-запитів та візуалізацію структури бази даних
6	Можливість інтеграції з бібліотеками та фреймворками	Для реалізації функціональності застосунку використовуються бібліотеки, такі як Jackson для роботи з JSON, тому IDE має підтримувати легку інтеграцію зі сторонніми бібліотеками
7	Підтримка тестування	Середовище розробки має забезпечувати зручні інструменти для написання та запуску тестів, що є важливим для забезпечення якості коду
8	Крос-платформенність	Оскільки застосунок розробляється для різних операційних систем, середовище розробки має бути доступним на різних платформах (Windows, macOS, Linux)
9	Документація та спільнота	Наявність детальної документації та активної спільноти є важливим фактором для швидкого вирішення проблем, що виникають під час розробки
10	Можливість розширення функціональності	Середовище розробки має підтримувати плагіни та розширення, що дозволяє адаптувати його під конкретні потреби проєкту

2.2.2 Аналіз доступних середовищ розробки

На ринку існує кілька популярних середовищ розробки для Java, які відповідають вищезазначеним критеріям. Серед них можна виділити IntelliJ IDEA, Eclipse та NetBeans. Кожне з цих середовищ має свої переваги та недоліки, які були враховані під час вибору.

IntelliJ IDEA є одним із найпопулярніших середовищ розробки для Java [54]. Воно надає потужний набір інструментів для написання, тестування та налагодження коду. Його переваги виділені в таблиці 2.2.

Eclipse є іншим популярним середовищем розробки для Java [55]. Він також надає широкий набір інструментів для розробки, але має деякі відмінні риси порівняно з IntelliJ IDEA (таблиця 2.3).

Таблиця 2.2 – Переваги середовища IntelliJ IDEA

№	Переваги	Опис
1	Підтримка сучасних версій Java	IntelliJ IDEA підтримує останні версії JDK, що дозволяє використовувати нові функції мови
2	Інтеграція з JavaFX	IDE надає зручні інструменти для розробки графічного інтерфейсу за допомогою JavaFX
3	Робота з базами даних	IntelliJ IDEA має вбудовану підтримку роботи з базами даних, включаючи SQLite, що дозволяє легко виконувати SQL-запити та візуалізувати структуру бази даних
4	Підтримка бібліотек та фреймворків	IDE підтримує легку інтеграцію з бібліотеками, такими як Jackson, через систему управління залежностями Maven або Gradle
5	Інструменти для тестування	IntelliJ IDEA надає зручні інструменти для написання та запуску тестів, що є важливим для забезпечення якості коду
6	Крос-платформенність	IDE доступне для Windows, macOS та Linux, що дозволяє розробляти застосунок на різних платформах
7	Документація та спільнота	IntelliJ IDEA має детальну документацію та активну спільноту, що спрощує вирішення проблем, що виникають під час розробки
8	Можливість розширення	IDE підтримує велику кількість плагінів, що дозволяє розширювати його функціональність

Таблиця 2.3 – Відмінності Eclipse від IntelliJ IDEA

№	Критерії відмінності	Опис
1	Підтримка Java	Eclipse також підтримує сучасні версії Java, але може вимагати додаткового налаштування для роботи з новими функціями мови
2	Інтеграція з JavaFX	Eclipse підтримує JavaFX, але інтеграція може бути менш зручною порівняно з IntelliJ IDEA
3	Робота з базами даних	Eclipse має плагіни для роботи з базами даних, але їх налаштування може бути більш складним
4	Підтримка бібліотек та фреймворків	Eclipse також підтримує інтеграцію з бібліотеками, але процес налаштування може бути більш трудомістким
5	Інструменти для тестування	Eclipse надає інструменти для тестування, але вони можуть бути менш інтуїтивно зрозумілими порівняно з IntelliJ IDEA

Продовження таблиці 2.3

№	Критерії відмінності	Опис
6	Крос-платформенність	Eclipse доступний для різних платформ, але може мати проблеми зі стабільністю на деяких системах
7	Документація та спільнота	Eclipse має активну спільноту, але документація може бути менш структурованою порівняно з IntelliJ IDEA
8	Можливість розширення	Eclipse підтримує плагіни, але їх інтеграція може вимагати додаткових зусиль

NetBeans є ще одним середовищем розробки для Java, яке також розглядалося під час вибору [56]. Воно має характеристики наведені в таблиці 2.4, котрі відрізняються від характеристик попередніх представлених середовищ для розробки. Дане середовище має свої переваги та незначні недоліки.

Таблиця 2.4 – Характеристики NetBeans

№	Характеристики	Опис
1	Підтримка Java	NetBeans підтримує сучасні версії Java, але може відставати від IntelliJ IDEA у підтримці нових функцій
2	Інтеграція з JavaFX	NetBeans має вбудовану підтримку JavaFX, але інтерфейс може бути менш зручним
3	Робота з базами даних	NetBeans надає інструменти для роботи з базами даних, але вони можуть бути менш потужними порівняно з IntelliJ IDEA
4	Підтримка бібліотек та фреймворків	NetBeans підтримує інтеграцію з бібліотеками, але процес може бути менш автоматизованим
5	Інструменти для тестування	NetBeans надає базові інструменти для тестування, але вони можуть бути менш функціональними
6	Крос-платформенність	NetBeans доступний для різних платформ, але може мати проблеми з продуктивністю
7	Документація та спільнота	NetBeans має документацію, але спільнота може бути менш активною порівняно з IntelliJ IDEA
8	Можливість розширення	NetBeans підтримує плагіни, але їх кількість та якість може бути нижчою

2.2.3 Обґрунтування вибору IntelliJ IDEA

Після аналізу доступних середовищ розробки було прийнято рішення використовувати IntelliJ IDEA як основне середовище розробки для проєкту. Цей вибір був обґрунтований причинами описаними в таблиці 2.5.

Таблиця 2.5 – Причини вибору IntelliJ IDEA

№	Причини	Опис
1	Повноцінна підтримка Java	IntelliJ IDEA забезпечує повну підтримку сучасних версій Java, що дозволяє використовувати нові функції мови, такі як лямбда-вирази, модульна система та інші
2	Зручний інтерфейс для розробки GUI	IntelliJ IDEA надає інтуїтивно зрозумілі інструменти для створення графічного інтерфейсу за допомогою JavaFX, що є критично важливим для інтерактивного застосунку
3	Інтеграція з базами даних	IDE забезпечує зручну роботу з SQLite, включаючи виконання SQL-запитів, візуалізацію структури бази даних та автоматичне завершення коду
4	Легка інтеграція з бібліотеками	IntelliJ IDEA підтримує інтеграцію з бібліотеками через Maven або Gradle, що дозволяє легко додавати та керувати залежностями проєкту
5	Потужні інструменти для тестування	IDE надає зручні інструменти для написання та запуску тестів, що є важливим для забезпечення якості коду
6	Крос-платформенність	IntelliJ IDEA доступний для Windows, macOS та Linux, що дозволяє розробляти застосунок на різних платформах без додаткових зусиль
7	Документація та спільнота	Наявність детальної документації та активної спільноти дозволяє швидко вирішувати проблеми, що виникають під час розробки
8	Можливість розширення функціональності	IntelliJ IDEA підтримує велику кількість плагінів, що дозволяє адаптувати середовище розробки під конкретні потреби проєкту
9	Стабільність та продуктивність	IDE забезпечує високу продуктивність навіть при роботі з великими проєктами, що є важливим для розробки інтерактивного застосунку

2.2.4 Використання додаткових інструментів

Окрім IntelliJ IDEA, для розробки проєкту було використано низку додаткових інструментів, які доповнюють функціональність основного середовища розробки (таблиця 2.6).

Таблиця 2.6 – Додаткові інструменти для розробки проєкту

№	Причини	Опис
1	Maven	Для управління залежностями проєкту було використано Maven. Цей інструмент дозволив легко додавати та оновлювати бібліотеки, такі як Jackson для роботи з JSON, що значно спростило процес розробки [57]
2	SQLite	Для зберігання даних було обрано SQLite як легку та ефективну базу даних. IntelliJ IDEA забезпечує зручну роботу з SQLite, що дозволило легко створювати та керувати структурою бази даних [58]
3	JavaFX	Для створення графічного інтерфейсу користувача було використано JavaFX. Цей фреймворк дозволив створити сучасний та інтуїтивно зрозумілий інтерфейс для застосунку [59]
5	JUnit	Для тестування коду було використано фреймворк JUnit. IntelliJ IDEA надає зручні інструменти для написання та запуску тестів, що дозволило забезпечити високу якість коду [60]
6	JMeter	Для навантажувального тестування було використано інструмент JMeter. Це програмне забезпечення розроблене для навантажувального тестування функціональної поведінки та виміру продуктивності. За його допомогою було перевірено роботу сервера при 1000 запитів [61]
7	Postman	Для тестування логіки додавання коментарів було використано інструмент Postman. Це програмне забезпечення призначене для тестування, розробки та документування API [62]

Вибір середовища розробки є критично важливим етапом у процесі створення програмного забезпечення. Для розробки інтерактивного застосунку для кондитерів на Java було обрано IntelliJ IDEA як основне середовище розробки. Цей вибір був обґрунтований потужними інструментами для розробки, зручною інтеграцією з JavaFX, підтримкою роботи з базами даних та

можливістю розширення функціональності за допомогою плагінів. Використання додаткових інструментів, таких як Maven, SQLite, JUnit, JMeter та Postman дозволило забезпечити ефективну розробку, тестування та підтримку проєкту.

2.3 Вибір технологій для реалізації інтерфейсу

Реалізація інтерфейсу інтерактивного застосунку для кондитерів потребувала використання сучасних технологій, які забезпечують зручність користувача, швидкість роботи та ефективну взаємодію з серверною частиною. Для досягнення цих цілей було обрано низку інструментів та підходів, які дозволяють створити інтуїтивно зрозумілий та функціональний інтерфейс. Нижче розглядаються основні технології, їх переваги та обґрунтування вибору.

2.3.1 JavaFX як основа графічного інтерфейсу

Для створення графічного інтерфейсу користувача (GUI) було обрано фреймворк JavaFX. Цей інструмент є сучасним наступником Swing та AWT, які раніше використовувалися для розробки графічних інтерфейсів на Java. JavaFX пропонує ряд переваг, які роблять його ідеальним вибором для розробки інтерактивних застосунків (таблиця 2.7) [63].

Таблиця 2.7 – Переваги JavaFX

№	Переваги	Опис
1	Підтримка FXML	JavaFX дозволяє використовувати FXML для опису структури інтерфейсу. Це дозволяє розділяти логіку програми та її візуальне представлення, що спрощує розробку та тестування
2	Крос-платформеність	JavaFX працює на різних платформах, включаючи Windows, macOS та Linux. Це забезпечує однакову поведінку застосунку на різних операційних системах

Продовження таблиці 2.7

№	Переваги	Опис
3	Модульність та гнучкість	JavaFX дозволяє розділяти інтерфейс на окремі компоненти, що спрощує розробку та підтримку коду. Наприклад, екрани входу, реєстрації, оновлення поточного паролю, списку рецептів та деталей рецепту реалізовані як окремі панелі, які можна легко змінювати та комбінувати
4	Підтримка CSS	JavaFX підтримує використання каскадних таблиць стилів (CSS) для оформлення інтерфейсу. Це дозволяє розробникам створювати стильні та візуально привабливі інтерфейси. У застосунку використано власну колірну схему, яка включає кольори, натхненні кондитерською тематикою (наприклад, шоколадний, карамельний, глазурний тощо). Це додає застосунку естетичної привабливості та створює відповідну атмосферу
5	Анімації та ефекти	JavaFX надає можливість додавання анімацій та ефектів, що робить інтерфейс більш динамічним. Хоча в даному застосунку анімації не є основним акцентом, їх можна легко інтегрувати для покращення користувацького досвіду

2.3.2 Використання ObservableList

Для роботи зі списками рецептів у застосунку використано ObservableList з бібліотеки JavaFX. Цей механізм дозволяє автоматично оновлювати інтерфейс при зміні даних. Наприклад, коли користувач додає новий рецепт або здійснює пошук, список рецептів оновлюється без необхідності перезавантаження сторінки. Це забезпечує плавну та зручну взаємодію з застосунком. ObservableList інтегрований з компонентами JavaFX, такими як ListView, що дозволяє відображати дані у вигляді списку [64]. Кожен елемент списку може бути налаштований за допомогою CellFactory, що дозволяє створювати власні віджети для відображення рецептів. У застосунку використано простий список з назвами рецептів, але можливе розширення для відображення додаткової інформації, такої як зображення тощо.

2.3.3 Використання CSS для стилізації інтерфейсу

Стилізація інтерфейсу є важливим фактором створення привабливого застосунку. У JavaFX CSS використовується для налаштування зовнішнього вигляду компонентів. У застосунку використано власну колірну схему, яка включає:

- колір какао (#301B28): використовується для заголовків та акцентних елементів;
- колір шоколаду (#523634): використовується для фону панелей та кнопок;
- колір карамелі (#B6452C): використовується для кнопок та інтерактивних елементів;
- колір глазури (#DDC5A2): використовується для фону основного інтерфейсу.

Ця колірна схема створює гармонійний та приємний для ока інтерфейс, який асоціюється з кондитерською тематикою. Крім того, CSS дозволяє легко змінювати стилі, що спрощує адаптацію застосунку до різних потреб.

2.3.4 Використання компонентів JavaFX

Для створення інтерактивних елементів інтерфейсу використано стандартні компоненти JavaFX, такі як Button, TextField, TextArea, ListView, ScrollPane тощо. Ці компоненти є базовими елементами будь-якого графічного інтерфейсу та забезпечують зручну взаємодію з користувачем.

Більш детальну інформацію щодо описаних раніше компонентів подано в таблиці 2.8.

Таблиця 2.8 – Компоненти інтерфейсу

№	Компоненти	Опис
1	Кнопки (Button)	Використовуються для виконання дій, таких як вхід, пошук тощо. Кожна кнопка має власний стиль, який відповідає загальній колірній схемі застосунку
2	Текстові поля (TextField, TextArea)	Використовуються для введення тексту, наприклад, назви рецепту, інгредієнтів, інструкцій тощо. Ці компоненти підтримують багаторядковий текст та автоматичне перенесення слів
3	Списки (ListView)	Використовуються для відображення списку рецептів. Кожен елемент списку може бути налаштований за допомогою CellFactory
4	Прокручувані панелі (ScrollPane)	Використовуються для відображення великих обсягів тексту, таких як інгредієнти та інструкції до рецептів

2.3.5 Використання JSON для обміну даними

Для обміну даними між клієнтською та серверною частинами застосунку було обрано формат JSON (JavaScript Object Notation). Цей формат є легким для читання як людьми, так і машинами, що робить його ідеальним вибором для передачі структурованих даних. У Java для роботи з JSON використана бібліотека Jackson, яка забезпечує ефективну серіалізацію та десеріалізацію об'єктів [65].

Серіалізація передбачає перетворення об'єктів Java у формат JSON для подальшої відправки на сервер.

Десеріалізація, навпаки, полягає у перетворенні JSON-даних у об'єкти Java для подальшої обробки на стороні клієнта.

Використання JSON забезпечує гнучкість у передачі даних та спрощує інтеграцію з іншими системами, що є важливим для сучасних застосунків, які потребують швидкого та ефективного обміну інформацією.

2.3.6 Використання HTTP-запитів для взаємодії з сервером

Для взаємодії з серверною частиною застосунку використано стандартні HTTP-запити (GET, POST, PUT, DELETE) (таблиця 2.9). Ці запити дозволяють виконувати основні операції з даними, такі як отримання списку рецептів, створення нового рецепту, оновлення та видалення існуючих рецептів [66].

Таблиця 2.9 – Використані HTTP-запити

№	Компоненти	Опис
1	GET-запити	Використовуються для отримання даних, наприклад, списку рецептів або деталей конкретного рецепту
2	POST-запити	Використовуються для надсилання даних на сервер, наприклад, при створенні нового рецепту
3	PUT-запити	Використовуються для оновлення існуючих даних, наприклад, при редагуванні рецепту
4	DELETE-запити	Використовуються для видалення даних, наприклад, при видаленні рецепту

Для роботи з HTTP-запитами використано клас `HttpURLConnection`, який є стандартним інструментом Java для роботи з мережевими з'єднаннями.

2.3.7 Використання ObjectMapper для роботи з JSON

Бібліотека Jackson, зокрема клас `ObjectMapper`, використовується для серіалізації та десеріалізації JSON. Цей інструмент дозволяє легко перетворювати об'єкти Java у JSON та навпаки. Наприклад, при створенні нового рецепту об'єкт `RecipeDTO` перетворюється у JSON-рядок, який потім відправляється на сервер.

`ObjectMapper` також підтримує роботу зі складними структурами даних, такими як списки та мапи, що робить його ідеальним вибором для роботи з JSON у застосунку [67].

2.3.8 Використання потоків для асинхронної роботи

Для забезпечення плавної роботи інтерфейсу використано асинхронну обробку HTTP-запитів. Це дозволяє виконувати мережеві операції у фоновому режимі, не блокуючи головний потік інтерфейсу. Наприклад, при завантаженні списку рецептів інтерфейс залишається відповідальним на дії користувача, навіть якщо запит до сервера займає деякий час.

Для реалізації асинхронної роботи використано механізм потоків Java. Кожен HTTP-запит виконується в окремому потоці, що дозволяє уникнути блокування інтерфейсу.

2.3.9 Використання Alert для відображення повідомлень

Для відображення повідомлень користувачу використано клас Alert з JavaFX. Цей компонент дозволяє показувати діалогові вікна з повідомленнями про помилки, підтвердження дій тощо. Наприклад, при невдалому вході або реєстрації користувач отримує відповідне повідомлення.

Alert підтримує різні типи повідомлень, такі як інформаційні, попередження, помилки тощо [68]. Це дозволяє забезпечити зручну взаємодію з користувачем.

2.3.10 Використання Scene та Stage для керування екранами

У JavaFX кожен екран застосунку представлений як Scene, яка містить усі елементи інтерфейсу. Керування екранами здійснюється за допомогою класу Stage, який представляє головне вікно застосунку. У застосунку використано кілька сцен для різних екранів, таких як екран входу, реєстрації, списку рецептів тощо.

Перехід між екранами здійснюється шляхом зміни сцени у головному вікні. Це дозволяє створювати багатоекранні застосунки зі зручною навігацією.

Вибір технологій для реалізації інтерфейсу інтерактивного кондитерського застосунку базувався на сучасних підходах до розробки графічних інтерфейсів. Використання JavaFX, ObservableList, CSS, JSON та інших інструментів дозволило створити зручний, швидкий та естетично привабливий інтерфейс, який задовольняє потреби користувачів. Кожен з обраних інструментів має свої переваги та забезпечує ефективну роботу застосунку на різних етапах взаємодії з користувачем.

2.4 Структурна схема розробки застосунку

Структурна схема розробки застосунку відображає основні етапи та взаємозв'язки між компонентами системи (рисунок 2.2). Для візуалізації структури використовується мова опису PlantUML [69], яка дозволяє детально відобразити логіку роботи застосунку, його модулі та їх взаємодію. Нижче наведено структурну схему, яка охоплює всі ключові етапи розробки, починаючи з ініціалізації бази даних, закінчуючи взаємодією клієнта з сервером.

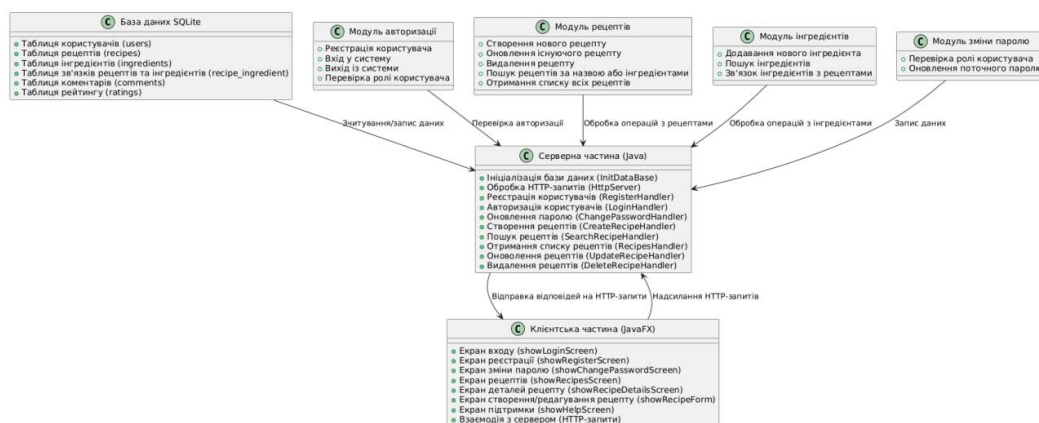


Рисунок 2.2 – Структурна схема розробки застосунку

Опис структурної схеми поданий в таблиці 2.10.

Таблиця 2.10 – Опис структурної схеми

№	Компоненти	Опис
1	База даних (SQLite)	База даних є основним сховищем інформації в застосунку. Вона містить шість основних таблиць:
1.1	users	зберігає дані користувачів (логін, пароль, роль)
1.2	recipes	містить інформацію про рецепти (назва, інгредієнти, інструкції, ідентифікатор автора)
1.3	ingredients	зберігає унікальні інгредієнти, які використовуються в рецептах
1.4	recipe_ingredients	зв'язує рецепти з інгредієнтами через зовнішні ключі
1.5	comments	зберігає коментарі авторизованих користувачів
1.6	ratings	зберігає рейтинг рецептів
2	Серверна частина (Java)	Серверна частина реалізована на Java з використанням вбудованого HTTP-сервера. Вона включає такі модулі:
2.1	Ініціалізація бази даних	створення таблиць та наповнення тестовими даними
2.2	Обробка HTTP-запитів	маршрутизація запитів до відповідних обробників
2.3	Реєстрація та авторизація	обробка запитів на реєстрацію, вхід у систему та оновлення паролю
2.4	Операції з рецептами	створення, оновлення, видалення та пошук рецептів
3	Клієнтська частина (JavaFX)	Клієнтська частина реалізована за допомогою JavaFX. Вона включає наступні екрани:
3.1	Екран входу	авторизація користувача
3.2	Екран реєстрації	створення нового облікового запису
3.3	Екран зміни паролю	зміна поточного паролю на новий
3.4	Екран рецептів	відображення списку рецептів з можливістю пошуку
3.5	Екран деталей рецепту	перегляд повної інформації про рецепт
3.6	Екран створення або редагування рецепту	додавання або зміна рецепту
3.7	Екран підтримки	відображення способів зв'язку з підтримкою (електронна пошта)
4	Модуль інгредієнтів	Модуль інгредієнтів відповідає за управління інгредієнтами: додавання нового інгредієнта; пошук інгредієнтів; зв'язок інгредієнтів з рецептами через таблицю recipe_ingredients

Продовження таблиці 2.10

№	Компоненти	Опис
5	Модуль авторизації	Цей модуль відповідає за управління користувачами: реєстрація нового користувача; вхід у систему з перевіркою логіну та пароллю; зміна поточного паролю на новий; вихід із системи; перевірка ролі користувача (гість, зареєстрований користувач)
6	Модуль рецептів	Модуль рецептів забезпечує всі операції, пов'язані з рецептами: створення нового рецепту; оновлення існуючого рецепту; видалення рецепту; пошук рецептів за назвою або інгредієнтами; отримання списку всіх рецептів

Взаємодія компонентів виглядає наступним чином:

- база даних ↔ сервер: серверна частина взаємодіє з базою даних через JDBC, виконуючи запити на зчитування та запис даних;
- сервер ↔ клієнт: клієнтська частина надсилає HTTP-запити до сервера, який обробляє їх та повертає відповіді у форматі JSON;
- модулі авторизації, рецептів та інгредієнтів: ці модулі інтегровані в серверну частину та використовуються для обробки відповідних операцій.

Деталізація етапів розробки представлена в таблиці 2.11.

Таблиця 2.11 – Деталізація етапів розробки

№	Етапи	Опис
1	Ініціалізація бази даних	На цьому етапі створюються таблиці в базі даних, якщо вони ще не існують. Виконується наповнення тестовими даними, включаючи користувачів та рецепти
2	Реалізація серверної частини	Серверна частина включає створення HTTP-сервера, реєстрацію обробників для різних маршрутів (наприклад, /api/auth/register, /api/recipes/create) та обробку запитів
3	Розробка клієнтської частини	Клієнтська частина реалізована за допомогою JavaFX. Включає створення інтерфейсів для входу, реєстрації, перегляду рецептів та їх редагування

Продовження таблиці 2.11

№	Етапи	Опис
4	Інтеграція модулів	Модулі авторизації, рецептів та інгредієнтів інтегровані в серверну частину. Вони забезпечують логіку обробки запитів та взаємодію з базою даних
5	Тестування та налагодження	На завершальному етапі проводиться тестування всіх компонентів системи, включаючи перевірку коректності роботи бази даних, серверної та клієнтської частин

Ця структурна схема відображає основні етапи розробки застосунку та його ключові компоненти, що забезпечує зрозумілість та ефективність процесу розробки.

3 ОСНОВНІ РІШЕННЯ ЩОДО РЕАЛІЗАЦІЇ КОМПОНЕНТІВ СИСТЕМИ

3.1 Проектування моделей даних для рецепта

Проектування моделей даних є ключовим етапом розробки будь-якої інформаційної системи, оскільки саме вони визначають структуру зберігання, обробки та взаємодії даних. У контексті інтерактивного застосунку для кондитерів, модель даних для рецепта має враховувати специфіку кулінарних рецептів, їхню структуру та взаємозв'язки з іншими сутностями системи, такими як користувачі та інгредієнти.

Рецепт у кондитерському застосунку є центральною сутністю, яка містить інформацію про страви, їхні інгредієнти, інструкції з приготування та інші метадані. Для ефективного зберігання та обробки рецептів необхідно визначити основні атрибути цієї сутності, що представлені в таблиці 3.1.

Таблиця 3.1 – Основні атрибути сутності

№	Атрибути	Опис
1	Ідентифікатор (ID)	Унікальний числовий ідентифікатор рецепту, який дозволяє однозначно ідентифікувати кожен рецепт у системі. Цей атрибут є первинним ключем у відповідній таблиці бази даних
2	Назва (Title)	Назва рецепту, яка описує страву. Наприклад, "Шоколадний торт" або "Ванільні кекси". Цей атрибут є текстовим полем і повинен бути унікальним для уникнення дублювання
3	Інгредієнти (Ingredients)	Список інгредієнтів, необхідних для приготування страви. Цей атрибут може зберігатися у вигляді текстового поля або бути нормалізованим у окрему таблицю для підтримки зв'язків між рецептами та інгредієнтами
4	Інструкції (Instructions)	Детальний опис кроків для приготування страви. Цей атрибут є текстовим полем, яке може містити багаторядковий текст
5	Ідентифікатор користувача (User ID)	Зовнішній ключ, який пов'язує рецепт із користувачем, який його створив. Це дозволяє відстежувати авторство рецептів та забезпечує персоналізацію застосунку

Для забезпечення ефективного зберігання та мінімізації надмірності даних необхідно застосувати принципи нормалізації. У контексті рецептів це передбачає розділення даних на декілька таблиць (таблиця 3.2, рисунок 3.1).

Таблиця 3.2 – Таблиці даних у контексті рецептів

№	Таблиці	Опис
1	Таблиця рецептів (Recipes)	Містить основну інформацію про рецепти, таку як ідентифікатор, назву, інструкції та ідентифікатор користувача
2	Таблиця інгредієнтів (Ingredients)	Містить унікальні інгредієнти, які використовуються у рецептах. Кожен інгредієнт має свій ідентифікатор та назву
3	Таблиця зв'язків рецептів та інгредієнтів (Recipe_Ingredients)	Ця таблиця реалізує зв'язок "багато-до-багатьох" між рецептами та інгредієнтами. Вона містить ідентифікатори рецептів та інгредієнтів, що дозволяє одному рецепту мати багато інгредієнтів, а одному інгредієнту бути частиною багатьох рецептів

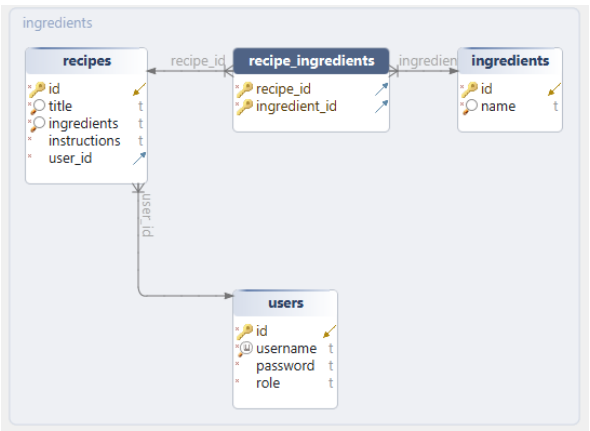


Рисунок 3.1 – Структура БД

Такий підхід дозволяє уникнути дублювання даних та забезпечує гнучкість у роботі з інгредієнтами. Наприклад, якщо інгредієнт "цукор" використовується у багатьох рецептах, він зберігається лише один раз у таблиці інгредієнтів, а зв'язки з рецептами реалізуються через таблицю Recipe_Ingredients. Модель даних для рецептів передбачає наявність зв'язків між різними сутностями (таблиця 3.3).

Таблиця 3.3 – Зв'язки між різними сутностями

№	Зв'язки	Опис
1	Зв'язок "один-до-багатьох" між користувачами та рецептами	Кожен користувач може створити багато рецептів, але кожен рецепт належить лише одному користувачу. Цей зв'язок реалізується через зовнішній ключ <code>user_id</code> у таблиці рецептів
2	Зв'язок "багато-до-багатьох" між рецептами та інгредієнтами	Рецепт може містити багато інгредієнтів, а інгредієнт може використовуватися у багатьох рецептах. Цей зв'язок реалізується через проміжну таблицю <code>Recipe_Ingredients</code>

Ці зв'язки забезпечують цілісність даних та дозволяють ефективно виконувати запити, такі як пошук рецептів за інгредієнтами або відображення списку рецептів, створених конкретним користувачем.

Однією з ключових функцій кондитерського застосунку є можливість пошуку рецептів за різними критеріями, такими як назва або інгредієнти. Для забезпечення швидкого пошуку необхідно створити індекси на відповідних полях:

- індекс на полі `title` у таблиці `Recipes` – це дозволяє швидко знаходити рецепти за назвою;
- індекс на полі `name` у таблиці `Ingredients` – це прискорює пошук інгредієнтів за назвою;
- індекси на полях `recipe_id` та `ingredient_id` у таблиці `Recipe_Ingredients` – це оптимізує запити, пов'язані з пошуком рецептів за інгредієнтами та навпаки.

Інгредієнти та інструкції зберігаються у вигляді текстових даних, що вимагає особливої уваги до їхньої обробки. Для зручності користувачів інгредієнти можуть зберігатися у форматі, де кожен інгредієнт представлений окремим рядком. Наприклад:

- 200 г чорного шоколаду;
- 250 г масла;
- 200 г цукру;
- 4 яйця;

- 150 г борошна тощо.

Цей формат дозволяє легко читати та редагувати інгредієнти, а також автоматично обробляти їх під час пошуку або фільтрації.

У сучасних застосунках важливо враховувати можливість підтримки багатомовності, тому у майбутніх версіях застосунку варто приділити цьому особливу увагу. Для рецептів це може включати:

- назви рецептів та інгредієнтів у різних мовах – це дозволяє користувачам з різних країн знаходити рецепти на їхній рідній мові;
- переклад інструкцій – інструкції з приготування можуть бути доступні у кількох мовах, що робить застосунок більш зручним для міжнародної аудиторії.

Для реалізації цієї функціональності можна використовувати окремі таблиці для перекладів, де кожен рецепт та інгредієнт мають відповідні записи для кожної мови.

Крім основних даних, рецепти можуть містити додаткові метадані, які подані в таблиці 3.4.

Таблиця 3.4 – Метадані, які можуть містити рецепти

№	Зв'язки	Опис
1	Час створення та оновлення	У майбутніх версіях це дозволить відстежувати, коли рецепт був створений або змінений
2	Рейтинг та відгуки	Користувачі можуть оцінювати рецепти та залишати коментарі, що додає соціальну складову до застосунку
3	Категорії та теги	У майбутніх версіях рецепти можуть бути класифіковані за категоріями (наприклад, "торти") або тегами (наприклад, "шоколад")

Ці метадані дозволяють покращити пошук та фільтрацію рецептів, а також забезпечують додаткову інформацію для користувачів.

Для підтримки цілісності даних у базі даних необхідно використовувати обмеження та правила, що представлені в таблиці 3.5.

Таблиця 3.5 – Обмеження та правила

№	Назва	Опис
1	Обмеження унікальності	Назви рецептів та інгредієнтів повинні бути унікальними, щоб уникнути дублювання
2	Каскадне видалення	При видаленні рецепту повинні автоматично видалятися всі пов'язані записи у таблиці Recipe_Ingredients
3	Обмеження на значення	Наприклад, поле user_id у таблиці рецептів не може бути порожнім, оскільки кожен рецепт повинен мати автора

Проектування моделі даних для рецептів у застосунку для кондитерів є комплексним завданням, яке вимагає врахування багатьох факторів, таких як структура даних, зв'язки між сутностями, оптимізація для пошуку та підтримка масштабованості. Використання нормалізованої структури бази даних, індексів та обмежень дозволяє забезпечити ефективне зберігання та обробку даних, що є критично важливим для успішної роботи застосунку.

3.1.1 Реалізація реєстрації користувача та авторизації

Реєстрація та авторизація користувачів є важливими елементами будь-якої інтерактивної системи, яка працює з обліковими записами. У контексті застосунку для кондитерів ці механізми забезпечують персоналізацію досвіду користувачів, дозволяючи їм створювати, редагувати та зберігати рецепти, а також взаємодіяти з іншими користувачами. Реалізація цих функцій базується на сучасних підходах до розробки програмного забезпечення, включаючи клієнт-серверну архітектуру, REST API та використання баз даних.

Система реєстрації та авторизації побудована на клієнт-серверній моделі. Клієнтська частина, розроблена за допомогою JavaFX, забезпечує інтерфейс для взаємодії з користувачем. Серверна частина, написана на Java, обробляє запити, зберігає дані в базі даних SQLite та генерує відповіді. Клієнт і сервер взаємодіють через HTTP-запити, використовуючи JSON для передачі даних.

Процес реєстрації починається з введення обов'язкових даних, таких як ім'я користувача та пароль. Клієнтська частина перевіряє коректність введених даних, включаючи мінімальну довжину пароля та унікальність імені користувача. Після цього дані передаються на сервер через POST-запит.

Серверна частина отримує дані, перевіряє їх на валідність та унікальність імені користувача в базі даних. Якщо перевірки успішні, дані користувача зберігаються в таблиці users, де кожен запис містить унікальний ідентифікатор, ім'я користувача та роль, наприклад "REGISTERED". Після успішної реєстрації сервер повертає клієнту токен авторизації, який використовується для подальшої ідентифікації користувача.

Основні кроки реєстрації представлені в таблиці 3.6.

Таблиця 3.6 – Основні кроки реєстрації

№	Кроки	Опис
1	Введення даних	Користувач вводить ім'я та пароль у відповідні поля форми
2	Перевірка на клієнті	Клієнтська частина перевіряє, чи відповідають дані вимогам (наприклад, пароль має бути не менше 6 символів)
3	Відправка запиту на сервер	Дані передаються на сервер через HTTP POST-запит
4	Перевірка на сервері	Сервер перевіряє унікальність імені користувача та зберігає дані в базі даних
5	Генерація токена	Після успішної реєстрації сервер генерує токен, який повертається клієнту
6	Збереження токена	Клієнт зберігає токен для подальшої авторизації

Авторизація користувача відбувається після успішної реєстрації або при повторному вході в систему. Користувач вводить свої облікові дані (ім'я та пароль), які передаються на сервер через POST-запит. Сервер перевіряє введені дані, порівнюючи їх із записами в базі даних. Якщо дані вірні, сервер генерує токен авторизації та повертає його клієнту.

Токен авторизації використовується для ідентифікації користувача при кожному наступному запиті до сервера. Це дозволяє забезпечити безпеку та захист даних, оскільки кожен запит перевіряється на наявність дійсного токена.

Основні кроки авторизації описані в таблиці 3.7.

Таблиця 3.7 – Основні кроки авторизації

№	Кроки	Опис
1	Введення даних	Користувач вводить ім'я та пароль у форму входу
2	Відправка запиту на сервер	Дані передаються на сервер через HTTP POST-запит
3	Перевірка на сервері	Сервер порівнює введені дані з даними в базі даних
4	Генерація токена	Якщо дані вірні, сервер генерує токен
5	Повернення токена	Токен повертається клієнту та зберігається для подальшого використання

Токени авторизації є основним механізмом забезпечення безпеки в системі. Вони генеруються сервером після успішної реєстрації або авторизації та містять інформацію про користувача, таку як ім'я та роль. Кожен запит до сервера, який вимагає авторизації, повинен включати токен у заголовку Authorization. Сервер перевіряє токен на валідність перед обробкою запиту.

Система реєстрації та авторизації включає механізми обробки помилок, які забезпечують зручність для користувача. Наприклад, якщо користувач вводить неправильний пароль або ім'я, сервер повертає відповідне повідомлення про помилку, що дозволяє швидко виправити помилку та повторити спробу.

Система підтримує різні ролі користувачів, такі як гість та зареєстрований користувач. Гість має обмежений доступ до функцій застосунку, тоді як зареєстрований користувач може створювати, редагувати та видаляти рецепти. Роль користувача зберігається в базі даних і використовується для контролю доступу до функцій системи.

Для збереження даних користувачів використовується реляційна база даних SQLite. Таблиця users містить унікальний ідентифікатор користувача, ім'я

користувача, пароль та роль. Кожен запит до бази даних виконується через підготовлені SQL-запити, що забезпечує захист від SQL-ін'єкцій.

Реалізація реєстрації та авторизації є важливим етапом розробки, який забезпечує безпеку, зручність та персоналізацію досвіду користувачів. Використання сучасних підходів, таких як токени авторизації та REST API, дозволяє створити надійну та ефективну систему управління обліковими записами. Ці механізми є основою для подальшого розширення функціональності застосунку, включаючи додавання нових ролей, інтеграцію з соціальними мережами та впровадження додаткових заходів безпеки.

3.2 Реалізація механізму публікації рецептів

Сучасні кулінарні застосунки набирають популярності, дозволяючи користувачам знаходити нові рецепти та ділитися власними досягненнями. Ключові функції, наприклад, як публікація рецептів, забезпечують інтерактивність та персоналізацію, роблячи застосунок зручним і корисним. Реалізація основних механізмів у застосунку для кондитерів на Java відображена в UML-діаграмах (Додаток Г).

Система побудована на клієнт-серверній архітектурі: сервер обробляє запити, зберігає дані та реалізує логіку застосунку, а клієнтська частина забезпечує інтерфейс для користувача. Для зберігання даних використовується SQLite, що дозволяє ефективно керувати інформацією про користувачів, рецепти та інгредієнти.

Механізм публікації рецептів є однією з ключових функцій застосунку. Він дозволяє користувачам додавати нові рецепти, які стають доступними для інших користувачів. Процес публікації включає кілька етапів, що представлені в таблиці 3.8.

Таблиця 3.8 – Етапи публікації рецептів

№	Етапи	Опис
1	Введення даних користувачем	Користувач заповнює форму, яка включає назву рецепту, список інгредієнтів та інструкції з приготування. Назва рецепту повинна бути унікальною, щоб уникнути дублювання. Інгредієнти вводяться у вигляді текстового списку, де кожен інгредієнт розділений новим рядком. Інструкції також вводяться у текстовому форматі
2	Перевірка даних	Перед збереженням рецепту система перевіряє введені дані на коректність. Наприклад, перевіряється заповнення всіх обов'язкових полів, назва на унікальність
3	Збереження рецепту	Після успішної перевірки дані зберігаються в базі даних. Для цього використовується SQL-запит, який вставляє новий запис у таблицю recipes. Кожен рецепт пов'язується з користувачем, який його опублікував, через поле user_id. Це дозволяє відстежувати авторство рецептів
4	Обробка інгредієнтів	Окремо обробляються інгредієнти, які введені користувачем. Кожен інгредієнт розбивається на окремі компоненти (наприклад, назва інгредієнта та його кількість). Назви інгредієнтів зберігаються в окремій таблиці ingredients, якщо вони ще не існують. Потім створюються зв'язки між рецептом та інгредієнтами в таблиці recipe_ingredients. Це дозволяє ефективно здійснювати пошук рецептів за інгредієнтами
5	Повернення результату	Після успішного збереження рецепту система повертає користувача на сторінку рецептів, де користувач може переглянути свій рецепт у списку опублікованих

Для реалізації публікації рецептів використовуються SQL-запити. Система забезпечує коректну обробку помилок, наприклад, повідомляє, якщо користувач намагається додати вже існуючий рецепт. Для валідації даних застосовуються регулярні вирази та обмеження на рівні бази даних. Наприклад, поле title у таблиці recipes має унікальність, що запобігає дублюванню рецептів. Інтерфейс для публікації рецептів реалізований у вигляді зручних форм та кнопок. Користувач може додавати нові рецепти, переглядати їх та видаляти власні за потреби. Ці механізми є ключовими для інтерактивності та персоналізації застосунку, роблячи його зручним і корисним для користувачів.

Використання SQLite та Java дозволяє ефективно реалізувати функції та забезпечити стабільну роботу.

3.3 Розробка функцій пошуку та коментування рецептів

Функціональність пошуку та коментування рецептів є ключовими елементами інтерактивного застосунку для кондитерів, які забезпечують зручність користувачів та підвищують їх задоволення від використання системи. Ці функції дозволяють швидко знаходити потрібні рецепти та ділитися своїми думками щодо вже існуючих рецептів.

3.3.1 Функція пошуку рецептів

Пошук рецептів є ключовою функцією застосунку, що дозволяє користувачам швидко знаходити рецепти за ключовими словами, інгредієнтами або назвами. Система аналізує запит, порівнюючи його з індексованими даними, та враховує синоніми, морфологічні форми слів і частоту їх використання для підвищення точності результатів. Для оптимізації пошуку реалізовано пошук за інгредієнтами: користувач може ввести інгредієнт, який його цікавить, і система знайде рецепти, що містять його. Це дозволяє знаходити рецепти з тим продуктом, котрий користувач хоче обов'язково бачити в своєму десерті.

3.3.2 Функція коментування рецептів

Коментування рецептів сприяє соціальній взаємодії, дозволяючи користувачам залишати відгуки, поради чи запитання. Це створює спільноту, де люди діляться досвідом і вдосконалюють кулінарні навички. Кожен коментар може включати текст і оцінку, створюючи діалоги для уточнення деталей чи

обміну ідеями. Користувачі можуть переглядати коментарі інших, знаходячи найкорисніші для себе.

3.3.3 Інтеграція функцій пошуку та коментування

Функції пошуку та коментування об'єднані в єдиний інтерфейс для зручності користувачів. Це дозволяє легко переходити від пошуку до перегляду коментарів без зайвих кроків. Наприклад, після пошуку рецептів можна прочитати відгуки. Таке поєднання робить застосунок зручним і ефективним інструментом для роботи з рецептами.

Функції пошуку та коментування забезпечують швидкий доступ до рецептів, їх відбір за критеріями та обмін думками. Їхня реалізація базується на продуманому дизайні, що гарантує інтуїтивний і комфортний інтерфейс.

4 ЕКСПЛУАТАЦІЯ, ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ ПРОГРАМИ

4.1 Призначення та умови застосування програми

Сучасні технології впливають на різні сфери життя, включаючи кулінарію. Інтерактивний застосунок для кондитерів на Java є інноваційним рішенням для створення, зберігання та пошуку рецептів. Він призначений для користувачів будь-якого рівня досвіду та забезпечує зручний інтерфейс для роботи з рецептами. Основна мета програми – надати зручний інструмент для роботи з рецептами. Вона дозволяє:

- зберігати та організовувати рецепти: користувачі можуть додавати, редагувати та структурувати рецепти за назвами, інгредієнтами, інструкціями та інформацією про автора;
- шукати рецепти: програма забезпечує швидкий пошук за ключовими словами, назвами чи інгредієнтами;
- ділитися рецептами: користувачі можуть створювати облікові записи, додавати свої рецепти та обмінюватися ними;
- оптимізувати приготування: чіткі інструкції та списки інгредієнтів допомагають ефективно планувати процес;
- забезпечувати доступність: програма підтримує роботу для зареєстрованих користувачів та гостей.

Програма розроблена для використання в різних умовах, що робить її універсальним інструментом для широкого кола користувачів. Основні умови застосування показані в таблиці 4.1.

Таблиця 4.1 – Основні умови застосування

№	Переваги	Опис
1	Особисте використання	Користувачі можуть використовувати програму для особистих потреб, таких як планування домашньої випічки або пошук нових ідей для приготування страв
2	Професійне використання	Кондитери та кулінари можуть використовувати програму для організації своєї роботи, додавання професійних рецептів та обміну досвідом з колегами
3	Навчальний процес	Програма може бути використана у навчальних закладах для вивчення основ кондитерської справи. Вона дозволяє студентам знаходити та аналізувати рецепти, що сприяє кращому засвоєнню матеріалу
4	Комерційне використання	Кафе, пекарні та кондитерські можуть використовувати програму для ведення внутрішньої бази рецептів, що дозволяє оптимізувати робочий процес та підвищити ефективність виробництва
5	Мобільність	Програма може бути використана на різних пристроях, включаючи комп'ютери та ноутбуки, що забезпечує доступ до рецептів у будь-якому місці та в будь-який час

Програма надає широкий спектр функціональних можливостей, які забезпечують зручність та ефективність її використання (таблиця 4.2).

Таблиця 4.2 – Основні функціональні можливості

№	Переваги	Опис
1	Реєстрація та автентифікація користувачів	Користувачі можуть створювати облікові записи, що дозволяє їм додавати свої рецепти та отримувати доступ до них з будь-якого пристрою
2	Додавання та редагування рецептів	Користувачі можуть додавати нові рецепти, вказуючи назву, список інгредієнтів, інструкції з приготування та іншу необхідну інформацію. Також є можливість редагувати вже існуючі рецепти
3	Пошук рецептів	Програма забезпечує швидкий пошук рецептів за ключовими словами, назвами або інгредієнтами. Це дозволяє користувачам знаходити потрібну інформацію за лічені секунди
4	Доступ для гостей	Програма дозволяє гостям переглядати рецепти без необхідності реєстрації, що робить її доступною для широкого кола користувачів

Для коректної роботи програми необхідно забезпечити відповідні технічні умови, що показані в таблиці 4.3.

Таблиця 4.3 – Технічні вимоги

№	Вимоги	Опис
1	Операційна система	Програма може працювати на будь-якій операційній системі, яка підтримує Java (Windows, macOS, Linux)
2	Java Runtime Environment (JRE)	Для запуску програми необхідно встановити Java Runtime Environment версії 8 або вище
3	База даних	Програма використовує SQLite для зберігання даних, що дозволяє їй працювати без необхідності встановлення додаткового програмного забезпечення для управління базою даних
4	Інтернет-з'єднання	Хоча програма може працювати в локальному режимі, для повноцінного використання всіх функцій (наприклад, синхронізації даних між пристроями) рекомендується стабільне інтернет-з'єднання
5	Апаратні вимоги	Програма не вимагає значних апаратних ресурсів і може працювати на більшості сучасних пристроїв

Програма має низку переваг, які роблять її привабливою для користувачів (таблиця 4.4).

Таблиця 4.4 – Переваги програми

№	Переваги	Опис
1	Простота використання	Інтуїтивно зрозумілий інтерфейс дозволяє користувачам швидко освоїти програму та почати її використовувати
2	Гнучкість	Програма підходить як для особистого, так і для професійного використання, що робить її універсальним інструментом
3	Мобільність	Можливість використовувати програму на різних пристроях забезпечує доступ до рецептів у будь-якому місці
4	Безпека	Програма забезпечує захист даних користувачів за допомогою автентифікації
5	Постійне оновлення	Програма може бути оновлена з урахуванням нових вимог користувачів та технологічних трендів

Інтерактивний застосунок для кондитерів є сучасним та ефективним інструментом для роботи з рецептами. Він призначений для широкого кола користувачів і може бути використаний в різних умовах, від домашнього використання до професійної діяльності. Програма забезпечує зручний інтерфейс, широкий спектр функціональних можливостей та високу продуктивність, що робить її незамінним помічником для всіх, хто цікавиться кондитерською справою.

4.2 Вимоги до інтерфейсу застосунку

Інтерфейс застосунку – ключовий елемент взаємодії, що має бути зручним, інтуїтивним та ефективним для професіоналів і аматорів кондитерської справи. Основні вимоги: ергономічність, візуальна привабливість тощо. Ергономічність забезпечує логічне розташування елементів управління, що дозволяє користувачам швидко орієнтуватися. Для досягнення цього необхідно враховувати фактори, що подані в таблиці 4.5.

Таблиця 4.5 – Елементи ергономічності

№	Елементи	Опис
1	Інтуїтивна навігація	Інтерфейс повинен бути побудований таким чином, щоб користувач міг легко знаходити потрібні функції. Наприклад, головне меню має містити чіткі та зрозумілі пункти, такі як "Рецепти", "Пошук", "Створити рецепт". Кожен розділ повинен мати логічну структуру
2	Мінімалізм та простота	Інтерфейс не повинен бути перевантажений елементами управління. Кожна кнопка, поле введення чи інший елемент повинен мати чітке призначення. Наприклад, форма створення рецепту має містити лише необхідні поля: назва, інгредієнти, інструкції. Зайві елементи можуть відволікати користувача та ускладнювати процес взаємодії
3	Послідовність дій	Кожен процес у застосунку, наприклад, реєстрація, пошук рецепту чи створення нового рецепту, повинен бути побудований у вигляді послідовних кроків.

Візуальний дизайн інтерфейсу відіграє важливу роль у залученні користувачів та створенні позитивного враження від використання застосунку. Для кондитерського застосунку важливо створити теплу та затишну атмосферу, що нагадує про домашню кухню та солодощі.

Деталі розписані в таблиці 4.6.

Таблиця 4.6 – Елементи візуальної привабливості

№	Елементи	Опис
1	Кольорова схема	Використання теплих та насичених кольорів, таких як шоколадний, карамельний, бежевий, дозволяє створити відповідну атмосферу. Наприклад, основним кольором може бути бежевий (#DDC5A2), що символізує глазур, а акцентними кольорами — темно-коричневий (#301B28) та червоний (#B6452C), що асоціюються з шоколадом та ягодами
2	Типографіка	Шрифти повинні бути легкими для читання та відповідати загальній стилістиці застосунку. Використання заокруглених шрифтів з елементами рукописного стилю може додати інтерфейсу домашнього затишку. Розмір шрифту повинен бути достатньо великим для комфортного читання.
3	Іконки та ілюстрації	Використання іконок, що відображають основні функції застосунку (наприклад, іконки зірочок для оцінки рецепту), дозволяє зробити інтерфейс більш зрозумілим. В майбутньому при розширенні функціоналу, ілюстрації, такі як зображення тортів, печива чи інших десертів, можуть підкреслити тематику застосунку

Інтерфейс повинен забезпечувати доступ до всіх функціональних можливостей застосунку, включаючи пошук рецептів, створення нових рецептів, редагування та видалення існуючих (таблиця 4.7).

Таблиця 4.7 – Функціональність

№	Елементи	Опис
1	Пошук рецептів	Інтерфейс пошуку повинен бути простим та ефективним. Користувач повинен мати можливість шукати рецепти за ключовими словами, інгредієнтами або назвами. Результати пошуку повинні відображатися у вигляді списку
2	Створення та редагування рецептів	Форма створення рецепту повинна бути інтуїтивною та містити всі необхідні поля: назва, інгредієнти, інструкції. Редагування рецепту повинно бути доступним лише для автора рецепту, що забезпечує захист від несанкціонованих змін
3	Управління профілем	Користувач повинен мати можливість редагувати свої дані – наприклад, змінювати пароль. Інтерфейс профілю повинен бути простим та зрозумілим, з чіткими кнопками для виконання основних дій

Інтерфейс застосунку повинен бути доступним для всіх користувачів, незалежно від їхнього рівня технічної підготовки чи фізичних обмежень. Елементи доступності показані в таблиці 4.8.

Таблиця 4.8 – Елементи доступності

№	Елементи	Опис
1	Підтримка різних пристроїв	Застосунок повинен коректно відображатися на екранах різних розмірів, включаючи ноутбуки та комп'ютери
2	Підтримка доступності для людей з обмеженими можливостями	Інтерфейс повинен підтримувати різні функції, наприклад, високу контрастність. Це дозволяє зробити застосунок доступним для користувачів з порушенням зору

Інтерфейс інтерактивного застосунку для кондитерів повинен бути гнучким та здатним адаптуватися до змін у функціональності застосунку або до нових вимог користувачів.

Елементи адаптивності застосунку для кондитерів представлені в таблиці 4.9.

Таблиця 4.9 – Елементи адаптивності

№	Елементи	Опис
1	Можливість оновлення	Інтерфейс повинен бути розроблений таким чином, щоб дозволяти легко додавати нові функції чи змінювати існуючі без порушення загальної структури. Наприклад, додавання нових категорій рецептів або нових способів пошуку не повинно вимагати повної переробки інтерфейсу
2	Зворотний зв'язок	Інтерфейс повинен забезпечувати можливість отримання зворотного зв'язку від користувачів. Наприклад, кнопка "Підтримка" для відправки пропозицій дозволяє користувачам впливати на розвиток застосунку

Інтерфейс повинен забезпечувати захист персональних даних користувачів та запобігати несанкціонованому доступу до їхніх рецептів та інформації.

Захист даних: форми реєстрації та авторизації повинні використовувати виключення для захисту паролів та інших конфіденційних даних, наприклад перевірка унікальності логіну користувача. Користувачі повинні бути впевнені, що їхні дані захищені.

Контроль доступу: інтерфейс повинен забезпечувати доступ до редагування та видалення рецептів лише для їхніх авторів. Це дозволяє уникнути несанкціонованих змін та захищає авторські права користувачів.

Для забезпечення якості інтерфейсу необхідно проводити регулярне тестування (таблиця 4.10).

Таблиця 4.10 – Тестування інтерфейсу

№	Елементи	Опис
1	Юзабіліті-тестування	Перевірка зручності використання інтерфейсу на реальних користувачах. Це дозволяє виявити проблеми, які можуть бути не помітні на етапі розробки
2	Тестування на різних апаратах	Перевірка коректного відображення інтерфейсу на різних екранах комп'ютерів та ноутбуків
3	Тестування доступності	Перевірка підтримки функцій доступності для людей з обмеженими можливостями

Вимоги до інтерфейсу застосунку для кондитерів включають ергономічність, візуальну привабливість, функціональність, доступність, адаптивність та безпеку. Реалізація цих вимог дозволяє створити зручний, привабливий та ефективний інтерфейс, що задовольняє потреби користувачів та сприяє популярності застосунку.

4.3 Умови виконання програми

Для успішного виконання програми "Інтерактивний застосунок для кондитерів" необхідно забезпечити відповідні технічні та програмні умови, які дозволять користувачам ефективно взаємодіяти з системою та отримувати максимальну користь від її функціональних можливостей. Умови виконання програми включають як мінімальні вимоги до апаратного забезпечення, так і налаштування програмного середовища, що забезпечує стабільну роботу застосунку. Для запуску програми необхідний пристрій, який відповідає таким мінімальним характеристикам, що представлені в таблиці 4.11.

Таблиця 4.11 – Апаратні вимоги

№	Елемент	Вимоги
1	Процесор	Процесор з тактовою частотою не менше 1.8 ГГц (наприклад, Intel Core i3 або аналогічний). Це забезпечить достатню продуктивність для обробки запитів та рендерингу інтерфейсу користувача
2	Оперативна пам'ять	Не менше 4 ГБ оперативної пам'яті. Це дозволить програмі ефективно працювати з великими наборами даних, такими як списки рецептів та інші елементи
3	Жорсткий диск	Вільний простір на диску не менше 500 МБ для зберігання бази даних, файлів програми та тимчасових даних
4	Монітор	Рекомендується роздільна здатність екрану не менше 1280x720 пікселів для комфортного відображення
5	Мережевий інтерфейс	Наявність стабільного підключення до Інтернету для роботи з серверною частиною програми. Це особливо важливо для клієнтського застосунку, який взаємодіє з сервером через HTTP-запити

Ці вимоги є мінімальними та забезпечують базову роботу програми. Для більш комфортного використання рекомендується використовувати більш потужні пристрої, особливо якщо планується одночасна робота з великою кількістю рецептів або інших даних.

Програма розроблена для роботи в середовищі Java, тому необхідно забезпечити програмні умови, які описані в таблиці 4.12.

Таблиця 4.12 – Програмні вимоги

№	Компоненти	Вимоги
1	Java Runtime Environment (JRE)	Для запуску клієнтського застосунку необхідно встановити Java Runtime Environment версії 11 або новішої. Це забезпечує підтримку всіх функцій, які використовуються в програмі, включаючи роботу з HTTP-запитами та обробку JSON
2	Java Development Kit (JDK)	Для розробників, які планують модифікувати або розширювати програму, необхідно встановити JDK версії 11 або новішої. Це дозволить компілювати та запускати програму з вихідних кодів
3	База даних SQLite	Серверна частина програми використовує SQLite для зберігання даних. Ця база даних не вимагає окремого встановлення, оскільки вона вбудована в програму. Однак необхідно забезпечити доступ до файлу бази даних (confectionery.db) для читання та запису
4	Бібліотеки	Програма використовує такі бібліотеки, як Jackson для роботи з JSON та JavaFX для створення графічного інтерфейсу користувача. Ці бібліотеки повинні бути включені до проєкту або встановлені окремо, якщо програма запускається з вихідних кодів
5	Операційна система	Програма підтримує роботу на таких операційних системах, як Windows, macOS та Linux. Для кожного з цих середовищ необхідно забезпечити відповідну версію Java

Оскільки програма складається з клієнтської та серверної частин, важливим питанням є налаштування мережевого середовища (таблиця 4.13).

Для коректного запуску програми необхідно виконати кроки, описані в таблиці 4.14.

Таблиця 4.13 – Мережеві вимоги

№	Компоненти	Вимоги
1	Серверна частина	Серверна частина програми працює на порту 8080. Необхідно забезпечити, щоб цей порт був відкритий для вхідних з'єднань. Якщо сервер запускається в локальній мережі, слід переконатися, що брандмауер або мережеві налаштування не блокують доступ до цього порту
2	Клієнтська частина	Клієнтський застосунок взаємодіє з сервером через HTTP-запити. Для цього необхідно забезпечити стабільне підключення до Інтернету або локальної мережі, де розгорнуто сервер
3	CORS (Cross-Origin Resource Sharing)	Серверна частина підтримує CORS, що дозволяє клієнтському застосунку взаємодіяти з сервером з різних доменів. Це важливо для розробки та тестування програми в різних середовищах

Таблиця 4.14 – Кроки налаштування середовища

№	Кроки	Опис
1	Завантаження та встановлення Java	Якщо Java не встановлена на комп'ютері, слід завантажити та встановити останню версію з офіційного сайту Oracle або OpenJDK
2	Запуск сервера	Серверна частина програми запускається з вихідного коду або з попередньо скомпільованого JAR-файлу. Для запуску сервера необхідно виконати команду <code>java -jar Server.jar</code> у терміналі або командному рядку
3	Запуск клієнта	Клієнтський застосунок також запускається з вихідного коду або JAR-файлу. Для запуску необхідно виконати команду <code>java -jar Client.jar</code> . Якщо використовується JavaFX, слід переконатися, що відповідні бібліотеки підключені до проєкту
4	Налаштування бази даних	Програма використовує SQLite, тому файл бази даних (<code>confectionery.db</code>) повинен бути доступний для читання та запису. Якщо база даних відсутня, програма автоматично створить її при першому запуску

Для розробників, які працюють з вихідним кодом програми, необхідно забезпечити умови, що показані в таблиці 4.15.

Таблиця 4.15 – Умови для розробників

№	Компоненти	Вимоги
1	Інтегроване середовище розробки (IDE)	Рекомендується використовувати IntelliJ IDEA, Eclipse або іншу IDE, яка підтримує Java. Це дозволить зручно редагувати код, компілювати та запускати програму
2	Система контролю версій	Для зручності розробки рекомендується використовувати Git для керування версіями коду. Це дозволить відстежувати зміни та співпрацювати з іншими розробниками
3	Тестування	Для тестування програми рекомендується використовувати фреймворки, такі як JUnit для модульного тестування та Selenium для тестування інтерфейсу користувача

Для користувачів, які використовують програму, важливо забезпечити конкретні умови для її коректної роботи. Дані умови детально описані в таблиці 4.16.

Для тестування програми необхідно забезпечити умови показані в таблиці 4.17.

Таблиця 4.16 – Умови для користувачів

№	Компоненти	Вимоги
1	Простота інтерфейсу	Графічний інтерфейс користувача розроблений з урахуванням зручності використання. Користувачі можуть легко реєструватися, входити в систему, додавати рецепти та шукати їх за назвами
2	Документація	Для полегшення використання програми рекомендується надати коротку інструкцію або довідку, яка пояснює основні функції та можливості застосунку
3	Підтримка	Рекомендується надати користувачам можливість звертатися до розробників через електронну пошту або інші канали зв'язку у разі виникнення проблем

Таблиця 4.17 – Умови для тестування

№	Компоненти	Вимоги
1	Тестові дані	Програма містить тестові дані, які дозволяють перевірити всі основні функції. Ці дані включають тестових користувачів, рецепти та інгредієнти
2	Автоматизоване тестування	Для перевірки стабільності програми рекомендується використовувати автоматизовані тести, які охоплюють всі основні сценарії використання
3	Ручне тестування	Окрім автоматизованих тестів, важливо провести ручне тестування для перевірки інтерфейсу користувача та взаємодії з сервером

Для проведення експериментального дослідження програми необхідно забезпечити конкретні умови. Дані умови представлені в наступній таблиці 4.18.

Таблиця 4.18 – Умови для експериментального дослідження

№	Компоненти	Вимоги
1	Збір даних	Програма дозволяє збирати дані про використання, такі як кількість рецептів, популярність інгредієнтів та активність користувачів. Ці дані можуть бути використані для аналізу ефективності програми
2	Аналіз продуктивності	Для оцінки продуктивності програми рекомендується використовувати інструменти, такі як JMeter або VisualVM, які дозволяють вимірювати час виконання запитів та використання ресурсів
3	Звітність	Результати експериментального дослідження повинні бути оформлені у вигляді звіту, який включає опис методів, отримані результати та висновки

Програма може бути масштабована для роботи з більшою кількістю користувачів та даних. Для цього необхідно забезпечити наступні умови з таблиці 4.19.

Таблиця 4.19 – Умови для масштабування

№	Компоненти	Вимоги
1	Використання потужнішого серверного обладнання	Для обробки великої кількості запитів рекомендується використовувати сервери з більшою кількістю ядер процесора та оперативної пам'яті
2	Оптимізація бази даних	Для покращення продуктивності у майбутньому можна використовувати інші СУБД, такі як PostgreSQL або MySQL, які підтримують більш складні запити та індексацію
3	Використання хмарних технологій	У наступних оновленнях програму можна розгорнути в хмарному середовищі, наприклад, AWS або Google Cloud, що дозволить автоматично масштабувати ресурси в залежності від навантаження

Для забезпечення безпеки програми необхідно враховувати деякі нюанси. Дані нюанси представлені в наступній таблиці 4.20.

Таблиця 4.20 – Умови для безпеки

№	Компоненти	Вимоги
1	Обмеження доступу	Серверна частина програми повинна бути налаштована таким чином, щоб обмежити доступ до неавторизованих користувачів
2	Захист даних	Програма використовує просту автентифікацію за допомогою логіну та пароля. Для підвищення безпеки в майбутніх версіях рекомендується використовувати шифрування паролів та HTTPS для захисту переданих даних
3	Регулярне оновлення	Для запобігання вразливостям рекомендується регулярно оновлювати бібліотеки та залежності, які використовуються в програмі

Умови виконання програми "Інтерактивний застосунок для кондитерів" включають широкий спектр технічних, програмних та мережевих вимог, які забезпечують стабільну та ефективну роботу системи. Відповідне налаштування використовуваного середовища, забезпечення безпеки застосунку та масштабованості дозволяють програмі функціонувати в різних умовах та задовольняти потреби користувачів.

4.4 Методика тестування основних функцій: пошук, коментарі

Тестування є важливим етапом розробки будь-якого програмного забезпечення, оскільки воно дозволяє виявити наявні помилки, перевірити коректність роботи функціональних модулів програми та забезпечити стабільність системи.

Для тестування даного інтерактивного застосунку для кондитерів була використана достатня кількість різноманітних програм, що надавали багатий вибір інструментів.

У контексті інтерактивного застосунку для кондитерів, основним функціям, таким як пошук та коментарі приділяється особлива увага, оскільки вони є ключовими для забезпечення зручності користувачів.

Методика тестування цих функцій, яка включає підходи, інструменти та етапи перевірки описана в таблиці 4.21.

Тестування основних функцій застосунку, таких як пошук та коментарі, є важливим етапом для забезпечення якості та стабільності системи. Використання комбінації ручного та автоматизованого тестування дозволяє виявити помилки на різних рівнях, від логіки до інтерфейсу користувача.

Інструменти, такі як JUnit, Postman та JMeter, забезпечують ефективну перевірку функціональності та продуктивності. Результати тестування дозволяють впевнитися, що застосунок відповідає вимогам користувачів та забезпечує зручний та безпечний досвід взаємодії.

Таблиця 4.21 – Методика тестування функцій

№	Функція	Опис	Етапи	Інструменти
1	Пошук рецептів	Функція пошуку рецептів є однією з найважливіших у застосунку, оскільки вона дозволяє користувачам швидко знаходити потрібні рецепти за ключовими словами, інгредієнтами або назвами. Для тестування цієї функції використовується комбінація ручного та автоматизованого тестування	Тестування валідних даних: – перевірка коректності пошуку за назвою рецепту: наприклад, введення назви "Шоколадний торт" має повернути відповідний рецепт; – перевірка пошуку за інгредієнтами: наприклад, введення "цукор" має повернути всі рецепти, які містять цей інгредієнт; – перевірка пошуку за частковим введенням: наприклад, введення "шоко" має повернути всі рецепти, які містять цей фрагмент у назві або описі. Тестування невалідних даних: – перевірка поведінки системи при введенні неіснуючих назв або інгредієнтів;	JUnit для автоматизованого тестування логіки пошуку. Ручне тестування інтерфейсу користувача. JMeter для тестування продуктивності та навантаження.

Продовження таблиці 4.21

№	Функція	Опис	Етапи	Інструменти
			– перевірка реакції на порожній запит: наприклад, система має повернути всі доступні рецепти. Тестування продуктивності: – перевірка швидкості пошуку при великій кількості рецептів: система має забезпечувати швидкий пошук навіть при наявності тисяч записів; – перевірка стабільності при одночасному виконанні багатьох пошукових запитів. Тестування інтерфейсу: – перевірка коректності відображення результатів пошуку: вони мають бути відсортовані за релевантністю та відповідати введеним критеріям.	

Завершення таблиці 4.21

№	Функція	Опис	Етапи	Інструменти
2	Коментарі до рецептів	Функція коментування дозволяє зареєстрованим користувачам залишати відгуки до рецептів, сприяючи соціальній взаємодії та обміну досвідом	Тестування додавання коментарів: – перевірка можливості додавання коментарів лише зареєстрованими користувачами (наявність токена авторизації); – перевірка відмови в додаванні порожніх коментарів (<code>text.trim().isEmpty()</code>). Тестування безпеки: – перевірка захисту від SQL-ін'єкцій через використання параметризованих запитів. Тестування відображення коментарів: – перевірка коректності відображення коментарів для заданого рецепту через параметр <code>recipeId</code>; – перевірка сортування коментарів за датою додавання (найновіші зверху).	Postman для тестування логіки додавання коментарів та їх перегляду. Postman для тестування API

4.5 Результати роботи алгоритму пошуку та фільтрації

Алгоритм пошуку є ключовим елементом застосунку, забезпечуючи зручний доступ до рецептів. Його основна мета — надати користувачам можливість швидко знаходити рецепти за ключовими словами, інгредієнтами чи іншими параметрами.

Алгоритм реалізований за допомогою SQL-запитів, що дозволяють шукати рецепти за назвою, інгредієнтами чи їх комбінацією. Використання індексів прискорює пошук та оптимізує запити. Наприклад, таблиця `recipe_ingredients` зв'язує рецепти з інгредієнтами через унікальні ідентифікатори, що дозволяє швидко знаходити рецепти з певними інгредієнтами навіть у великій базі даних.

Для пошуку за назвою використовується оператор `LIKE`, який знаходить рецепти з ключовими словами у назві. Наприклад, пошук за словом "шоколадний" повертає всі рецепти з цим словом у назві.

Ефективність алгоритму оцінювалася за швидкістю виконання запитів, точністю результатів та зручністю використання (таблиця 4.22).

Таблиця 4.22 – Ефективність алгоритму

№	Компоненти	Опис
1	Швидкість пошуку	Середній час виконання запиту становить близько 50-100 мілісекунд для пошуку за назвою та 100-200 мілісекунд для пошуку за інгредієнтами. Це забезпечує миттєвий відгук на запити користувачів, що є критично важливим для зручності використання застосунку
2	Точність результатів	Алгоритм забезпечує високу точність пошуку, повертаючи лише релевантні результати. Наприклад, пошук за інгредієнтом «шоколад» повертає всі рецепти, які містять цей інгредієнт, без зайвих результатів
3	Зручність використання	Алгоритм підтримує пошук за частковим співпадінням, що дозволяє користувачам знаходити рецепти навіть за неповними даними. Наприклад, пошук за словом "торт" поверне всі рецепти, назви яких містять це слово, незалежно від їх повної форми

Для демонстрації роботи алгоритму розглянуто кілька прикладів пошуку, що висвітлені в таблиці 4.23.

Алгоритм пошуку покращує користувацький досвід, забезпечуючи швидкий доступ до рецептів. Користувачі можуть знаходити їх за параметрами, що робить застосунок зручним як для професіоналів, так і для аматорів. Підтримка часткового пошуку дозволяє знаходити рецепти навіть за неповними даними, що особливо корисно, якщо точна назва невідома.

Для підвищення ефективності алгоритму було використано кілька методів оптимізації, що подані в таблиці 4.24.

Таблиця 4.23 – Приклади пошуку

№	Пошук	Результат
1	Пошук за назвою	Користувач вводить слово "Шоколадний". Алгоритм повертає всі рецепти, назви яких містять це слово, наприклад, "Шоколадний торт", "Шоколадні мафіни", "Шоколадний фондан". Час виконання запиту становить близько 60 мілісекунд
2	Пошук за інгредієнтами	Користувач вводить інгредієнт "яблука". Алгоритм повертає рецепти, які містять цей інгредієнт, наприклад, "Яблучний штрудель", "Зефір". Час виконання запиту становить близько 120 мілісекунд

Таблиця 4.24 – Методи оптимізації алгоритму

№	Методи	Результат
1	Індексація	Створено індекси для полів title та ingredients у таблиці recipes, а також для поля name у таблиці ingredients. Це дозволяє прискорити пошук за цими полями
3	Оптимізація SQL-запитів	Використовуються оптимізовані SQL-запити, які мінімізують кількість операцій, що виконуються на сервері. Наприклад, для пошуку за інгредієнтами використовується JOIN між таблицями recipes, recipe_ingredients та ingredients, що дозволяє швидко знаходити рецепти за інгредієнтами

Алгоритм пошуку забезпечує швидкий і точний доступ до рецептів, підвищуючи зручність використання застосунку. Оптимізація через індексацію

та ефективні SQL-запити гарантує швидкий відгук на запити користувачів. У майбутньому можна додати пошук за категоріями, часом приготування або рівнем складності для покращення користувацького досвіду.

4.6 Висновки за отриманими результатами тестування

Тестування інтерактивного застосунку для кондитерів, розробленого на мові програмування Java, дозволило виявити ключові елементи його функціональності, продуктивності та зручності використання. Отримані результати демонструють, що програма успішно виконує свої основні завдання, забезпечуючи користувачам зручний інтерфейс для роботи з рецептами, пошуку інформації та взаємодії з базою даних. Проте, тестування також виявило певні області, які потребують покращення для забезпечення більш стабільної та ефективної роботи застосунку.

Отримані результати описані в таблиці 4.25.

Таблиця 4.25 – Висновки за отриманими результатами тестування

№	Критерій	Висновок за результатом
1	Зручність використання	Інтерфейс застосунку інтуїтивно зрозумілий, але користувачі відзначили необхідність покращення редактора рецептів
2	Функціональність застосунку	Тестування підтвердило коректну роботу основних модулів: реєстрація, автентифікація, зміна паролю, створення, редагування та видалення рецептів, а також пошук за ключовими словами та інгредієнтами. Виявлено, що при великій кількості запитів на пошук від клієнтів не сповільнюється
3	Продуктивність та стабільність	Застосунок стабільно працює при одночасній роботі до 50 користувачів (час відгуку 200-300 мс). При збільшенні навантаження до 1000 користувачів час відгуку зростає до 1 секунди. Для покращення продуктивності запропоновано використовувати кешування та масштабування серверної частини

Продовження таблиці 4.25

№	Критерій	Висновок за результатом
4	Безпека даних	Застосунок має захист від SQL-ін'єкцій. Однак відсутність шифрування паролів та двофакторної автентифікації може бути недоліком для користувачів, які працюють з конфіденційною інформацією
5	Експериментальне дослідження	Застосунок коректно працює на більшості сучасних пристроїв, але на старих пристроях з обмеженими ресурсами спостерігаються незначні затримки. Для покращення сумісності запропоновано оптимізувати передачу даних та використовувати ефективніші алгоритми обробки

Результати тестування підтвердили, що інтерактивний застосунок для кондитерів є функціональним, стабільним і зручним у використанні. Проте для його подальшого вдосконалення необхідно покращити пошуковий алгоритм, щоб забезпечити ефективну роботу з великими обсягами даних, а також впровадити механізми кешування та масштабування серверної частини, що підвищить продуктивність.

Додатково варто удосконалити інтерфейс створення рецептів, зокрема, надати можливість завантаження зображень. Посилення заходів безпеки за рахунок додаткових методів автентифікації також зробить застосунок надійнішим. Окрім цього, важливо оптимізувати його роботу для пристроїв із обмеженими ресурсами, що дозволить розширити аудиторію користувачів. Усі ці покращення сприятимуть підвищенню конкурентоспроможності застосунку та забезпеченню комфортного користування навіть для найвибагливіших споживачів.

ВИСНОВКИ

Розробка інтерактивного застосунку для кондитерів на мові програмування Java стала результатом комплексного підходу до вирішення актуальних завдань у сфері кулінарії, зокрема кондитерської справи. Застосунок створено з метою забезпечення зручного доступу до рецептів, полегшення процесу їх створення, редагування та пошуку, а також для формування спільноти користувачів, які можуть ділитися своїм досвідом та натхненням. У ході роботи було реалізовано функціональний інструмент, що відповідає сучасним вимогам до програмного забезпечення, зокрема щодо зручності інтерфейсу, швидкості роботи та безпеки даних.

Одним із ключових досягнень проєкту є створення архітектури застосунку, яка поєднує серверну частину з клієнтським інтерфейсом. Використання мови Java дозволило забезпечити стабільність роботи програми. Серверна частина, реалізована за допомогою вбудованого HTTP-сервера, забезпечує зберігання даних у SQLite базі даних, що дозволяє ефективно керувати рецептами та користувачами. Клієнтська частина, розроблена з використанням JavaFX, надає інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам легко взаємодіяти із застосунком. Використання таких технологій, як JSON для обміну даними між клієнтом і сервером, а також ObservableList для динамічного оновлення інтерфейсу, значно підвищило ефективність роботи програми.

Важливим напрямком розробки стало забезпечення функціональності, яка відповідає потребам користувачів. Застосунок дозволяє реєструватися та авторизуватися, змінювати пароль в акаунті, створювати та редагувати рецепти, а також здійснювати пошук за ключовими словами та інгредієнтами. Реалізація механізму пошуку, який враховує як назви рецептів, так і інгредієнти, забезпечує швидкий доступ до необхідної інформації.

Інтерфейс застосунку розроблено з урахуванням сучасних стандартів юзабіліті. Використання CSS для стилізації елементів інтерфейсу дозволило

створити привабливий та естетичний дизайн, який сприяє комфортній роботі з програмою. Крім того, реалізація асинхронних операцій за допомогою потоків забезпечила плавну роботу застосунку навіть при виконанні складних запитів до бази даних.

Тестування застосунку підтвердило його стабільність та ефективність. Було проведено комплексне тестування основних функцій, таких як реєстрація, авторизація, створення рецептів, пошук. Результати тестування показали, що застосунок коректно працює в різних умовах та забезпечує високу швидкість обробки запитів. Крім того, було виявлено та усунуто потенційні вразливості, що стосуються безпеки даних, зокрема запобігання SQL-ін'єкціям.

Розроблений застосунок має значний потенціал для подальшого вдосконалення. Серед можливих напрямів розвитку можна виділити додавання функцій створення списку улюблених рецептів та рекомендацій на основі вподобань користувачів, а також інтеграцію з соціальними мережами для зручного поширення рецептів. Крім того, можливе розширення функціональності шляхом додавання можливості онлайн-замовлення інгредієнтів або проведення віртуальних майстер-класів.

У цілому, розроблений інтерактивний застосунок для кондитерів є сучасним інструментом, який відповідає потребам як професійних кондитерів, так і аматорів. Він сприяє розвитку кулінарних навичок, полегшує доступ до інформації та створює умови для обміну досвідом між користувачами. Реалізація проекту підтвердила ефективність використання мови Java та сучасних технологій для створення інтерактивних застосунків, що мають практичне значення у повсякденному житті.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Google Play – Tasty [Electronic resource]. – Access mode: <https://play.google.com/store/apps/details/Tasty?id=com.buzzfeed.tasty&hl=pl&hl=uk&pli=1>.
2. Tasty [Electronic resource]. – Access mode: <https://tasty.co/>.
3. Google Play – Kitchen Stories: Recipes [Electronic resource]. – Access mode: <https://play.google.com/store/apps/details?id=com.ajnsnewmedia.kitchenstories&hl=pl&hl=uk>.
4. Google Play – Dessert Recipes [Electronic resource]. – Access mode: <https://play.google.com/store/apps/details?id=com.dilstudio.dessertrecipes&hl=pl&hl=uk>.
5. Great British Chefs [Electronic resource]. – Access mode: <https://www.greatbritishchefs.com/>.
6. Cake Master [Electronic resource]. – Access mode: <https://www.cake-masters.com/>.
7. Google Play – Dr. Oetker Rezeptideen [Electronic resource]. – Access mode: <https://play.google.com/store/apps/details?id=de.oetker.android.rezeptideen&hl=pl>.
8. Betty Crocker [Electronic resource]. – Access mode: <https://www.bettycrocker.com/>.
9. Wikipedia – Great British Chefs [Electronic resource]. – Access mode: https://en.wikipedia.org/wiki/Great_British_Chefs.
10. Great British Chefs – FAQs [Electronic resource]. – Access mode: <https://www.greatbritishchefs.com/faqs>.
11. Oracle – Understanding New User Registration [Electronic resource]. – Access mode: https://docs.oracle.com/cd/E56917_01/cs9pbr4/eng/cs/lscs/concept_UnderstandingNewUserRegistration-438000.html#topofpage.

12. QAGroup – Аутентифікація і авторизація: що це і в чому відмінність [Електрон. ресурс]. – Режим доступу: <https://qagroup.com.ua/publications/autentyfikatsiia-i-avtoryzatsiia/>.

13. EBSCO Connect – What is guest access and how do I configure it for users? [Electronic resource]. – Access mode: https://connect.ebsco.com/s/article/What-is-guest-access-and-how-do-I-configure-it-for-users?language=en_US.

14. Geeks for geeks – What is Multimedia? [Electronic resource]. – Access mode: <https://www.geeksforgeeks.org/what-is-multimedia/>.

15. Dspace – Захист даних користувачів в інформаційних системах [Електрон. ресурс]. – Режим доступу: <https://dspace.onua.edu.ua/server/api/core/bitstreams/976197cc-4ae9-4ca5-8f24-0b28f1baa1dc/content>.

16. Monday – Restricting only one of several guests to view only [Electronic resource]. – Access mode: <https://community.monday.com/t/restricting-only-one-of-several-guests-to-view-only/80946>.

17. Foxminded – Основи побудови SQL запитів [Електрон. ресурс]. – Режим доступу: <https://foxminded.ua/sql-zapyty/>.

18. Smashing magazine – How To Optimize Progressive Web Apps: Going Beyond The Basics [Electronic resource]. – Access mode: <https://www.smashingmagazine.com/2020/12/progressive-web-apps/>.

19. Шлякбित्रаф – Переклад чи локалізація [Електрон. ресурс]. – Режим доступу: <https://sbt.localization.com.ua/article/lokalizatsiya-onlajnovikh-videoihor/>.

20. Wezom – Як використовувати Google Analytics [Електрон. ресурс]. – Режим доступу: <https://wezom.com.ua/ua/blog/google-analytics>.

21. Wezom – Статистика пошукових запитів. Навіщо дивитися і які послуги використовувати [Електрон. ресурс]. – Режим доступу: <https://wezom.com.ua/ua/blog/kak-posmotret-statistiku-poiskovyh-zaprosov>.

22. Mark Web – Інтеграція соціальних мереж в дизайн веб-сайту: Поради та стратегії [Електрон. ресурс]. – Режим доступу:

<https://www.markweb.pro/integraciya-socialnih-merezh-v-dizajn-veb-sajtu-poradi-ta-strategii/>.

23. Activtrack – Productivity vs. Efficiency: Differences and Formulas [Electronic resource]. – Access mode: <https://www.activtrak.com/solutions/productivity-management/productivity-vs-efficiency/>.

24. Wikipedia – Response time (technology) [Electronic resource]. – Access mode: [https://en.wikipedia.org/wiki/Response_time_\(technology\)](https://en.wikipedia.org/wiki/Response_time_(technology)).

25. IBM – System request processing scenarios [Electronic resource]. – Access mode: <https://www.ibm.com/docs/en/i/7.4?topic=sessions-system-request-processing-scenarios>.

26. VPN Unlimited Оптимізація бази даних [Електрон. ресурс]. – Режим доступу: <https://www.vpnunlimited.com/ua/help/cybersecurity/database-optimization>.

27. Tseivo – Що таке Scalability (масштабованість)? [Електрон. ресурс]. – Режим доступу: <https://tseivo.com/b/memecode/t/eznj5bkeqz>.

28. Aerospike – Vertical vs. horizontal scaling explained [Electronic resource]. – Access mode: <https://aerospike.com/blog/vertical-vs-horizontal-scaling/>.

29. Host zealot – Розподілена система зберігання даних: Типи та реальні приклади [Електрон. ресурс]. – Режим доступу: <https://www.hostzealot.com.ua/blog/about-servers/rozpodilena-sistema-zberigannya-danix-tipi-ta-realni-prikladi>.

30. Кушнір, К. С. Золотухіна, К. І. Дослідження продуктивності та зручності використання мультимедійних додатків. Технологія і техніка друкарства. – 2021 – №3(73), С. 91–105. [Електрон. ресурс]. – Режим доступу: [https://doi.org/10.20535/2077-7264.3\(73\).2021.246426](https://doi.org/10.20535/2077-7264.3(73).2021.246426).

31. Погляди – Як розробити інтуїтивний і зручний користувацький інтерфейс для вебсайту [Електрон. ресурс]. – Режим доступу: <https://thepage.ua/ua/experts/yak-rozrobiti-intuyitivnij-i-zruchnij-koristuvackij-interfejs-dlya-vebsajtu>.

32. Tech Target – Mobile computing [Electronic resource]. – Access mode: <https://www.techtarget.com/searchmobilecomputing/definition/nomadic-computing>.
33. Verbo Labs – What is Localization? A Complete Guide to Reaching Global Audiences [Electronic resource]. – Access mode: <https://www.verbolabs.com/what-is-localization/>.
34. Ada.gov – Guidance on Web Accessibility and the ADA [Electronic resource]. – Access mode: <https://www.ada.gov/resources/web-guidance/>.
35. Tetrade – Why Is Application Security Important? [Electronic resource]. – Access mode: <https://tetrade.io/faq/why-is-application-security-important/>.
36. Auth0 – What is Authentication? [Electronic resource]. – Access mode: <https://auth0.com/intro-to-iam/what-is-authentication>.
37. Information security – Password reset vs change [Electronic resource]. – Access mode: <https://security.stackexchange.com/questions/214597/password-reset-vs-change>.
38. Adamant – Резервне копіювання даних і чому це важливо [Електрон. ресурс]. – Режим доступу: <https://adamant.ua/blog/rezervne-kopiyuvannya-danykh-i-chomu-tse-vazhlyvo>.
39. Tech target – Reliability of computers [Electronic resource]. – Access mode: <https://www.techtarget.com/whatis/definition/reliability>.
40. LinkedIn – Ensuring 24/7 Operations With Continuous Integration And Continuous Deployment [Electronic resource]. – Access mode: <https://www.linkedin.com/pulse/ensuring-247-operations-continuous-integration-deployment-yzl2c>.
41. Soft den – Моніторинг і логування для захисту бізнес-процесів [Електрон. ресурс]. – Режим доступу: <https://soft-den.com/monitoring-and-logging>.
42. Sii – Error handling – how to create clear messages [Electronic resource]. – Access mode: <https://sii.pl/blog/en/error-handling-how-to-create-clear-messages/>.

43. EMB Global – Software Support: Common Issues and Solutions [Electronic resource]. – Access mode: <https://blog.emb.global/software-support-issues-and-solutions/>.

44. Core – Technical Support: What It Is and How It Affects User Experience [Electronic resource]. – Access mode: <https://oncoredevit.com/news-and-insights/technical-support/technical-support-what-it-is-and-how-it-affects-user-experience/>.

45. Trucell – The Importance of Regular Software Updates [Electronic resource]. – Access mode: <https://www.trucell.com.au/blog/the-importance-of-regular-software-updates/>.

46. Tech Smith How to Build the Best User Manual [Electronic resource]. – Access mode: https://www.techsmith.com/blog/user-documentation/?srsltid=AfmBOoq3a7Y_TUQ16H-982fEvxj-jpZzsFyc-xbYbVOIvCFGpgrw5BTp.

47. Tech Target – Compatibility [Electronic resource]. – Access mode: <https://www.techtarget.com/whatis/definition/compatibility>.

48. DOU – Linux, Windows чи macOS? Сеньйор-розробники та інженери — про улюблені операційні системи [Електрон. ресурс]. – Режим доступу: <https://dou.ua/lenta/articles/linux-windows-or-macos/>.

49. Research Gate – [Electronic resource]. – Access mode: https://www.researchgate.net/publication/383127665_Exploring_the_Integration_of_Artificial_Intelligence_in_Retail_Operations.

50. Wezom – Коли та чому Java використовується для розробки? [Електрон. ресурс]. – Режим доступу: <https://wezom.com.ua/ua/blog/kogda-i-pochemu-java-ispolzuetsja-dlja-razrabotki-prilozhenij>.

51. Highload – Java для початківців: чи складно вивчити програмування на цій мові [Електрон. ресурс]. – Режим доступу: <https://highload.tech/uk/java-dlya-pochatktivtsiv-chy-skladno-vyvchyty-programuvannya-na-tsiy-movi/>.

52. Java Rush – Концепція об'єктно-орієнтованого програмування JAVA [Електрон. ресурс]. – Режим доступу:

<https://javarush.com/ua/groups/posts/uk.1265.koncepcja-obktno-orntovanogo-programuvannja-java>.

53. Resarch Gate – Багатопотокове програмування на Java [Електрон. ресурс]. – Режим доступу: https://www.researchgate.net/publication/356356477_Bagatopotokove_programmuвання_anna_na_Java.

54. Java Rush – Створення проєкту в IntelliJ IDEA [Електрон. ресурс]. – Режим доступу: <https://javarush.com/ua/groups/posts/uk.2311.stvorennja-proektu-v-intellij-idea>.

55. UA5.org – Популярні середовища розробки мов програмування [Електрон. ресурс]. – Режим доступу: <https://ua5.org/osnprog/1600-populyarni-seredovyshha-rozrobky-mov-programuvannya.html>.

56. Vukladach – Розробка об'єктно орієнтованих програм мовою java та їх документування [Електрон. ресурс]. – Режим доступу: https://vukladach.pp.ua/MyWeb/manual/ingenerija_programnogo_zabezpegennia/ob'yektno%20oriyentovane%20prohramuvannya/OB'YEKTNO%20ORIYENTOVAN'E%20PROHRAMUVANNYA/rozdil%201/1_4.html.

57. Guru99 – У чому полягає Maven Java (Archітекстура) [Електрон. ресурс]. – Режим доступу: <https://www.guru99.com/uk/maven-tutorial.html>.

58. JetBrains – UltimateSQLite [Electronic resource]. – Access mode: <https://www.jetbrains.com/help/idea/sqlite.html>.

59. Iwanoff – Створення програм графічного інтерфейсу користувача [Електрон. ресурс]. – Режим доступу: http://iwanoff.inf.ua/prev/java_ua/LabTraining04.html.

60. Фреймворк Junit [Електрон. ресурс]. – Режим доступу: https://studopedia.com.ua/1_246440_freymvork-JUnit.html.

61. The Apache Software Foundation – Apache JMeter [Electronic resource]. – Access mode: <https://jmeter.apache.org/>.

62. Postman API platform [Electronic resource]. – Access mode: <https://www.postman.com/>.

63. Java Rush – Введення в Java FX [Електрон. ресурс]. – Режим доступу: <https://javarush.com/ua/groups/posts/uk.2560.vvedennja-v-java-fx>.

64. Oracle – Interface ObservableList<E> [Electronic resource]. – Access mode:

<https://docs.oracle.com/javase/8/javafx/api/javafx/collections/ObservableList.html>.

65. Foxminded – Як парсити JSON у Java? [Електрон. ресурс]. – Режим доступу: <https://foxminded.ua/yak-parsyty-json-u-java/>.

66. Foxminded – REST API як спосіб спілкування компонент веб-додатків [Електрон. ресурс]. – Режим доступу: <https://foxminded.ua/shcho-take-rest-api/>.

67. Foxminded – Як оновити json-файл у Java [Електрон. ресурс]. – Режим доступу: <https://foxminded.ua/yak-onovyty-json-fail-u-java/>.

68. JavaScript.info – Особливості JavaScript [Електрон. ресурс]. – Режим доступу: <https://uk.javascript.info/javascript-specials>.

69. PlantUML – PlantUML at a Glance [Electronic resource]. – Access mode: <https://plantuml.com/>.

ДОДАТОК А
Технічне завдання

Вступ

Назва програми: інтерактивний застосунок для кондитерів.

У сучасному світі кондитерське мистецтво набуває все більшої популярності, а рецепти стають важливим джерелом натхнення для кулінарів. Проте, зберігання, пошук та обмін рецептами може бути складним завданням без належного інструментарію. Інтерактивний застосунок для кондитерів розроблено для полегшення процесу створення, зберігання, пошуку та обміну рецептами між користувачами. Програма дозволяє користувачам реєструватися, створювати рецепти, редагувати їх, шукати за ключовими словами та ділитися своїми кулінарними досягненнями.

Основною термінологією, яка використовується в програмі, є такі поняття, як "рецепт", "користувач", "авторизація", "пошук", "редагування", "коментування", "оцінювання". Ці терміни охоплюють основні функціональні можливості програми, які спрямовані на полегшення роботи з рецептами.

Необхідність розробки даного програмного забезпечення виникла через відсутність зручного інструменту для зберігання та пошуку рецептів, який би дозволяв користувачам легко додавати, редагувати та знаходити рецепти за ключовими словами. Програма стане корисним інструментом для кондитерів, кулінарів та любителів готувати, які хочуть зберігати свої рецепти в одному місці та ділитися ними з іншими.

Сфера застосування програми охоплює як домашніх користувачів, так і професійних кулінарів. Вона може бути використана для особистого використання, а також у навчальних закладах для вивчення основ кулінарного мистецтва.

A.1 Підстава для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу на тему «Програмна реалізація інтерактивного застосунку для кондитерів»,

затверджене наказом Національного університету «Запорізька політехніка» № 209 від 28 квітня 2025 р.

A.2 Призначення розробки

Програма має два основні призначення:

- зберігання та управління рецептами: користувачі можуть створювати нові рецепти, редагувати існуючі та видаляти їх, рецепти зберігаються в базі даних, що дозволяє їх швидко знаходити та використовувати;
- пошук та обмін рецептами: користувачі можуть шукати рецепти за ключовими словами, що дозволяє знаходити потрібні рецепти серед великої кількості даних, рецепти можуть бути доступні для перегляду іншим користувачам, що сприяє обміну кулінарним досвідом.

Програма розроблена для надання зручного інструменту для роботи з рецептами, що дозволяє користувачам ефективно зберігати, редагувати та знаходити потрібні рецепти.

A.3 Вимоги до програми або програмного продукту

Програмне забезпечення повинно відповідати ряду важливих вимог.

Функціональні характеристики:

- реєстрація та авторизація користувачів: користувачі можуть зареєструватися, вказавши ім'я користувача та пароль, після реєстрації користувачі можуть увійти в систему за допомогою своїх облікових даних;
- створення та редагування рецептів: користувачі можуть створювати нові рецепти, вказуючи назву, інгредієнти та інструкції, користувачі можуть редагувати свої рецепти, змінюючи назву, інгредієнти або інструкції;

- пошук рецептів: користувачі можуть шукати рецепти за ключовими словами, що дозволяє знаходити потрібні рецепти серед великої кількості даних;
- видалення рецептів: користувачі можуть видаляти свої рецепти, якщо вони більше не потрібні;
- перегляд рецептів: користувачі можуть переглядати рецепти, створені іншими користувачами;
- коментування рецептів: користувачі можуть додавати коментарі до рецептів, висловлюючи свої думки чи поради, користувачі можуть переглядати коментарі, залишені іншими користувачами до рецептів.

Надійність:

- програма повинна мати захист від некоректного введення даних користувачем, наприклад, при введенні пустих полів або некоректних символів;
- продукт має коректно працювати при обробці великих обсягів даних, щоб забезпечити стабільне функціонування при роботі з тисячами користувачів.

Умови експлуатації:

- платформи: Windows, Linux, macOS;
- сумісність із Java 11 і новішими версіями.

Технічні засоби:

- комп'ютер з процесором не нижче Intel Core i3 і 4 Гб оперативної пам'яті;
- установка Java Runtime Environment (JRE) версії 11 або новішої.

Інформаційна та програмна сумісність:

- використання стандартних бібліотек Java без потреби у встановленні сторонніх модулів;
- програма повинна бути легкодоступною для установки та використання, з чіткою інструкцією для користувача.

Маркування та упаковка: програмне забезпечення повинно бути упаковане в zip-архів з усіма необхідними файлами та інструкцією для установки.

Транспортування та зберігання: програма повинна зберігатися на захищених носіях, щоб уникнути механічних пошкоджень та електромагнітних впливів.

A.4. Вимоги до програмної документації

Документація повинна містити:

- опис функціоналу програми: детальний опис функцій програми, включаючи реєстрацію, авторизацію, створення, редагування, пошук та видалення рецептів;
- інструкцію користувача: послідовність дій для реєстрації, авторизації, створення рецептів, їх редагування та пошуку;
- технічну документацію: опис архітектури програми, включаючи основні модулі, деталі використання бібліотек для роботи з базою даних, опис тестів, що проводились для перевірки стабільності програми;
- звіт з результатами тестування: результати тестування продуктивності програми, включаючи час виконання операцій з рецептами.

A.5 Техніко-економічні показники

Орієнтовна економічна ефективність програми полягає в її здатності заощаджувати час користувачів на пошук та зберігання рецептів. Програма дозволить користувачам ефективно керувати своїми рецептами, що може підвищити продуктивність їхньої роботи на 20-30%.

A.6 Стадії та етапи розробки

Аналіз вимог і планування (тиждень): вивчення потреб користувачів та визначення функціональних вимог до програми.

Проектування архітектури програми (тиждень): визначення основних модулів програми, включаючи модулі для роботи з базою даних, авторизації, створення та пошуку рецептів.

Реалізація програми (3 тижні): написання коду для реалізації функцій програми.

Тестування та виправлення помилок (2 тижні): перевірка програми на коректність роботи з великими обсягами даних.

Підготовка документації (2 тижні): створення інструкцій для користувачів, технічної документації та звіту з результатами тестування.

A.7. Порядок контролю та приймання

Контроль виконання програми буде здійснюватися на всіх етапах розробки. Приймання програми відбуватиметься комісією, до складу якої увійдуть викладачі кафедри та фахівці з програмування. Протягом приймання програми проведуть тестування на відповідність вимогам та перевірку документації.

Критерії приймання продукту:

- точність виконання операцій: програма повинна коректно виконувати всі заявлені функції, включаючи реєстрацію, авторизацію, створення, редагування, пошук та видалення рецептів;
- здатність обробляти великі обсяги даних: продукт повинен стабільно працювати з тисячами клієнтів без суттєвих затримок або помилок;
- надання результатів у зручному вигляді: результати пошуку повинні бути представлені у вигляді зручного списку, що дозволяє користувачеві легко знаходити потрібні рецепти;

– завершена документація: документація повинна містити опис функцій програми, інструкції для користувачів та технічну інформацію щодо роботи продукту.

ДОДАТОК Б**Покроковий опис роботи програми**

Крок 1: Запуск програми.

Програма запускається з головного методу main класу Client. На цьому етапі ініціалізується графічний інтерфейс користувача за допомогою JavaFX. Створюється головне вікно застосунку з назвою "Інтерактивний застосунок для кондитерів", яке має розміри 800x600 пікселів. Встановлюється колірна схема інтерфейсу, що відповідає тематиці кондитерських виробів. Після ініціалізації інтерфейсу програма переходить до відображення стартового екрану входу.

Крок 2: Відображення стартового екрану входу.

На екрані входу користувач бачить форму для авторизації (рисунок Б.1), яка включає поля для введення логіну та паролю, а також кнопки для входу, реєстрації та входу як гість. Користувач може вибрати один із варіантів:

- увійти в систему, ввівши свої облікові дані;
- зареєструватися, якщо він новий користувач;
- увійти як гість, щоб переглядати рецепти без можливості їх редагування чи створення.

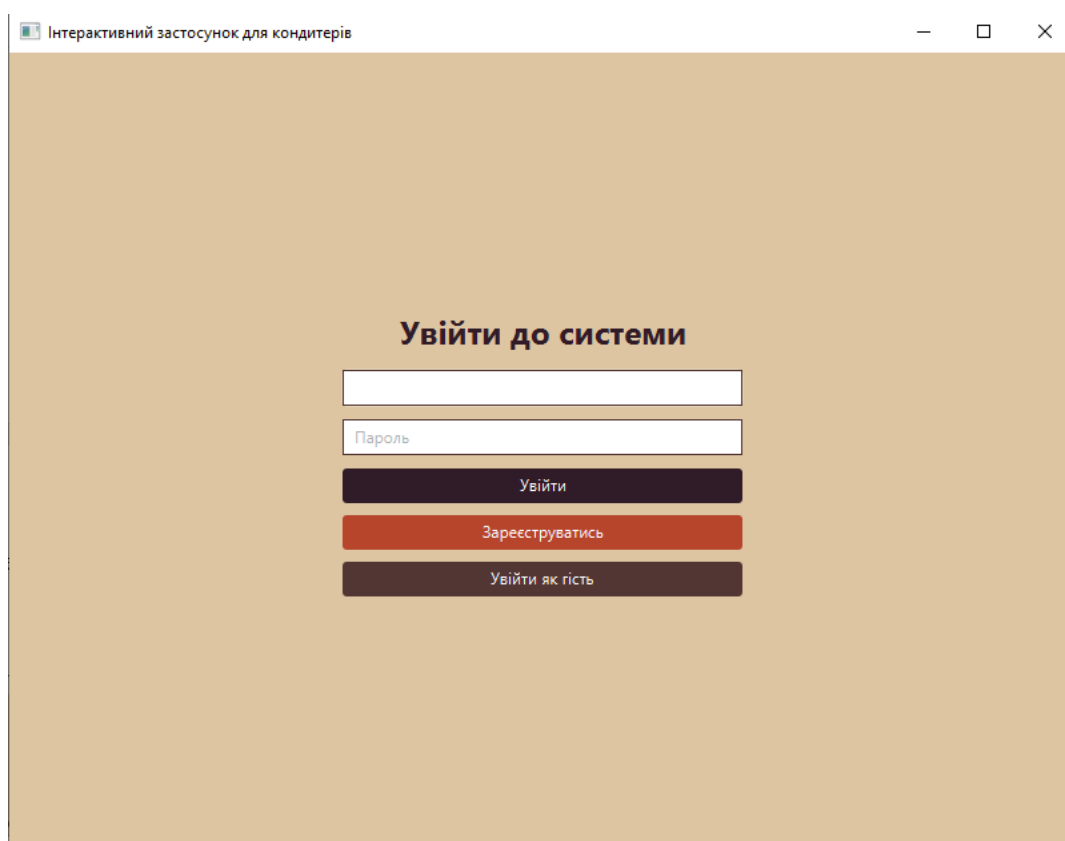


Рисунок Б.1 – Форма авторизації користувача

Крок 3: Авторизація або реєстрація користувача.

Якщо користувач обирає вхід або реєстрацію, програма відправляє HTTP-запит на сервер для перевірки облікових даних. У разі успішної авторизації сервер повертає токен доступу, який зберігається в програмі для подальших запитів. Після цього інтерфейс оновлюється, і користувач переходить до головного екрану зі списком рецептів.

Крок 4: Відображення головного екрану з рецептами.

На головному екрані відображається список рецептів, які завантажуються з сервера (рисунок Б.2). Кожен рецепт містить назву, інгредієнти та інструкції. Користувач може використовувати поле пошуку для фільтрації рецептів за ключовими словами. Для зареєстрованих користувачів також доступна кнопки "Створити новий рецепт" та "Змінити пароль".

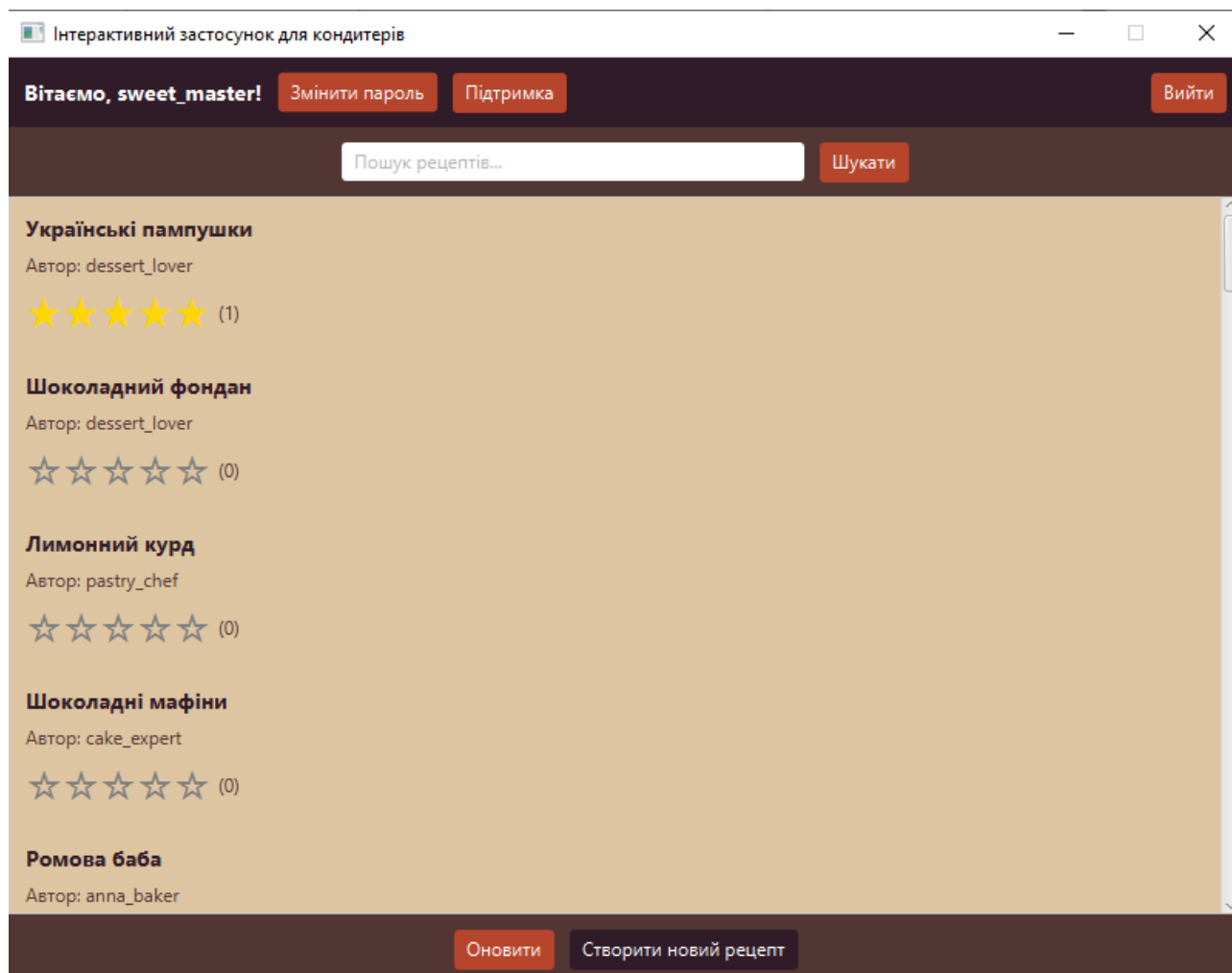


Рисунок Б.2 – Відображення головного екрану авторизованого користувача

Крок 5: Зміна паролю.

Кожен зареєстрований користувач має можливість змінити поточний пароль на новий за потреби. Для цього потрібно натиснути на кнопку "Змінити пароль" у верхньому лівому куточку. Відкриється форма з полями для вводу даних: поточний пароль, новий пароль та його підтвердження (рисунок Б.3). Для того, щоб підтвердити зміну паролю, потрібно на формі натиснути на кнопку "Змінити пароль". Якщо оновлення паролю не потрібне – то в такому випадку можна натиснути на кнопку "Назад".

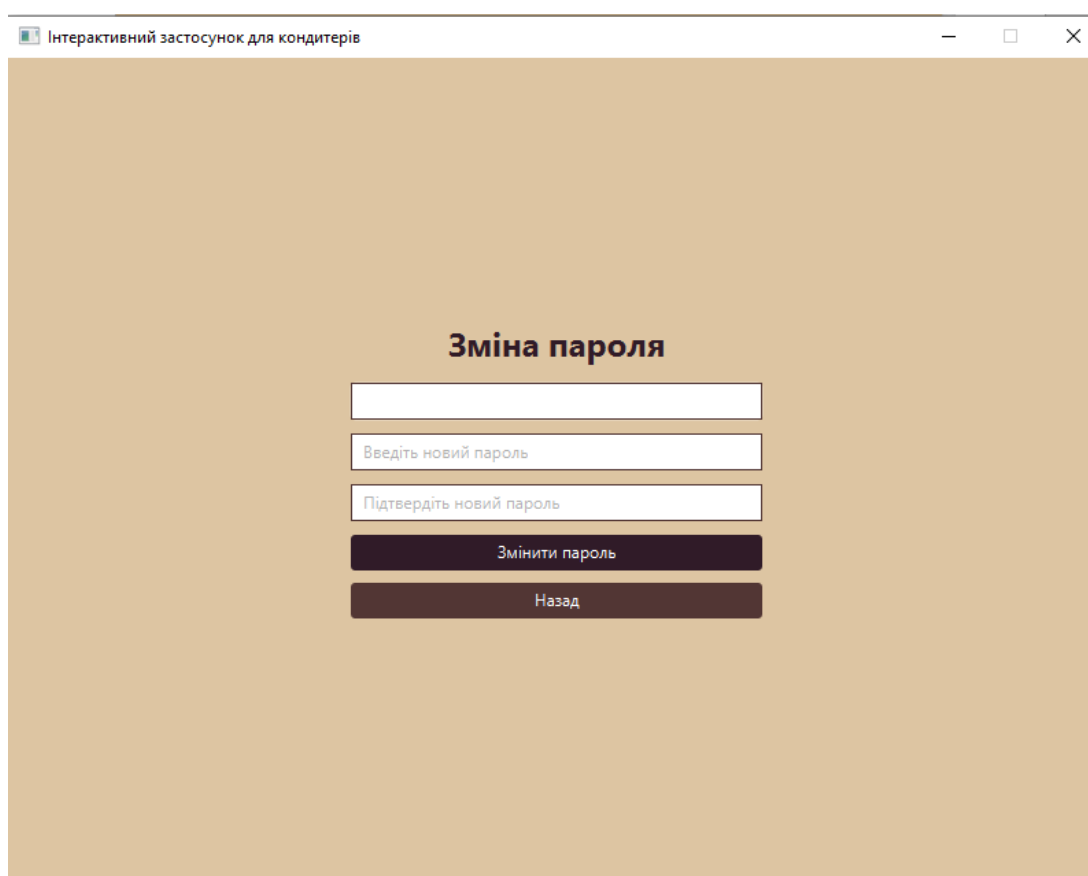
The image shows a web browser window titled "Інтерактивний застосунок для кондитерів". The main content area has a light beige background. In the center, the title "Зміна пароля" is displayed in bold black text. Below the title are three input fields: the first is empty, the second contains the placeholder text "Введіть новий пароль", and the third contains the placeholder text "Підтвердіть новий пароль". Below these fields are two dark brown buttons with white text: "Змінити пароль" and "Назад".

Рисунок Б.3 – Відображення вікна оновлення паролю

Крок 6: Перегляд деталей рецепту

При виборі конкретного рецепту зі списку відкривається сторінка з детальною інформацією (рисунок Б.4). Тут відображаються всі необхідні дані: назва, інгредієнти, інструкції та ім'я автора. Якщо користувач є автором рецепту, він може редагувати або видалити його.

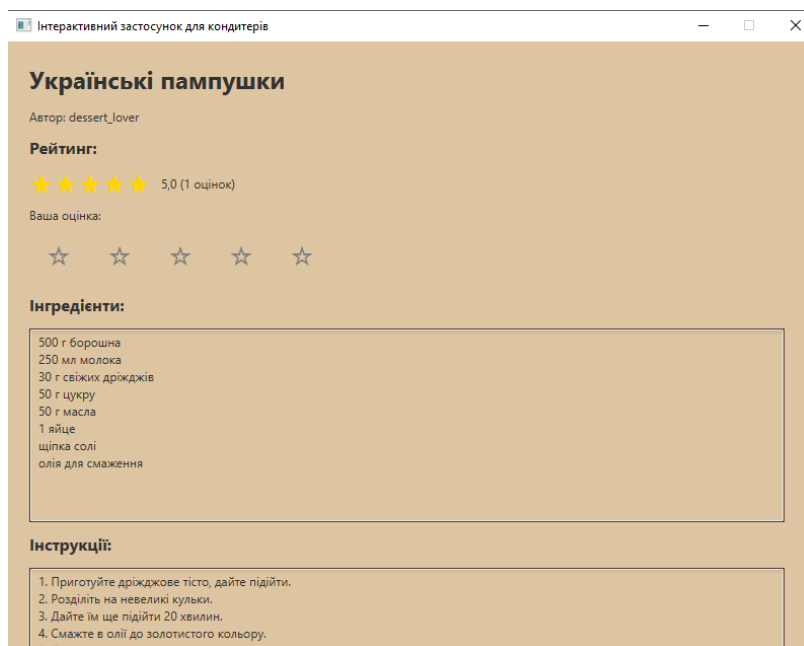


Рисунок Б.4 – Відображення деталей рецепту

Крок 7: Створення або редагування рецепту

Зареєстровані користувачі можуть створювати нові унікальні рецепти або редагувати існуючі, якщо вони є авторами (рисунок Б.5, рисунок Б.6, рисунок Б.7). Для цього відкривається форма, де можна ввести назву рецепту, інгредієнти та інструкції. Після збереження дані відправляються на сервер, і список рецептів оновлюється.

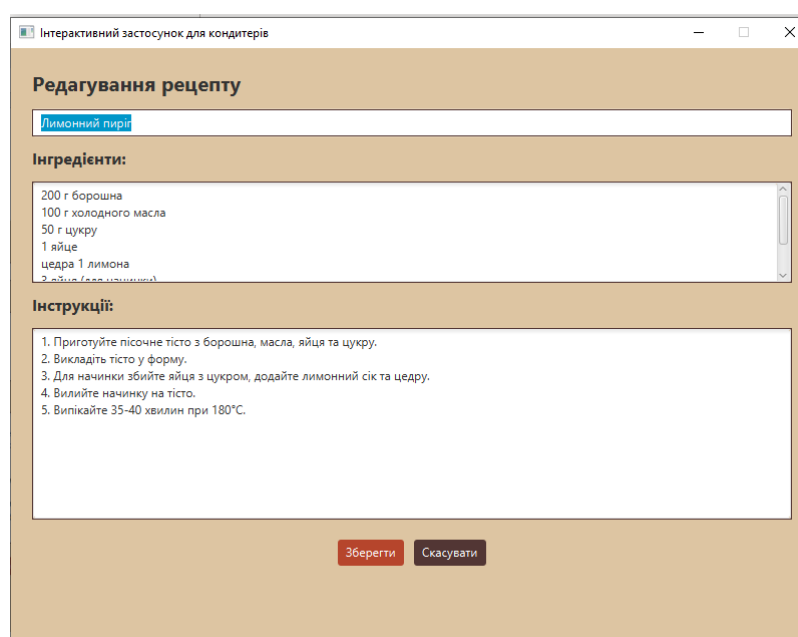


Рисунок Б.5 – Редагування існуючого рецепту

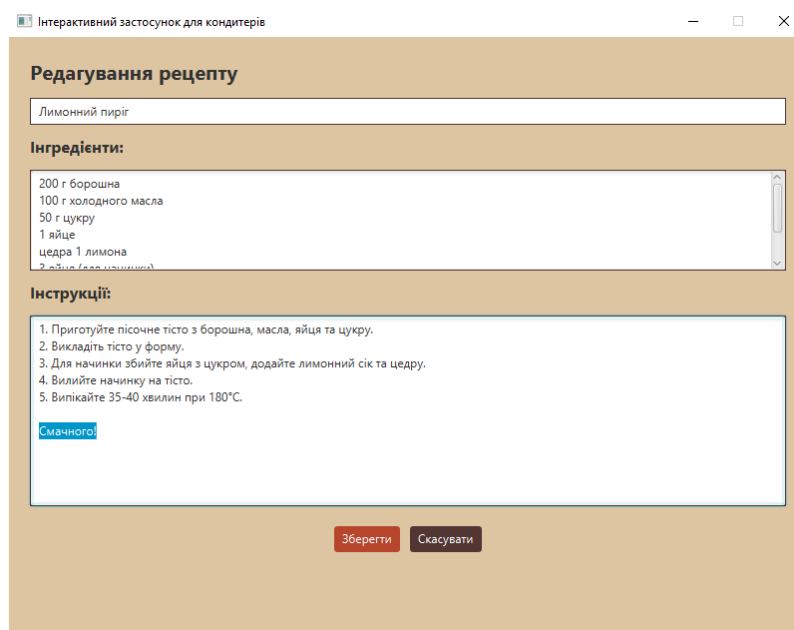


Рисунок Б.6 – Внесення змін до інструкцій при редагуванні рецепту

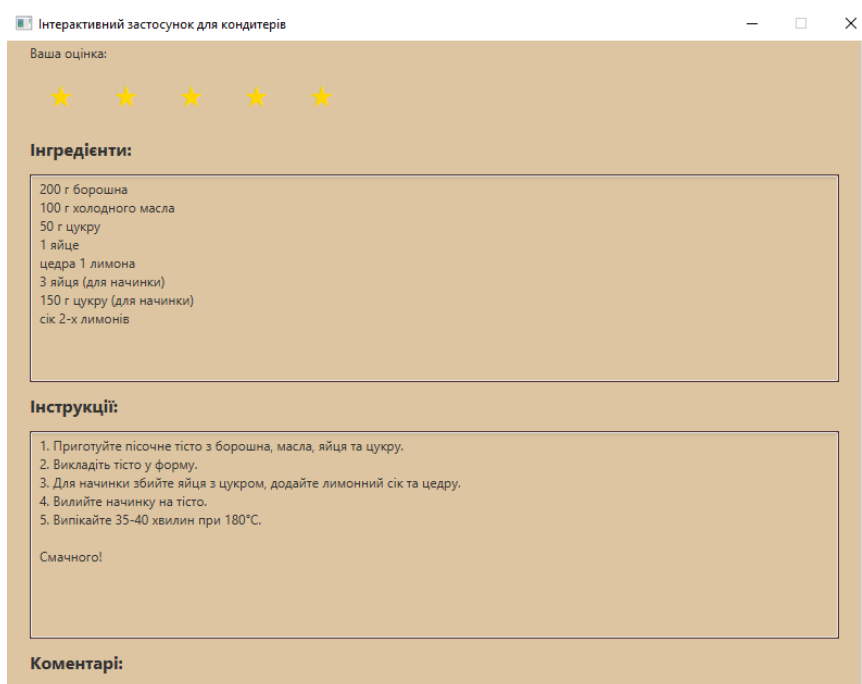


Рисунок Б.7 – Перегляд збережених змін після редагування

Крок 8: Пошук рецептів

Користувач може вводити ключові слова в поле пошуку, щоб знайти рецепти, які містять ці інгредієнти або назви (рисунок Б.8, рисунок Б.9). Програма відправляє запит на сервер, який повертає відфільтрований список рецептів.

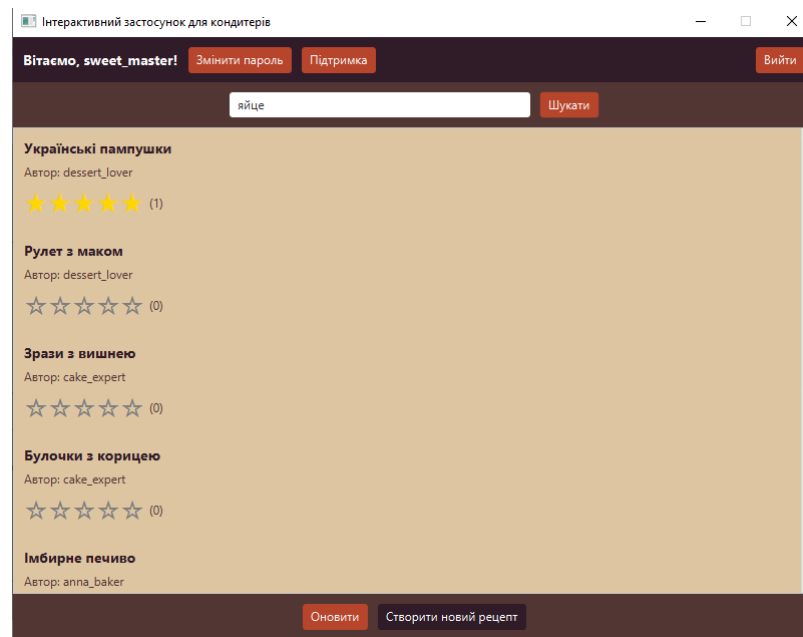


Рисунок Б.8 – Пошук рецептів за словом (складником рецепту)

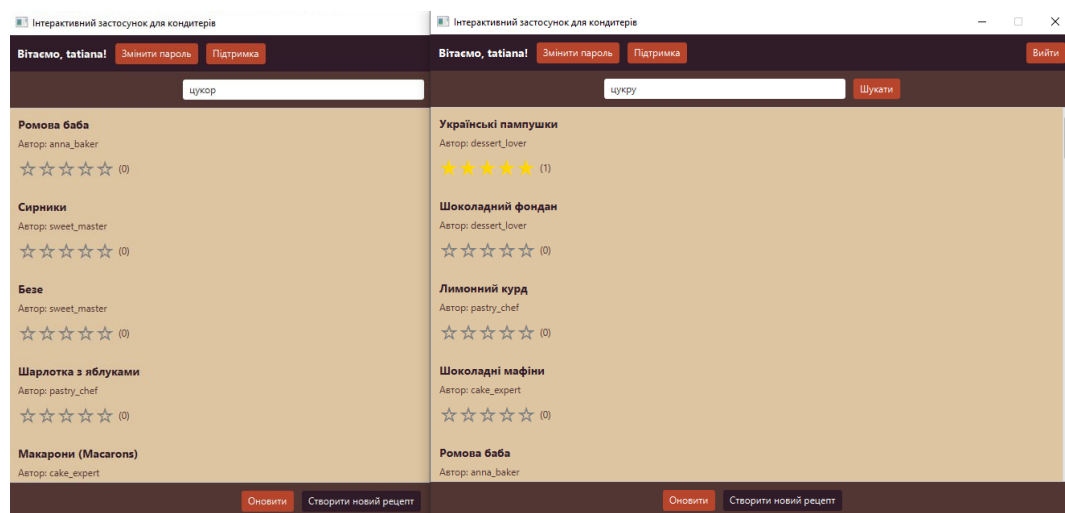


Рисунок Б.9 – Відображення різниці пошуку при відмінюванні слова

Крок 9: Коментування рецептів

Зареєстровані користувачі можуть залишати коментарі до рецептів на сторінці деталей рецепту (рисунок Б.10). Для цього передбачено текстове поле, де користувач вводить свій коментар, і кнопка "Додати коментар". Після натискання кнопки програма відправляє POST-запит на сервер із текстом коментаря та ідентифікатором рецепту. У разі успіху коментар додається до бази даних і відображається у списку коментарів під рецептом у реальному часі (рисунок Б.11). Користувачі також можуть переглядати всі існуючі коментарі

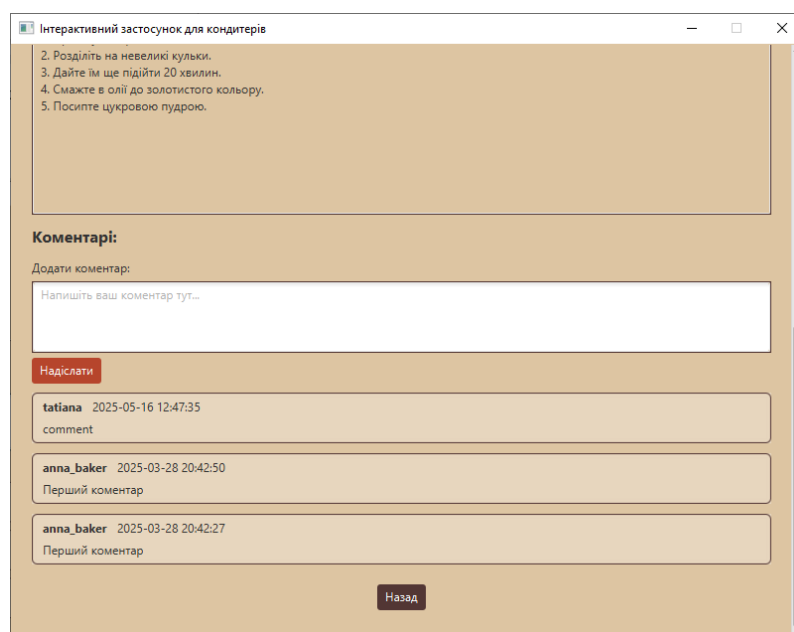
до рецепту, які завантажуються з сервера за допомогою GET-запиту при відкритті сторінки рецепту. Кожен коментар супроводжується іменем автора та датою створення.

Крок 10: Вихід із системи

Користувач може вийти зі свого облікового запису, натиснувши кнопку "Вийти". Після цього токен доступу видаляється, і програма повертається до стартового екрану входу.

Крок 11: Завершення роботи програми

Користувач може закрити вікно програми, що призводить до завершення її роботи. Усі дані зберігаються на сервері, тому при наступному запуску користувач зможе продовжити роботу з того місця, де він зупинився.



Інтерактивний застосунок для кондитерів

2. Розділіть на невеликі кульки.
3. Дайте їм ще підійти 20 хвилин.
4. Смажте в олії до золотистого кольору.
5. Посипте цукровою пудрою.

Коментарі:

Додати коментар:

Напишіть ваш коментар тут...

Надіслати

tatiana 2025-05-16 12:47:35
comment

anna_baker 2025-03-28 20:42:50
Перший коментар

anna_baker 2025-03-28 20:42:27
Перший коментар

Назад

Рисунок Б.10 – Форма для додавання коментаря до рецепту



Коментарі:

Додати коментар:

Напишіть ваш коментар тут...

Надіслати

anna_baker 2025-03-28 20:49:08
Тут коментар

Назад

Рисунок Б.11 – Відображення доданого коментаря під рецептом

ДОДАТОК В
Фрагменти тексту програми

```

public class Server {
    private static final int PORT = 8080;
    private static final String DB_PATH = "confectionery.db";

    public static void main(String[] args) throws Exception {
        // Ініціалізація бази даних
        initDatabase();

        // Створення HTTP сервера
        HttpServer server = HttpServer.create(new
InetSocketAddress(PORT), 0);

        // Реєстрація обробників шляхів
        server.createContext("/api/auth/register", new
RegisterHandler());
        server.createContext("/api/auth/login", new
LoginHandler());
        server.createContext("/api/changePassword", new
ChangePasswordHandler());
        server.createContext("/api/recipes/create", new
CreateRecipeHandler());
        server.createContext("/api/recipes", new
RecipesHandler());
        server.createContext("/api/recipes/search", new
SearchRecipesHandler());
        server.createContext("/api/recipes/comment", new
CommentHandler());
        server.createContext("/api/recipes/rate", new
RatingHandler());

        // Запуск сервера
        server.start();
        System.out.println("[SERVER] Сервер запущено на порту " +
PORT);

        private static void initDatabase() {
            System.out.println("[SERVER] Ініціалізація бази даних");
        }
    }
}

```

```

// Перевіряємо існування БД
boolean isNewDatabase = !new File(DB_PATH).exists();

try (Connection conn = getConnection()) {
    // Включити foreign keys підтримку
    try (Statement stmt = conn.createStatement()) {
        stmt.execute("PRAGMA foreign_keys = ON;");
    }

    // Створення таблиці користувачів
    String createUserTable =
        "CREATE TABLE IF NOT EXISTS users (" +
            "id          INTEGER          PRIMARY          KEY
AUTOINCREMENT," +
            "username TEXT UNIQUE NOT NULL," +
            "password TEXT NOT NULL," +
            "role TEXT NOT NULL)";

    // Створення таблиці рецептів
    String createRecipeTable =
        "CREATE TABLE IF NOT EXISTS recipes (" +
            "id          INTEGER          PRIMARY          KEY
AUTOINCREMENT," +
            "title UNIQUE TEXT NOT NULL," +
            "ingredients TEXT NOT NULL," +
            "instructions TEXT NOT NULL," +
            "user_id INTEGER NOT NULL," +
            "FOREIGN KEY (user_id) REFERENCES
users(id))";

    // Створення таблиці коментарів
    String createCommentTable =
        "CREATE TABLE IF NOT EXISTS comments (" +
            "id          INTEGER          PRIMARY          KEY
AUTOINCREMENT," +

```

```

        "recipe_id INTEGER NOT NULL," +
        "user_id INTEGER NOT NULL," +
        "text TEXT NOT NULL," +
        "created_at      TIMESTAMP      DEFAULT
CURRENT_TIMESTAMP," +
        "FOREIGN KEY (recipe_id) REFERENCES
recipes(id) ON DELETE CASCADE," +
        "FOREIGN KEY (user_id) REFERENCES
users(id))";

```

```

// Створення таблиці оцінок
String createRatingTable =
    "CREATE TABLE IF NOT EXISTS ratings (" +
        "id      INTEGER      PRIMARY      KEY
AUTOINCREMENT," +
        "recipe_id INTEGER NOT NULL," +
        "user_id INTEGER NOT NULL," +
        "rating INTEGER NOT NULL CHECK(rating
BETWEEN 1 AND 5)," +
        "FOREIGN KEY (recipe_id) REFERENCES
recipes(id) ON DELETE CASCADE," +
        "FOREIGN KEY (user_id) REFERENCES
users(id)," +
        "UNIQUE(recipe_id, user_id))";

```

```

try (Statement stmt = conn.createStatement()) {
    stmt.execute(createUserTable);
    stmt.execute(createRecipeTable);
    stmt.execute(createCommentTable);
    stmt.execute(createRatingTable);
    System.out.println("[SERVER]          Таблиці
створено/перевірено успішно");
}

```

```

// Створення тестового користувача тільки для нової
БД

if (isNewDatabase) {

```

```

        // Перевіряємо, чи є вже користувач test
        String checkTestUser = "SELECT COUNT(*) FROM
users WHERE username = ?";

        try (PreparedStatement pstmt =
conn.prepareStatement(checkTestUser)) {
            pstmt.setString(1, "test");
            ResultSet rs = pstmt.executeQuery();
            if (rs.next() && rs.getInt(1) == 0) {
                String insertUser = "INSERT INTO users
(username, password, role) VALUES (?, ?, ?)";
                try (PreparedStatement insertStmt =
conn.prepareStatement(insertUser,
Statement.RETURN_GENERATED_KEYS)) {
                    insertStmt.setString(1, "test");
                    insertStmt.setString(2, "test");
                    insertStmt.setString(3,
"REGISTERED");

                    insertStmt.executeUpdate();

                    try (ResultSet generatedKeys =
insertStmt.getGeneratedKeys()) {
                        if (generatedKeys.next()) {
                            long userId =
generatedKeys.getLong(1);

                            System.out.println("[SERVER]
Створено тестового користувача 'test' з ID: " + userId);
                        }
                    }
                }
            }
        }
    } catch (SQLException e) {
        System.err.println("[SERVER] Помилка при
ініціалізації бази даних: " + e.getMessage());
        e.printStackTrace();
    }
}

```

```

    }
    debugDatabase();
}

private static void debugDatabase() {
    System.out.println("[SERVER] Дебаг бази даних");

    try (Connection conn = getConnection()) {
        // Перевірка структури таблиць
        String showTables = "PRAGMA table_info(users)";
        try (Statement stmt = conn.createStatement()) {
            ResultSet rs = stmt.executeQuery(showTables) {
                System.out.println("[SERVER] Структура таблиці
users:");

                while (rs.next()) {
                    System.out.println("          -      "      +
rs.getString("name") +
                                " (" + rs.getString("type") + ")" +
                                (rs.getInt("pk") == 1 ? " PRIMARY
KEY" : ""));
                }
            }

            showTables = "PRAGMA table_info(recipes)";
            try (Statement stmt = conn.createStatement()) {
                ResultSet rs = stmt.executeQuery(showTables) {
                    System.out.println("[SERVER] Структура таблиці
recipes:");

                    while (rs.next()) {
                        System.out.println("          -      "      +
rs.getString("name") +
                                    " (" + rs.getString("type") + ")" +
                                    (rs.getInt("pk") == 1 ? " PRIMARY
KEY" : ""));
                    }
                }
            }
        }
    }
}

```

```

showTables = "PRAGMA table_info(comments)";
try (Statement stmt = conn.createStatement());
    ResultSet rs = stmt.executeQuery(showTables)) {
    System.out.println("[SERVER] Структура таблиці
comments:");

    while (rs.next()) {
        System.out.println("          -      "      +
rs.getString("name") +
                                " (" + rs.getString("type") + ")" +
                                (rs.getInt("pk") == 1 ? " PRIMARY
KEY" : ""));
    }
}

showTables = "PRAGMA table_info(ratings)";
try (Statement stmt = conn.createStatement());
    ResultSet rs = stmt.executeQuery(showTables)) {
    System.out.println("[SERVER] Структура таблиці
ratings:");

    while (rs.next()) {
        System.out.println("          -      "      +
rs.getString("name") +
                                " (" + rs.getString("type") + ")" +
                                (rs.getInt("pk") == 1 ? " PRIMARY
KEY" : ""));
    }
}

// Виведення всіх користувачів
String selectUsers = "SELECT * FROM users";
try (Statement stmt = conn.createStatement());
    ResultSet rs = stmt.executeQuery(selectUsers))
{
    System.out.println("[SERVER]          Список
користувачів:");

```

```

        while (rs.next()) {
            if (rs.getInt("id") != 0 ||
rs.getString("username").isEmpty())
                System.out.println("      - ID: " +
rs.getInt("id") +
                ",      Ім'я: " +
rs.getString("username") +
                ",      Роль: " +
rs.getString("role"));
        }
    }

    // Виведення всіх рецептів
    String selectRecipes = "SELECT r.*, u.username FROM
recipes r JOIN users u ON r.user_id = u.id";
    try (Statement stmt = conn.createStatement();
        ResultSet rs =
stmt.executeQuery(selectRecipes)) {
        System.out.println("[SERVER] Список рецептів:");
        int count = 0;
        while (rs.next()) {
            count++;
            System.out.println("      - ID: " +
rs.getInt("id") +
                ", Назва: " + rs.getString("title")
+
                ", Автор ID: " + rs.getInt("user_id")
+
                ",      Автор: " +
rs.getString("username"));
        }
        if (count == 0) {
            System.out.println("  Рецептів не знайдено");
        }
    }
}

```

```

        } catch (SQLException e) {
            System.err.println("[SERVER] Помилка при дебагу бази
даних: " + e.getMessage());
            e.printStackTrace();
        }
    }

    private static void sendResponse(HttpExchange exchange, int
statusCode, String response) throws IOException {
        exchange.getResponseHeaders().set("Content-Type",
"application/json; charset=UTF-8");
        exchange.getResponseHeaders().add("Access-Control-Allow-
Origin", "*");
        exchange.getResponseHeaders().add("Access-Control-Allow-
Methods", "GET, POST, PUT, DELETE, OPTIONS");
        exchange.getResponseHeaders().add("Access-Control-Allow-
Headers", "Content-Type, Authorization");

        byte[] responseBytes =
response.getBytes(StandardCharsets.UTF_8);
        exchange.sendResponseHeaders(statusCode,
responseBytes.length);

        try (OutputStream os = exchange.getResponseBody()) {
            os.write(responseBytes);
        }
    }

    private static String readRequestBody(HttpExchange exchange)
throws IOException {
        try (InputStream inputStream = exchange.getRequestBody();
            BufferedReader reader = new BufferedReader(new
InputStreamReader(inputStream, StandardCharsets.UTF_8))) {

            StringBuilder requestBody = new StringBuilder();
            String line;
            while ((line = reader.readLine()) != null) {

```

```

        requestBody.append(line);
    }
    return requestBody.toString();
}

// Обробник OPTIONS запитів для CORS
private static void handleOptions(HttpExchange exchange)
throws IOException {
    exchange.getResponseHeaders().add("Access-Control-Allow-
Origin", "*");
    exchange.getResponseHeaders().add("Access-Control-Allow-
Methods", "GET, POST, PUT, DELETE, OPTIONS");
    exchange.getResponseHeaders().add("Access-Control-Allow-
Headers", "Content-Type, Authorization");
    exchange.sendResponseHeaders(204, -1);
}

// Обробник реєстрації
static class RegisterHandler implements HttpHandler {
    if (exchange.getRequestMethod().equals("OPTIONS")) {
        handleOptions(exchange);
        return;
    }
    try {
        String requestBody = readRequestBody(exchange);
        System.out.println("[SERVER] Запит на реєстрацію: " +
requestBody);

        Map<String, String> requestData =
objectMapper.readValue(requestBody, Map.class);
        String username = requestData.get("username");
        String password = requestData.get("password");

        if (username == "" || password == "") {
            sendResponse(exchange, 400, "{\"error\":\"Логін та
пароль обов'язкові\"}");

```

```

        System.out.println("[SERVER] Запит на реєстрацію  

скасовано");
        return;
    }

    if (password.length() <= 5) {
        sendResponse(exchange, 400, "{\"error\": \"Пароль не  

може бути менше 6 символів\"}");
        System.out.println("[SERVER] Запит на реєстрацію  

скасовано");
        return;
    }

    try (Connection conn = getConnection()) {
        // Перевірка, чи існує користувач
        String checkUser = "SELECT COUNT(*) FROM  

users WHERE username = ?";
        try (PreparedStatement pstmt =  

conn.prepareStatement(checkUser)) {
            pstmt.setString(1, username);
            ResultSet rs = pstmt.executeQuery();
            if (rs.next() && rs.getInt(1) > 0) {
                sendResponse(exchange, 400,  

"{\"error\": \"Користувач з таким ім'ям вже існує\"}");
                return;
            }
        }

        // Додавання користувача
        String insertUser = "INSERT INTO users  

(username, password, role) VALUES (?, ?, ?)";
        long userId;
        try (PreparedStatement pstmt =  

conn.prepareStatement(insertUser,  

Statement.RETURN_GENERATED_KEYS)) {
            pstmt.setString(1, username);
            pstmt.setString(2, password);

```

```

        pstmt.setString(3, "REGISTERED");
        pstmt.executeUpdate();

        try        (ResultSet        generatedKeys        =
pstmt.getGeneratedKeys()) {
            if (generatedKeys.next()) {
                userId        =
generatedKeys.getLong(1);

                System.out.println("[SERVER]
Користувач зареєстрований з ID: " + userId);
            } else {
                throw        new        SQLException("Не
вдалося отримати ID для нового користувача");
            }
        }

        // Відповідь з токеном (простий токен - ім'я
користувача)

        Map<String,        Object>        responseData        =        new
HashMap<>();

        responseData.put("token", username);
        responseData.put("username", username);
        responseData.put("role", "REGISTERED");

        sendResponse(exchange,        200,
objectMapper.writeValueAsString(responseData));

    } catch (SQLException e) {
        System.err.println("[SERVER] Помилка SQL при
реєстрації: " + e.getMessage());
        e.printStackTrace();
        sendResponse(exchange,        500,
"{\"error\": \"Внутрішня помилка сервера\"}");
    }

```

```

        } catch (Exception e) {
            System.err.println("[SERVER]      Помилка      при
реєстрації: " + e.getMessage());
            e.printStackTrace();
            sendResponse(exchange, 400, "{\"error\":\"" +
e.getMessage() + "\"}");
        }
    }
}

// Метод для обробки пустого пошукового запиту - повертає всі
рецепти
private void handleGetAllRecipes(HttpExchange exchange)
throws IOException {
    try {
        System.out.println("[SERVER] Запит на отримання
списку рецептів");

        try (Connection conn = getConnection()) {
            // Включаємо середній рейтинг в запит
            String getRecipes =
                "SELECT      r.id,      r.title,
r.ingredients, r.instructions, u.username, " +
                "(SELECT  AVG(rating)  FROM
ratings rt WHERE rt.recipe_id = r.id) as avg_rating, " +
                "(SELECT  COUNT(id)    FROM
ratings rt WHERE rt.recipe_id = r.id) as rating_count " +
                "FROM recipes r JOIN users u
ON r.user_id = u.id " +
                "ORDER BY r.id DESC";

            System.out.println("[SERVER] SQL запит для
отримання рецептів: " + getRecipes);

            List<Map<String, Object>> recipesList = new
ArrayList<>();

```

```

        try (Statement stmt = conn.createStatement();
              ResultSet      rs      =
stmt.executeQuery(getRecipes)) {

            int count = 0;
            while (rs.next()) {
                count++;
                Map<String, Object> recipe = new
HashMap<>();

                recipe.put("id", rs.getLong("id"));
                recipe.put("title",
rs.getString("title"));
                recipe.put("ingredients",
rs.getString("ingredients"));
                recipe.put("instructions",
rs.getString("instructions"));
                recipe.put("authorUsername",
rs.getString("username"));

                // Додавання рейтингу
                double      avgRating      =
rs.getDouble("avg_rating");

                if (rs.isNull()) avgRating = 0;
                recipe.put("averageRating",
avgRating);

                recipe.put("ratingCount",
rs.getInt("rating_count"));

                recipesList.add(recipe);
            }
            System.out.println("[SERVER]    Кількість
рядків у результаті запиту: " + count);
        }

        System.out.println("[SERVER]    Знайдено    " +
recipesList.size() + " рецептів");

```

```

        sendResponse(exchange, 200,
objectMapper.writeValueAsString(recipesList));

        } catch (SQLException e) {
            System.err.println("[SERVER] Помилка SQL при
отриманні рецептів: " + e.getMessage());
            e.printStackTrace();
            sendResponse(exchange, 500,
"{\"error\":\"Внутрішня помилка сервера\"}");
        }

        } catch (Exception e) {
            System.err.println("[SERVER] Помилка при
отриманні рецептів: " + e.getMessage());
            e.printStackTrace();
            sendResponse(exchange, 400, "{\"error\":\"" +
e.getMessage() + "\"}");
        }
    }

    }

    public class Client extends Application {

        // Константи
        private static final String BASE_URL =
"http://localhost:8080/api";
        private static final String AUTH_URL = BASE_URL + "/auth";
        private static final String RECIPES_URL = BASE_URL +
"/recipes";
        private static final String SEARCH_URL = RECIPES_URL +
"/search";

        // Колірна схема застосунку
        private static final String COLOR_COCOA = "#301B28";
        private static final String COLOR_CHOCOLATE = "#523634";

```

```

private static final String COLOR_TOFFEE = "#B6452C";
private static final String COLOR_GLAZE = "#DDC5A2";

// Стан користувача
private String token = null;
private String username = null;
private UserRole role = UserRole.GUEST;

public void start(Stage primaryStage) {
    System.out.println("[CLIENT] Запуск клієнта");
    this.primaryStage = primaryStage;

    // Встановлення заголовка
    primaryStage.setTitle("Інтерактивний застосунок для кондитерів");

    // Створення головного макету
    mainLayout = new BorderPane();

    // Застосування стилів для головного макету
    mainLayout.setStyle("-fx-background-color: " + COLOR_GLAZE + ";");

    // Показ стартового екрану
    showLoginScreen();

    // Налаштування і показ основного вікна
    Scene scene = new Scene(mainLayout, 800, 600);
    primaryStage.setScene(scene);
    primaryStage.show();
    System.out.println("[CLIENT] Головне вікно відображено");
}

private void login(String username, String password) throws
Exception {

```

```

        System.out.println("[CLIENT] Відправка запиту на вхід для
користувача: " + username);

        Map<String, String> requestData = new HashMap<>();
        requestData.put("username", username);
        requestData.put("password", password);

        String response = sendPostRequest(AUTH_URL + "/login",
requestData);

        System.out.println("[CLIENT] Отримано відповідь: " +
response);

        Map<String, Object> responseMap =
objectMapper.readValue(response, Map.class);

        this.token = (String) responseMap.get("token");
        this.username = (String) responseMap.get("username");
        this.role = UserRole.valueOf((String)
responseMap.get("role"));

        System.out.println("[CLIENT] Результат входу: token=" +
this.token + ", username=" + this.username + ", role=" + this.role);

        private void register(String username, String password)
throws Exception {

            System.out.println("[CLIENT] Відправка запиту на
реєстрацію для користувача: " + username);

            Map<String, String> requestData = new HashMap<>();
            requestData.put("username", username);
            requestData.put("password", password);

            String response = sendPostRequest(AUTH_URL + "/register",
requestData);

            System.out.println("[CLIENT] Отримано відповідь: " +
response);

            Map<String, Object> responseMap =
objectMapper.readValue(response, Map.class);

```

```

        this.token = (String) responseMap.get("token");
        this.username = (String) responseMap.get("username");
        this.role = UserRole.valueOf((String)
responseMap.get("role"));

        System.out.println("[CLIENT] Результат реєстрації:
token=" + this.token + ", username=" + this.username + ", role=" +
this.role);
    }

    private void loadRecipes() {
        try {
            System.out.println("[CLIENT] Завантаження списку
рецептів");

            String response = sendGetRequest(RECIPES_URL);
            System.out.println("[CLIENT] Отримано відповідь від
сервера, розмір: " + response.length() + " байт");

            List<Map<String, Object>> recipesList =
objectMapper.readValue(response, List.class);

            System.out.println("[CLIENT] Кількість завантажених
рецептів: " + recipesList.size());

            this.recipesList.clear();

            for (Map<String, Object> map : recipesList) {
                RecipeDTO recipe = new RecipeDTO();

                recipe.setId(Long.valueOf(map.get("id").toString()));
                recipe.setTitle((String) map.get("title"));
                recipe.setIngredients((String)
map.get("ingredients"));
                recipe.setInstructions((String)
map.get("instructions"));
                recipe.setAuthorUsername((String)
map.get("authorUsername"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }

```

```

        // Додаємо дані про рейтинг
        if (map.containsKey("averageRating")) {
            double                avgRating                =
Double.parseDouble(map.get("averageRating").toString());
            recipe.setAverageRating(avgRating);
        } else {
            recipe.setAverageRating(0.0);
        }

        if (map.containsKey("ratingCount")) {
            int                    ratingCount                =
Integer.parseInt(map.get("ratingCount").toString());
            recipe.setRatingCount(ratingCount);
        } else {
            recipe.setRatingCount(0);
        }
        this.recipesList.add(recipe);
    }
} catch (Exception e) {
    System.err.println("[CLIENT] Помилка завантаження
рецептів: " + e.getMessage());
    e.printStackTrace();
    showError("Помилка завантаження рецептів: " +
e.getMessage());
}
}

```

```

public class Seed {
    private static final String DB_PATH = "confectionery.db";

    // Дані тестових користувачів
    private static final String[][] USERS = {
        {"anna_baker", "password123", "REGISTERED"},
        {"sweet_master", "confection", "REGISTERED"},
        {"pastry_chef", "bakery456", "REGISTERED"},
    }
}

```

```

        {"dessert_lover", "cookietime", "REGISTERED"},
        {"cake_expert", "frosting789", "REGISTERED"}
    };

    public static void main(String[] args) {
        // Перевірка існування файлу БД
        if (new File(DB_PATH).exists()) {
            // Файл БД вже існує
            System.out.println("[SEED] База даних вже існує. Ви впевнені, що хочете її перезавантажити?");
            System.out.println("[SEED] Натисніть 'y' для продовження або будь-яку іншу клавішу для виходу.");

            Scanner scanner = new Scanner(System.in);
            String input = scanner.nextLine();

            if (!input.equalsIgnoreCase("y")) {
                System.out.println("[SEED] Процес скасовано. База даних не змінена.");
                return;
            }

            // Видалення існуючого файлу БД
            boolean deleted = new File(DB_PATH).delete();
            if (!deleted) {
                System.out.println("[SEED] Не вдалося видалити існуючу базу даних. Можливо вона використовується іншим процесом.");
                return;
            }
        }

        // Метод для виведення статистики про створену БД
        private static void printStats(Connection conn) throws SQLException {
            System.out.println("\n[SEED] Статистика бази даних:");

```

```

    try (Statement stmt = conn.createStatement()) {
        // Кількість користувачів
        try (ResultSet rs = stmt.executeQuery("SELECT COUNT(*) as
count FROM users")) {
            if (rs.next()) {
                System.out.println("Користувачів:      "      +
rs.getInt("count"));
            }
        }

        // Кількість рецептів
        try (ResultSet rs = stmt.executeQuery("SELECT COUNT(*) as
count FROM recipes")) {
            if (rs.next()) {
                System.out.println("Рецептів:      "      +
rs.getInt("count"));
            }
        }

        // Кількість унікальних інгредієнтів
        try (ResultSet rs = stmt.executeQuery("SELECT COUNT(*) as
count FROM ingredients")) {
            if (rs.next()) {
                System.out.println("Унікальних інгредієнтів: " +
rs.getInt("count"));
            }
        }

        // Рецепти по користувачах
        try (ResultSet rs = stmt.executeQuery(
            "SELECT u.username, COUNT(r.id) as recipe_count "
+
            "FROM users u " +
            "LEFT JOIN recipes r ON u.id = r.user_id "
+

```

```

        "GROUP BY u.id " +
        "ORDER BY recipe_count DESC")) {
    System.out.println("\nРозподіл        рецептов        по
користувачах:");
    while (rs.next()) {
        System.out.println(rs.getString("username") + ": "
+ rs.getInt("recipe_count") + " рецептов");
    }
}

// Топ-10 найпопулярніших інгредієнтів
try (ResultSet rs = stmt.executeQuery(
    "SELECT        i.name,        COUNT(ri.recipe_id)        as
recipe_count " +
        "FROM ingredients i " +
        "JOIN  recipe_ingredients ri  ON  i.id =
ri.ingredient_id " +
        "GROUP BY i.id " +
        "ORDER BY recipe_count DESC " +
        "LIMIT 10")) {
    System.out.println("\nТоп-10        найпопулярніших
інгредієнтів:");
    while (rs.next()) {
        System.out.println(rs.getString("name") + ": " +
rs.getInt("recipe_count") + " рецептов");
    }
}
}

```

ДОДАТОК Г

Uml-діаграми



Рисунок Г.1 – Діаграма компонентів

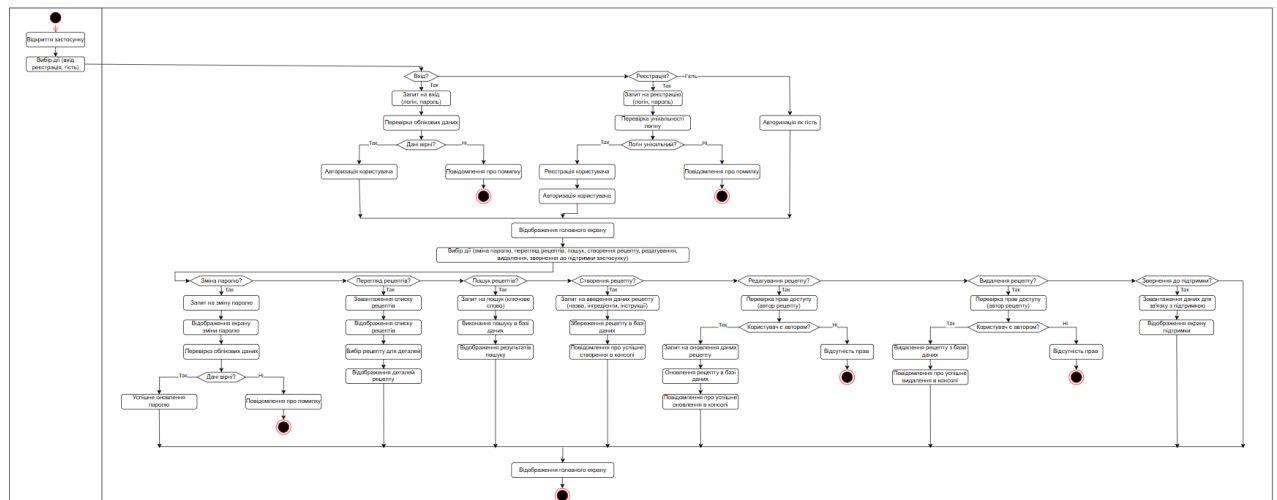


Рисунок Г.2 – Діаграма діяльності

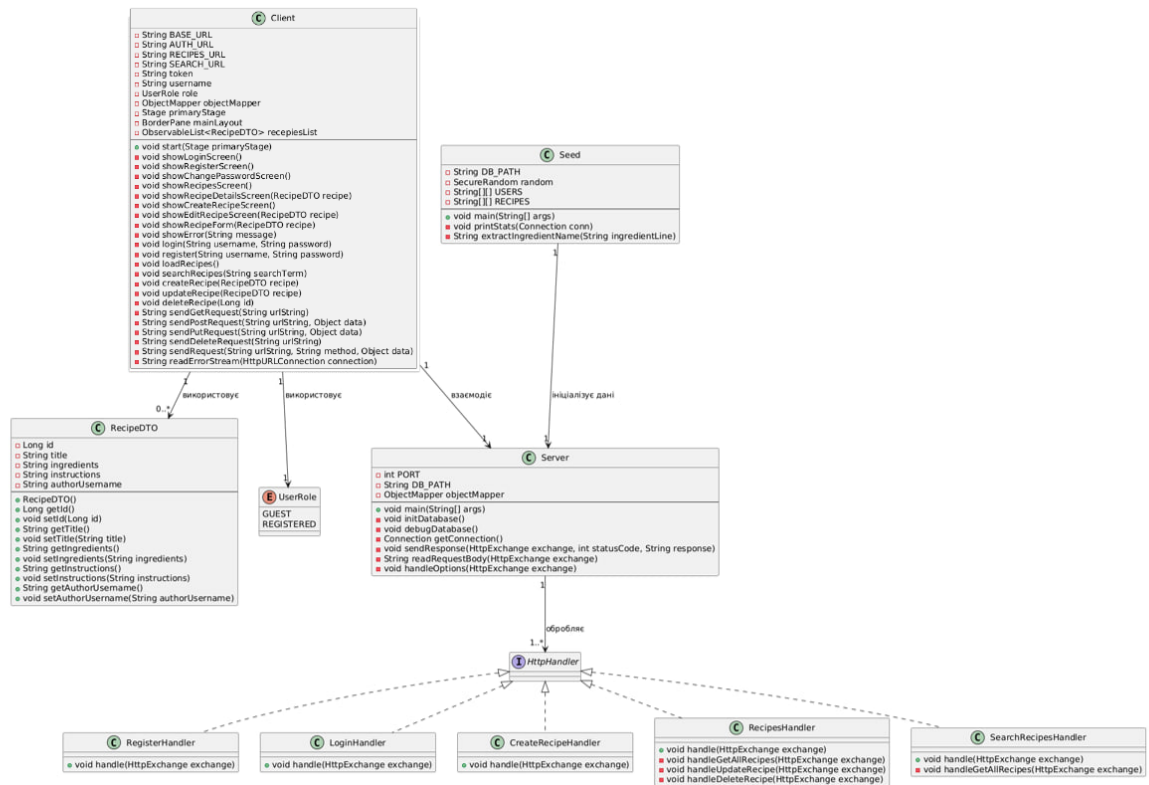


Рисунок Г.3 – Діаграма класів

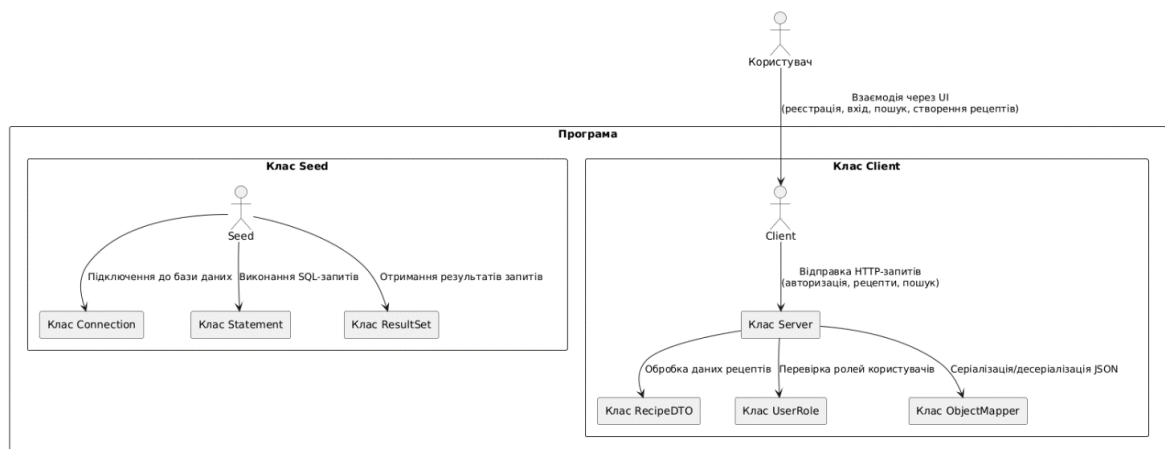


Рисунок Г.4 – Діаграма кооперацій

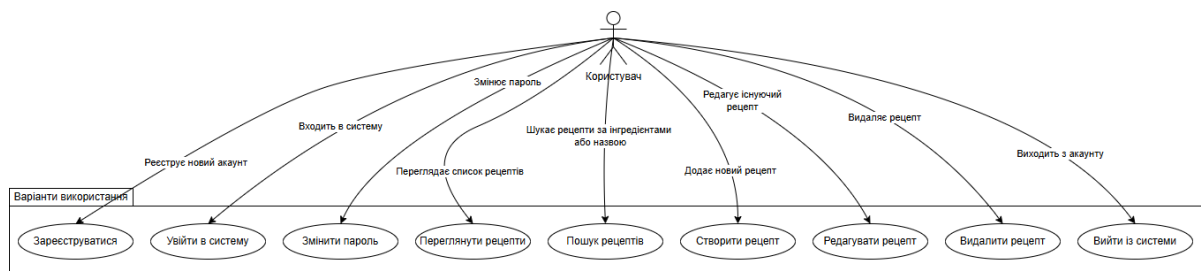


Рисунок Г.5 – Діаграма прецедентів (варіантів використання)



Рисунок Г.6 – Діаграма послідовності

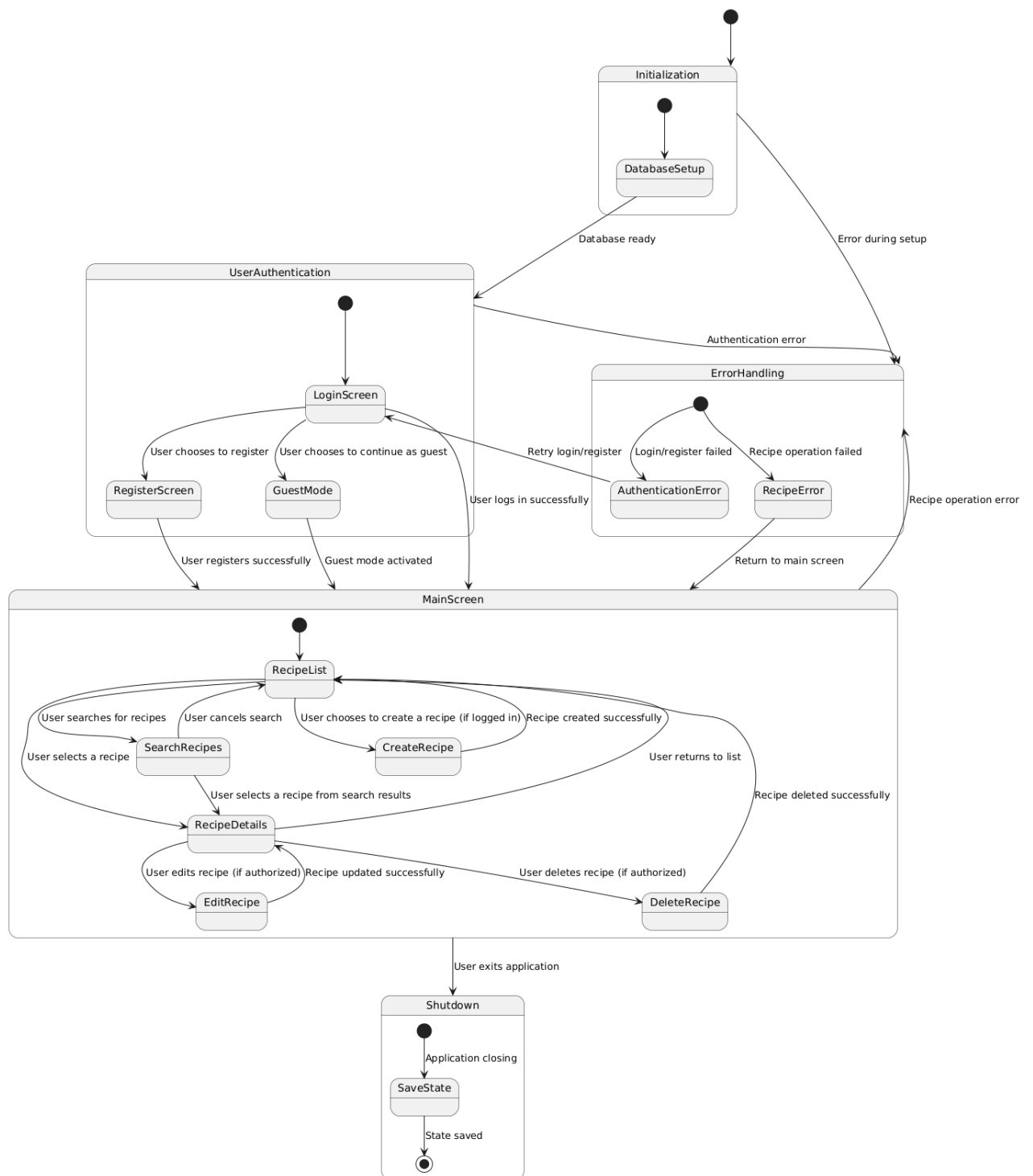


Рисунок Г.7 – Діаграма станів

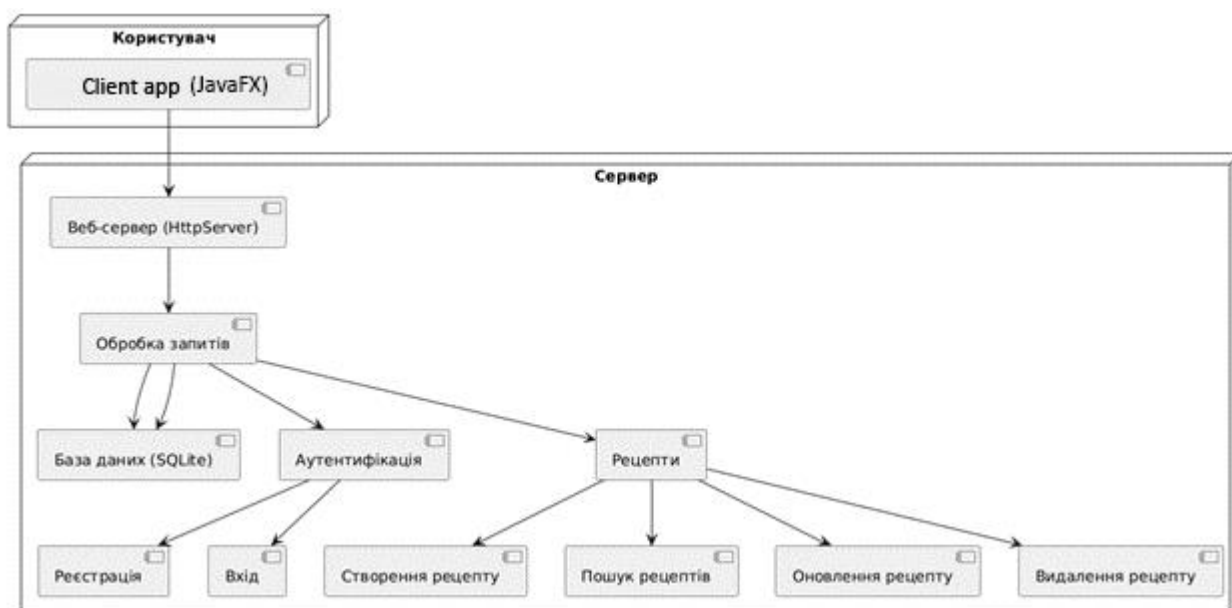


Рисунок Г.8 – Діаграма розгортання

ДОДАТОК Д
Слайди презентації

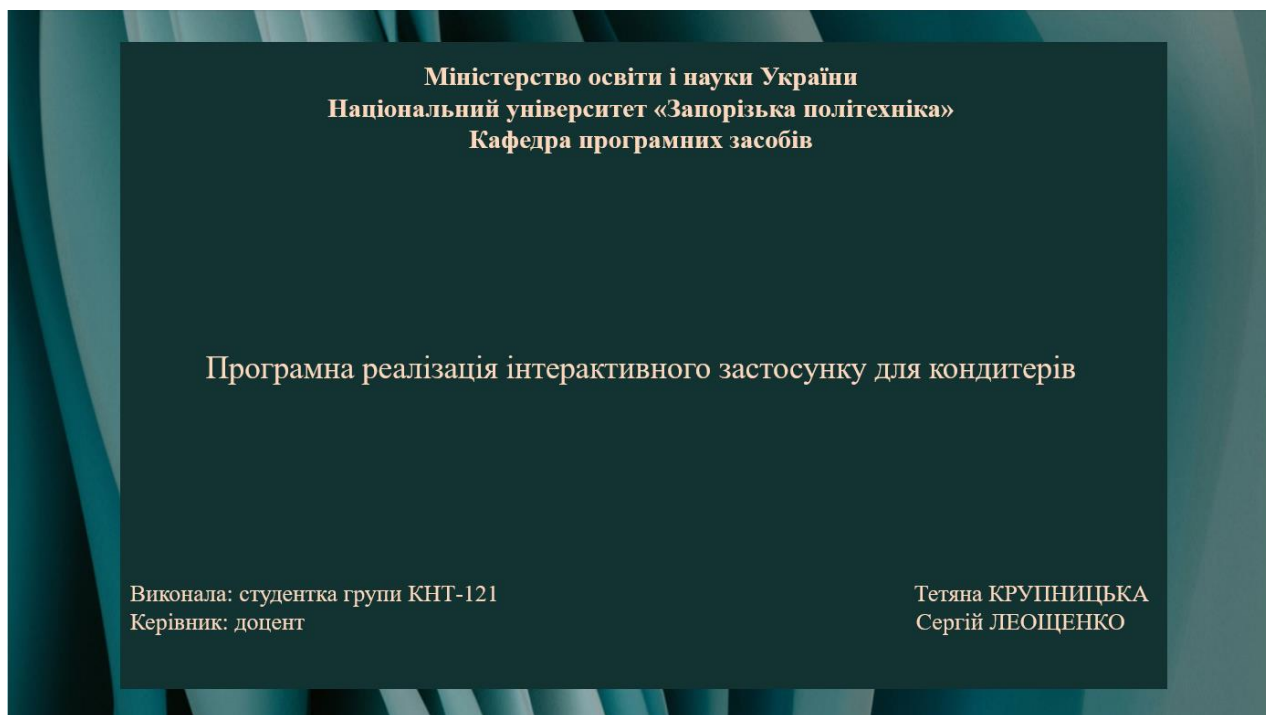


Рисунок Д.1 – Слайд 1

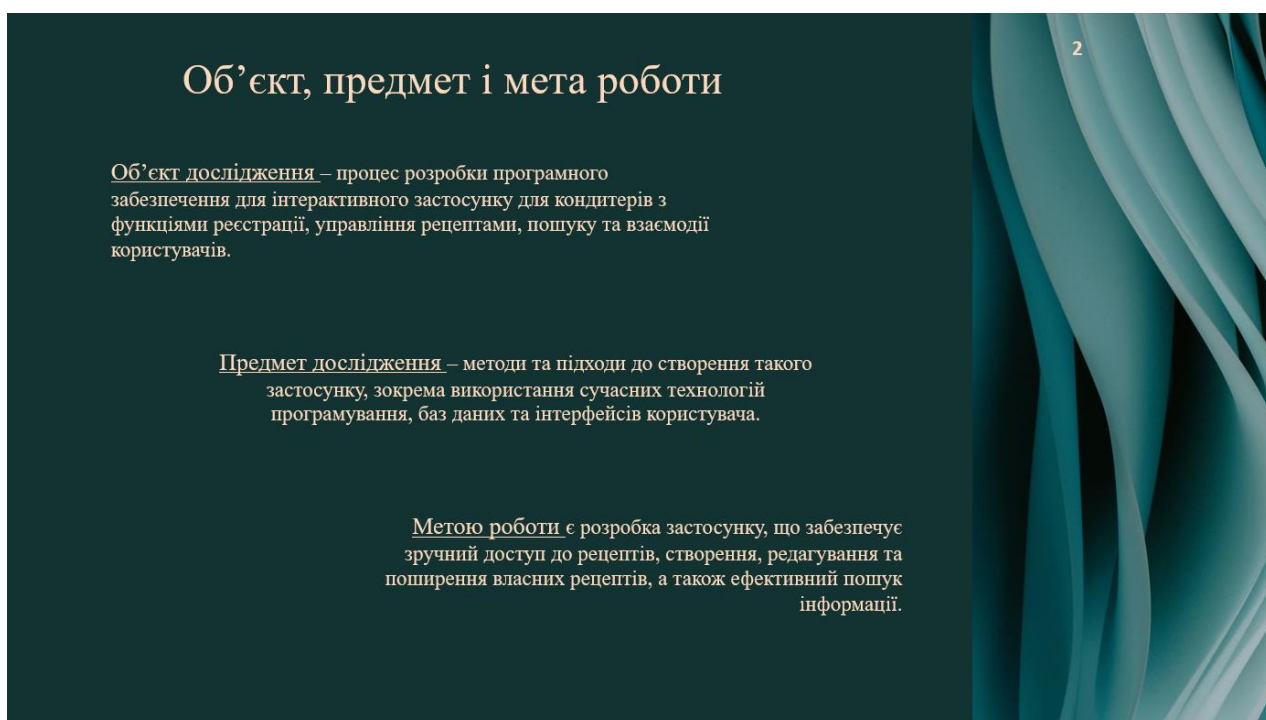


Рисунок Д.2 – Слайд 2

3

Завдання до роботи

Для досягнення поставленої мети у кваліфікаційній роботі необхідно виконати наступні пункти:

- ▶ - виконати аналіз предметної області та застосунків для кондитерської справи;
- ▶ - виконати проєктування програмного забезпечення для інтерактивного застосунку для кондитерів;
- ▶ - реалізувати програмне забезпечення для інтерактивного застосунку для кондитерів;
- ▶ - виконати тестування розробленого програмного забезпечення інтерактивного застосунку для кондитерів.

Рисунок Д.3 – Слайд 3

4

Порівняння програмного забезпечення для кондитерської справи

Функція	Tasty	Kitchen Stories	Desert Recipes	Great British Chefs	Cake Master	Dr. Oetker Rezeptideas	Betty Crocker Cookbook
Покрокові інструкції	Так	Так	Так	Так	Так	Так	Так
Відеоуроки	Так	Так	Ні	Так	Так	Так	Так
Створення власної кулінарної книги	Ні	Так	Ні	Ні	Ні	Ні	Так
Інтерактивні майстер-класи	Так	Так	Ні	Ні	Так	Так	Ні
3D-моделі для декорування	Ні	Ні	Ні	Ні	Так	Ні	Ні
Віртуальний дизайн	Ні	Ні	Ні	Ні	Так	Ні	Ні
Спільнота	Так	Так	Так	Так	Так	Так	Так
Професійні поради	Так	Так	Так	Так	Так	Так	Так

Рисунок Д.4 – Слайд 4

5

Вимоги до програми

При розробці інтерактивного застосунку для кондитерів треба забезпечити наступні функціональні вимоги:

- розробити реєстрацію та автентифікацію користувачів;
- можливість управління рецептами: створення, редагування, видалення, перегляд;
- пошук рецептів за назвою або інгредієнтами;
- зручний інтерфейс у застосунку;
- захист даних користувачів за допомогою токенів та обмежений доступ до функцій для гостьового режиму;
- можливість змінити пароль за потреби;
- коментування рецептів та оцінка для авторизованих користувачів.

Рисунок Д.5 – Слайд 5

6

Обґрунтування вибору мови програмування

Критерій порівняння мов програмування	Java	C++	C#
Кросплатформенність	Висока (працює всюди, де є JVM)	Середня (залежить від компілятора)	Середня
ООП	Чистий ООП	Мультипарадигмовий	Чистий ООП
Безпека	Висока (керована пам'ять)	Низька (ручне керування пам'яттю)	Висока (керована пам'ять)
Стандартна бібліотека	Дуже багата	Базова	Багата
Підтримка сучасних технологій	Хороша	Відмінна	Відмінна
Створення інтерфейсу	Swing, JavaFX	Qt, ImGui (потребує сторонні бібліотеки)	WPF (Windows), MAUI, Avalonia
Спільнота та підтримка	Велика спільнота та хороша документація	Велика спільнота, але фрагментована	Велика спільнота

Рисунок Д.6 – Слайд 6

Обґрунтування вибору середовища розробки

Критерій порівняння середовищ розробки	IntelliJ IDEA	Eclipse	NetBeans
Підтримка Java	Підтримує останні версії JDK	Підтримує сучасні версії Java, але може вимагати додаткового налаштування	Підтримує сучасні версії Java, але може відставати від IntelliJ IDEA у підтримці нових функцій
Інтеграція з JavaFX	Зручна інтеграція інструментів для розробки	Менш зручна інтеграція, порівняно з IntelliJ IDEA	Вбудована підтримка, але інтерфейс менш зручний
Робота з базами даних	Вбудована підтримка роботи з базами даних, включаючи SQLite	Має плагіни для роботи з базами даних, але їх налаштування може бути більш складним	Надає інструменти для роботи з базами даних, але вони можуть поступатися IntelliJ IDEA
Підтримка бібліотек та фреймворків	Підтримує легку інтеграцію з бібліотеками	Підтримує інтеграцію з бібліотеками, але процес налаштування може бути складнішим	Підтримує інтеграцію з бібліотеками, але процес може бути менш автоматизованим
Інструменти для тестування	Надає зручні інструменти для написання та запуску тестів	Надає менш інтуїтивно зрозумілі інструменти для тестування, порівняно з IntelliJ IDEA	Надає базові інструменти для тестування, але вони можуть бути менш функціональними
Кросплатформенність	IDE доступне для Windows, macOS та Linux	Eclipse доступний для різних платформ, але може мати проблеми зі стабільністю на деяких системах	Доступний для різних платформ, але може мати проблеми з продуктивністю
Документація та спільнота	Має детальну документацію та активну спільноту	Має активну спільноту, але документація може бути менш структурованою	Має документацію, але спільнота може бути менш активною

Рисунок Д.7 – Слайд 7

Структурна схема застосунку

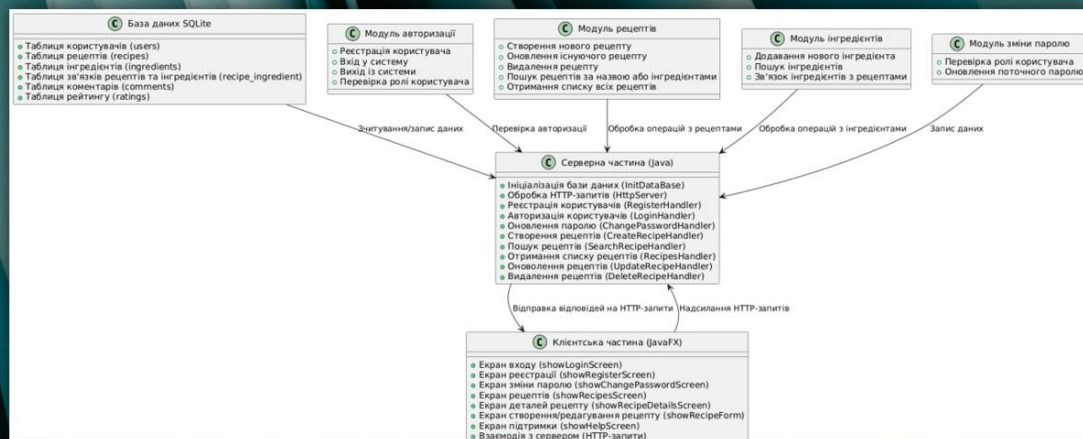


Рисунок Д.8 – Слайд 8

11

Робота з програмою. Вікно авторизації та головне

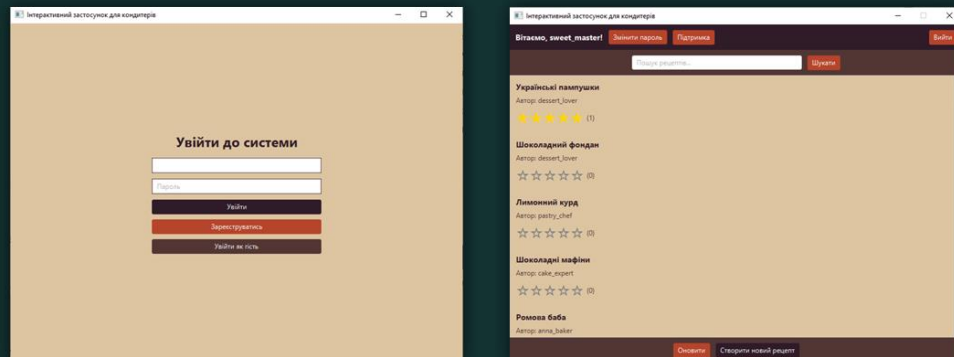


Рисунок Д.11 – Слайд 11

12

Робота з програмою. Перегляд деталей рецепту, редагування, створення нового

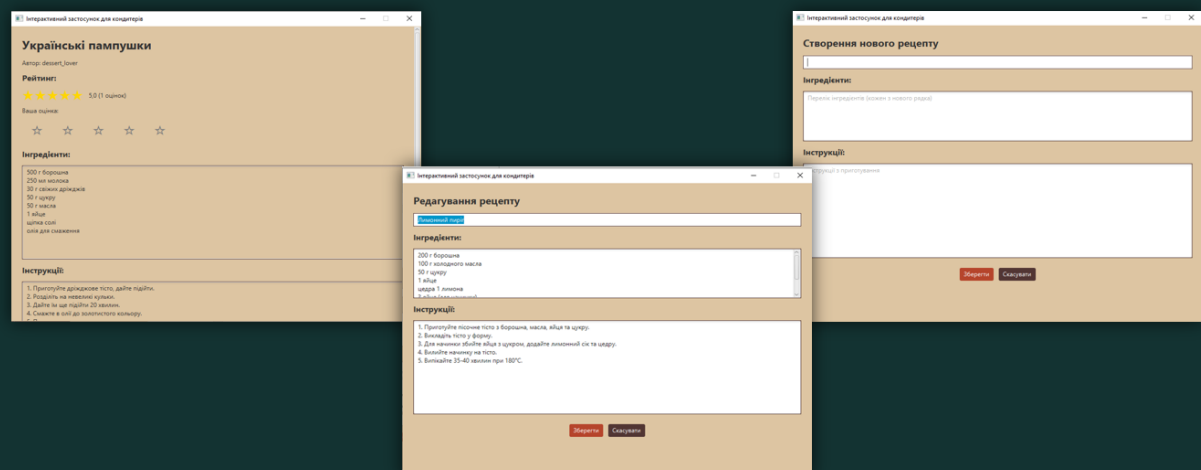


Рисунок Д.12 – Слайд 12

13

Робота з програмою. Пошук за категорією, інгредієнтом, додавання коментарів

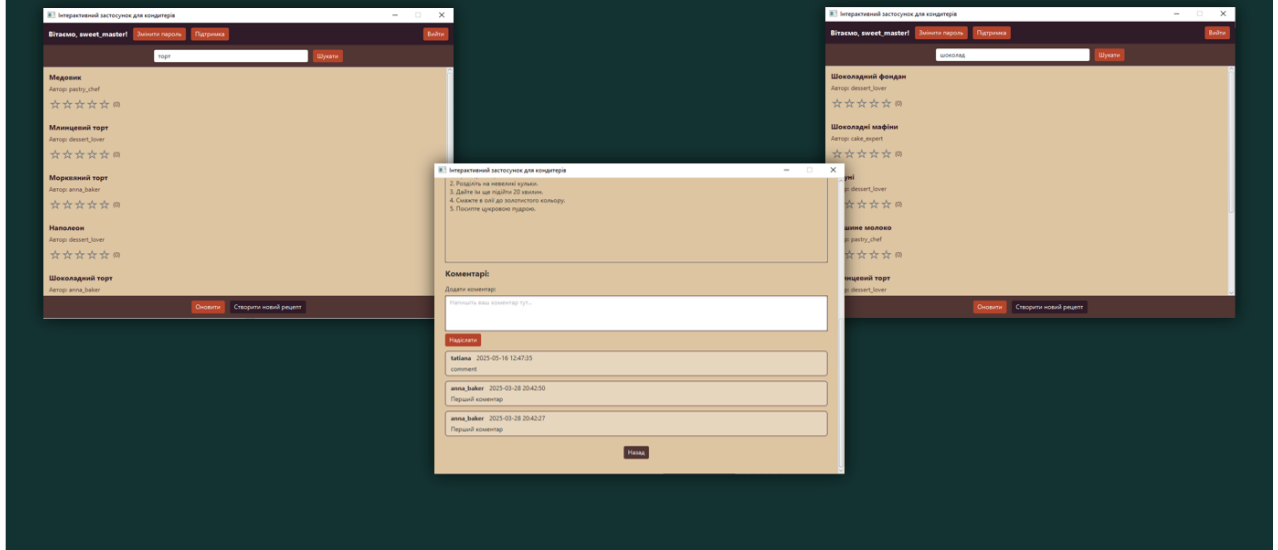


Рисунок Д.13 – Слайд 13

14

Висновки

У процесі виконання дипломної роботи було проаналізовано стадії розробки програмного забезпечення для інтерактивного кондитерського застосунку.

Після аналізу доступних мов програмування та середовищ для розробки даного застосунку було обрано Java та IntelliJ IDEA відповідно. Архітектура програмного забезпечення поєднує серверну частину з клієнтським інтерфейсом. Серверна частина, реалізована за допомогою вбудованого HTTP-сервера, забезпечує зберігання даних у SQLite базі даних, що дозволяє ефективно керувати рецептами та користувачами. Клієнтська частина, розроблена з використанням JavaFX, надає інтуїтивно зрозумілий інтерфейс, який дозволяє користувачам легко взаємодіяти із застосунком.

Тестування застосунку підтвердило його стабільність та ефективність. Було проведено комплексне тестування основних функцій, таких як реєстрація, авторизація, створення рецептів, пошук, коментування. Результати тестування показали, що застосунок коректно працює в різних умовах та забезпечує високу швидкість обробки запитів.

Рисунок Д.14 – Слайд 14