МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ІМ. Ю.М. ПОТЕБНІ
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ
КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему **Особливості використання мови програмування Golang для побудови вебсайту на прикладі тематичного блогу**

Виконала: студентка 2 курсу, групи 8.1212-іпз-2
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Інженерія програмного забезпечення
(код і назва освітньої програми)
В.С. Оглобліна
(ініціали та прізвище)
Керівник доцент, к.т.н., О.М.Міхайлуца
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ «Дісітел»
П.О.Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

Кафедра Електроніки, інформаційних систем та програмного забезпечення

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)

Освітня програма ___Інженерія програмного забезпечення_____
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри_____Т.В. Критська
“_01_” __жовтня_____2023_року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Оглобліній Валерії Сергіївні
(прізвище, ім’я, по батькові)

1.Тема роботи __Особливості використання мови програмування Golang для побудови вебсайту на прикладі тематичного блогу_____

керівник роботи ___Міхайлуца Олена Миколаївна, доцент, к.т.н.
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від 09.10.2023 р. № 1577-с

2. Строк подання студентом кваліфікаційної роботи 1 березня 2024 р.

3. Вихідні дані магістерської роботи

- комплект нормативних документів ;
- технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

- огляд та збір літератури стосовно теми кваліфікаційної роботи;
- огляд та аналіз існуючих рішень та аналогів;
- дослідження особливостей розробки вебсайтів;
- створення програмного продукту та його опис;
- дослідження поставленої проблеми та розробка висновків та пропозицій.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
слайдів презентації

6. Консультанти розділів магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата Завдання прийняв

7. Дата видачі завдання 01.10.2023

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1	Аналіз предметної області. Огляд існуючих вебсайтів з тематикою кулінарного блогу	02.10-12.10.23	виконано
2	Формулювання основної задачі та визначення цілей дослідження	12.10-14.10.23	виконано
3	Аналіз тематичної літератури. Дослідження основних етапів розробки вебсайту	14.10-25.10.23	виконано
4	Проектування структури та основного функціоналу блогу	25.10-02.11.23	виконано
5	Розробка користувацького інтерфейсу кулінарного блогу	03.11-11.11.23	виконано
6	Реалізація основних функцій вебсайту	11.11-01.12.23	виконано
7	Робота з базою даних. Створення структури даних та їх взаємозв'язків	30.11-14.12.23	виконано
8	Інтеграція функціоналу вебсайту з користувацьким інтерфейсом	14.12-28.12.23	виконано
9	Тестування загальної роботи вебсайту	28.12.23-04.01.24	виконано
10	Демонстрація результатів реалізації системи науковому керівнику та узгодження етапів дослідження	05.01-08.01.24	виконано
11	Проведення дослідження реалізації вебсайту на мовах програмування Go та PHP	09.01-20.01.24	виконано
12	Опис результатів дослідження	20.01-29.01.24	виконано
13	Оформлення звіту	29.01-10.02.24	виконано

Студент _____ В.С. Оглобліна
(підпис) (прізвище та ініціали)

Керівник роботи _____ О.М. Міхайлуца
(підпис) (прізвище та ініціали)

Нормоконтроль пройдено

Нормоконтролер _____ І.А. Скрипник
(підпис) (прізвище та ініціал)

АНОТАЦІЯ

Сторінок: 75

Рисунків: 16

Таблиць: 2

Джерел: 35

Оглобліна В.С. Особливості використання мови програмування Golang для побудови вебсайту на прикладі тематичного блогу: кваліфікаційна робота магістра спеціальності 121 «Інженерія програмного забезпечення» / наук. керівник О.М. Міхайлуца. Запоріжжя : ЗНУ, 2024. 75 с.

Метою кваліфікаційної роботи є дослідження особливостей та оптимальних підходів у розробці вебсайтів, зокрема використання та порівняння різних мов програмування, засобів frontend та backend розробки, а також розробка тематичного кулінарного блогу.

В процесі дослідження було проаналізовано основні етапи розробки вебсайтів, а також досліджено технології реалізації користувацького інтерфейсу та функціоналу кулінарного блогу.

Розроблено вебсайт на тему кулінарний блог, що складається з серверної частини на Golang, API та клієнтського інтерфейсу, що забезпечує можливість розміщення дописів та взаємодії з ними після створення особистого профілю.

Ключові слова: вебзастосунок, Golang, PostgreSQL, PHP, RESTful API, frontend, backend, HTML, CSS, JavaScript, server, JSON, кулінарний блог, користувач, адміністратор, інтерфейс, дизайн

SUMMARY

Pages: 75

Figures: 16

Tables: 2

Sources: 35

Ohloblina V.S. Features of using the Golang programming language to build a website on the example of a thematic blog: qualification work of the master of specialty 121 "Software Engineering" / Science head O.M. Mikhailutsa. Zaporizhzhia : ZNU, 2024. 75 p.

The aim of the research is to study the features and optimal approaches to website development, including the use and comparison of different programming languages, frontend and backend development tools, as well as the development of a thematic culinary blog.

The study analyzed the main stages of website development and investigated the technologies for implementing the user interface and functionality of a blog.

A culinary blog website has been developed, consisting of a Golang server side, API and client interface that allows users to post posts and interact with them after creating a personal profile.

Keywords: web application, Golang, PostgreSQL, PHP, RESTful API, frontend, backend, HTML, CSS, JavaScript, server, JSON, cooking blog, user, administrator, interface, design.

ЗМІСТ

ВСТУП

Актуальність теми

В нашому інформаційному суспільстві, де доступ до знань і спілкування став важливою частиною повсякденного життя, тематичні блоги відіграють ключову роль у створенні, обміні та поширенні контенту. Основним завданням блогів є поширення інформації та експертних думок у конкретних галузях. Вони дозволяють авторам, які мають глибокі знання у певній області, ділитися своїм експертним баченням із широкою аудиторією. Такі блоги можуть стати цінним ресурсом для навчання та поглиблення знань. Також тематичні блоги створюють унікальні спільноти інтересів. Вони дозволяють людям, що ділять спільний інтерес або хобі, об'єднуватися в інтернеті та обмінюватися думками, досвідом та інформацією. Крім того блоги є ефективним інструментом для особистого та професійного розвитку. Створення і управління блогом може допомогти розвинути навички написання, аналізу та візуалізації інформації. Водночас, це може стати платформою для будування особистого бренду власника блогу та встановлення контактів у відповідній галузі.

Харчування є невід'ємною частиною людського життя з багатьма аспектами, що впливають на наше фізичне, емоційне і соціальне благополуччя. IRI повідомляє, що 56% споживачів набридли страви, які їм зазвичай подають у ресторанах, і лише 14% шукають натхнення для приготування їжі в супермаркеті. Надихаючи ентузіастів кулінарії рецептами виробники продуктів харчування та ритейлери отримують значну вигоду від повернення інвестицій. Важливим питанням є те, де ці рецепти повинні знаходитися. IRI стверджує, що обмін рецептами серед сім'ї та друзів є основним способом, за допомогою якого споживачі дізнаються про нові рецепти, потім вони шукають їх в Інтернеті, після чого йдуть YouTube та кулінарні книги. Найменш популярними джерелами є Facebook та Instagram [19].

Виходячи з дослідження проведеного одним з найпопулярніших каналів пошуку рецептів є саме пошукова система Google, а тому розробка кулінарного блогу є досить актуальним завданням. Вебсайт з рецептами надає користувачам широкий доступ до різноманітності кулінарних ідей та варіацій. Завдяки такому сайту, користувачі можуть легко знайти рецепти на будь-який смак та з різних кухонь світу. Це допомагає розширити горизонти та спробувати щось нове, незалежно від географічного розташування.

Тож у сучасному світі, де інтернет став неодмінною частиною нашого повсякденного життя, розробка вебсайту на прикладі тематичного блогу з рецептами є дуже актуальним завданням. Тематикою блогу було обрано кулінарію, оскільки харчування не тільки задовольняє нашу фізіологічну потребу, але й стає важливим аспектом нашого життя, стилю та культури. Кулінарні блоги надають різноманітні рецепти, поради з готування, демонструють нові тенденції в кулінарному світі, використовують стильні фотографії та відеоматеріали. Отже, такий вебсайт буде мати свою аудиторію та потенційних покупців які зацікавлені у сфері кулінарії та бажають просувати свої послуги онлайн.

Мета і завдання дослідження

Мета дослідження полягає у виявленні особливостей та оптимальних підходів у розробці тематичного кулінарного блогу, зокрема використання та порівняння різних мов програмування, засобів frontend та backend розробки.
Для вирішення задачі дослідження поставлено наступні завдання:

1. Аналіз наукових та практичних джерел для визначення тенденцій розвитку вебсайтів.
2. Огляд популярних вебсайтів з рецептами для ідентифікації основних компонентів, які привертають користувачів.
3. Розробка рекомендацій щодо покращення кулінарних вебсайтів, враховуючи виявлені особливості та попередній аналіз.

4. Дослідження сучасних інструментів та технологій frontend розробки.

5. Порівняння мов програмування Golang та PHP як засобів backend розробки.

6. Вивчення функціональностей та можливостей бази даних PostgreSQL.

7. Розробка прототипу та реалізація основного функціоналу кулінарного блогу, використовуючи визначені методи та засоби розробки.

Об’єкт дослідження

Об’єктом дослідження є процес розробки та оптимізації вебсайту на тематику кулінарного блогу.

Предмет дослідження

Предметом дослідження є інструменти та методи розробки вебсайтів, їхня структура та функціональність, а також можливості покращення та оптимізації існуючих ресурсів.

Методи дослідження

Дослідження здійснено за допомогою аналізу літературних джерел, аналізу та порівняння існуючих вебсайтів, порівняння мов програмування, а також використання емпіричних методів, таких як створення прототипу тематичного блогу з використанням Golang, для визначення практичної ефективності мови. Тож використовувались наукові методи аналізу та порівняння, а також експериментальний підхід у процесі розробки прототипу вебсайту.

Наукова новизна одержаних результатів

Наукова новизна одержаних результатів дослідження полягає у впровадженні мови програмування Golang для розробки вебсайту, а саме

розглядання його особливостей, ефективності та придатності для побудови вебсайтів. Розгляд можливостей створення тематичного блогу з використанням Golang дозволяє визначити, наскільки ця мова відповідає потребам конкретних проектів, таких як блоги.

Практичне значення одержаних результатів

Практичне значення одержаних результатів дослідження полягає у тому, що розроблений кулінарний блог слугує науковою демонстрацією використання Golang та PostgreSQL, і стає практично значущим рішенням для тих, хто бажає створити власний вебсайт з рецептами. Дослідження порівняння мов програмування та вибір оптимальних інструментів може слугувати важливим внеском для розробників, що вивчають оптимальні шляхи розробки вебсайтів.

Апробація одержаних результатів

Результати дослідження були представлені на Міжнародній науково-практичній конференції Інженерного навчально-наукового інституту Запорізького національного університету «Геостратегічні трансформації та траєкторія національної безпеки в контексті відбудови та сталого розвитку України» [29], а також на IV Міжнародній науково-практичній конференції «Стратегічні пріоритети розвитку підприємництва, торгівлі та біржової діяльності» НУ «Запорізька політехніка» [30].

Глосарій

Архітектура програмного забезпечення — це високорівневий план або концепція, яка визначає структуру, компоненти, модулі, інтерфейси та взаємодію всіх елементів програмного забезпечення.

Архітектурний стиль — це сукупність визначень та обмежень, які визначають структуру та взаємодію компонентів системи.

Бенчмарк-тест — це спеціальні програми або тестові сценарії, розроблені для вимірювання продуктивності комп’ютерної системи або окремих її компонентів.

Блогінг — це процес створення, публікації та управління онлайн-журналом, відомим як блог.

Вебсервер — це програмне забезпечення яке надає доступ до вебсторінок та інших вебресурсів за допомогою протоколів передачі гіпертексту (HTTP).

Горутини — це легкі потоки виконання в мові програмування Go, що дозволяють одночасне виконання функцій і є ключовою функцією для створення паралельних і масштабованих програм.

Інтерфейс користувача — це механізм взаємодії між користувачем та комп'ютерною програмою чи пристроєм.

Кешування — це процес зберігання копій даних або результатів певних обчислень, щоб зменшити час доступу до них при подальших запитаннях.

Масштабованість — це здатність системи ефективно збільшувати свою продуктивність або обсяг обробки для відповіді на зростаюче навантаження.

Соціальна мережа — вебплатформа, що дозволяє користувачам створювати особисті сторінки, спілкуватися з іншими користувачами, обмінюватися інформацією, створювати та споживати контент.

Фреймворк — це набір кодових структур, бібліотек, класів та інших компонентів, які надають базові функції та можливості для розробки програмного забезпечення.

Функціонал — це сукупність можливостей та операцій, які виконує програма.

Хостинг — це послуга, яка надається провайдером, для зберігання і надання доступу для перегляду в Інтернеті файли вебсайту.

AJAX (Asynchronous JavaScript and XML) — це техніка, яка використовується в розробці вебдодатків для асинхронної взаємодії з сервером.

ANSI SQL (Structured Query Language) — це стандарт мови запитів до баз даних, розроблений і прийнятий Інститутом національних стандартів (ANSI).

DNS-записи (Domain Name System records) — це дані, що зберігаються в системі доменних імен (DNS) та пов'язані з конкретним доменом.

FTP-клієнт FTP (File Transfer Protocol) — це програма, яка дозволяє користувачеві взаємодіяти з FTP-сервером для передачі файлів між своїм комп'ютером і сервером.

HTTP-обробник — це фрагмент програмного коду або програма, яка обробляє HTTP-запити, що надходять від клієнтів і повертає відповіді.

Joomla — це система управління контентом (CMS), яка дозволяє створювати та управляти веб-сайтами, включаючи їхні структуру, контент, розширення та інші функціональні можливості.

JSON (JavaScript Object Notation) — це формат обміну даними, що використовується для представлення структури даних і є мовонезалежним.

XML (eXtensible Markup Language) — це розширювана мова розмітки для представлення структурованих даних у вигляді тексту.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ АКТУАЛЬНОСТІ РОЗРОБКИ ТЕМАТИЧНОГО БЛОГУ У ФОРМАТІ ВЕБСАЙТУ

1.1 Аналіз літературних джерел

У безмежному просторі цифрових технологій створення кухонного блогу — це захоплююча подорож у світ кулінарних розповідей. За останні кілька років інтерес до кулінарії значно зріс, тому кулінарний блог може стати вигідним бізнесом або засобом додаткового заробітку. Це може бути корисно для тих, хто займається кулінарією вдома або мріє відкрити власну кулінарну студію чи ресторан. Кулінарні блоги можуть допомогти популяризувати місцеві продукти та традиції готування їжі, що важливо для підтримки місцевої економіки та/або розвитку туризму. Кулінарні компанії та ресторани, які шукають партнерів для просування своїх продуктів та послуг в Інтернеті також зацікавлені у співпраці з власником кулінарного блогу.

Ведення блогів стало широко розповсюдженим соціальним явищем, що впливає на різні аспекти, такі як політика, бізнес та комунікація. Значна кількість людей займається блогінгом як розвагою, використовуючи блоги як платформу для обміну думками, емоціями та поглядами на теми, які їх цікавлять. Блог — це вебсайт зі статтями, що зазвичай ведеться однією людиною у формі щоденника або журналу. Автор може дозволити іншим читачам взаємодіяти з ним і писати власні коментарі до блогу. Також блог можна тлумачити як видавничу систему, яка дозволяє додавати фотографії, відео та аудіо [9].

Програмне забезпечення для ведення блогів полегшує завантаження різноманітних медіа, таких як фотографії, музика та відео. Незалежно від того, чи використовують люди блоги для отримання доходу, чи для проведення дозвілля, їх об'єднує участь у блогосфері — мережі блогів, яка надає людям можливість висловити свою думку та швидко і широко поширювати інформацію [18].

Особистий вебсайт з блогом або ведення блогу у соціальних мережах виконують різні функції, і кращий вибір може залежати від індивідуальних потреб та цілей користувача. Блог у популярних соціальних мережах має свої переваги, такі як широкий охоплення, простота використання та можливість взаємодії з великою аудиторією. Однак, блог у вигляді вебсайту має декілька переваг, що робить його зручним для розробки та користування [24]:

- Простота навігації — вебсайт дозволяє створити логічну структуру розділів і сторінок, що полегшує навігацію користувачам. Рецепти можуть бути організовані за категоріями, тегами або іншими параметрами, що робить пошук більш зручним.
- Гнучкість дизайну — розробка вебсайту дозволяє створити унікальний дизайн, враховуючи потреби та стиль блогу, що дає можливість підкреслити брендування та створити привабливе візуальне враження для відвідувачів.
- Функціональні можливості — вебсайт надає широкі можливості для імплементації функціоналу, таких як пошук, коментування, рейтинги, підписка, інтеграція з соціальними медіа тощо. Це дозволяє створити більш інтерактивне та привабливе середовище для користувачів.
- Більше контролю — власний вебсайт дає більший контроль над вмістом, візуальним оформленням та функціоналом блогу. Також є можливість налаштувати свої правила та вимоги щодо публікації контенту, монетизації, аналітики та інших аспектів управління блогом.
- Відсутність обмежень алгоритмів — на власному вебсайті немає алгоритмів, що визначають, хто і як бачить розміщений контент, що надає повний контроль над тим, як контент доступний та розповсюджується
- Масштабованість — вебсайт надає можливість розширити свій блог у майбутньому. Розвиваючи блог можна додавати нові функції, сторінки, розділи або розширювати свою аудиторію. З плином часу, коли блог зростатиме, вебсайт забезпечить більшу гнучкість.

В сучасному світі важко собі уявити людину яка ніколи не стикалася з певними інтернет ресурсами. Вебсистеми супроводжують людину на кожному її кроці — вони використовуються для вирішення великої кількості завдань, та мають широкий спектр застосування. Тому важливо розуміти визначення вебсайту, основні характеристики та історію виникнення і розвитку.

Вебсайт — це набір вебсторінок, які зазвичай пов'язані між собою і розміщені на вебсервері. Доступ до цих сторінок здійснюється через Інтернет за допомогою веббраузера [21]. Кожна вебсторінка містить такий контент, як текст, зображення, відео або інтерактивні елементи, зазвичай ці сторінки написані такими мовами, як HTML, CSS та JavaScript.

Кожен вебсайт має унікальне доменне ім'я, яке слугує його адресою в Інтернеті (наприклад, www.example.com) [33]. Доменні імена реєструються та управляються через реєстраторів доменів. Вебсайти розміщуються на вебсерверах, які зберігають файли вебсайту та надають їх користувачам, коли вони заходять на сайт. Послуги вебхостингу надають інфраструктуру та технології, необхідні для розміщення вебсайтів.

Вебсайти включають різні функціональні можливості, такі як форми для введення даних користувачем, можливості пошуку, функції електронної комерції для онлайн-покупок, інтеграція з соціальними мережами, відтворення мультимедіа та інтерактивні елементи (анімації або ігри). Вебсайти зазвичай мають навігаційне меню або посилання, які дозволяють користувачам переходити між різними сторінками та розділами сайту. Чітка та інтуїтивно зрозуміла структура навігації має важливе значення для зручності користувачів. Дизайн та верстка вебсайту сприяють його візуальній привабливості та зручності використання. Сюди входять такі аспекти, як кольорові схеми, типографіка, інтервали та розташування елементів контенту. З поширенням мобільних пристроїв стало важливим щоб вебсайти були адаптивними, тобто підлаштовувалися під різні розміри екранів і пристрої. Адаптивний дизайн забезпечує однаковий користувацький досвід на настільних комп'ютерах, ноутбуках, планшетах і смартфонах.

Загалом, вебсайти відіграють життєво важливу роль у цифровому світі, слугуючи платформою для спілкування, поширення інформації, комерції, розваг та багато іншого. Вони стали невід'ємною частиною сучасного суспільства, дозволяючи людям, компаніям, організаціям та громадам спілкуватися та взаємодіяти в глобальному масштабі.

Історія створення вебсайтів — це захоплююча подорож, яка охоплює кілька десятиліть, позначена значними технологічними досягненнями та змінами в практиці веброзробки [34]. Перша реалізація Всесвітньої павутини WorldWideWeb (WWW) відбулась у 1990 році. Тім Бернерс-Лі, британський вчений-комп'ютерник, розробив концепцію Всесвітньої павутини під час роботи в ЦЕРНі, а пізніше створив перший веббраузер і перший вебсервер. У 1991 р. Тім Бернерс-Лі представив мову розмітки гіпертексту (HTML) як спосіб створення та форматування документів з гіперпосиланнями. Це стало початком структурування контенту для Інтернету. У 1993 р. було випущено Браузер Mosaic, розроблений Національним центром суперкомп'ютерних програм (NCSA) [34]. Mosaic відіграв ключову роль у популяризації Інтернету серед широкої громадськості. Пізніше, впровадження JavaScript компанією Netscape у 1995 році дозволило розробникам створювати інтерактивний та динамічний контент на вебсторінках. Також з'явилися мови сценаріїв на стороні сервера, такі як PHP, що дозволило створювати більш динамічні вебсайти. Наприкінці 1990-х років розпочався бум Dot-Com, період стрімкого зростання інтернет-бізнесу. Було запроваджено каскадні таблиці стилів (CSS) для розділення структури та стилю вебдокументів, що підвищило гнучкість та можливості дизайну.

На початку 2000-х років з'явилися системи управління контентом (CMS), такі як WordPress та Joomla, що полегшили нетехнічним користувачам створення та управління вебсайтами. З'явився веб 2.0, який акцентував увагу на користувацькому контенті, соціальних мережах та інтерактивному вебдосвіді [14]. З поширенням мобільних пристроїв зросла важливість адаптивного дизайну. HTML5, представлений у 2008 році, приніс нові

семантичні елементи, вбудовану підтримку мультимедіа та вдосконалені API. У 2010-тих набули популярності односторінкові додатки SPA (Single Page Applications), керовані JavaScript фреймворками, такими як Angular, React та Vue.js. Ці фреймворки дозволили розробникам створювати динамічні вебдодатки з більш плавним користувацьким інтерфейсом [17]. Еволюція веброзробки триває і надалі, з постійними інноваціями в технологіях, інструментах і фреймворках.

1.2 Основні етапи створення вебсайтів

В сучасному цифровому віці вебсайти стали невід’ємною частиною нашого повсякденного життя, об'єднуючи інформацію, розваги та комунікацію. І щоб створити ефективний та функціональний вебсайт, важливим є процес його програмування. Цей процес можна розглядати як послідовність етапів, кожен з яких відіграє ключову роль у вирішенні великої кількості завдань та викликів [12].

Етап 1: планування та аналіз. Першим і, можливо, найважливішим етапом є планування та аналіз. Визначення мети вебсайту, аудиторії та основних функціональних вимог встановлює фундамент для подальшого розвитку [14]. Аналіз конкурентів та визначення унікальних рис допомагає сформулювати власну стратегію. Також необхідно обрати унікальне доменне ім’я. Доменне ім'я слугує цифровою адресою блогу [33]. Тому це має бути ім'я, яке не тільки запам'ятовується, але й відображає ідентичність сайту. Можливо включення ключових слів, пов'язаних з обраною нішею, для кращої видимості в пошукових системах.

Етап 2: дизайн та wireframing. Другий етап включає в себе розробку дизайну та створення wireframe. Дизайн визначає вигляд і відчуття вебсайту, включаючи колірну палітру, шрифти та інші важливі аспекти [2]. Перш ніж занурюватися у візуалізацію треба скласти план структури сайту.

Wireframe — це легкий макет вебсайту, який візуалізує розташування елементів та основну структуру [2]. Цей план допоможе вам керувати процесом створення сайту. На прикладі кулінарного блогу, можна виділити ключові розділи, такі як рецепти, кулінарні поради та розділ для взаємодії з аудиторією. Після можна переходити до наповнення вебсайту візуально привабливим дизайном. Також необхідно переконатись, що дизайн є адаптивним, забезпечуючи оптимальну роботу на різних пристроях.

Етап 3: розробка фронтенду (frontend). Розробка фронтенду — це процес створення користувацького інтерфейсу вебсайту, який взаємодіє з користувачем. Фронтенд відповідає за те, як виглядає і як взаємодіє користувач з вебсайтом. Розробка фронтенду включає в себе використання HTML, CSS та JavaScript для створення відмінного інтерфейсу [12].

HTML — це основна мова для створення контенту в Інтернеті. Використовуючи теги HTML, визначаються різні елементи сторінки, такі як заголовки, абзаци, зображення і форми. Його роль у структуруванні документів, полегшенні кросбраузерної сумісності, підтримці доступності та інтеграції з іншими технологіями робить його фундаментальним і незамінним інструментом для веброзробки [5].

Використання JavaScript необхідно для динамічної взаємодії, а CSS для стилізації з метою покращення візуальної привабливості вебсайту. Тобто JavaScript додає інтерактивність до вебсайту. Використовуючи JavaScript, можливо керувати елементами сторінки, обробляти події користувача, виконувати асинхронні запити до сервера та створювати багато інших динамічних функцій [5].

CSS використовується для задання кольорів, шрифтів, розташування елементів та інших візуальних аспектів сторінки [5]. Цей етап також охоплює розробку адаптивного дизайну для оптимального відображення на різних пристроях. Фронтенд часто взаємодіє з бекендом (серверною частиною) для отримання або відправки даних. З використанням асинхронних запитів (AJAX) можна ефективно обмінюватись інформацією.

Етап 4: проєктування бази даних. Робота з базою даних є ключовою частиною створення багатьох вебсайтів, оскільки вона дозволяє зберігати, оновлювати та отримувати доступ до інформації. Вибір правильної бази даних залежить від потреб розробника та технічних вимог. Популярні системи управління базами даних (СУБД) включають MySQL, PostgreSQL, SQLite, та MongoDB (для NoSQL даних) [12].

Створення схеми бази даних включає розробку структури даних та їх взаємозв'язків. Необхідно розробити таблиці, поля, індекси та ключі, що визначають організацію інформації. Далі за допомогою використання мов програмування та фреймворків база даних підключається до бекенду (логіки сервера).

SQL (Structured Query Language) використовується для взаємодії з базою даних. Це включає в себе запити для отримання, оновлення, вставки та видалення даних — CRUD (Create, Read, Update, Delete). Ці операції визначають основні дії, які можна виконати з даними в базі даних.

Забезпечити безпеку бази даних можна використовуючи параметризовані запити, обмеження доступу, шифрування та інші заходи безпеки, що допоможе уникнути атак та злому системи.

Для прискорення пошукових операцій використовується індексація та оптимізація запиів для підвищення продуктивності. Також можна розглянути використання кешування для запитів, які використовуються часто.

Крім цього, необхідно проводити регулярні резервні копії бази даних для захисту від втрати даних, а також переконатись, що процедури відновлення працюють правильно. ORM (Object-Relational Mapping) дозволяє взаємодіяти з базою даних, використовуючи об'єктно-орієнтований код, що спрощує роботу та підтримку. Працюючи з базою даних для створення вебсайту, важливо зрозуміти основні концепції, використовувати ефективні методи доступу до даних та дотримуватися кращих практик безпеки. Правильно налаштована та доглянута база даних є важливою складовою успішного вебпроекту.

Етап 5: розробка бекенду (backend). Розробка бекенду — це процес створення логіки та функціональності, які відповідають за обробку та управління даними на вебсайті. Бекенд відповідає за взаємодію з базою даних, обробку запитів від фронтенду, забезпечення безпеки та виконання бізнес-логіки. Розробка бекенду зазвичай включає в себе вибір мов програмування (таких як Python, PHP, Ruby), вибір фреймворку та роботу з базою даних [12].

У цьому дослідженні мовою було обрано Go. У сфері веброзробки Go, також відома як Golang, стала потужною та ефективною мовою програмування. Її простота, швидкість і масштабованість роблять її чудовим вибором для створення динамічних вебсайтів [17]. Цей етап буде включати в себе налаштування середовища розробки Go, ознайомлення з синтаксисом, структурою та системою управління пакетами Go.

API (Application Programming Interface) дозволяє фронтенду взаємодіяти з бекендом [17]. Необхідно визначити точки входу (endpoints) та формати обміну даними. У нашому випадку використовується архітектурний стиль gRPC для передачі даних між компонентами системи.

Оптимізувати роботу бекенду можна шляхом ефективного використання пам'яті та кешування даних. Зменшення частоти звернень до бази даних може значно покращити продуктивність.

Оскільки блог у вигляді вебсайту вимагатиме автентифікації користувачів, необхідно реалізувати безпечні функції реєстрації та входу. Необхідно реалізувати механізми аутентифікації для перевірки ідентичності користувачів та авторизації для надання прав доступу до певних ресурсів. Захистити вебсайт можливо за допомогою впровадження HTTPS, валідації даних та захисту від поширених вебвразливостей.

Етап 6: тестування та відлагодження. Перш ніж презентувати свій продукт світу необхідно провести ретельне тестування. Після завершення розробки обох сторін (фронтенду та бекенду), важливим етапом є тестування. Тестування включає в себе перевірку всіх функціональностей, адаптивності та безпеки вебсайту. Вебсайт має безперебійно функціонувати на різних

пристроях і браузерах. Також необхідно перевірити наявність непрацюючих посилань, і звернути увагу на загальний користувацький досвід. Відлагодження вирішує будь-які помилки та проблеми, що виникають під час тестування [12].

Етап 7: підключення до хостинг-платформи. Коли вебсайт готовий до оприлюднення настає час додавання його на хостинг, що зробить його доступним для користувачів в Інтернеті [14]. Необхідно обрати надійного хостинг-постачальника залежно від потреб та бюджету. Популярними хостинг-постачальники є Bluehost, HostGator, SiteGround, AWS, Heroku та інші. Далі можемо отримати від хостинг-постачальника дані для підключення, такі як ім'я хоста (hostname), ім'я користувача, пароль та ім'я бази даних.

За допомогою FTP-клієнта чи іншого інструменту завантажуємо файли вебсайту на сервер (зазвичай, це файли HTML, CSS, JavaScript, зображення та інші активи). Далі налаштовуємо DNS обраного домену, щоб вказати на сервер хостинг-постачальника. Це може зайняти трохи часу, оскільки DNS-записи потребують часу для поширення.

Також необхідно налаштувати БД за допомогою інструментів, які надає хостинг-постачальник, що включає імпорт структури бази даних та необхідних даних. Для забезпечення безпеки вебсайту рекомендується встановити SSL-сертифікат. Більшість хостинг-постачальників надають можливість безкоштовно отримати та встановити SSL [14].

Особливу увагу варто приділити перевірці функціонування вебсайту (коректність відображення сторінок, функціональність та зв'язок з базою даних). Кожен хостинг-постачальник може мати свої унікальні особливості та інтерфейс, тому слід використовувати документацію конкретного хостингу для найточніших інструкцій.

За необхідності можна зареєструвати вебсайт у пошукових системах, таких як Google, для забезпечення його індексації та відображення у пошукових результатах. Підвищити видимість блогу можна застосувавши основні методи пошукової оптимізації (SEO). Наприклад, відповідні ключові

слова, які пов'язані з тематикою вебсайту, інструменти, такі як Google Keyword Planner чи SEMrush, створити креативні унікальні метатег заголовка (title tag) та метатег опису (meta description) щоб привернути увагу користувачів [32].

Етап 8: підтримка та розвиток вебсайту. Створення вебсайту — це не завершений процес. Після успішного завершення тестування вебсайт готовий до розгортання. Оптимізація для продуктивності та безпеки важлива на цьому етапі. Це також включає встановлення систем аналізу відвідуваності та інші інструменти для відстеження та аналізу вебсайту. Після розгортання важливо надавати постійну підтримку, виправляти помилки, а також вносити оновлення та вдосконалення відповідно до змін потреб користувачів та технологічних стандартів. Тобто необхідно регулярно оновлювати контент та слідкувати за розвитком трендів [12].

Узагальнюючи, етапи програмування вебсайту є складним, але захоплюючим процесом, що вимагає творчого мислення, технічної експертизи та виважених стратегій. Справжнє мистецтво полягає в тому, як програмісти вдаються об'єднати функціональність, естетику та ефективність для створення вебсайтів, які не лише задовольняють технічні вимоги, але і залишають незабутній враження у користувачів.

1.3 Огляд популярних вебсайтів з рецептами

Популярність сайтів з рецептами постійно зростає в наш час. І це не дивно, адже такі сайти стають важливим джерелом інформації та натхнення для людей, які цікавляться готуванням та харчуванням. Сприяючи кулінарній творчості, ці сайти надають користувачам широкий вибір рецептів з усіх кухонь світу, від класичних до екзотичних страв. Вони розповідають про інгредієнти, методи приготування та кулінарні техніки, що допомагає навчитися новому і розширити свої кулінарні знання.

Найпопулярніші сайти з рецептами відзначаються своєю простотою використання та зручним пошуком. Це дозволяє користувачам швидко знайти

рецепт, який відповідає їхнім потребам, враховуючи обмеження дієт, складники або час приготування. Це особливо корисно для людей з особливими дієтичними потребами або алергіями. Фотографії високої якості, розташовані поряд з рецептами, стають справжнім джерелом натхнення. Вони допомагають візуалізувати кінцевий результат та створюють бажання спробувати приготувати цю страву самостійно. Багато сайтів також додають відео-інструкції, які крок за кроком показують процес приготування, спрощуючи його для користувачів. Відповідно до багатьох рейтингів [19, 35] трійкою найпопулярніших кулінарних вебсайтів є Cookpad.com, Allrecipes.com та Foodnetwork.com.

Cookpad.com — це популярний вебсайт і додаток для обміну рецептами, який дозволяє користувачам знаходити, створювати та ділитися рецептами зі спільнотою домашніх кухарів з усього світу. Він був заснований в Японії в 1997 році Акіміцу Сано і з тих пір поширився на інші країни, включаючи США, Індію, Іспанію, Великобританію, Бразилію, Індонезію та інші. В середньому в місяць сайт відвідує близько 115 мільйонів користувачів, з яких 47% чоловіків та 53% жінок [35].

На Cookpad користувачі можуть шукати рецепти за різними критеріями, такими як інгредієнти, час приготування, дієтичні обмеження та кухні. Вони також можуть завантажувати власні рецепти, додавати фотографії та писати детальні інструкції з приготування. Проте навігація по сайту є ускладненою адже, ні у хедері, ні у випадаючому меню немає швидкого доступу до категорій страв (див. Рис. 1.1).

Платформа заохочує взаємодію та залучення користувачів, дозволяючи їм коментувати рецепти, ставити запитання та ділитися своїм кулінарним досвідом. Cookpad має на меті надихнути та розширити можливості людей готувати вдома, надаючи платформу для обміну рецептами та відкриття нових. Він просуває ідею домашньої кухні як засобу формування здорових харчових звичок, економії коштів та створення відчуття спільноти серед кулінарів.

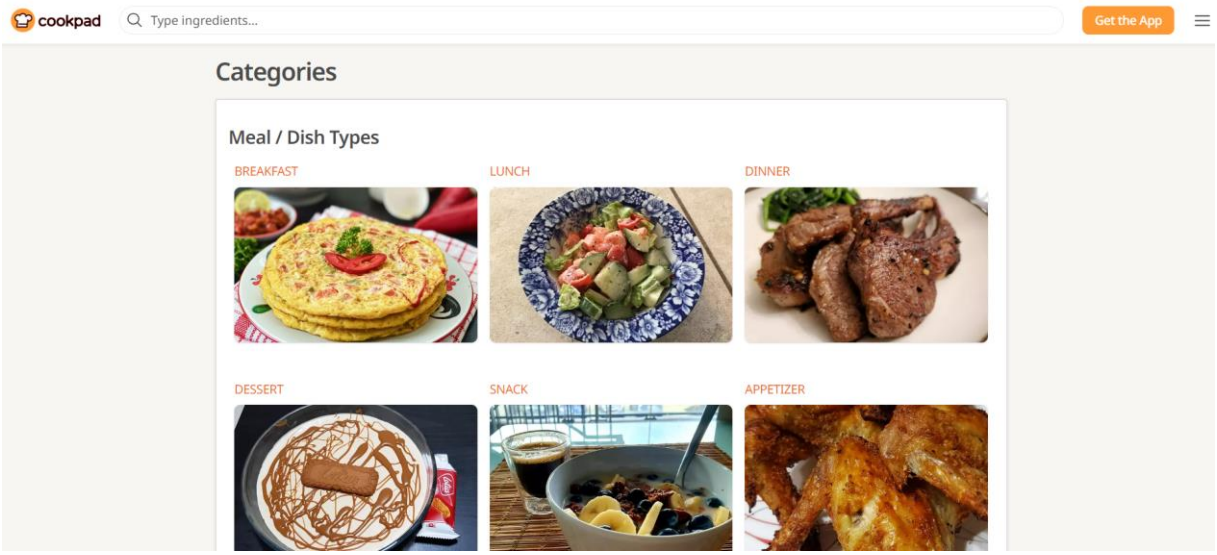

Рисунок 1.1 — *Інтерфейс вебсайту «Cookpad.com»*

Окрім вебсайту, Cookpad розробив мобільні додатки для пристроїв на iOS та Android, завдяки яким користувачам зручно отримувати доступ до рецептів та спілкуватися зі спільнотою на ходу (див. Рис. 1.2).

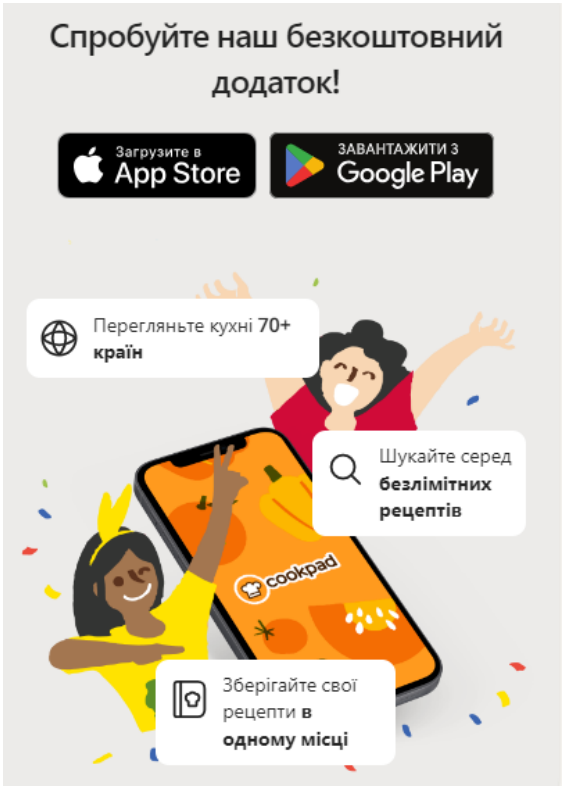

Рисунок 1.2 — *Реклама мобільного додатку «Cookpad.com»*

Allrecipes.com також був заснований у 1997 році, групою ентузіастів кулінарії на чолі з Тімом Хантом і зараз працює як дочірня компанія Meredith Corporation. Є популярним вебсайтом для обміну рецептами та онлайн-спільнотою, яка надає домашнім кухарям платформу для пошуку та обговорення рецептів. В середньому в місяць сайт відвідує близько 80 мільйонів користувачів, з яких 39% чоловіків та 61% жінок [35].

Allrecipes.com пропонує величезну колекцію рецептів, надісланих користувачами, з широкого спектру кухонь і категорій, включаючи закуски, основні страви, десерти та багато іншого. Користувачі можуть шукати рецепти за певними інгредієнтами, дієтичними вподобаннями, способами приготування та оцінками користувачів. На сайті також є корисні інструменти, такі як огляди рецептів, рейтинги та можливість зберігати рецепти в персоналізованих колекціях. Порівняно з вебсайтом Cookpad.com навігація по сайту є більш доступною та наповнення сайту детальніше, проте наявний лише англомовний варіант (див. Рис. 1.3).

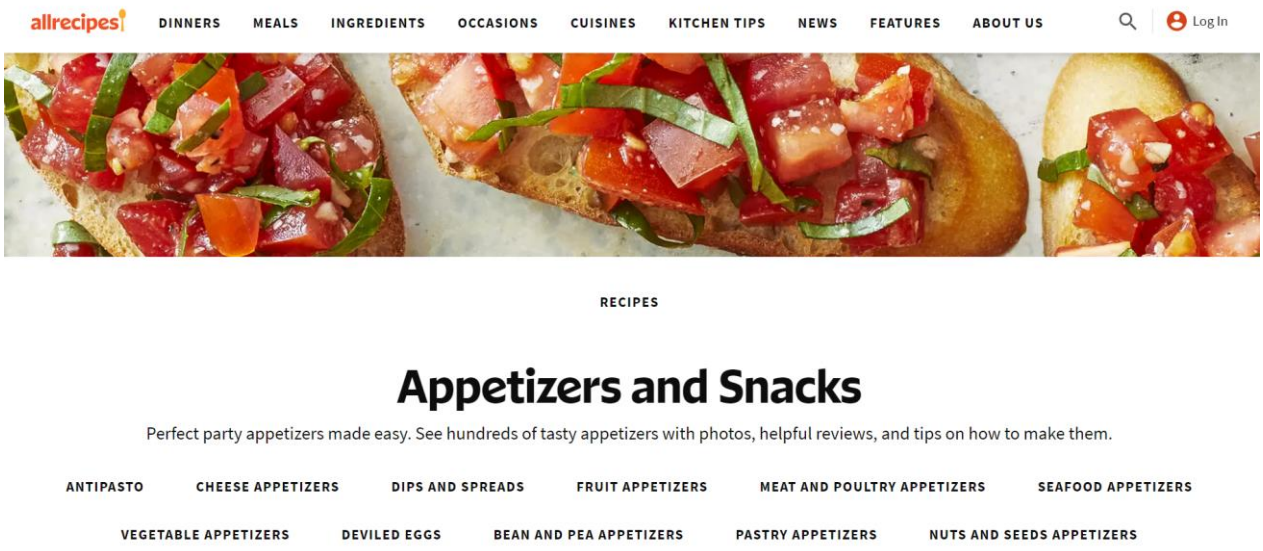

Рисунок 1.3 — *Інтерфейс вебсайту «Allrecipes.com»*

Одним з важливих аспектів Allrecipes.com є його активна та зацікавлена спільнота. Користувачі можуть створювати профілі, взаємодіяти з іншими кулінарами за допомогою коментарів і приватних повідомлень, а також брати

участь у форумах і дискусіях. Це дозволяє ділитися кулінарними порадами, варіаціями рецептів та брати участь у кулінарних бесідах (див. Рис. 1.4). Allrecipes.com розширив свою присутність за межі вебсайту, створивши також мобільні додатки для пристроїв iOS і Android, що полегшує користувачам доступ до рецептів і спілкування зі спільнотою на смартфонах і планшетах.

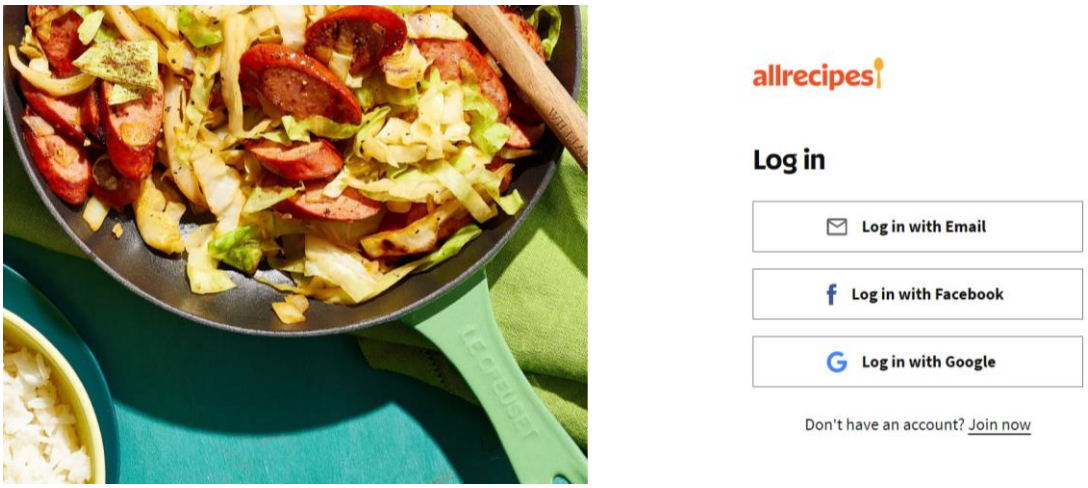

Рисунок 1.4 — *Сторінка реєстрації на вебсайті «Allrecipes.com»*

Foodnetwork.com — це офіційний вебсайт популярного американського телевізійного каналу Food Network, присвяченого їжі та кулінарії. Сайт слугує продовженням каналу, надаючи користувачам широкий спектр кулінарних ресурсів, рецептів, відео, статей та інформації, пов'язаної з їжею, технікою приготування їжі та різноманітними телевізійними шоу, що транслюються на телеканалі. В середньому в місяць сайт відвідує близько 50 мільйонів користувачів, з яких 37% чоловіків та 63% жінок [35].

Foodnetwork.com пропонує величезну колекцію рецептів від відомих шеф-кухарів, кулінарних експертів та телеведучих, які беруть участь у програмах Food Network. Користувачі можуть шукати рецепти за певними інгредієнтами, типами кухні, дієтичними вподобаннями та рівнями складності. На сайті також можна знайти кулінарні поради, технічні посібники та статті про харчові тенденції, розважальні ідеї та кухонні гаджети. Окрім рецептів і статей, Foodnetwork.com надає доступ до відеоконтенту, включаючи

кулінарні шоу, навчальні відео та закулісні зйомки. Користувачі можуть дивитися повні випуски своїх улюблених шоу, переглядати пропущені епізоди та досліджувати ексклюзивний онлайн-контент (див. Рис. 1.5).

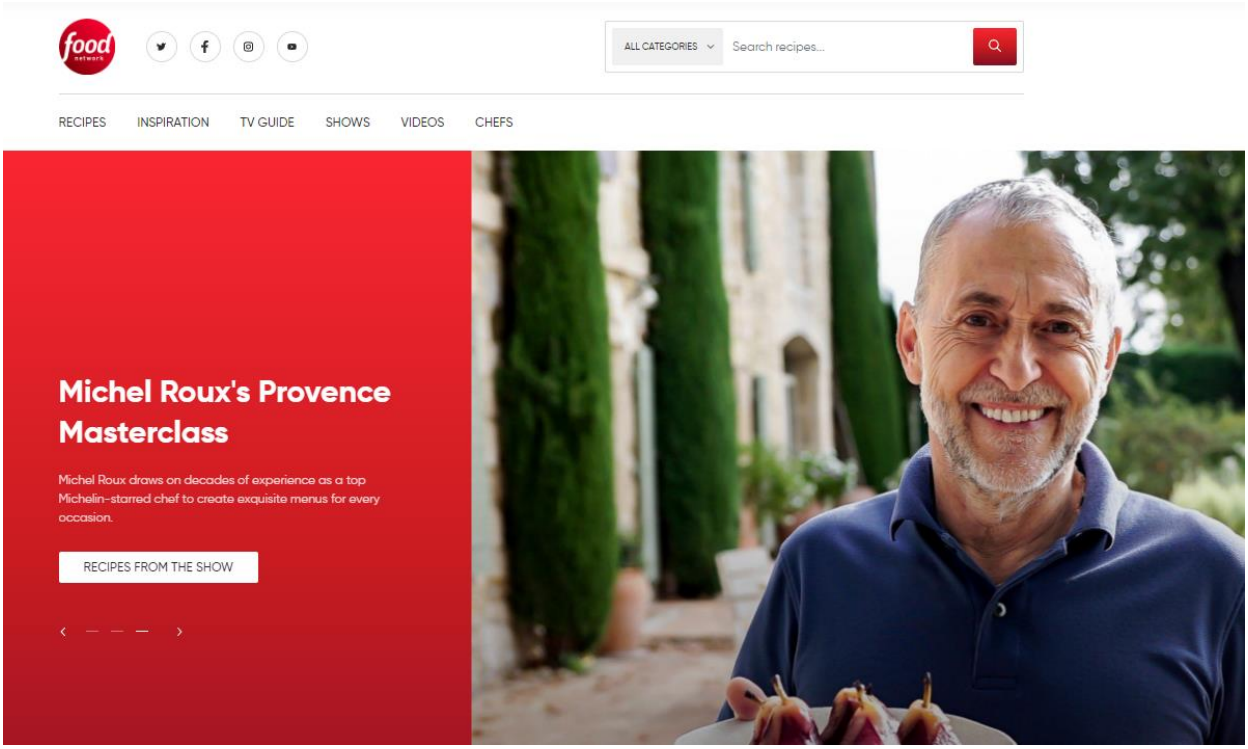

Рисунок 1.5 — *Інтерфейс вебсайту «Foodnetwork.com»*

На сайті також є інтерактивні функції, такі як форуми спільнот, де користувачі можуть спілкуватися, ділитися рецептами, ставити запитання та обговорювати теми, пов'язані з кулінарією. Foodnetwork.com постійно поповнюється новим контентом, включаючи сезонні рецепти, ідеї для свят та висвітлення спеціальних подій (див. Рис. 1.6). Це цінний ресурс для домашніх кулінарів та ентузіастів, які шукають натхнення, поради та розваги у світі їжі та кулінарії. Однією з ключових переваг популярних сайтів з рецептами є їхня доступність та безкоштовність. Багато з них надають великий обсяг інформації без необхідності підписки або платних послуг. Це дозволяє людям з усього світу мати доступ до різноманітних рецептів без фінансових обмежень. Популярність сайтів з рецептами також пояснюється зростаючим інтересом до здорового харчування.

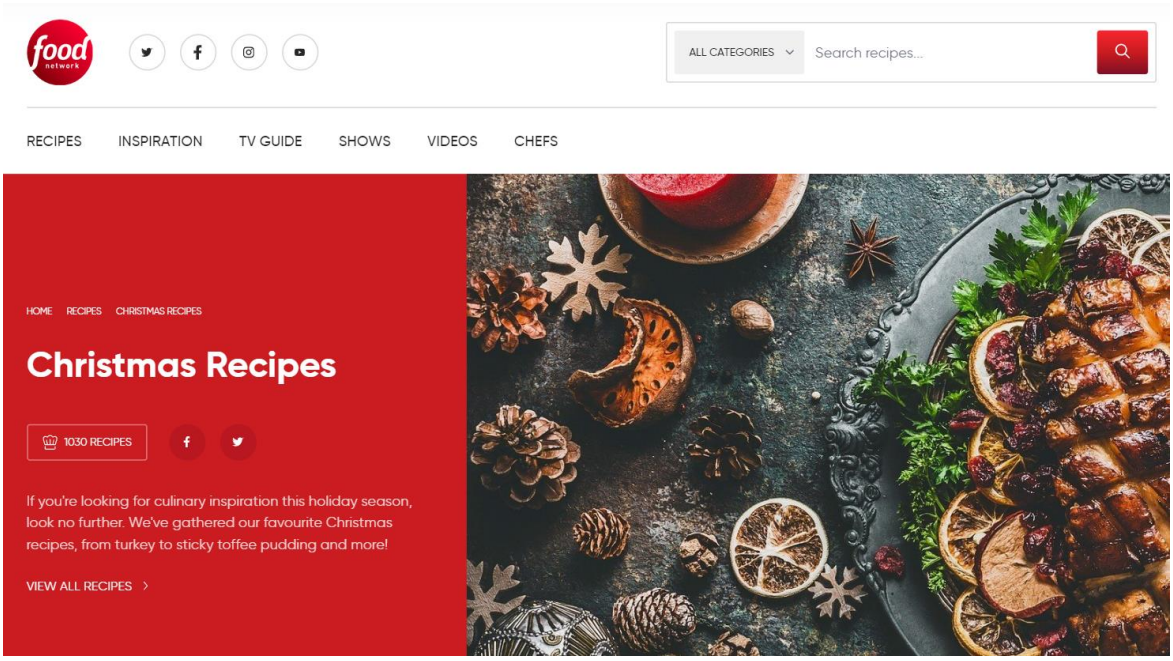

Рисунок 1.6 — *Тематичні рецепти на вебсайті «Foodnetwork.com»*

Окрім надання рецептів, багато кулінарних вебсайтів також створюють спільноту любителів готування. Це дозволяє користувачам обмінюватися своїми враженнями, думками та питаннями про рецепти і кулінарні техніки. На таких сайтах можна знайти коментарі, рецензії та поради від інших користувачів, що сприяє взаємодії та обміну досвідом. Важливо обирати надійні та впливові джерела, щоб мати впевненість у якості рецептів та інформації.

Загалом, популярність сайтів з рецептами зростає завдяки своїй доступності, широкому вибору рецептів, натхненню, спільноті та підтримці здорового харчування. На жаль, в Україні кількість вебсайтів з рецептами не така велика, порівняно з іншими країнами. Хоча кулінарна культура в Україні є багатою і різноманітною, існує певний дефіцит ресурсів, які пропонують широкий вибір рецептів та інформацію про готування. Проте, в останні роки спостерігається певне зростання інтересу до створення вебсайтів з рецептами в Україні. З'являються нові проекти та блоги, що пропонують різноманітні рецепти. Це свідчить про зростання кулінарного інтересу та бажання людей поділитися своїми кулінарними досягненнями через вебсайти. Для прикладу розглянемо вебсайт з рецептами Retsepty.online.ua (див. Рис. 1.7).

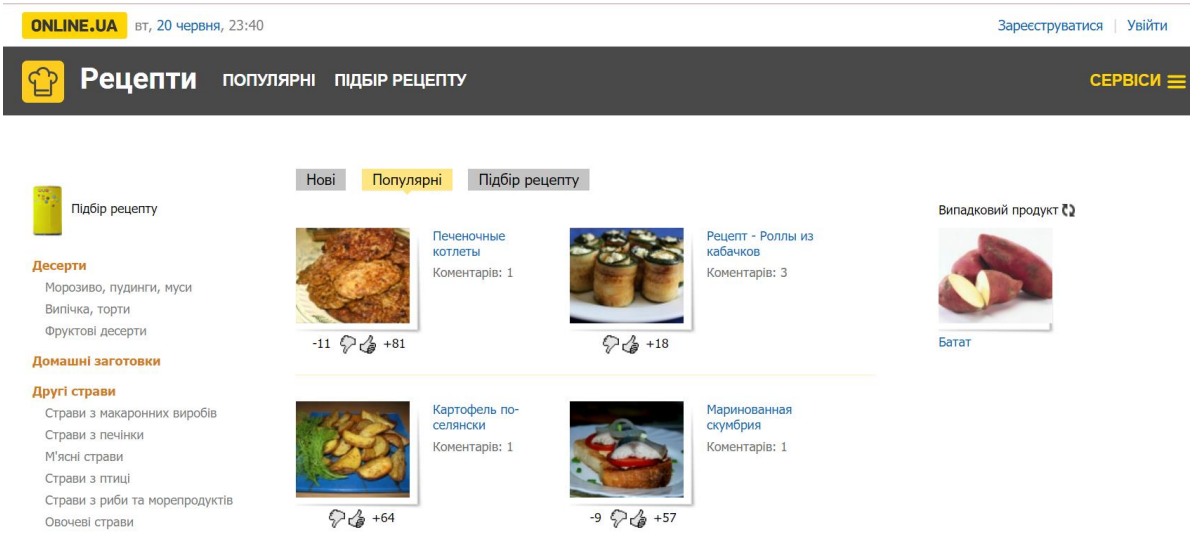

Рисунок 1.7 — *Інтерфейс вебсайту «Retsepty.online.ua»*

На цьому сайті представлені рецепти за категоріями десерти, закуски, перші страви, другі страви, салати які також містять інші підкатегорії. Ця панель закріплена у лівій частині сайту і завжди залишається доступною для користувача, проте, можна виділити недолік стосовно відсутності більш детального фільтру, наприклад, за інгредієнтами або часом приготування. Також на сайті можна створювати профіль, оцінювати та коментувати рецепти, а фішкою є підбір рецепту за окремими продуктами. В цілому сайт виконує основну функцію кулінарного вебсайту, а саме розміщення рецептів, однак потребує ще значного вдосконалення, такі як адаптація вебсторінок, їх наповнення та пересування між ними.

Проаналізувавши сучасні вебсайти з рецептами можна виокремити наступні аспекти, що потребують особливого підходу: адаптивність, категорії рецептів, навігація та фільтри у пошуку — вони грають ключову роль у покращенні користувацького досвіду та забезпеченні зручності використання.

1. Адаптивність вебсайту. У сучасному цифровому світі, де доступ до Інтернету здійснюється з різних пристроїв, включаючи комп'ютери, планшети та смартфони, адаптивність стає невід'ємною частиною розробки вебсайту. Кулінарний сайт повинен бути оптимізований для різних пристроїв і розмірів

екранів, забезпечуючи зручний перегляд і взаємодію з контентом незалежно від пристрою, який використовується. Адаптивний дизайн дозволяє автоматично налаштовувати розмір, розташування та зображення на вебсторінці, щоб забезпечити оптимальне візуальне відображення та зручне використання для користувачів.

2. Категорії рецептів. Організація рецептів за категоріями допомагає користувачам швидко знайти та вибрати потрібні страви. Розробка чіткої і логічної структури категорій дозволяє класифікувати рецепти за типом страви (супи, салати, головні страви, десерти і т.д.), складниками, кухнями або іншими характеристиками. Кожна категорія повинна мати свою окрему сторінку зі списком відповідних рецептів, що полегшує навігацію та забезпечує зручний доступ до бажаної інформації.

3. Навігація по сайту. Ефективна навігація є ключовим фактором успіху будь-якого вебсайту. Користувачі повинні легко знаходити потрібну інформацію, переміщуватися між різними сторінками та функціями, не гублячись на сайті. Зручне розташування меню, використання ярликів, пошукових полів і посилань на основні розділи допомагають полегшити навігацію по сайту. Також важливо забезпечити постійну наявність основного меню навігації на всіх сторінках сайту, щоб користувачі могли легко переходити між різними розділами.

4. Фільтри у пошуку. Кулінарний сайт може містити велику кількість рецептів, тому важливо надати користувачам можливість фільтрувати та шукати рецепти за певними критеріями. Фільтри у пошуку дозволяють користувачам обмежити результати до певного типу страви, складників, складності приготування, часу приготування та інших параметрів. Це дозволяє зменшити кількість результатів і швидше знайти бажану страву. Також важливо забезпечити зручну інтерактивну функцію пошуку, яка автоматично підказує варіанти при введенні запиту.

Звертаючи особливу увагу на ці моменти при розробці кулінарного сайту можна створити зручний і легкий у використанні ресурс для

користувачів. Ці аспекти допомагають полегшити пошук та вибір рецептів, забезпечують зручний доступ до необхідної інформації та покращують загальний досвід використання сайту та роблять його привабливішим. Такий блог має забезпечити аудиторію різноманітними та інноваційними рецептами, надавати детальні та корисні поради щодо процесу готування і сформувати активну та зацікавлену спільноти, де любителі готування зможуть ділитися своїм досвідом, обговорювати рецепти та надихати один одного.

1.4 Висновки до розділу 1

Отже, розглянувши явище блогінгу очевидно, що блоги стали важливою частиною інтернет-середовища. Блоги стають місцем, де автори можуть ділитися своїм досвідом, ідеями та знаннями з аудиторією. Їхній вплив на відвідувачів полягає в забезпеченні доступу до інформації, розважанні та комунікації. Переваги блогу у форматі вебсайту надають авторам можливість повніше виражати себе через дизайн та структуру сайту, що підкреслює індивідуальність та унікальність контенту. Також вебсайти надають ширший спектр можливостей для розширення аудиторії та взаємодії з читачами через коментарі, соціальні мережі та інші функції.

Було надано дефініцію вебсайтів, основні характеристики, історію виникнення та опис основних етапів. Тож, вебсайт являє собою зібрання вебсторінок, зазвичай розташованих на одному домені, що можна відвідати за допомогою веббраузера. Вебсайт може містити текст, графіку, мультимедіа, гіперпосилання та інші елементи. Історія розробки вебсайтів відображає динамічний шлях, сформований мінливими потребами користувачів і розвитком технологій.

Основними етапами розробки вебсайту для кулінарного блогу є: попередня розробка структури та дизайну сайту, вибір технологій, розробка frontend та backend, вибір бази даних та реалізація основного функціоналу, підключення до хостинг-платформи. Це вказує на необхідність

систематичного та компетентного підходу під час створення вебпростору для кулінарного блогу.

Проведений аналіз популярних кулінарних блогів дозволяє виділити їх сильні та слабкі місця. Очевидно, що успішні кулінарні блоги часто володіють візуально привабливим дизайном та якісним контентом. Однак завжди є можливість для покращення, а саме варто звернути увагу на такі аспекти: адаптивність дизайну для зручного перегляду та взаємодії на різних пристроях; сучасне оформлення з використанням якісних фотографій для створення візуально привабливого інтерфейсу, що привертатиме увагу користувачів; забезпечення простоти навігації та доступності контенту. Опанування основних етапів розробки та врахування найкращих практик існуючих кулінарних блогів може служити підґрунтям для створення успішного та привабливого вебсайту для кулінарного блогу.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ ЗАСОБІВ РОЗРОБКИ ТЕМАТИЧНИХ ВЕБСАЙТІВ

2.1 Огляд засобів frontend розробки

HTML (HyperText Markup Language) є основною мовою розмітки для створення вебсторінок. Використання HTML дозволяє створювати структуру та визначати вигляд вебсторінок. HTML має свої переваги і недоліки, які варто враховувати при розробці вебсайтів (див. табл. 1). Враховуючи їй HTML є необхідним і фундаментальним інструментом для створення вебсайтів. Він забезпечує структуру та базовий контент сторінок, а для досягнення бажаного вигляду та функціональності використовуються додаткові технології, такі як CSS та JavaScript [2].

Таблиця 1 — Переваги та недоліки HTML

Переваги	Недоліки
Сумісність з іншими технологіями: HTML ідеально поєднується з CSS (Cascading Style Sheets) для задання стилів та JavaScript для динамічного взаємодії з вебсторінками.	Відсутність динамічності: HTML не надає можливостей для складних інтерактивних функцій. Для цього потрібні додаткові технології, такі як JavaScript.
Простота в освоєнні: HTML має простий синтаксис, що дозволяє швидко оволодіти основними принципами розмітки вебсторінок.	Орієнтація на структуру: HTML зосереджується на структурі вебсторінки, а не на її вигляді. Це означає, що для досягнення бажаного вигляду може знадобитись додаткове використання CSS.
Доступність: HTML надає можливість створювати доступні вебсторінки, які можуть бути використовані людьми з різними обмеженнями.	
Відкритий стандарт: HTML є відкритим стандартом, що означає, що він є загальноприйнятим і підтримується всіма сучасними браузерами.	Велика кількість коду: Для створення складних вебсторінок може знадобитись велика кількість HTML-коду, що може призвести до збільшення обсягу файлів та швидкості завантаження сторінок.

CSS (Cascading Style Sheets) — це мова стилів, що використовується для опису зовнішнього вигляду та форматування вебдокументів, написаних у мові розмітки, зокрема HTML (або XHTML). Основна мета CSS полягає у відокремленні стилів від структури змісту, що дозволяє змінювати зовнішній вигляд вебсторінок без необхідності змінювати сам HTML-код. За допомогою CSS можна задавати кольори, шрифти, розташування елементів, розміри, фони, анімації та багато іншого [26].

Таблиця 2 — Переваги та недоліки CSS

Переваги	**Недоліки**
Розділення змісту та представлення: CSS дозволяє відокремити стиль вебсторінки від її змісту. Це полегшує зміну дизайну та стилів без впливу на сам контент.	Обмеженість взаємодії зі змістом: CSS може контролювати зовнішній вигляд елементів, але обмежений взаємодією зі змістом сторінки.
Консистентність та повторне використання: Використання CSS дозволяє створити однорідний вигляд для всього вебсайту, що забезпечує консистентність стилів. Крім того, стилі можна повторно використовувати на різних сторінках, що спрощує розробку та підтримку.	Складність підтримки різних пристроїв: Розробка адаптивного дизайну, який коректно відображається на різних пристроях і розмірах екрану, може бути викликом. Потрібно враховувати різні розміри екрану, орієнтацію, дотик і т.д.
Гнучкість та контроль: CSS надає широкий набір можливостей для стилізації елементів і розташування на сторінці. Це дозволяє розробникам мати повний контроль над виглядом і поведінкою елементів.	Складність підтримки різних пристроїв: Розробка адаптивного дизайну, який коректно відображається на різних пристроях і розмірах екрану, може бути викликом. Потрібно враховувати різні розміри екрану, орієнтацію, дотик і т.д.
Швидкість та продуктивність: Завантаження стилей через CSS-файли дозволяє браузеру кешувати їх, що призводить до більш швидкого завантаження сторінок і поліпшення продуктивності.	Сумісність зі старими браузерами: Деякі застарілі браузери можуть не підтримувати деякі нові функції CSS або інтерпретувати їх неправильно.

Загалом, CSS є потужним інструментом для стилізації та вигляду вебсайтів, але вимагає розуміння і досвіду для оптимального використання. Розробники повинні зважати на переваги та недоліки CSS та знаходити баланс між дизайном, функціональністю і підтримкою різних пристроїв. (див. табл. 2)

JavaScript (або скорочено JS) — це високорівнева мова програмування, яка використовується для розробки динамічного вебзмісту і взаємодії з користувачем на вебсторінках. JavaScript є інтерпретованою мовою, що означає, що код виконується безпосередньо в браузері, без необхідності попередньої компіляції. Він забезпечує можливість додавати функціональність, обробку подій, маніпуляцію елементами вебсторінок, анімацію, валідацію даних та багато іншого. JavaScript широко використовується для розробки вебдодатків, інтерактивних вебсторінок, анімації, вебігор та інших динамічних елементів на вебсайтах. Він є однією з ключових мов програмування для фронтенд-розробки [15].

Серед переваг JavaScript можна виділити:

- JavaScript підтримується всіма сучасними веббраузерами, що дозволяє використовувати його на будь-якій платформі або пристрої.
- JavaScript надає можливість створювати динамічний контент, реагувати на дії користувача та забезпечувати інтерактивну взаємодію на вебсторінках.
- За допомогою JavaScript можна розробляти власні функції, бібліотеки та фреймворки, що дозволяє розширити його можливості та пристосувати до конкретних потреб проекту.
- JavaScript підтримує різноманітні функції, такі як маніпуляція DOM, асинхронне програмування, робота з AJAX для обміну даними з сервером, анімація, валідація форм, маніпуляція зображеннями тощо.

Проте одночасно JavaScript належать певні недоліки: різні браузери можуть мати різні реалізації, що може призводити до несумісності та проблем з переносимістю; старі версії браузерів можуть не підтримувати нові функції; великі проекти можуть стати складними для управління і підтримки; якщо

великий обсяг обробки даних або обчислень потрібно здійснити на клієнтському боці, це може призвести до затримок та зниження продуктивності, особливо на слабших пристроях; а також вразливість до помилок, що можуть призвести до некоректної роботи або збоїв на вебсторінках.

Не зважаючи на ці недоліки, JavaScript залишається однією з найпопулярніших мов програмування для веброзробки, завдяки своїм широким можливостям та доступності. З правильним використанням і керуванням його недоліками, JavaScript може значно поліпшити функціональність та взаємодію на вебсайтах.

2.2 Порівняння Golang та PHP як засобів backend розробки

Серед популярних технологій, які реалізують створення вебсторінок із фрагментами коду, виконуваного на сервері, найпопулярнішою можна вважати некомерційну, вільно розповсюджувану технологію PHP (Personal Home Pages). Ця технологія заснована на використанні CGI-застосувань, що інтерпретують впроваджений у HTML-сторінку код на скриптовій мові. Головною особливістю мови PHP є її практичність. PHP надає програмісту інструмент для швидкого й ефективного вирішення поставлених завдань. Вона вирізняється винятковою гнучкістю до потреб розробника. Хоча PHP традиційно рекомендують використовувати у поєднанні з HTML-кодом, проте PHP з таким же успіхом інтегрується і в JavaScript, XML та інші мови Інтернет-програмування. Ці технології забезпечують сучасну функціональність, ефективний супровід процесів створення сайтів та їх наповнення інформаційними ресурсами [22].

Головними перевагами PHP є практичність, легкість у застосуванні, ефективність, продуктивність та гнучкість. PHP фреймворки за останній час набрали популярність і стали базовою платформою для розробки вебзастосунків. Використання цих систем, дозволяє економити велику

кількість часу, зменшити навантаження на процес розробки, позбавляючи від проблеми повторюваного коду, і швидко створювати якісні додатки. Між тим, використання PHP фреймворків робить процес створення програми значно більш легким і функціональним. Незважаючи на те, що PHP як і раніше є одною з найбільш часто використовуваних мов для створення вебсайтів, за роки існування вона завоювала собі репутацію мови з жахливою організацією несекьюрного повного дірок коду, недосвідченими розробниками, нестабільними бібліотеками і т.д.

За останні роки у сфері веброзробки значної популярності набула мова програмування Golang (або Go), розроблена в компанії Google. Розробники програмного забезпечення використовують Go у низці операційних систем і фреймворків для розробки вебдодатків, хмарних і мережевих служб та інших типів програмного забезпечення [20]. Go є статично типізованою мовою, в основі створення якої є мова програмування C. Через швидкий час запуску мови Go, низькі накладні витрати на виконання та можливість працювати без віртуальної машини (VM), вона стала дуже популярною мовою для написання мікросервісів та інших цілей. Крім того, Go використовується для паралельного програмування — стратегії виконання кількох завдань одночасно, не по порядку або в частковому порядку. Ось кілька переваг Golang перед PHP.

Продуктивність. Golang відомий своєю високою продуктивністю завдяки ефективному опрацюванню паралельних завдань і низькому рівню накладних витрат. Він пропонує безліч вбудованих функцій і бібліотек, які дають змогу створювати швидкі та чуйні вебдодатки.

Простота та ефективність коду. Golang має простий і зрозумілий синтаксис, який полегшує читання і написання коду. Він прагне до мінімалізму і уникає зайвої складності, що спрощує підтримку і розробку проєктів.

Масштабованість. Golang пропонує вбудовану підтримку паралельності та розподілених обчислень, що робить його чудовим вибором для розробки

високонавантажених вебдодатків. Він має вбудовані інструменти для управління горутинами (goroutines) і каналами (channels), що спрощує розробку масштабованих систем.

При розробці вебсайту продуктивність мови програмування зазвичай не є головним фактором для визначення вибору [24]. Обидві мови, PHP і Go (Golang), здатні обробити вимоги вебзастосунку з достатньою продуктивністю. PHP має велику базу користувачів і безліч готових рішень, таких як фреймворки (Laravel, Symfony), які можуть значно спростити розробку вебдодатків. PHP також має потужну підтримку баз даних і різноманітні бібліотеки для роботи із зображеннями, текстом та іншими типами даних, які можуть бути корисні для розробки вебсайтів. Go (Golang), з іншого боку, є компільованою мовою з фокусом на простоту та ефективність. Він має вбудовану підтримку багатопоточності та хорошу продуктивність завдяки своїй низькорівневій природі. Go також відомий своїм простим і зрозумілим синтаксисом, що може полегшити розробку і підтримку коду. У підсумку, вибір між PHP і Go для розробки вебсайту буде залежати від ваших уподобань, досвіду і доступних ресурсів. Якщо у вас є досвід роботи з PHP або перевага використовувати широко використовувані фреймворки, PHP може бути хорошим вибором. Якщо ви хочете використовувати компільовану мову з акцентом на ефективність і паралельне виконання, Go може бути більш підходящим варіантом.

Вибір засобу для написання вебсайту залежить від його наповнення і очікуваних функцій, а також від ваших уподобань, досвіду та вимог проекту. Зрештою, при створенні вебсайту обидві мови, PHP і Go, можуть бути досить ефективними. Для досягнення максимальної продуктивності й ефективності при розробці вебсайту нами було обрано Go.

Для успішної веборієнтованої системи критично важливо мати хорошу архітектуру. При написанні вебзастосунку виникає питання — який саме архітектурний стиль обрати для ефективної роботи, безпеки та швидкості. Одним з найпопулярніших архітектурних стилів є RESTful, який базується на

принципах роботи з мережевими протоколами для доступу до інформаційних ресурсів. Однак в останні роки став розповсюдженим стиль gRPC, який виступає надійною альтернативою RESTful API та пропонує безліч переваг для покращення продуктивності, ефективності та універсальності системи.

В основі gRPC лежить використання буферів протоколів (protobuf), протоколу бінарної серіалізації, що не залежить від мови. Protobuf пропонує компактний та ефективний формат для обміну даними, значно зменшуючи затримку мережі порівняно з текстовим форматом JSON, який використовується в RESTful API.

Архітектура gRPC також використовує можливості HTTP/2 для підтримки мультиплексування та двонаправленої потокової передачі. Це забезпечує одночасний зв’язок між клієнтом і сервером, покращуючи обмін даними в реальному часі та зменшуючи накладні витрати на встановлення кількох з’єднань. Це різко контрастує з RESTful API, які працюють на HTTP/1.1 і обмежені моделлю запит-відповідь.

Ще одна перевага gRPC — підтримка кількох мов програмування, що сприяє більш інклюзивному та універсальному середовищу розробки. Це доповнюється використанням мови визначення інтерфейсу (IDL) для визначення послуг, сприяння чіткому виконанню контрактів і зниження ризику неузгодженості в інтеграції API.

Нарешті, вбудовані функції gRPC, такі як розповсюдження термінів і контроль потоку, надають розробникам більший контроль над мережевими ресурсами та підвищують надійність системи. Ці аспекти малюють чітку, детальну картину того, чому gRPC може бути більш вигідним вибором для певних програм, особливо тих, які вимагають високої продуктивності, ефективного обміну даними та підтримки різноманітних мов програмування.

В рамках проведеного дослідження розглянуто застосування архітектурного стилю gRPC для побудови вебсайту на прикладі кулінарного блогу. На початковому етапі у застосунку були визначені gRPC сервіси користувачів, адміністраторів, рецептів, інгредієнтів та коментарів тощо.

Наступним кроком згенеровано код за допомогою компілятора protoc, тобто було визначено методи мікросервісів для реалізації. Кожен метод представляє віддалений виклик процедури (RPC), який буде доступний іншим сервісам. Таким чином після генерації коду отримано read-only інтерфейс клієнта, який в подальшому є можливість реалізувати на зручній для розробника мові програмування, яку підтримує gRPC.

Якщо важлива ефективність, асинхронна взаємодія та суворий контракт, то gRPC може бути більш вдалим вибором. В той час як RESTful підходить для більш простих проектів, що легко розуміються і розширюються. Важливо також враховувати, що ці два стилі не є взаємовиключними, кожен з них можна використовувати для певних аспектів системи, тобто деякі проекти можуть навіть комбінувати обидві технології в залежності від контексту та потреб. До того ж, на відміну від REST, gRPC — рішення, яке не залежить від платформи та мови. Отже, для реалізації даного проекту було використано RESTful.

2.3 Огляд бази даних PostgreSQL для розробки кулінарного блогу

PostgreSQL (часто званий просто Postgres) — це потужна, відкрита реляційна база даних з акцентом на розширюваність та відповідність стандартам. Вона надає надійне зберігання та управління структурованими даними, дозволяючи розробникам створювати та масштабувати програми [6].

Ось деякі ключові особливості PostgreSQL:

- Надає реляційну модель зберігання даних, дозволяючи організовувати інформацію в таблиці з певними відносинами та зв’язками.
- Має потужну систему розширень, що дозволяє додавати нові функціональні можливості, типи даних та оператори. Це робить його гнучким та здатним адаптуватися до різних вимог застосунків.

- PostgreSQL активно підтримує безліч стандартів SQL та забезпечує відповідність ANSI SQL. Він також пропонує розширені можливості, такі як віконні функції, загальні таблиці виразів (CTE) та багато іншого.
- Підтримує масштабування як вертикально (підвищення продуктивності одному сервері) і горизонтально (розподіл даних на кілька серверів). Це дозволяє збільшувати пропускну здатність та обробляти великі обсяги даних.
- Забезпечує ACID-властивості (атомарність, узгодженість, ізольованість та довговічність) для забезпечення надійності та цілісності даних. Він підтримує транзакції, точки збереження (savepoints) та механізми відновлення після збоїв.
- PostgreSQL сумісний з багатьма мовами програмування та надає інтерфейси для роботи з ними, включаючи Python, Java, C++, PHP та інші.

Приклади використання PostgreSQL для розробки кулінарного блогу:

Зберігання рецептів. Можна створити таблицю «Рецепти» з колонками, такими як «Назва», «Інгредієнти», «Інструкції» та «Категорії». PostgreSQL дозволяє зберігати структуровані дані, такі як ці рецепти, і забезпечує швидкий доступ до них через SQL-запити.

Керування користувачами. Можна створити таблицю «Користувачі» для зберігання даних про користувачів вашого блогу, включаючи імена, адреси електронної пошти та паролі. PostgreSQL має вбудовану підтримку шифрування паролів, що допоможе забезпечити безпеку ваших користувачів.

Коментарі та відгуки. Можна створити таблицю «Коментарі» для зберігання коментарів і відгуків користувачів щодо рецептів. Кожен коментар може мати посилання на відповідний рецепт та користувача. PostgreSQL дозволяє зберігати зв'язки між таблицями, що допоможе вам встановити зв'язок між коментарями та рецептами.

Пошук і фільтрація. PostgreSQL надає потужну мову запитів SQL, яку можна використовувати для пошуку і фільтрації рецептів за різними

критеріями, наприклад, за назвою або категорією, використовуючи запити SELECT з використанням WHERE-умов.

Резервне копіювання та відновлення. PostgreSQL надає засоби для резервного копіювання бази даних, що є важливим для забезпечення безпеки вашої інформації. Можна регулярно створювати резервні копії бази даних і відновлювати їх у разі потреби.

2.4 Висновки до розділу 2

Отже, у дослідженні було розглянуто основні засоби розробки вебсайтів та надано характеристики технологіям, що використовувались у процесі створення тематичного блогу. Для побудови зручного інтерфейсу користувача використовуються HTML — для створення структури сторінок, CSS — для стилізації та вигляду та JavaScript — для взаємодії та динамічної поведінки. Для розробки серверної частини було обрано мову програмування Go (Golang), що характеризується ефективністю, швидкодією та простотою використання. Базою даних є PostgreSQL — потужна, відкрита реляційна база даних з акцентом на розширюваність, надає надійне зберігання та управління структурованими даними, дозволяючи розробникам створювати та масштабувати програми. Вибір перелічених технологій забезпечує високоякісне та продуктивне розроблення вебсайту з можливістю оптимізації продуктивності та підвищення масштабованості.

РОЗДІЛ 3 РОЗРОБКА ВЕБСАЙТУ НА ТЕМУ «КУЛІНАРНИЙ БЛОГ»

3.1 Основний функціонал кулінарного блогу

Проаналізувавши сучасні вебсайти з рецептами було виявлено їх спільні риси та слабкі сторони, які ми врахували під час розробки власного кулінарного блогу. Окрім розміщення різноманітних рецептів блог також має на меті надавати кулінарні поради, надихати людей експериментувати зі стравами, створити спільноту любителів готування. Крім того під час розробки сайту особливу увагу було приділено адаптивності, розподілу рецептів по категоріям, навігації та пошуку по сайту для покращення користувацького досвіду.

Тому кулінарний бог «KitcheNerd» містить такі необхідні функціональні можливості:

- Хедер сайту: містить необхідні для навігації розділи («головна сторінка», «категорії рецептів», «блог», «контакти» тощо).
- Головна сторінка: починається зі слайдеру з випадковими рецептами, що оновлюються щодня, нижче розміщено рецепти з актуальними сезонними стравами, а також корисні поради із розділу «блог».
- Створення особистого профілю: користувачі можуть створювати свої профілі на сайті, де можна зберігати улюблені рецепти, коментувати та оцінювати рецепти, обмінюватися думками з іншими користувачами та ін.
- Пошук по сайту: пошук за ключовими словами також є корисною функцією для навігації по сайту.
- Сторінка з рецептами: всі рецепти поділені на категорії, після вибору категорії доступний перелік рецептів.
- Контакти: користувачі мають змогу надіслати власні рецепти для розміщення їх на сайті через форму зворотного зв’язку, а також підписатись на сторінку блогу у соціальних мережах, посилання на які закріплено у футері.

- Блог: у цьому розділі розміщено корисні поради стосовно догляду за кухнею та приготування страв, а також статті з новинами та трендами для натхнення користувачів.

Крім того, вебсайт «KitcheNerd» відповідає наступним ергономічним вимогам: використання чітких зображень, легка та зрозуміла навігація, адаптивний дизайн, який забезпечує оптимальний перегляд та взаємодію на різних пристроях (комп'ютери, планшети та мобільні телефони), інтуїтивний інтерфейс, мінімалістичний та збалансований дизайн, що не завантажує користувача зайвими деталями, підтримка основних веббраузерів та забезпечення сумісності з різними платформами, зручні форми для вводу даних тощо.

Нижче наведено схематичне зображення головної сторінки вебсайту, яка візуалізує її структуру та компоненти (див. Рис. 3.1).

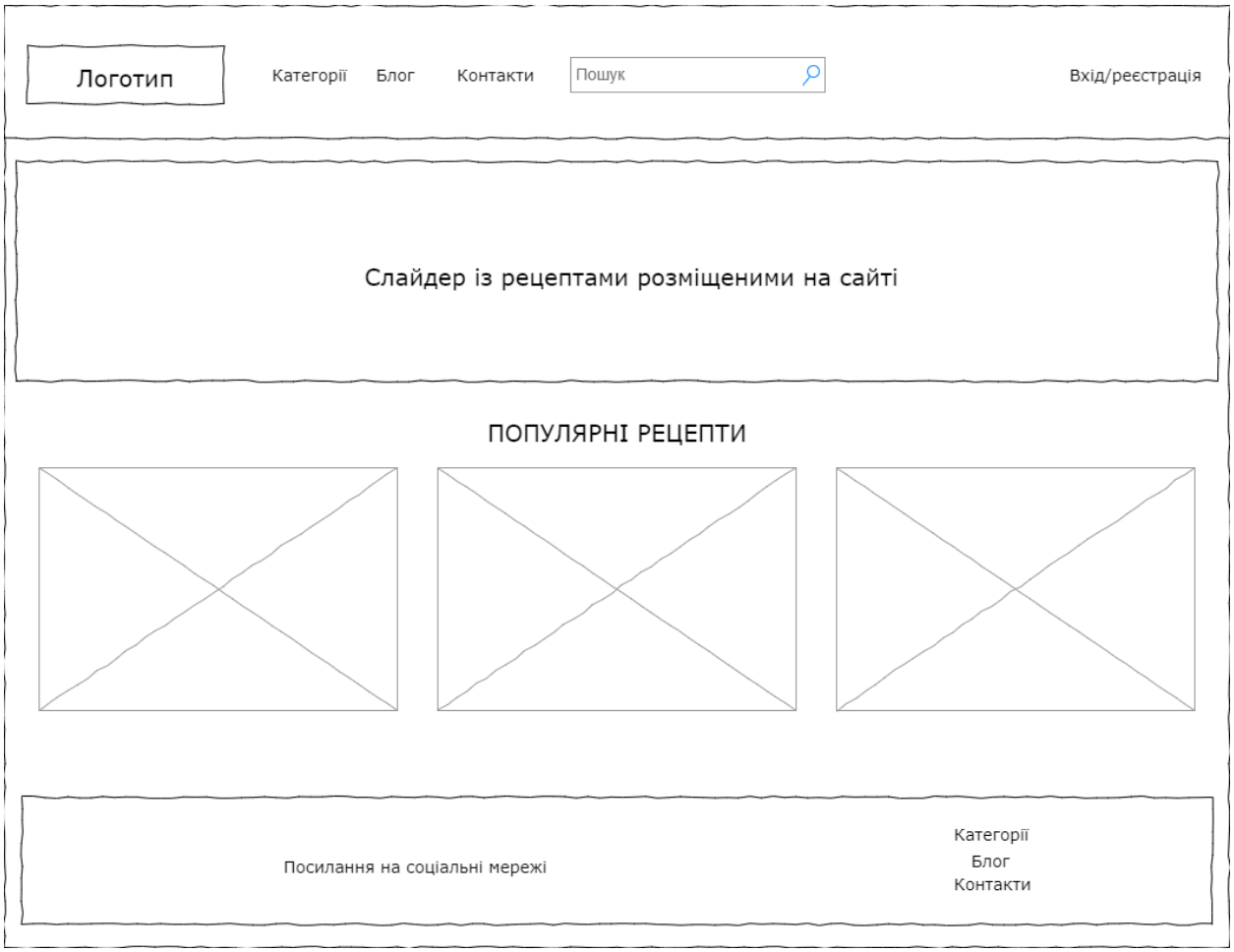

Рисунок 3.1 — *Wireframe для початкової сторінки сайту*

3.2 Проектування вебсайту «Кулінарний блог «KitcheNerd»

Для того аби окреслити основні компоненти, що містить вебсайт було розроблено логічну структуру кулінарного блогу (див. Рис. 3.2). Ця структура має на меті організацію інформації так, щоб користувачі легко знаходили та розуміли контент.

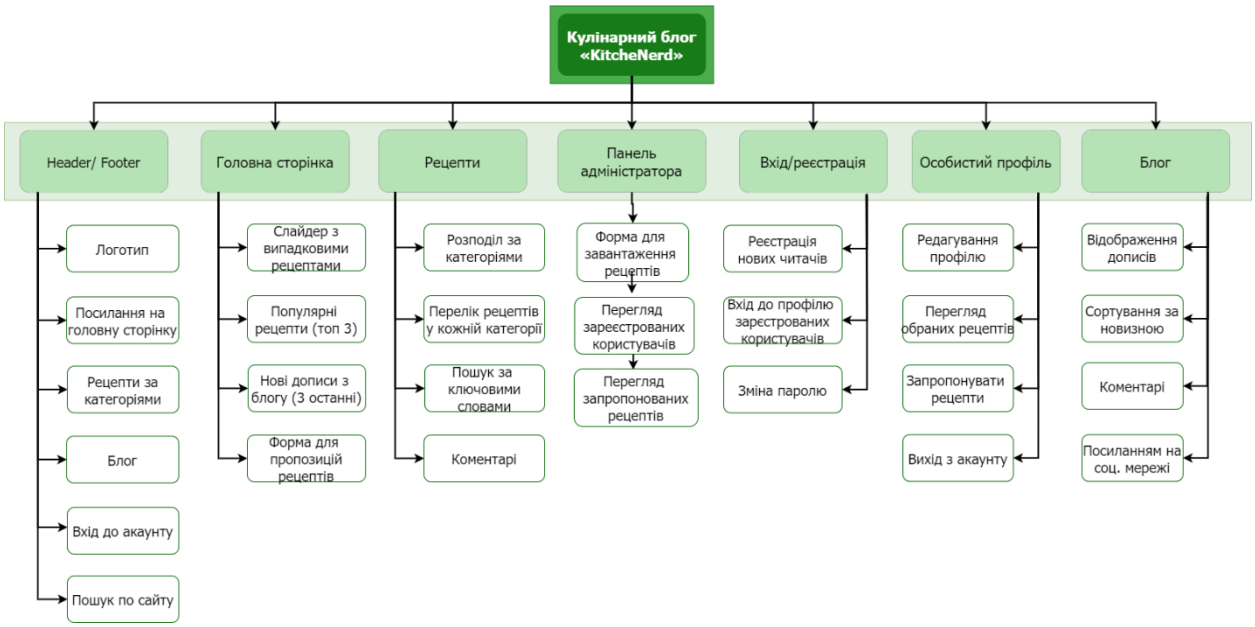

Рисунок 3.2 — *Логічна структура вебсайту «KitcheNerd»*

Основними засобами для навігації сайтом слугують header та footer, що містять посилання на головні розділи сайту, а також контакти автора блогу та зовнішні посилання на додаткові соціальні мережі.

На головній сторінці розміщено популярні або останні рецепти та кулінарні поради, та форма для надання власних пропозицій рецептів.

На сторінці «входу» нові відвідувачі мають можливість зареєструватись на сайті, а зареєстровані користувачі входять до профілю або можуть змінити пароль.

Після входу в особистий профіль користувач має змогу редагувати його, переглядати збережені рецепти та надсилати пропозиції рецептів адміністратору. Адміністратор в свою чергу може переглядати надіслані

пропозиції, керувати користувачами та завантажувати нові рецепти на сторінку рецептів та нові дописи у блог.

Одним з етапів розробки блогу було створення UML-діаграми випадків використання (див. Рис. 3.3). Діаграма варіантів використання UML (Unified Modeling Language — уніфікована мова моделювання) — це візуальне представлення взаємодії між акторами (користувачами) і системою, що розглядається. Вона відображає функціональність або поведінку системи з точки зору користувача.

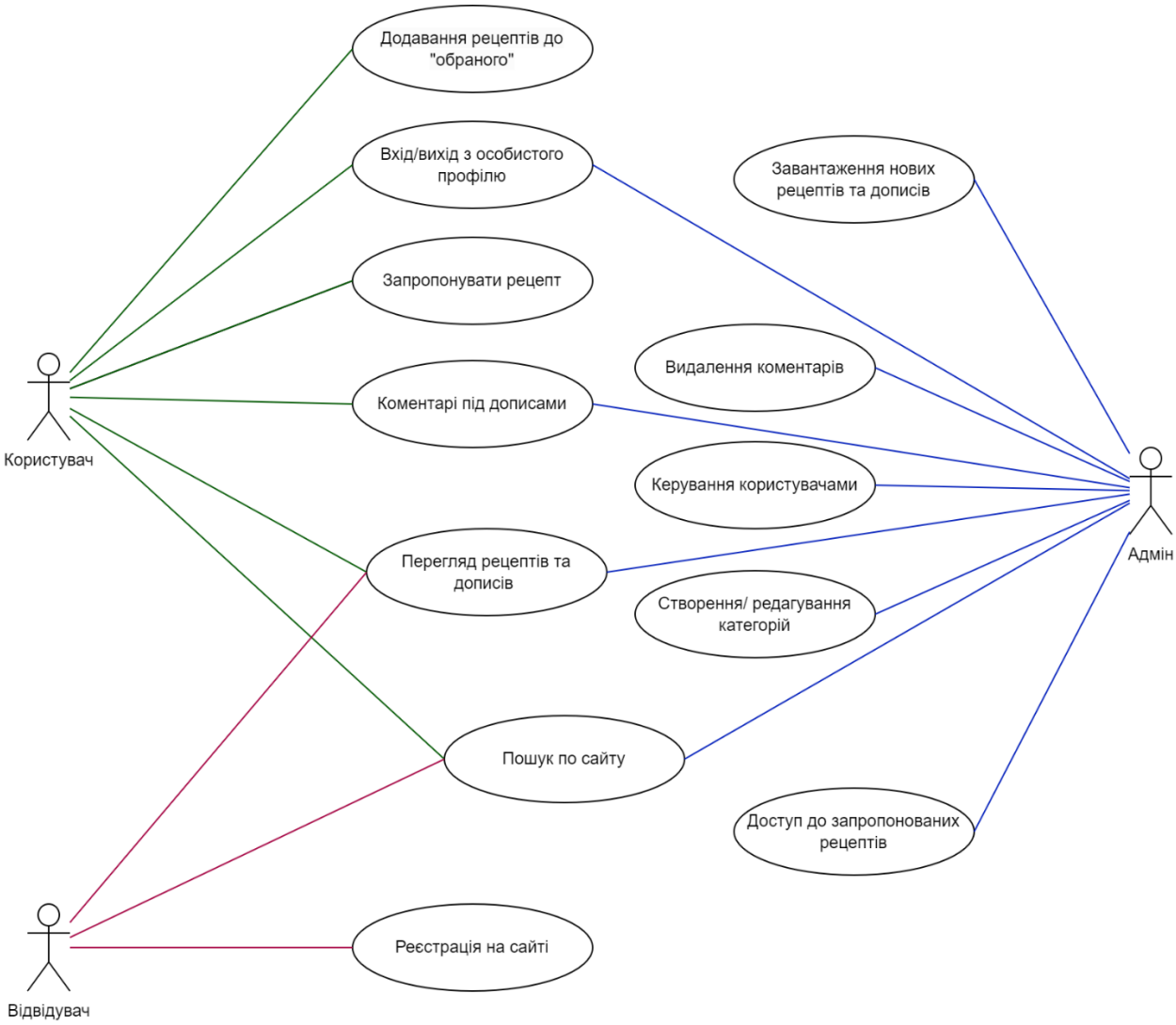

Рисунок 3.3 — *Use case діаграма вебсайту «KitcheNerd»*

Діаграми варіантів використання відображають функціональні вимоги до системи і допомагають визначити, як різні актори взаємодіють з системою для досягнення конкретних цілей або завдань. Діаграми варіантів використання надають огляд функціональності системи, показуючи різні функції або можливості, які вона пропонує, і те, як користувачі або зовнішні системи взаємодіють з нею. Вони слугують інструментом комунікації між зацікавленими сторонами, допомагаючи прояснити вимоги, визначити межі системи, а також підтримувати процеси розробки і тестування [13].

Тож, як було описано вище, кулінарний блог надає функціонал для створення облікового запису користувача, розміщення нових дописів, коментування, можливість надсилання пропозицій з рецептами, управління профілем користувача тощо. Ці основні характеристики блогу включають авторизацію користувачів для доступу до функцій вебсайту. Головними акторами є адміністратор блогу (керує користувачами та дописами), користувачі вебсайту (зареєстровані) та відвідувачі вебсайту (незареєстровані).

Випадки використання виключно для користувачів: додавання рецептів до «обраного» — коли користувач має бажання зберегти рецепт, є можливість додати його до особистого профілю помітивши символом серця, а також доступ до форми «запропонувати рецепт», який може надалі буде прийнятий або відхилений адміністратором для розміщення на сайті.

Випадки використання виключно для адміністратора: завантаження нових рецептів (для цього необхідно додати фото, опис, інгредієнти, етапи приготування), додавання статей у блог (кожна стаття має заголовок і тіло), видалення коментарів залишених користувачами, отримання рецептів надісланих користувачами та відвідувачами вебсайту, керування користувачами (обмеження/блокування особистих профілів).

Спільні функції для адміністратора та користувача: вхід до особистого профілю за допомогою зареєстрованого імені користувача та пароля, доступ

до свого профілю для подальших налаштувань, можливість залишати коментарі під дописами та рецептами у блозі.

Випадки використання доступні для всіх акторів: перегляд завантажених рецептів, а також пошук по всьому блогу і доступ до опублікованих статей.

Для кращого розуміння взаємодії між об'єктами було розроблено діаграму класів (див. Рис. 3.4).

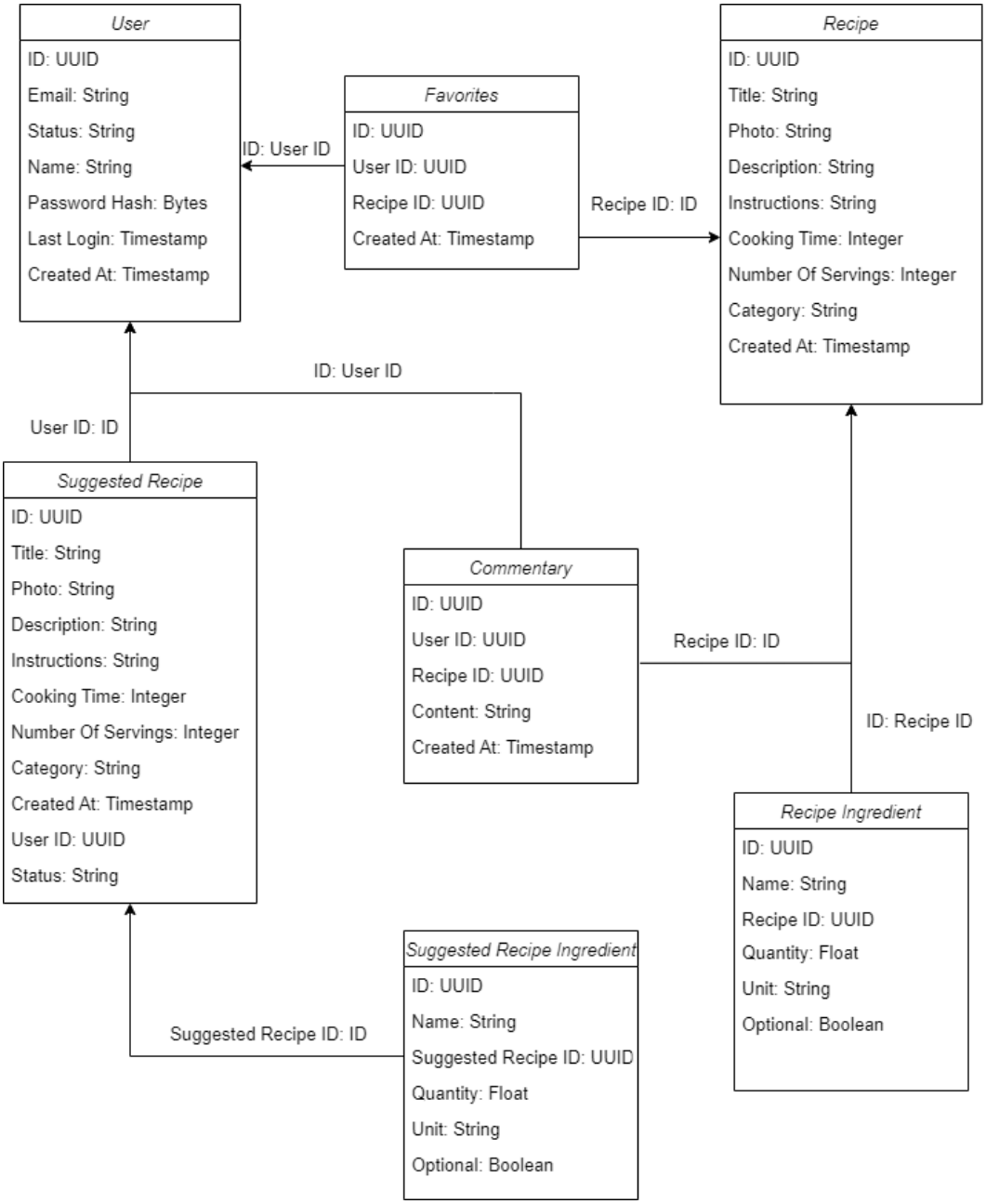

Рисунок 3.4 — *Діаграма класів вебсайту «KitcheNerd»*

Діаграма класів — це структурна діаграма, яка відображає класи, їх атрибути, методи та їх взаємодії в системі. Класи представлені прямокутниками з назвою вгорі, розділеним на три частини: ім'я класу, атрибути та методи. Атрибути вказують на властивості класу. Методами є операції або функції, які може виконувати клас. З'єднання між класами, які вказують на взаємодію або спадкування.

У наведеній діаграмі класів: клас User представляє користувачів системи, клас Recipe представляє рецепти, клас Recipe Ingredient представляє інгредієнти рецепту та їхні кількості та одиниці вимірювання, клас Favorites представляє рецепти, які були відмічені користувачами як обрані, клас Suggested Recipe представляє рецепти, які можуть бути запропоновані користувачами, клас Suggested Recipe Ingredient представляє інгредієнти запропонованого рецепту та їхні кількості та одиниці вимірювання, клас Commentary представляє коментарі до рецептів.

3.3 Програмна реалізація вебсайту

Головним завданням вебсайту є розміщення рецептів від імені адміністратора або надсилання пропозицій рецептів від користувачів. Розглянемо код, що створює запропонований рецепт і пов’язані з ним інгредієнти, а також взаємодіє з базою даних для зберігання даних.

Лістинг 1 визначає структуру `RecipeRequest`, яка представляє запит на створення рецепту. Він містить такі поля, як назва, фото, опис, інструкції, час приготування, кількість порцій, категорія, інгредієнти та ідентифікатор користувача. Теги json використовуються для вказівки назв полів JSON під час маршалінгу та демаршалінгу даних JSON.

Лістинг 1 — Структура запиту на створення рецепту

```
type RecipeRequest struct {
    Title    string        `json:"title"`
    PhotoBase64   string   `json:"photoBase64"`
```

```
    Description   string    `json:"description"`
    Instructions  string    `json:"instructions"`
    CookingTime   int       `json:"cookingTime"`
    NumberOfServings  int  `json:"servings"`
    Category  recipes.Category  `json:"category"`
    Ingredients   []recipes.RecipeIngredient
`json:"ingredients"`
    UserID   uuid.UUID     `json:"userID"`
}
```

Лістинг 2 у обробнику HTTP декодує вхідне тіло запиту JSON у структуру `RecipeRequest`. Потім він створює об’єкт `SuggestedRecipe` із декодованих даних запиту та викликає метод `SuggestRecipe` (див. Лістинг 3) структури `Service` для обробки створення запропонованого рецепту та пов’язаних з ним інгредієнтів.

Лістинг 2 — Зв’язок між сервером та користувачем для пропонування створення нового рецепту

```
ctx := r.Context()
var request *RecipeRequest
if err := json.NewDecoder(r.Body).Decode(&request); err !=
nil {
   c.serveError(w, http.StatusBadRequest,
ErrUsers.Wrap(err))
   log.Println(err)
   return
}

recipe := recipes.NewSuggestedRecipe(request.Title,
request.PhotoBase64, request.Description,
request.Instructions, request.CookingTime,
request.NumberOfServings, request.Category, request.UserID)

err := c.recipes.SuggestRecipe(ctx, recipe,
request.Ingredients)
if err != nil {
       c.serveError(w, http.StatusInternalServerError,
ErrUsers.Wrap(err))
       log.Println(err)
       return
}
```

Метод `SuggestRecipe` у структурі `Service` спочатку встановлює інгредієнти рецепту, а потім викликає метод `CreateSuggestedRecipe` структури `recipesDB`, щоб вставити запропонований рецепт у базу даних (див. Лістинг 3). Він також викликає метод `CreateSuggestedIngredients` для створення пов’язаних інгредієнтів.

Лістинг 3 — Метод сервісу, що зв’язує контролер з базою даних

```
func (service *Service) SuggestRecipe(ctx context.Context,
recipe *SuggestedRecipe, ingredients []RecipeIngredient)
error {
   recipe.Ingredients = ingredients
   err := service.recipes.CreateSuggestedRecipe(ctx,
recipe)
   if err != nil {
      return err
   }

   err = service.CreateSuggestedIngredients(ctx, recipe.ID,
recipe.Ingredients)
   if err != nil {
      return err
   }

   return nil
}
```

Метод `CreateSuggestedRecipe` у структурі `recipesDB` вставляє запропонований рецепт у таблицю `suggested_recipes` бази даних за допомогою запиту `SQL INSERT`. Якщо під час роботи з базою даних виникає помилка, вона повертає помилку, обгорнуту спеціальним типом помилки.

Лістинг 4 — Метод бази даних для створення запропонованого рецепту

```
func (s *recipesDB) CreateSuggestedRecipe(ctx
context.Context, recipe *recipes.SuggestedRecipe) error {
   recipe.CreatedAt = time.Now()

   _, err := s.pool.Exec(ctx, "INSERT INTO
suggested_recipes (id, title, photo, description,
instructions, cooking_time, number_of_servings, category,
created_at, user_id, status) VALUES ($1, $2, $3, $4, $5,
```

```
$6, $7, $8, $9, $10, $11)",
      recipe.ID, recipe.Title, recipe.PhotoBase64,
recipe.Description, recipe.Instructions,
recipe.CookingTime, recipe.NumberOfServings,
recipe.Category, recipe.CreatedAt, recipe.UserID,
recipe.Status)
   if err != nil {
      return ErrRecipes.Wrap(err)
   }

   return nil
}
```

Також однією з важливих функцій є збереження користувачами обраних рецептів. Лістинг 5 реалізує HTTP обробник для дій з обраною стравою (рецептом) користувача у системі рецептів. Основними діями, які він обробляє, є додавання рецепту до обраних та видалення його з обраних.

Лістинг 5 — Зв’язок між сервером та користувачем для додавання або видалення рецептів до обраних

```
func (c *Recipes) Favorite(w http.ResponseWriter, r
*http.Request) {
   ctx := r.Context()
   vars := mux.Vars(r)
   id, err := uuid.Parse(vars["id"])
   if err != nil {
      c.serveError(w, http.StatusBadRequest, err)
      return
   }

   var request struct {
      UserID uuid.UUID `json:"userID"`
   }
   if err := json.NewDecoder(r.Body).Decode(&request); err
!= nil {
      log.Println(err)
      c.serveError(w, http.StatusBadRequest,
ErrRecipes.Wrap(err))
      return
   }

   switch r.Method {
   case http.MethodPost:
      err = c.recipes.CreateFavorite(ctx, request.UserID,
id)
```

```
      if err != nil {
         c.serveError(w, http.StatusInternalServerError,
ErrRecipes.Wrap(err))
         return
      }
   case http.MethodDelete:
      err = c.recipes.DeleteFavorite(ctx, request.UserID,
id)
      if err != nil {
         c.serveError(w, http.StatusInternalServerError,
ErrRecipes.Wrap(err))
         return
      }
   }

}
```

У методі `Favorite` структура `Recipes` реалізує метод `Favorite`, який є HTTP обробником. Він отримує дані від клієнта через `http.Request`, а потім в залежності від методу запиту (`POST` або `DELETE`), викликає відповідний метод обробки в базі даних (додати до обраних або видалити з обраних). Параметри запиту визначаються як `UserID`, що отримується через розпакування JSON тіла запиту.

У випадку `POST` запиту, метод `CreateFavorite` викликається для додавання рецепту до обраних (див. Лістинг 6). Він використовує підготовлений SQL-запит для вставки нового запису в базу даних, представленої таблицею `favorites`.

Лістинг 6 — Метод бази даних для додавання обраних рецептів

```
func (s *recipesDB) CreateFavorite(ctx context.Context,
favorite *recipes.Favorite) error {
   query := `INSERT INTO favorites (id, user_id, recipe_id,
created_at)
             VALUES ($1, $2, $3, $4)`
   _, err := s.pool.Exec(ctx, query, favorite.ID,
favorite.UserID, favorite.RecipeID, favorite.CreatedAt)
   if err != nil {
      return ErrRecipes.Wrap(err)
   }
```

```
    return nil
}
```

У випадку `DELETE` запиту, метод `DeleteFavorite` викликається для видалення рецепту з обраних (див. Лістинг 7). Він використовує підготовлений SQL-запит для видалення запису з таблиці `favorites`.

Лістинг 7 — Метод бази даних для видалення обраних рецептів

```
func (s *recipesDB) DeleteFavorite(ctx context.Context,
userID, recipeID uuid.UUID) error {
    query := `DELETE FROM favorites WHERE user_id = $1 AND
recipe_id = $2`
    _, err := s.pool.Exec(ctx, query, userID, recipeID)
    if err != nil {
        return ErrRecipes.Wrap(err)
    }

    return nil
}
```

В обох випадках, якщо відбувається помилка під час виконання SQL-запиту, вона обгортається в екземпляр `ErrRecipes` і повертається. Цей код показує обробку HTTP запитів, які маніпулюють даними в базі даних. Він використовує підготовлені SQL-запити для запобігання SQL-ін'єкціям та використовує JSON для обміну даними між клієнтом і сервером.

Для отримання доступу до перелічених вище функцій сайту користувач має авторизуватись. Лістинг 8 перевіряє чи існує сесія з іменем `login`, якщо не існує, то код перенаправляє на сторінку логіну. Далі ми перевіряємо чи є у цієї сесії значення `userID` користувача, що намагається зайти на сторінку, на якій стоїть middleware і якщо все успішно, то користувачу буде дозволено на неї зайти. Middleware діє як посередник між різними компонентами програмного забезпечення. Це програмне забезпечення дозволяє виконувати різноманітні завдання, такі як обробка запитів, маніпулювання даними, фільтрація вхідних та вихідних даних, аутентифікація, авторизація та багато

іншого. Якщо користувач не автентифікований або виникає помилка, він перенаправляється на сторінку входу в систему за допомогою редіректу HTTP.

Лістинг 8 — Middleware перевірка наявності користувача в сесії

```
func (c *Auth) AuthMiddleware(handler http.Handler)
http.Handler {
   return http.HandlerFunc(func(w http.ResponseWriter, r
*http.Request) {
      ctx := r.Context()
      session, err := Store.Get(r, "login")
      if err != nil {
         c.serveError(w, http.StatusInternalServerError,
ErrAuth.Wrap(err))
         return
      }

      userID, ok := session.Values["user_id"].(string)
      if !ok {
         log.Println(ErrAuth.New("could not get value from
session"))
         http.Redirect(w, r, "/auth/login",
http.StatusSeeOther)
         return
      }

      userUUID, err := uuid.Parse(userID)
      if err != nil {
         log.Println(ErrAuth.Wrap(err))
         http.Redirect(w, r, "/auth/login",
http.StatusSeeOther)
         return
      }

      ctx = context.WithValue(ctx, KeyUserID, userUUID)
      handler.ServeHTTP(w, r.Clone(ctx))
   })
}
```

3.4 Висновки до розділу 3

Отже, кулінарний блог «KitcheNerd» було розроблено на підставі проведеного аналізу сучасних вебсайтів з рецептами. Основними функціональними можливостями сайту є створення особистого профілю,

пропозиції рецептів (для адміністратора можливість прийняти/відхилити), додавання рецептів до списку обраного, пошук по сайту, коментування рецептів.

Вебсайт відповідає сучасним ергономічним вимогам, включаючи чіткі зображення, легку навігацію, інтуїтивний інтерфейс, адаптивний та мінімалістичний дизайн. Головні елементи навігації, хедер та футер, містять посилання на головні розділи сайту та контактні дані автора блогу. Головна сторінка демонструє популярні рецепти, останні рецепти та дописи у блозі.

Було наведено діаграму випадків використання сайту, що допомагає визначити структуру та функціональність системи. А у діаграмі класів представлено визначені класи, такі як User, Recipe, Recipe Ingredient, Suggested Recipe, Suggested Recipe Ingredient, Commentary та інші, що визначають основні об'єкти системи та їхні взаємовідносини.

В підрозділі реалізації програмного забезпечення було описано кілька ключових етапів, включаючи створення та обробку рецептів, управління обраними рецептами, аутентифікацію та авторизацію користувачів. Кодові лістинги і пояснення надають уявлення про те, як функції були реалізовані на рівні коду. Загалом, розроблений сайт враховує потреби користувачів, забезпечуючи не тільки рецепти, але й кулінарні поради та спільноту.

РОЗДІЛ 4 РЕАЛІЗАЦІЯ ВЕБСАЙТУ НА ТЕМУ «КУЛІНАРНИЙ БЛОГ» З ВИКОРИСТАННЯМ МОВИ ПРОГРАМУВАННЯ GOLANG

4.1 Огляд користувацького інтерфейсу вебсайту

Результатом проведеного дослідження є розроблений кулінарний блог. Нижче представлено користувацький інтерфейс основних вебсторінок сайту. Головна сторінка вебсайту є ключовим елементом, що вражає користувачів і визначає їхнє перше враження від сайту. Перейти на початкову сторінку можна натиснувши на логотип, що розміщений у лівому верхньому кутку хедера. Хедер в свою чергу відображається на всіх сторінках та є основною навігаційною панеллю для користувачів. Тут вони можуть перейти до переліку всіх рецептів або окремої категорії, переглянути дописи у блозі, робити пошук за ключовими словами або увійти до особистого профілю.

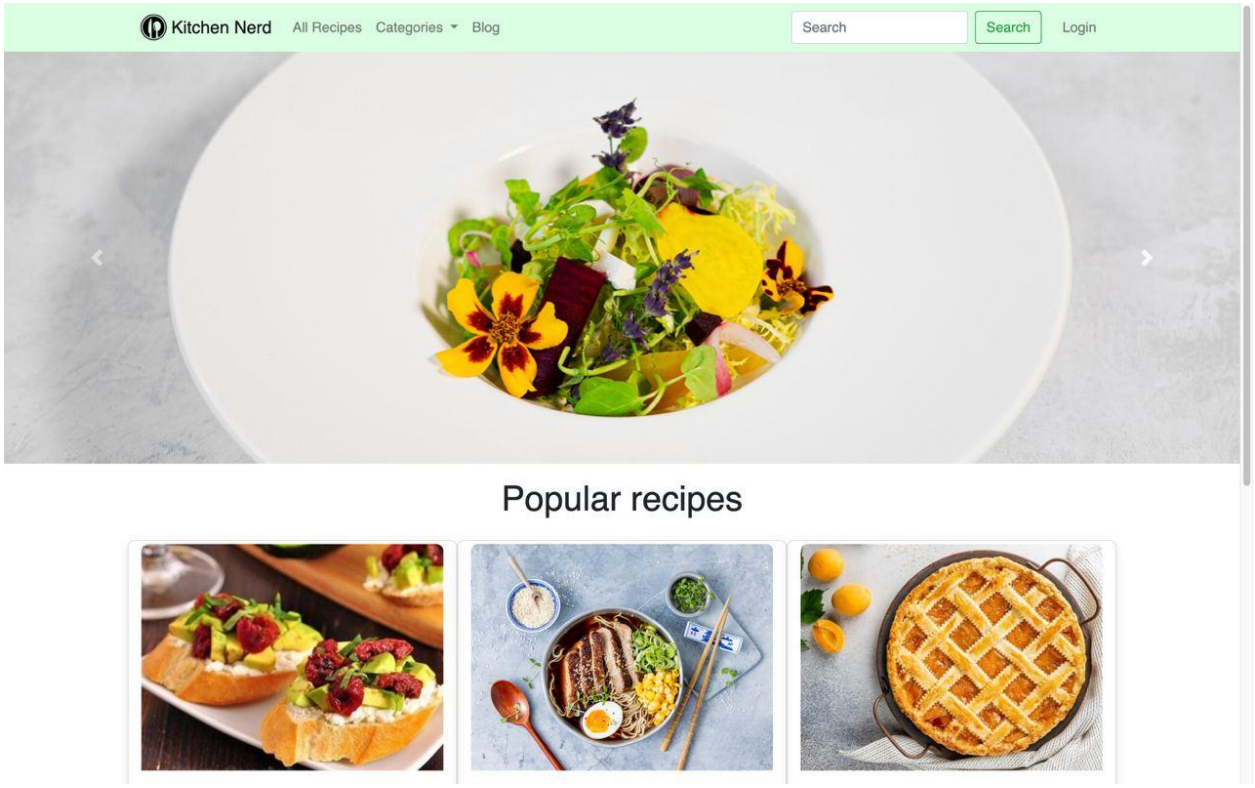

Рисунок 4.1 — *Інтерфейс початкової сторінки сайту*

Також на початковій сторінці розміщено слайдер з випадковими рецептами, що обрав автор блогу, після чого відображаються популярні рецепти. Популярними рецептами є топ 3 з усіх «вподобаних» рецептів, що були додані користувачами до списку обраних. Можливість обрати рецепт є після того як відвідувач сайту створює особистий профіль та заходить на сторінку з рецептом, що сподобався (див Рис. 4.2).

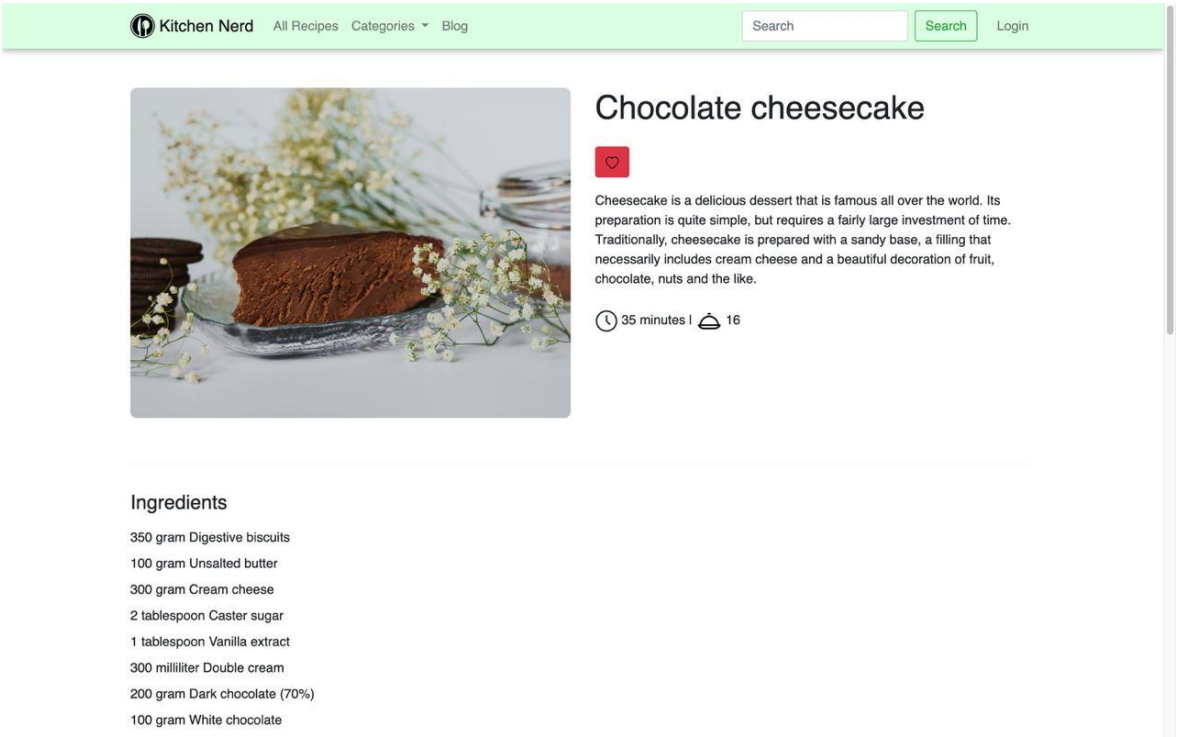

Рисунок 4.2 — *Інтерфейс сторінки з рецептом*

Після входу до облікового запису користувачу доступно більше функцій. Перейшовши на сторінку особистого профілю користувач має доступ до вподобаних рецептів, а також може редагувати особисту інформацію та виходити з профілю (див Рис. 4.3). З метою спрощення навігації серед обраних рецептів було додано пошук за ключовими словами. Однією з функцій що стає доступною лише після реєстрації на сайті є пропонування власного рецепту для публікації на сайті. Після надсилання заповненої форми рецепт стає доступним для адміністратора, який може його відхилити або прийняти.

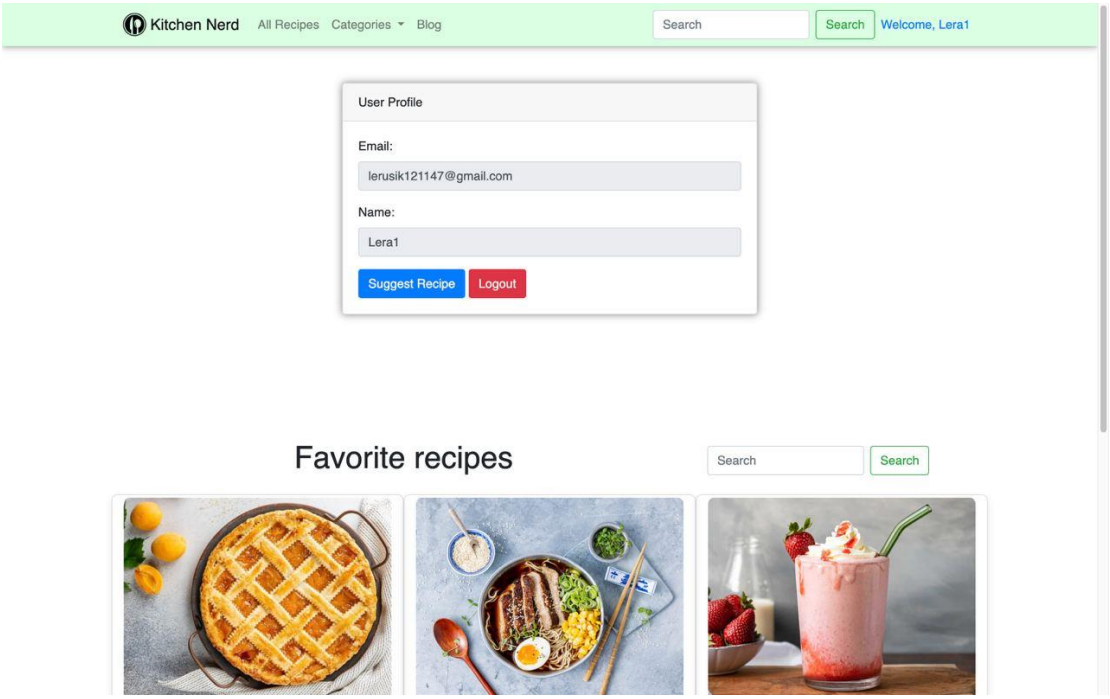

Рисунок 4.3 — *Інтерфейс сторінки особистого профілю користувача*

Адміністратор (автор блогу) має більше функцій, порівняно з користувачем, але вебсторінки виглядають аналогічно. Однією з найважливіших функцій є завантаження рецептів на сайт, сторінка є аналогічною до форми для пропозицій рецептів від користувачів (див. Рис. 4.4).

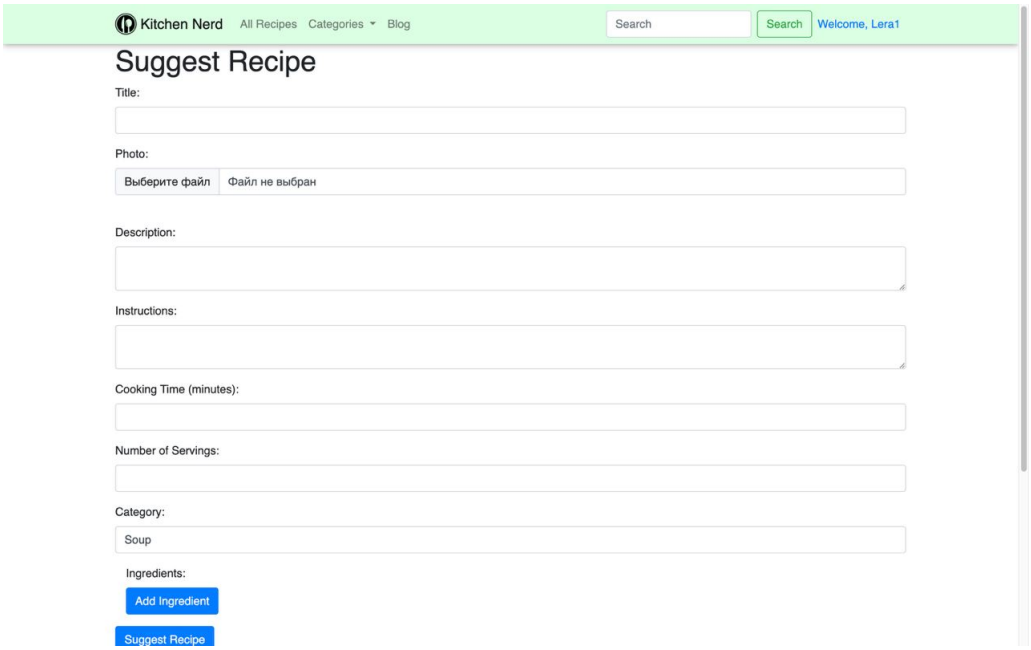

Рисунок 4.4 — *Інтерфейс сторінки для завантаження рецепту*

Також адміністратор має змогу переглядати перелік зареєстрованих користувачів та керувати ними. Як зазначалось раніше адміністратор може приймати або відхиляти запропоновані рецепти для розміщення на сайті. Крім того, адміністратор може видаляти коментарі, залишені користувачами під публікаціями з рецептами.

Додатково під час розробки блогу було використано фреймворк Bootstrap для забезпечення адаптивності вебсайту. На рисунку 4.5 представлено відображення інтерфейсу початкової сторінки сайту на прикладі мобільного пристрою. Отже, користувачі з різними типами пристроїв зможуть користуватись створеною програмою без будь-яких обмежень.

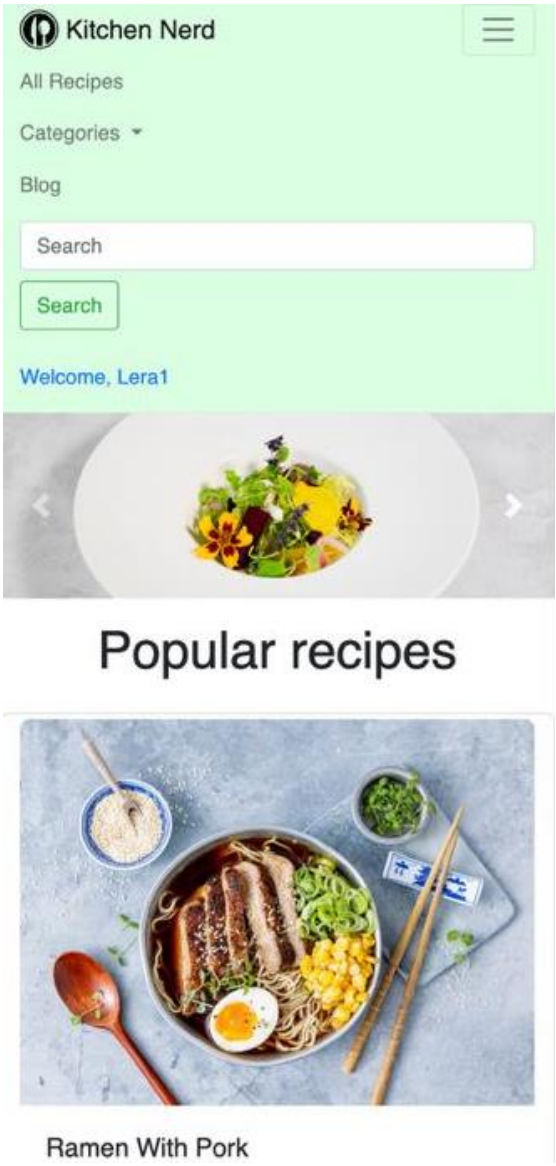

Рисунок 4.5 — *Відображення початкової сторінки на мобільному пристрої*

4.2 Порівняння реалізацій за допомогою мов програмування Golang та PHP

В ході виконання роботи було розроблено програмне забезпечення на мові програмування Golang. За основу сайту було взято створений раніше кулінарний блог на мові PHP, тому зараз на меті є порівняння ефективності розробки сайту на цих двох мовах. Почнемо порівняння на прикладі функції, що відображає найпопулярніші рецепти на сайті — перші три рецепти за кількістю уподобань з усіх, що були обрані користувачами.

Лістинг 9 повертає топ 3 рецепти за кількістю уподобань. Ми маємо сутність `favorites`, цей код рахує усі `recipe_id` у таблиці та сортує від найбільшої кількості уподобань до найменшої та повертає ліміт з трьох перших елементів вибірки (TOP 3).

Лістинг 9 — Метод бази даних для вибірки трьох найпопулярніших рецептів реалізований на Go

```
func (s *recipesDB) Popular(ctx context.Context)
([]*recipes.Recipe, error) {
     query := `SELECT r.id, r.title, r.photo, r.description,
r.instructions, r.cooking_time, r.number_of_servings,
r.category, r.created_at,
       COUNT(f.recipe_id) AS likes_count
FROM recipes r
LEFT JOIN favorites f ON r.id = f.recipe_id
GROUP BY r.id, r.title, r.photo, r.description,
r.instructions, r.cooking_time, r.number_of_servings,
r.category, r.created_at
ORDER BY likes_count DESC
LIMIT 3; `
     rows, err := s.pool.Query(ctx, query)
     if err != nil {
          log.Println(err)
          return nil, ErrRecipes.Wrap(err)
     }
     defer rows.Close()
     var count int
     var list []*recipes.Recipe
     for rows.Next() {
          recipe := new(recipes.Recipe)
```

```
          if err = rows.Scan(&recipe.ID, &recipe.Title,
&recipe.PhotoBase64, &recipe.Description,
&recipe.Instructions, &recipe.CookingTime,
&recipe.NumberOfServings, &recipe.Category,
&recipe.CreatedAt, &count); err != nil {
               log.Println(err)
               return nil, ErrRecipes.Wrap(err)
          }
          ingredients, err := s.GetIngredients(ctx,
recipe.ID)
          if err != nil {
               log.Println(err)

               return nil, ErrRecipes.Wrap(err)
          }
          recipe.Ingredients = ingredients
          list = append(list, recipe)
     }
     return list, nil
}
```

Для порівняння нижче наведено функцію `Popular` написану на PHP (див. Лістинг 10).

Лістинг 10 — Функція для вибірки трьох найпопулярніших рецептів реалізована на PHP

```
<?php
class RecipesDB {
    private $pdo; // Assuming you have a PDO object for
database connection

    public function __construct($pdo) {
        $this->pdo = $pdo;
    }
    public function Popular() {
        $query = 'SELECT r.id, r.title, r.photo,
r.description, r.instructions, r.cooking_time,
r.number_of_servings, r.category, r.created_at,
                          COUNT(f.recipe_id) AS likes_count
                  FROM recipes r
                  LEFT JOIN favorites f ON r.id =
f.recipe_id
                  GROUP BY r.id, r.title, r.photo,
r.description, r.instructions, r.cooking_time,
r.number_of_servings, r.category, r.created_at
```

```
                    ORDER BY likes_count DESC
                    LIMIT 3';
        $stmt = $this->pdo->prepare($query);
        $stmt->execute();
        $list = [];
        while ($row = $stmt->fetch(PDO::FETCH_ASSOC)) {
            $recipe = new Recipe();
            $recipe->ID = $row['id'];
            $recipe->Title = $row['title'];
            $recipe->PhotoBase64 = $row['photo'];
            $recipe->Description = $row['description'];
            $recipe->Instructions = $row['instructions'];
            $recipe->CookingTime = $row['cooking_time'];
            $recipe->NumberOfServings =
$row['number_of_servings'];
            $recipe->Category = $row['category'];
            $recipe->CreatedAt = $row['created_at'];

            $ingredients = $this->GetIngredients($recipe-
>ID);
            $recipe->Ingredients = $ingredients;

            $list[] = $recipe;
        }
        return $list;
    }
    private function GetIngredients($recipeID) {
        // Implement the logic to retrieve ingredients
based on recipeID
        // Return an array of ingredients
    }
}
class Recipe {
    public $ID;
    public $Title;
    public $PhotoBase64;
    public $Description;
    public $Instructions;
    public $CookingTime;
    public $NumberOfServings;
    public $Category;
    public $CreatedAt;
    public $Ingredients;
}

// Usage example:
 $recipesDB = new RecipesDB();
 $popularRecipes = $recipesDB->Popular();
```

```
?>
```

Порівнюючи лістинги 9 та 10 помітно, що код написаний на Го менший за обсягом. Це означає, що у Go використовується більш лаконічний і статичний синтаксис, в той час як PHP має більш багатослівний синтаксис з динамічною типізацією. Go явно повертає помилку як частину сигнатури функції і використовує метод `Wrap` для обгортання помилок, а PHP покладається на виключення для обробки помилок. Обидві мови використовують цикли для циклічного перегляду набору результатів, створення об'єктів рецептів та отримання інгредієнтів.

Також Go компілюється у машинний код, що потенційно забезпечує кращу продуктивність у порівнянні з інтерпретованими мовами, якою є PHP. Бенчмарк-тести — це спеціальні програми або тестові сценарії, розроблені для вимірювання продуктивності комп’ютерної системи або окремих її компонентів.

Нижче наведено код методу `List` на Go. (див. Лістинг 11) В даному методі `List` ми робимо запит до бази даних і отримуємо список усіх рецептів. Після цього у вкладеному циклі ми беремо з бази даних інгредієнти для кожного рецепта. Далі повертаємо назви рецептів з інгредієнтами на frontend.

Лістинг 11 — Реалізація методу для вибірки всіх рецептів на Golang

```
func (s *recipesDB) List(ctx context.Context, filter
*recipes.Filter) ([]*recipes.Recipe, error) {
    query := `SELECT DISTINCT recipes.id, title, photo,
description, instructions, cooking_time,
number_of_servings, category, created_at
              FROM recipes`

    var params []interface{}
    if filter.Category != "" {
        query += ` WHERE category LIKE $1 `
        params = append(params, "%"+filter.Category+"%")
    }
    if filter.Search != "" {
        query += ` LEFT JOIN recipe_ingredients ri on
recipes.id = ri.recipe_id WHERE (lower(title)  LIKE $1) OR
```

```
(lower(instructions) LIKE $1 ) OR (lower(category) LIKE $1)
OR (lower(ri.name) LIKE $1) `
          params = append(params, "%"+filter.Search+"%")
     }
     query += ` ORDER BY created_at DESC `
     rows, err := s.pool.Query(ctx, query, params...)
     if err != nil {
          return nil, ErrRecipes.Wrap(err)
     }
     defer rows.Close()

     var list []*recipes.Recipe
     for rows.Next() {
          recipe := new(recipes.Recipe)
          if err = rows.Scan(&recipe.ID, &recipe.Title,
&recipe.PhotoBase64, &recipe.Description,
&recipe.Instructions, &recipe.CookingTime,
&recipe.NumberOfServings, &recipe.Category,
&recipe.CreatedAt); err != nil {
               return nil, ErrRecipes.Wrap(err)
          }

          ingredients, err := s.GetIngredients(ctx,
recipe.ID)
          if err != nil {
               return nil, ErrRecipes.Wrap(err)
          }

          recipe.Ingredients = ingredients
          list = append(list, recipe)
     }

     return list, nil
}
```

Для дослідження продуктивності кожної з цих мов нами було проведено бенчмарк-тести на прикладі методу `List`. Лістинг 12 відображає код бенчмарк-тесту роботи методу `List` на мові програмування Golang.

Лістинг 12 — Бенчмарк-тест роботи методу для вибірки всіх рецептів, реалізованого на Golang

```
package test
import (
     "context"
     "fmt"
```

```
     "log"
     "time"

     "recipes"
)

func TestListRecipes() {
     recipesDB := &recipes.RecipesDB{}

     startTime := time.Now()

     // Run the List method 1000 times for testing
     for i := 0; i < 1000; i++ {
          _, err := recipesDB. List (context.Background())
          if err != nil {
               log.Fatal(err)
          }
     }

     elapsedTime := time.Since(startTime)
     fmt.Printf("List Method took %s\n", elapsedTime)
}
```

Результат: бенчмарк-тест методу `List` написаного на мові Golang зайняв 0.02 секунди. Після було проведено бенчмарк-тест роботи методу `List` на мові програмування PHP (див. Лістинг 13).

Лістинг 13 — Бенчмарк-тест роботи функції для вибірки всіх рецептів, реалізованої на PHP

```
<?php

require_once 'Recipe.php'; // Include the file containing
the Recipe class
require_once 'RecipesDB.php'; // Include the file
containing the RecipesDB class

$recipesDB = new RecipesDB($your_pdo_object); // Initialize
recipesDB object

$startTime = microtime(true);

// Run the List method 1000 times for testing
for ($i = 0; $i < 1000; $i++) {
    $popularRecipes = $recipesDB->List();
```

```
}

$elapsedTime = microtime(true) - $startTime;
echo "PHP Method took " . $elapsedTime . " seconds\n";
?>
```

Результат бенчмарк-тесту: бенчмарк-тест методу `List` написаного на мові PHP зайняв 0.35 секунд. Як ми можемо побачити, чим більше елементів у циклі, тим більше часу буде витрачатись на виконання. В даному випадку час росте квадратично зі збільшенням вхідних даних.

Тобто при збільшенні об’ємів даних ефективність знижається. Але як ми можемо побачити — Golang справляється з цим в рази швидше на відміну від PHP. А все тому, що PHP має динамічну типізацію, що може призвести до виявлення помилок під час виконання, а не під час компіляції, що може мати критичні наслідки. Головною причиною цього є те, що PHP є інтерпретованою мовою програмування, а Go компільованою, як зазначалось вище. Крім того, горутини в Go надають легкі потоки, що робить багатозадачність більш ефективною та простою для використання.

При розробці вебсайтів на двох цих мовах програмування ми можемо виділити деякі суттєві відмінності. У Go, якщо відбувається помилка, програма може продовжити виконання, обробляючи помилку або ігноруючи її. Це пов'язано з тим, що Go дозволяє обробляти помилки вручну за допомогою механізму повернення помилки (error handling). Це дає більшу гнучкість в керуванні помилками у коді, але вимагає більше уваги обробки помилок та запобігання падінню програми. У PHP, з іншого боку, помилка зазвичай призводить до того, що програма зупиняється, і виводить повідомлення про помилку на екран, вказуючи на конкретний рядок коду, де сталася помилка. Це полегшує виявлення та виправлення помилок, але призводить до переривання роботи програми у разі виникнення помилки.

Golang є безпечнішим, він абсолютно захищений від SQL ін’єкцій, на відміну від PHP. SQL-ін’єкція — це метод отримання несанкціонованого доступу (різновид злому) до бази даних, при якому шкідливий код виконується

прямо з поля вводу звичайної форми. Якщо SQL-ін’єкція пройшла успішно, неавторизовані особи зможуть читати, створювати, оновлювати або навіть видаляти записи з таблиць бази даних.

При роботі з базою даних Go пропонує більш високу продуктивність, завдяки своїй компіляції в машинний код і ефективній реалізації багатопоточності. Це може бути особливо корисно при обробці великих об’ємів даних або високонавантажених застосунків. Тобто використовуючи Go, можна уникнути накладних витрат, зв’язаних з інтерпретацією, яка присутня в PHP. Оскільки Go є статично типізованою мовою, це дозволяє виявити помилки на етапі компіляції, що облегшує рефакторинг коду. Це може зменшити кількість помилок при роботі з базами даних.

4.3 Висновки до розділу 4

В даному розділі було розглянуто користувацький інтерфейс вебсайту кулінарного блогу, який був розроблений на основі проведеного дослідження. Головною метою було створення зручного інтерфейсу для користувачів, що б дозволило їм легко знаходити і переглядати рецепти, а також активно взаємодіяти з контентом блогу. На головній сторінці сайту відображається основний контент, що робить перебування користувача на сайті більш привабливим. Зручна навігаційна панель у хедері дозволяє легко переходити між різними розділами сайту, включаючи сторінки рецептів, блогу та особистого профілю користувача. Після здійснення входу до облікового запису користувач має доступ до додаткових функцій, таких як перегляд вподобаних рецептів та можливість надсилання власних рецептів для публікації. Для забезпечення адаптивності вебсайту до різних екранних розмірів був використаний фреймворк Bootstrap.

Також у цьому розділі було проведено порівняльний аналіз реалізацій функціональностей вебсайту на мовах програмування Golang та PHP. Отже, код, написаний на мові Golang, виявився меншим за обсягом порівняно з PHP,

що свідчить про більш лаконічний та статичний синтаксис мови Golang у порівнянні з більш багатошаровим синтаксисом PHP з динамічною типізацією. Обидві мови використовують цикли для циклічного перегляду набору результатів та створення об'єктів, проте Golang компілюється у машинний код, що потенційно забезпечує кращу продуктивність у порівнянні з інтерпретованою мовою PHP.

Були проведені бенчмарк-тести на прикладі методу, що відображає список усіх рецептів для обох мов програмування. Результати показали, що метод на Golang виконується значно швидше (0.02 секунди) порівняно з PHP (0.35 секунди), особливо при збільшенні об'єму даних. Далі були розглянуті особливості обробки помилок в обох мовах. У Golang помилки обробляються вручну за допомогою механізму повернення помилки, що дозволяє більшу гнучкість в керуванні помилками, а PHP зазвичай призводить до зупинки програми у разі помилки. На основі порівняльного аналізу можна зробити висновок, що мова програмування Golang є більш безпечною та продуктивною для розробки вебзастосунків у порівнянні з PHP, особливо при роботі з великими об'ємами даних та обробці помилок.

ВИСНОВКИ

Розробка вебсайту має актуальність в сучасному світі, оскільки Інтернет є одним з найпоширеніших засобів комунікації і надання інформації. Створення тематичного блогу є поширеним завданням для багатьох веброзробників. Тематикою розробки вебсайту було обрано кулінарний блог, що потенційно має широку цільову аудиторію та різноманітні можливості отримання економічної та соціальної вигоди. Під час проведеного дослідження було виконано такі основні етапи розробки вебсайту: попередня розробка структури та дизайну сайту, вибір технологій, розробка frontend та backend, вибір бази даних та реалізація основного функціоналу, підключення до хостинг-платформи.

У межах роботи було досліджено можливості використання мови програмування Golang для побудови вебсайту, проаналізовано переваги та недоліки її використання у порівнянні з іншими мовами програмування, а також способи використання цієї мови для розробки тематичного блогу. У результаті дослідження Golang виявився більш продуктивним у порівнянні з PHP. Він компілюється у машинний код, що забезпечує швидке виконання програм. Вбудована підтримка для багатозадачності (goroutines) та ефективна робота з паралелізмом сприяють ефективному використанню ресурсів сервера. Синтаксис Go є досить простим і чистим, тому він є зручним у програмуванні та розумінні коду. Go поставляється з безліччю вбудованих пакетів у стандартній бібліотеці, що спрощує вирішення багатьох завдань без необхідності використання сторонніх бібліотек. Статична типізація в Go допомагає запобігати помилкам на етапі компіляції, що підвищує надійність коду. А також, Go абсолютно захищений від SQL ін’єкцій, на відміну від PHP.

У роботі розглянуто такі аспекти, як швидкість та ефективність розробки, безпека та стабільність, масштабованість та інші важливі фактори, що впливають на якість вебсайту. Було проаналізовано три найпопулярніші кулінарні вебсайти Cookpad.com, Allrecipes.com та Foodnetwork.com. У них

присутні спільні слабкі сторони, що потребують особливої уваги при розробці кулінарного блогу: адаптивність, категорії рецептів, навігація та пошук по сайту — вони грають ключову роль у покращенні користувацького досвіду та забезпеченні зручності використання.

Всі ці аспекти ми врахували під час розробки власного кулінарного блогу «KitcheNerd». Окрім розміщення різноманітних рецептів блог також має на меті надавати кулінарні поради, надихати людей експериментувати зі стравами, створити спільноту любителів готування. Звичайно, в майбутньому цей перелік можна розширювати постійно удосконалюючи можливості вебсайту та надаючи користувачам ще більше цікавого та корисного контенту. Також однією з корисних функцій для зареєстрованих користувачів може бути можливість складення власного раціону харчування на день/тиждень/місяць.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Alonso G., Casati F., Kuno H., Machiraju V. Web Services. Heidelberg: Springer, 2004. 374 p.

2. Bickner C. Web design on a shoestring. Indianapolis: New Riders, 2003. 215 p.

3. Chen, X., Ji, Z., Fan, Y., & Zhan, Y. Restful API Architecture Based on Laravel Framework. *Journal of Physics Conference*. 2012, Vol. 910, P. 2-6.

4. Chuck M., Bill K. HTML: The Definitive Guide. New York: O'Reilly, 1997. 531 p.

5. Cunliffe, D. Developing usable Web sites – a review and model, *Internet Research*, 2000. Vol. 10 No. 4, P. 295-308.

6. Douglas K. PostgreSQL: A Comprehensive Guide to Building, Programming, and Administering PostgreSQL Databases. Carmel: Sams Publishing 2020. 1032 p.

7. Indrasiri K., Kuruppu D. gRPC: Up and Running. Building Cloud Native Applications with Go and Java for Docker and Kubernetes. New York: O'Reilly Media, 2020. 205 p.

8. Karpinski R. Beyond HTML. New York: McGraw-Hill, 1996. 472 p.

9. Kreutz C. Exploring the pontentials of blogging for development. *Participatory Learning and Action*, 2009. No. 59, P. 28-31.

10. Leon J. D. Security with Go: Explore the power of Golang to secure host, web, and cloud services. Birmingham: Packt Publishing, 2018. 340 p.

11. Lynch P. J. Web Style Guide: Basic Design Principles for Creating Web Sites. New Haven, London: Yale University Press, 1999. 165 p.

12. McFedries P. Web Coding & Development All-in-One For Dummies. Hoboken: John Wiley & Sons. 2018, 848 p.

13. Nishadha. Use Case Diagram Tutorial. Creately: вебсайт. URL: https://creately.com/guides/use-case-diagram-tutorial/ (дата звернення: 10.12.2023).

14. Nixon R. Learning PHP, MySQL, JavaScript, and CSS: A Step-by-Step Guide to Creating Dynamic Websites. O'Reilly Media, 2012. 586 p.

15. Research of main trends of modern web sites development. Kotenko N. et al. *Cybersecurity: Education, Science, Technique*. 2019. No. 5, P. 6–15.

16. Riva C. G. Special Topics in Information Technology. Milan: Springer, 2023. 152 p.

17. Sau Sheong Chang. Go Web Programming. New York: Manning. 2016. 312 p.

18. Sethi, S. K. Blog/Web Log — A new easy and interactive website building tool for a non-net savvy radiologist. *Journal of Thoracic Imaging*, 2007. No 22(2), P. 115–119.

19. Treadway T. Where Do Consumers Go for Recipes? We Have the Answer! Creative Energy: вебсайт. URL: https://creativeenergy.agency/food/where-do-consumers-go-for-recipes-we-have-the-answer/ (дата звернення: 13.06.2023).

20. Varghese S. Web Development with Go: Building Scalable Web Apps and RESTful Services. Berkeley: Apress, 2015, 314 p.

21. Wilson D.W. Lin X. Longstreet P. Sarker S. Web 2.0: A Definition, Literature Review, and Directions for Future Research. *AMCIS 2011 Proceedings - All Submissions.* 2011. No 368r.

22. Zandstra M. PHP Objects, Patterns, and Practice. In Apress eBooks. 2010. 556 p.

23. Біляєва К.Є. Рибальченко О.Г. Використання мов програмування Golang та Java в якості серверних мов для розробки web-застосунків. *Комп'ютерні інтелектуальні системи та мережі:* матеріали XIII всеукраїнської наук.-практ. web конф, м. Кривий Ріг, 2020. С. 86-89.

24. Глушаков С. В., Жакін В. А., Хачиров Т. З. Програмування WEB-сторінок: підручник. Харків: Фоліо, 2004. 390 с.

25. Жеребецька Д., Думанський Н. Роль вебсайту організації у створенні позитивного іміджу. *Інформація, комунікація, суспільство 2019*: матеріали 8-ї міжнар. наук. конф. ІКС-2019, м. Львів, 2019. С. 57–58.

26. Кузьменко А. О., Яремко С. А. Аналітичний огляд сучасних технічних засобів розробки вебсайтів. *Математичні методи, моделі та інформаційні технології в управлінні підприємством* : тези доп. III студент. вуз. наук. конференції. м. Вінниця, 2018. С. 147–149.

27. Лісовська А., Калита А. Контент вебсайтів і їхня структура. *Молодий вчений.* 2019. № 10(74). С. 166-170

28. Лютий С. Go vs Javascript. На чому писати IoT проекти. Blogchain: вебсайт. URL: https://blogchain.com.ua/go-vs-javascript-na-chomy-pisati-iot-proekti/ (дата звернення: 07.09.2023).

29. Оглобліна В.С., Міхайлуца О.М. Порівняння мов програмування Golang та PHP як засобів створення вебсайтів. *Геостратегічні трансформації та траєкторія національної безпеки в контексті відбудови та сталого розвитку України:* матеріали міжнар. наук.-практ. конф., 25-26 травня 2023 м. Запоріжжя: ІННІ ім. Ю. М. Потебні ЗНУ, 2023

30. Оглобліна В.С., Міхайлуца О.М. Посилення конкурентних переваг підприємства за рахунок ефективної розробки вебсайту. *Стратегічні пріоритети розвитку підприємництва, торгівлі та біржової діяльності*: матеріали IV міжнар. наук.-практ. конф, 10-11травня 2023 року. м.Запоріжжя: НУ «Запорізька політехніка», 2023. С.272-273.

31. Пасічник Н. Р., Дивак М. П. Метод та алгоритм побудови структур тематичних вебсайтів на основі онтологічного підходу. *Наукові праці ДонНТУ Серія "Інформатика, кібернетика та обчислювальна техніка".* 2012. № 15 (203). С. 184–189.

32. Пелещишин А. М. Позиціонування сайтів у глобальному інформаційному середовищі: монографія. Львів: вид-во Львів. політехніки, 2007. 258 с.

33. Петрів М. В. Доменне ім’я як засіб індивідуалізації в цифровому середовищі. *Часопис цивілістики* : наук.-практ. журн. м. Одеса: Гельветика, 2019. № 33. С. 92-96

34. Спасибо І. А. Щодо історії виникнення глобальної мережі Інтернет. *Право та інновації*, 2014. № 3. С. 15-25

35. Топ вебсайтів у категорії кулінарії. Similarweb: вебсайт. URL: https://www.similarweb.com/top-websites/food-and-drink/cooking-and-recipes/ (дата звернення: 13.06.2023)

Декларація
академічної доброчесності
здобувача ступеня вищої освіти ЗНУ

Я, Оглобліна Валерія Сергіївна, студентка 2 курсу, денної форми навчання, Інженерного навчально-наукового інституту ім. Ю.М. Потебні ЗНУ, спеціальність 121 Інженерія програмного забезпечення, адреса електронної пошти lerusik121147@gmail.com,

- підтверджую, що написана мною кваліфікаційна робота на тему: **«Особливості використання мови програмування Golang для побудови вебсайту на прикладі тематичного блогу»** відповідає вимогам академічної доброчесності та не містить порушень, що визначені у ст. 42 Закону України «Про освіту», зі змістом яких ознайомлений;

- заявляю, що надана мною для перевірки електронна версія роботи є ідентичною її друкованій версії;

- згодна на перевірку моєї роботи на відповідність критеріям академічної доброчесності у будь-який спосіб, у тому числі за допомогою Інтернет - системи, в також архівування моєї роботи у базі даних цієї системи.

Дата 01.03.2024 Підпис _____ Оглобліна Валерія Сергіївна
(студент)

Дата 01.03.2024 Підпис _____ Міхайлуца Олена Миколаївна
(науковий керівник)