МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДОНЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ВАСИЛЯ СТУСА
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ І ПРИКЛАДНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

МАРКУШ ЮРІЙ ВАСИЛЬОВИЧ

Допускається до захисту:
в.о. завідувача кафедри
інформаційних технологій,
д-р. техн. наук, професор
_____Наталія ВЕСЕЛОВСЬКА
«____»_____ 2025 р.

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ БРОНЮВАННЯ МІСЦЬ В ЗАКЛАДАХ ГРОМАДСЬКОГО ХАРЧУВАННЯ

Спеціальність 122 Комп’ютерні науки

Кваліфікаційна (магістерська) робота

Науковий керівник:
Наталія ВЕСЕЛОВСЬКА,
професор кафедри інформаційних технологій,
д-р. техн. наук, професор

Оцінка: _____ / _____ / _____
(бали/за шкалою ЄКТС/за національною шкалою)

Голова ЕК: _____
(підпис)

Вінниця — 2025

АНОТАЦІЯ

Маркуш Ю.В.

Інформаційна технологія бронювання місць в закладах громадського харчування. Спеціальність 122 «Комп'ютерні науки», освітня програма «Комп'ютерні технології обробки даних (Data Science)». Донецький національний університет імені Василя Стуса, Вінниця, 2025.

Магістерська робота спрямована на розробку інформаційної технології сервісу онлайн-бронювання місць у закладах громадського харчування. Основна мета дослідження полягає у вдосконаленні функціонування програмного додатку шляхом інтеграції сучасних технологічних рішень, що забезпечують підвищення швидкості його завантаження та надійності роботи для кінцевих користувачів.

У процесі дослідження здійснено аналіз існуючих технологій, визначено їхні переваги та обмеження, а також сформовано систематизований перелік вимог до розроблюваної системи. Це дозволило обґрунтувати вибір оптимальних інструментів і методів для створення ефективного та зручного сервісу бронювання.

Ключові слова: інформаційна технологія, бронювання, заклад громадського харчування, SILOS, React, React Native, Java Script.

ABSTRACT

Markush Y.V.

Information technology for booking seats in catering establishments.

Specialty 122 “Computer Science”, educational program “Computer Technologies of Data Processing (Data Science)”. Vasyl' Stus Donetsk National University, Vinnytsia, 2025. The work is devoted to the creation of information technology for the development of the client part of the restaurant reservation service. The purpose of the master's thesis is to improve the application by using modern technologies to develop the client part, which will increase the download speed of the application and its reliability for customers. The advantages and disadvantages of each of them are analyzed, the list of requirements is created. The client part itself was developed and integrated with the server part.

Keywords: interface, client part, React, React Native, Java Script

ЗМІСТ

ВСТУП

Актуальність теми дослідження. У сучасному світі мережа Інтернет стала невід’ємною складовою повсякденного життя людини. Вона забезпечує доступ до новітніх технологій, які активно впроваджуються у виробничих процесах, медичній практиці, сфері мистецтва, редагуванні графічних матеріалів, системах зберігання документації та, передусім, у побуті [1–36].

Ще донедавна виконання багатьох завдань вимагало значних часових і фізичних витрат. Наприклад, для оплати комунальних рахунків необхідно було витрачати години у чергах банківських установ; переклад текстів здійснювався за допомогою громіздких словників, які доводилося шукати в бібліотеках; асортимент товарів був обмежений локальними магазинами.

Сьогодні ж завдяки Інтернету та численним онлайн-сервісам користувач отримує можливість здійснювати оплату рахунків, виконувати переклади, замовляти будь-які товари незалежно від географічних меж. Це не лише значно спрощує повсякденні процеси, а й відкриває нові перспективи для розвитку суспільства та підвищення якості життя.

Реаліє сьогодення є той факт, що процеси цифровізації охоплюють практично всі сфери діяльності людини. Те, що ще десять-п'ятнадцять років тому виконувалося вручну, вимагало паперових журналів, тривалих телефонних розмов або особистої присутності, сьогодні поступово переходить у формат зручних електронних сервісів. Інформаційні технології стали не просто допоміжним інструментом, а необхідною умовою швидкої комунікації, ефективної організації роботи та якісного надання послуг.

Одним із напрямів, де потреба в автоматизації виявилася особливо гострою, є процес бронювання місць. Незалежно від того, йдеться про квитки в кінотеатр,

місця в готелі, посадкові місця у транспорті чи резервування зали для події — люди прагнуть отримати можливість забронювати потрібний ресурс швидко, у зручний для себе час і без додаткових клопотів. Сучасне життя надто динамічне, тому очікування в телефонній черзі або необхідність їхати особисто в касу вже виглядають застарілими та незручними. Традиційні способи бронювання, які десятиліттями здавалися нормою, — телефонний дзвінок, запис у журналі, спілкування з адміністратором — сьогодні не можуть забезпечити ані достатньої точності, ані оперативності. Вони створюють ризики людських помилок, дублювання резервувань та непорозумінь. Крім того, вони вимагають додаткових витрат часу як від клієнтів, так і від працівників установи. Саме тому розробка інформаційної системи бронювання місць стала логічним кроком еволюції таких процесів. Сучасна ІС дозволяє автоматизувати більшість дій, зробити їх прозорими, передбачуваними та зручними [21-45].

Автоматизована система дає змогу користувачам самостійно обирати час, місце, тип послуги, бачити актуальну доступність та одразу отримувати підтвердження бронювання. Для організацій це означає зменшення навантаження на персонал, оптимізацію робочих процесів, точніший облік зайнятості ресурсів та можливість аналізувати попит на послуги.

Метою даної роботи є детальний аналіз особливостей створення інформаційної системи бронювання місць, визначення її ключових структурних компонентів, основних функціональних можливостей, а також виявлення переваг, які отримують організації та користувачі після впровадження подібної системи. Розуміння цих аспектів дозволяє побудувати технічно грамотну, зручну та ефективну ІС, що відповідатиме вимогам сучасного цифрового середовища.

Метою та завданнями дослідження є вдосконалення системи онлайн-бронювання місць у закладах громадського харчування шляхом розширення її

функціональних можливостей та підвищення якості клієнтського інтерфейсу. Реалізація поставленої мети передбачає створення зручного, надійного та ефективного інструменту для користувачів, що забезпечить швидкий доступ до сервісу та комфорт у процесі взаємодії з ним.

Для досягнення визначеної мети необхідно виконати такі завдання:

- здійснити аналіз існуючих рішень у сфері систем бронювання та визначити їхні сильні й слабкі сторони;
- сформувати вимоги до функціональних та технічних характеристик розроблюваної системи;
- розробити архітектуру клієнтської частини з урахуванням сучасних технологій та принципів оптимізації;
- забезпечити підвищення швидкодії та надійності роботи додатку;
- провести тестування та оцінку ефективності запропонованих рішень.

Об’єкт дослідження – процес створення та впровадження онлайн-сервісу для бронювання місць у закладах громадського харчування.

Предмет дослідження – методи та інструменти розробки інформаційної технології, що забезпечує функціонування онлайн-сервісу бронювання.

Методи дослідження. У роботі застосовано методи моделювання архітектури програмних систем, методи проєктування та розробки користувацького інтерфейсу, а також інструменти аналізу ефективності програмних рішень.

Наукова новизна. Уперше запропоновано та реалізовано онлайн-сервіс для бронювання місць у закладах громадського харчування, який дозволяє здійснювати резервування столиків без необхідності телефонних дзвінків до адміністрації

закладу. Сервіс функціонує як у веб-браузері, так і у мобільному додатку, що розширює можливості користувачів та підвищує зручність взаємодії.

Практичне значення. Розроблений сервіс забезпечує можливість онлайн-бронювання з різних пристроїв, що сприяє оптимізації роботи закладів громадського харчування та підвищенню рівня клієнтського обслуговування.

Достовірність результатів. Теоретичні положення та висновки підтверджуються зіставленням отриманих результатів із відомими рішеннями, а також збіжністю даних математичного моделювання з результатами практичного впровадження програмних засобів.

Особистий внесок магістранта. Усі результати, наведені у магістерській кваліфікаційній роботі, отримані автором самостійно.

1. КОМПЛЕКСНИЙ АНАЛІЗ КОНТЕКСТУ ТА СЕРЕДОВИЩА РОЗРОБКИ

1.1 Аналіз проблеми розробки

Інформаційна система бронювання місць — це комплексне програмне рішення, яке поєднує в собі програмне забезпечення, бази даних та інструменти взаємодії з користувачем, що забезпечують повну автоматизацію процесів пошуку, вибору та резервування вільних місць у певному об’єкті або під час конкретного заходу. Така система дозволяє замінити традиційні ручні методи обліку місць — таблиці, журнали, паперові бланки — на сучасні цифрові механізми, які працюють точно, швидко та стабільно. Завдяки цьому зменшується ймовірність помилок, підвищується оперативність обслуговування та забезпечується комфорт для кінцевого користувача [19-34].

Сучасні користувачі очікують швидкого доступу до інформації, можливості забронювати місце в будь-який час та з будь-якого пристрою. Традиційні методи — телефонні дзвінки, журнали запису або особисте звернення — сьогодні виглядають повільними, незручними та вразливими до помилок. Тому автоматизована система забезпечує значно вищий рівень точності, надійності та ефективності.

Основним призначенням такої системи є комплексне управління процесом бронювання. Зокрема, вона виконує такі функції:

- *Надання користувачеві можливості перегляду доступних місць у режимі реального часу.*

Система постійно оновлює інформацію про зайняті й вільні місця, дозволяючи користувачу бачити актуальні дані без затримок. Це особливо важливо при високому попиті, коли місця швидко розбираються. Відображення може здійснюватися у вигляді інтерактивної схеми залу, списку або календаря.

- *Забезпечення швидкого та зручного бронювання.*

Користувач може обрати конкретне місце, вказати тип квитка або номеру, заповнити мінімальну кількість полів і миттєво створити бронювання. Часто система підтримує різні способи оплати, зберігає історію замовлень, надсилає підтвердження електронною поштою чи в особистий кабінет. Усе це робить процес інтуїтивно простим і доступним навіть для користувачів без спеціальних навичок.

- *Запобігання дублюванню резервувань.*

Завдяки чіткій взаємодії з базою даних система блокує можливість того, що одне й те саме місце буде заброньоване декількома людьми одночасно. Використовуються механізми транзакцій, перевірки зайнятості та синхронізації даних, що особливо важливо для великих об’єктів з високим навантаженням.

- *Ведення обліку, статистики та контролю для адміністрації.*

Адміністратори можуть переглядати загальну кількість бронювань, рівень заповнюваності залів або номерів, фінансові показники, популярність певних подій чи рейсів. Звіти можуть формуватися автоматично, що спрощує управління та дозволяє приймати обґрунтовані рішення щодо оптимізації роботи.

Інформаційні системи бронювання місць сьогодні мають дуже широке застосування в різних галузях. Вони активно використовуються транспортними компаніями для продажу квитків на авіа-, залізничні та автобусні рейси; у кінотеатрах та театрах для вибору місць на сеанси; у готелях і хостелах для резервування номерів; у навчальних центрах для реєстрації на курси чи події; у спортивних комплексах для бронювання кортів, тренажерних залів або місць на змагання [27-37].

Такі системи також поширені в сучасних коворкінгах, конференц-залах, оздоровчих комплексах і навіть закладах освіти, де потрібно керувати доступом до

аудиторій чи ресурсів. Завдяки своїй універсальності, масштабованості та гнучкості ІС бронювання місць стає важливим елементом цифрової інфраструктури, що значно підвищує ефективність роботи організацій та якість обслуговування клієнтів.

Веб-додаток функціонує за принципом архітектури «клієнт–сервер», що передбачає розподіл його на дві взаємопов'язані частини. **Клієнтська складова** відповідає за користувацький інтерфейс, формування запитів до сервера та обробку отриманих відповідей. **Серверна складова** приймає запити, виконує необхідні обчислення, генерує веб-сторінки та передає їх клієнту через мережу з використанням протоколу HTTP.

Веб-інтерфейс являє собою сукупність засобів, що забезпечують взаємодію користувача з веб-додатком або веб-сайтом у середовищі браузера. Його широке поширення зумовлене стрімким розвитком глобальної мережі та масовим використанням веб-браузерів [1]. Однією з ключових вимог до веб-інтерфейсів є забезпечення єдиного зовнішнього вигляду та стабільної функціональності незалежно від типу браузера.

Технології реалізації веб-інтерфейсів:

- Найбільш поширеним підходом є використання **HTML**, **CSS** та **JavaScript**. Проте відмінності у реалізації специфікацій HTML, CSS та DOM у різних браузерах створюють труднощі під час розробки та підтримки додатків. Додатковим фактором є можливість користувача змінювати параметри браузера (шрифти, кольори, налаштування сценаріїв), що може впливати на коректність роботи інтерфейсу.
- Альтернативний, хоча менш універсальний підхід, полягає у застосуванні **Adobe Flash**, **Silverlight** або **Java-аплетів**. Ці технології, що підтримуються більшістю браузерів через плагіни, надають

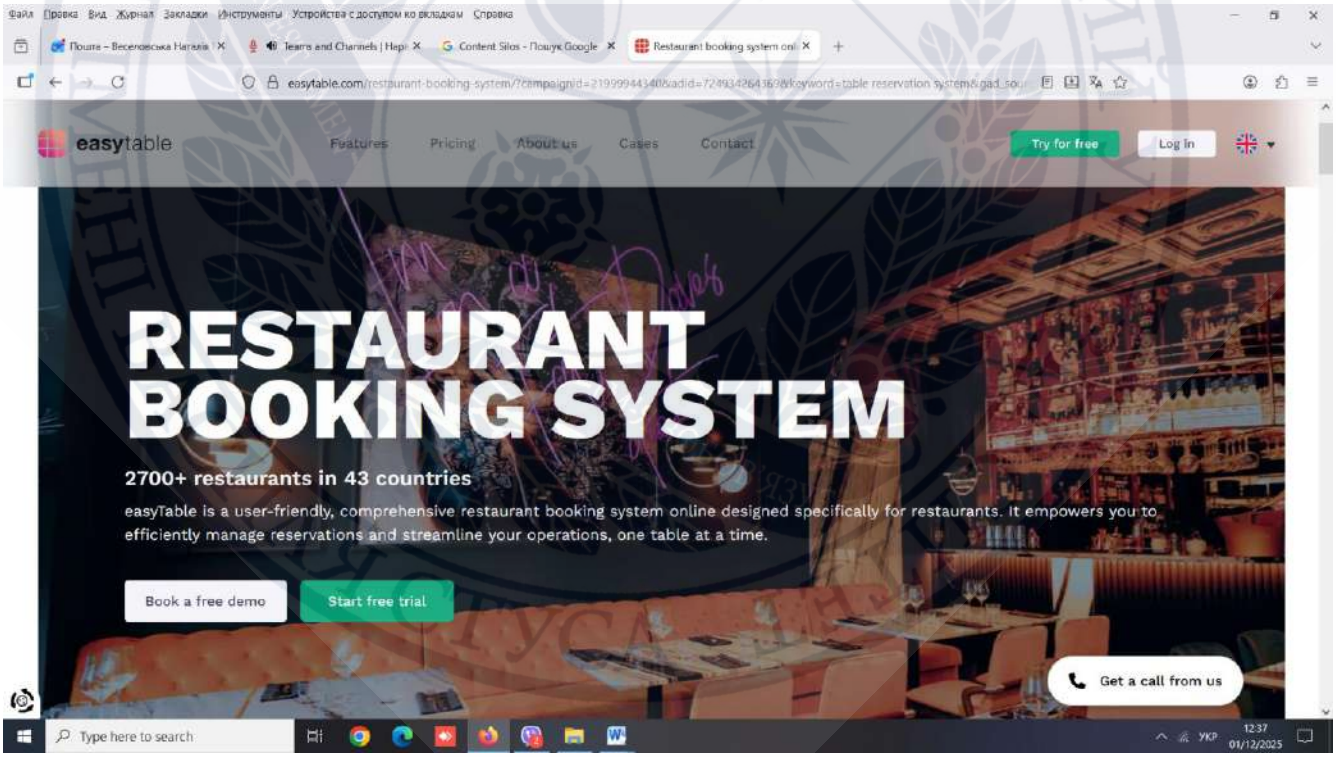
розробнику ширший контроль над інтерфейсом та дозволяють уникати проблем сумісності. Водночас вони можуть спричиняти власні труднощі через відмінності у реалізації на стороні клієнта.

- Для підвищення ефективності розробки активно використовуються додаткові інструменти:
- **SASS** – засіб для оптимізації структури CSS-коду та спрощення його редагування;
- **jQuery** – бібліотека JavaScript, що значно скорочує час написання коду без втрати функціональності;
- **Emmet** – плагін для текстових редакторів, який автоматизує створення шаблонних конструкцій HTML/CSS.

Сучасні тенденції. Значної популярності набуває технологія **Ajax**, яка дозволяє оновлювати лише необхідні дані інтерфейсу без повного перезавантаження сторінки. Це забезпечує інтерактивність, підвищує продуктивність та покращує користувацький досвід.

1.2 Комплексний аналіз переваг та обмежень

Ринок онлайн-сервісів для бронювання місць у закладах громадського харчування активно розвивається у багатьох країнах світу, що підтверджується численними угодами з придбання подібних проєктів провідними IT-компаніями. Так, індійська платформа пошуку ресторанів **Zomato** здійснила купівлю американського сервісу **NexTable**, який спеціалізується на онлайн-бронюванні столиків, за суму близько **52 млн доларів США** [27–37]. Такий стратегічний крок був спрямований на посилення конкурентних позицій компанії та створення альтернативи вже відомим міжнародним сервісам **OpenTable** (Priceline) та **SeatMe** (Yelp).



Перевагами даних сервісів в Україні є:

1. Залучення додаткової аудиторії;

2. Оптимізація заповнення залу за рахунок запровадження системи знижок у години низького попиту;

3. Нові моделі монетизації;

4. Зниження ризиків;

5. Робота з власною аудиторією на сайті [3].

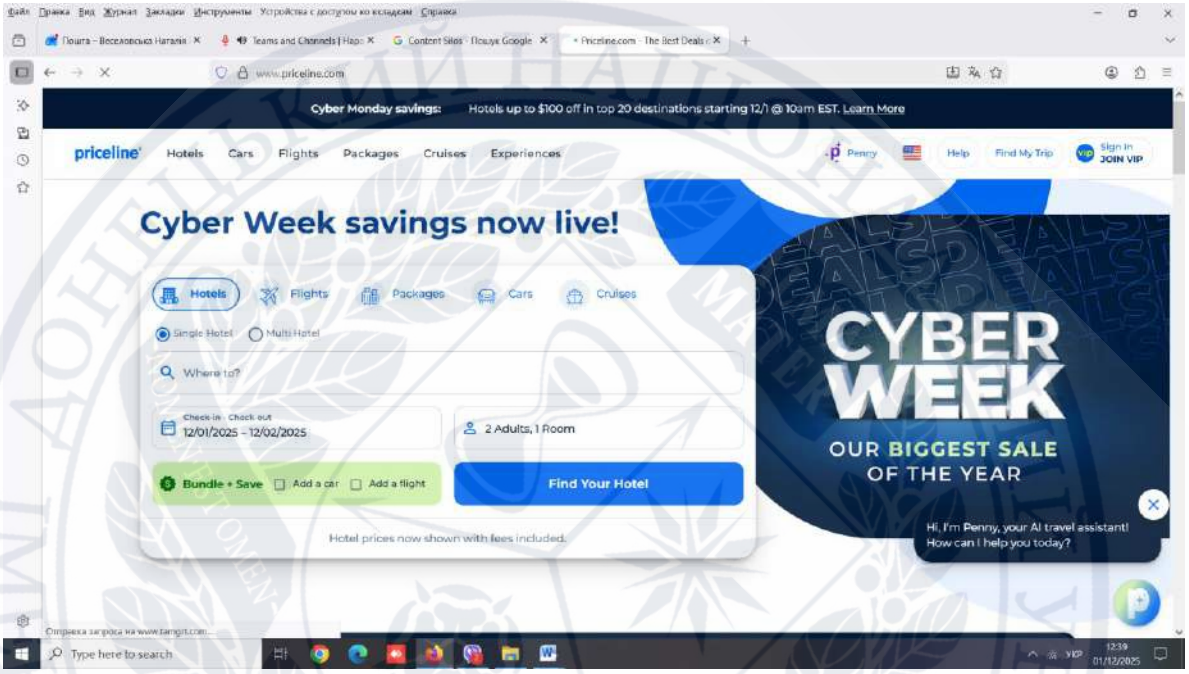
За такою схемою працює, наприклад, сервіс Killer Rezzy. Killer Rezzy — це сервіс бронювання столиків у ресторанах, який працює шляхом заздалегідь купівлі доступних бронювань та їх перепродажу з націнкою, надаючи клієнтам доступ до популярних і часто ексклюзивних закладів, які важко забронювати самостійно. Сервіс спочатку працював переважно в Нью-Йорку, Гамптонсі та Брукліні.

Як працює сервіс:

- **Бронювання та перепродаж:** Killer Rezzy заздалегідь бронює столики в популярних ресторанах (часто тих, з якими має партнерські угоди).
- **Продаж клієнтам:** Потім ці бронювання перепродаються за вищою ціною клієнтам, які хочуть потрапити до певного ресторану.
- **Партнерські та непартнерські ресторани:** Сервіс може продавати бронювання як для партнерських ресторанів, так і для тих, з якими не має офіційної угоди. У другому випадку бронювання оформлюється на ім'я іншого клієнта.
- **Конкуренція з іншими сервісами:** Такий підхід створює конкуренцію з іншими сервісами бронювання, такими як OpenTable.

У OpenTable здійснюються попереднє бронювання місць у закладах громадського харчування на найбільш затребувані дати, після чого реалізують ці броні повторно з додатковою націнкою. У такій ситуації самі ресторани не

отримують жодної додаткової вигоди, хоча для клієнтів це може здаватися зручним способом потрапити до бажаного закладу у святковий день. Водночас адміністрація ресторану часто навіть не усвідомлює факту штучного завищення вартості бронювання [21–41].



1.3 Порівняльний аналіз існуючих онлайн-платформ для резервування місць у закладах громадського харчування

На сучасному етапі розвитку інформаційних технологій вже функціонують онлайн-сервіси, що пропонують вирішення окресленої проблеми. Як приклад можна розглянути систему для бронювання столиків, наведено на рисунку 1.2.

На рисунку 1.2 наведено приклад одного з інтерфейсних вікон сервісу «Максимум».

Серед основних переваг системи можна виділити:

- наявність детального опису функціональних можливостей;

- зручний та інтуїтивно зрозумілий інтерфейс користувача;
- реалізований механізм обліку часу;
- підтримка створення кількох окремих проєктів.

До недоліків належать:

- відсутність інтерактивної моделі залу;
- нерозроблений функціонал для формування персоналізованих рекомендацій користувачам;
- неможливість самостійного додавання нових закладів до системи.

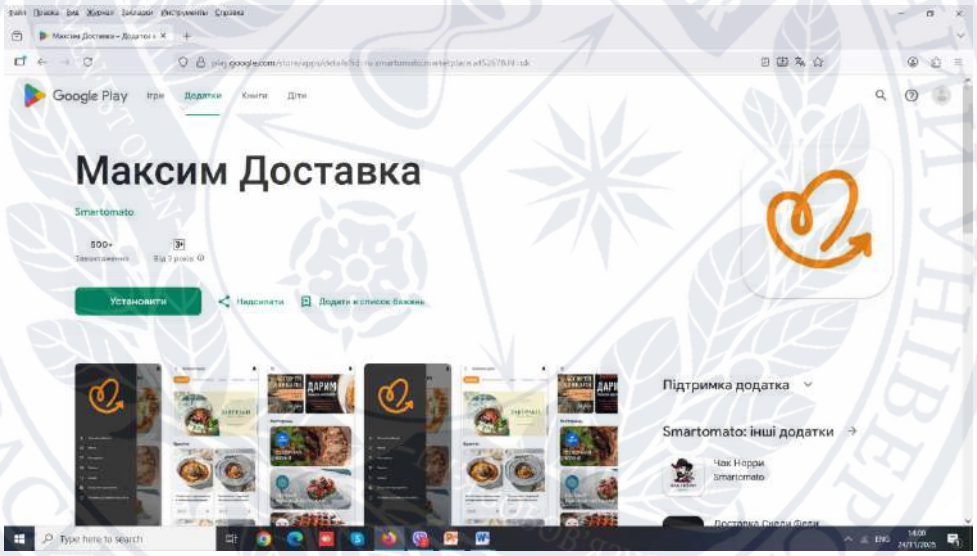


Рисунок 1.2. – Титульна сторінка сервісу Максимум у Google Play.

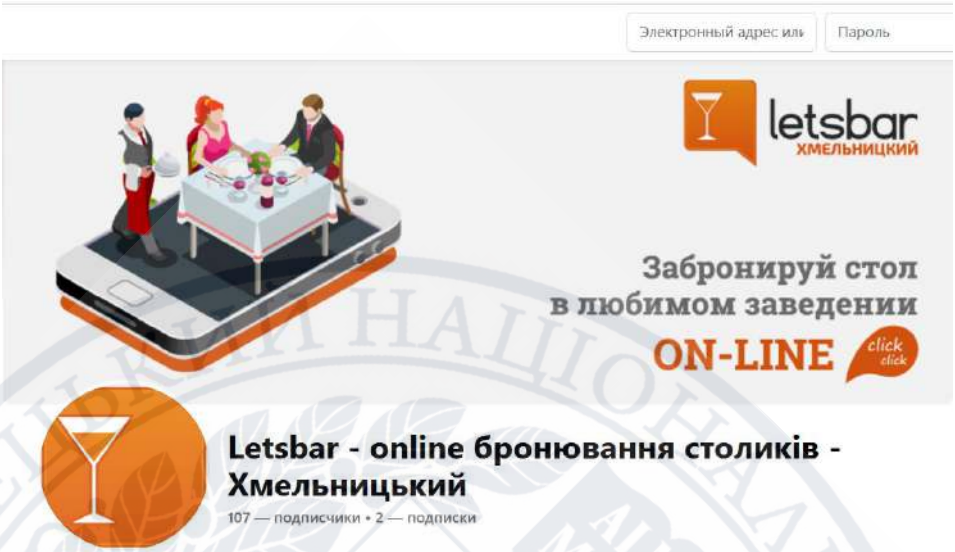


Рисунок 1.3. - Сервіс online бронювання столиків, що діє у м. Хмільницький

Ще одним прикладом рішення зазначеної проблеми виступає сервіс **«Letsbar»**. Дана система надає користувачам можливість попередньо обрати заклад, який вони планують відвідати, та здійснити онлайн-бронювання місця у зручному для них режимі. Інтерфейс одного з вікон сервісу «Letsbar» наведено на рисунку 1.4.

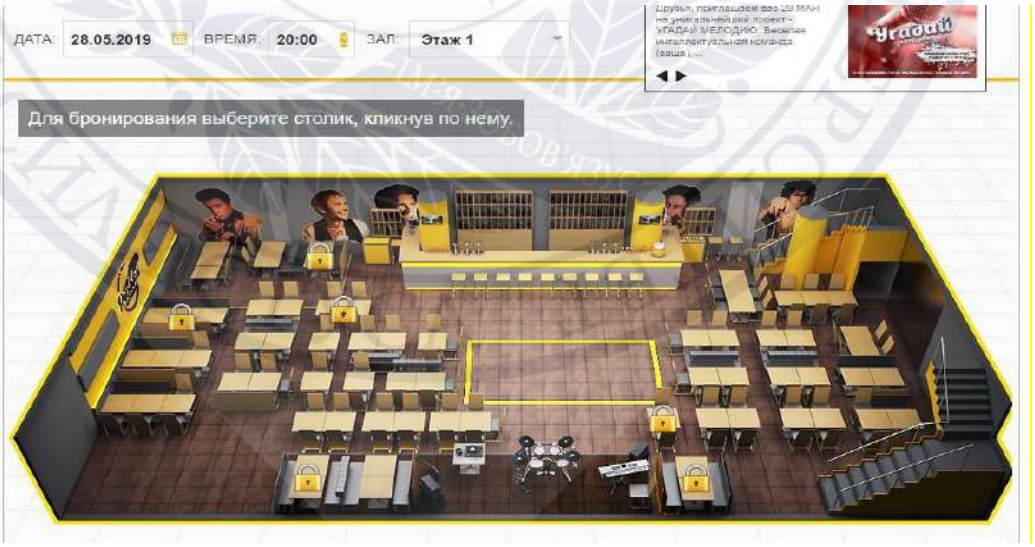


Рисунок 1.4 – Робоче вікно додатку «Letsbar»

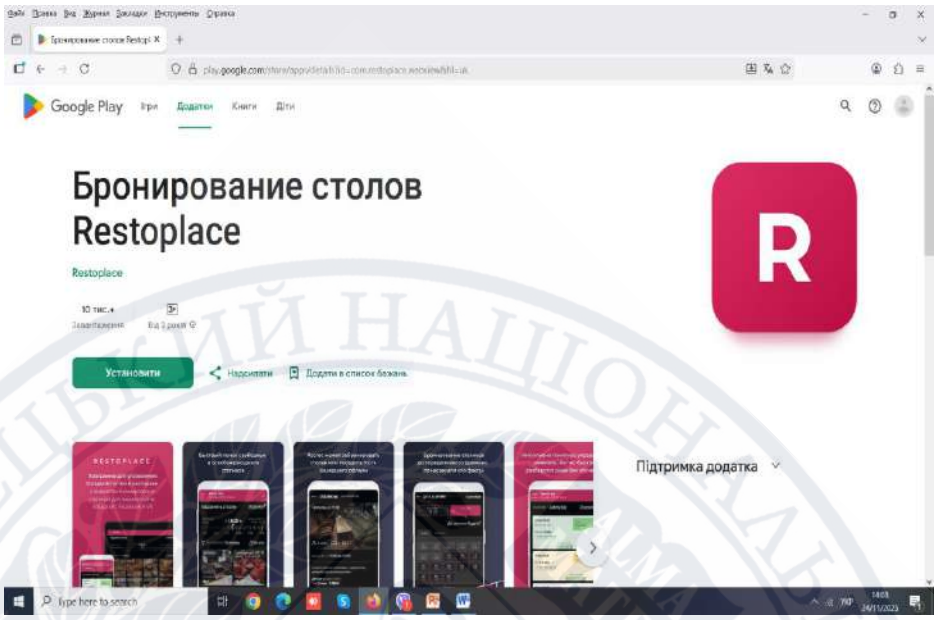


Рисунок 1.5. – Титульна сторінка сервісу Restoplace

1.4. **Висновки**

У розділі було проведено аналіз основної проблематики інформаційної технології онлайн-сервісів для бронювання місць в закладах громадського харчування.

2. ПРОЕКТУВАННЯ АРХІТЕКТУРИ ТА ВИЗНАЧЕННЯ ЕТАПІВ СТВОРЕННЯ СИСТЕМИ

2.1. Фази розробки та проектування

Процес бронювання має низку важливих особливостей, які необхідно ретельно враховувати під час розробки інформаційної системи. Кожен із цих аспектів впливає на архітектуру майбутньої програми, її логіку роботи, вимоги до бази даних та інтерфейсу. Ігнорування хоча б одного з них може призвести до збоїв, неточностей, незручностей для користувачів або навіть до повної непрацездатності системи в умовах реального навантаження [15 - 45].

1. Наявність обмеженої кількості місць.

У будь-якому об’єкті, де проводиться бронювання, завжди існує фізична межа ресурсів. Це можуть бути місця в актовому чи концертному залі, посадкові місця у літаку чи автобусі, столики в ресторані або номери в готелі. Кількість таких ресурсів не може бути збільшена в реальному часі, тому система повинна дуже чітко знати, скільки саме місць доступно, які вже зайняті, а які — ні. Крім того, у деяких випадках є різні категорії місць: наприклад, VIP-зони в кінотеатрі, різні класи обслуговування в транспорті, однокімнатні і двокімнатні номери в готелі. Усе це потребує гнучкої структури даних та грамотної логіки розподілу місць.

2. Динамічність бронювань.

Процес резервування має дуже активний та непередбачуваний характер. Ситуація може змінитися буквально щосекунди: місце, яке щойно було вільне, хтось може тут же забронювати або навіть придбати. Саме тому інформаційна система повинна оперувати даними в режимі реального часу. Важливо забезпечити швидку синхронізацію, щоб інформація на екрані користувача завжди відповідала фактичному стану бази даних, а не була застарілою на кілька хвилин. Особливо це

критично під час пікових навантажень — наприклад, у сезон відпусток, у години пік для транспортних перевізників або на популярні кінопрем’єри.

3. Потреба в точності даних.

Однією з найбільших загроз для систем бронювання є можливість дублювання резервувань. Наприклад, двоє людей можуть одночасно натиснути кнопку “Забронювати”, і без належних механізмів контролю система може помилково видати одному з них зайняте місце. Щоб уникнути цього, необхідно використовувати транзакції, блокування записів, механізми перевірки зайнятості місця перед фінальним підтвердженням. Крім того, система повинна зберігати тільки актуальні й коректні дані, адже найменша неточність може призвести до конфліфтних ситуацій між клієнтами та адміністрацією.

4. Різні категорії користувачів.

Інформаційна система бронювання рідко обмежується тільки клієнтами. Зазвичай вона має щонайменше дві групи користувачів — звичайних відвідувачів або покупців квитків та адміністраторів, які керують усіма процесами. До цього можуть додаватися зареєстровані клієнти, які мають особистий кабінет; менеджери чи співробітники, які відповідають за розклад, ціноутворення, зміну статусів; касири; працівники служби підтримки. Для кожної групи необхідно передбачити різні права доступу, інтерфейс та можливості. Це важлива частина безпеки та зручності системи.

5. Потреба у веденні історії та звітності.

Система бронювання — це не лише інструмент для резервування місць, а й потужний засіб аналітики. Вона повинна зберігати історію всіх бронювань: хто, коли і яке місце резервував, чи була оплата, чи було скасування. Адміністрація може використовувати ці дані для оцінки завантаженості об’єкта, аналізу прибутку, визначення популярних подій або періодів пікової активності. Крім того, ведення

такої історії допомагає у вирішенні спірних ситуацій, плануванні маркетингових кампаній та підвищенні якості обслуговування.

Додатковим аспектом є *зручність взаємодії з інтерфейсом*.

Користувачі повинні легко знаходити потрібну інформацію, розуміти структуру залу, без труднощів орієнтуватися в статусах місць. Простота інтерфейсу скорочує час на бронювання та підвищує задоволеність сервісом.

Безпека також є критично важливою.

Системи бронювання працюють із персональними даними, інколи з платіжними операціями. Тому потрібні шифрування, захищені протоколи, захист баз даних, правильне розмежування прав доступу.

Ще один важливий фактор — *стійкість до високих навантажень.*

Під час пікових періодів тисячі користувачів можуть одночасно працювати із системою. Вона повинна залишатися стабільною, не «падати», не працювати повільно. Для цього використовують оптимізовані БД, кешування, балансування.

Крім того, *система повинна бути масштабованою.*

Організація може розширюватися, додавати нові зали, маршрути, види послуг. Тому архітектура ІС має бути гнучкою, дозволяти оновлення та зміни без суттєвої перебудови всієї структури.

Таким чином, ефективна інформаційна система бронювання повинна відповідати дуже високим вимогам. Вона має бути одночасно надійною, швидкою, технічно стабільною та при цьому простою і зрозумілою для користувачів. Лише поєднання цих характеристик дозволить створити сервіс, який дійсно полегшить процес бронювання, мінімізує помилки й забезпечить комфорт як для клієнтів, так і для адміністрації.

Клієнтська частина веб-додатку – це інтерфейс, який користувач безпосередньо бачить та з яким взаємодіє у браузері. Веб-проєкт формується з

різноманітних елементів: текстових блоків, графічних зображень, гіперпосилань, списків, форм введення даних, таблиць та інших компонентів. Основне завдання веброзробки полягає у правильному розташуванні цих елементів на сторінці та наданні їм необхідних властивостей і стилістичних характеристик [4].

Наприклад, текстовий фрагмент може бути перетворений у заголовок або список та розміщений у центрі екрана; зображення можна масштабувати, перетворити на інтерактивне посилання та розташувати у потрібному місці. Таким чином, створення клієнтської частини веб-додатку фактично є процесом редагування текстових і графічних компонентів сторінки, що здійснюється не традиційними інструментами, а шляхом написання спеціального програмного коду мовами **HTML, CSS та JavaScript**.

При розробці інформаційної системи бронювання місць важливо визначити вимоги, яким вона повинна відповідати, щоб бути ефективною, надійною та зручною для користувачів. Усі вимоги можна умовно поділити на дві великі групи: функціональні, які описують конкретні можливості системи, та нефункціональні, що визначають якісні характеристики, пов’язані з продуктивністю, безпекою, зручністю та надійністю.

Функціональні вимоги

Функціональні вимоги описують що саме повинна робити система — тобто перелік її основних можливостей і дій, які користувач може виконувати.

1. Можливість перегляду доступних місць.

Система має відображати актуальний стан місць у режимі реального часу, незалежно від того, йдеться про зал кінотеатру, літак чи номерний фонд готелю. Користувач повинен бачити, які місця доступні, які вже заброньовані або продані. Дані повинні бути максимально точними та швидко оновлюватися.

2. Вибір конкретного місця на схемі або у списку.

Для зручності користувача система повинна містити схему залу або структуру об’єкта з чітким позначенням місць. Важливо, щоб користувач міг вибрати саме те місце, яке йому подобається, а не отримувати місця автоматично. Схема має бути зрозумілою, з наочними кольорами або позначками.

3. Реєстрація та авторизація користувачів.

Потрібно забезпечити можливість створення облікових записів, вхід до системи, відновлення пароля. Для зареєстрованих користувачів можуть бути доступні додаткові можливості — перегляд історії бронювань, повторне бронювання, налаштування профілю чи персональні пропозиції.

4. Створення, редагування та скасування бронювання.

Користувач повинен мати змогу легко оформити бронювання, переглянути його деталі та за необхідності змінити або скасувати. Це важлива функція, оскільки реальні плани людей можуть змінюватися, і система повинна бути готовою до таких коригувань без проблем і збоїв.

5. Адміністрування системи.

Для персоналу організації має бути доступний окремий модуль або панель керування. Адміністратори повинні мати змогу додавати нові зали, маршрути, рейси, сеанси, встановлювати ціни, змінювати розклад, переглядати статистику та керувати користувачами. Важливо забезпечити гнучкий та зрозумілий інтерфейс, щоб адміністрування не вимагало спеціальної технічної підготовки.

6. Ведення статистики та формування звітів.

Система повинна автоматично збирати дані про кількість бронювань, популярність окремих заходів чи маршрутів, завантаженість у різні періоди, прибуток тощо. Ці дані допомагають адміністрації аналізувати діяльність, планувати роботу, оптимізувати ціни та покращувати сервіс.

Нефункціональні вимоги

Нефункціональні вимоги визначають якість роботи системи та її відповідність сучасним технічним і користувацьким стандартам. Вони не описують конкретні функції, але без них система не може працювати ефективно [27-37].

1. Зручний та інтуїтивно зрозумілий інтерфейс.

Для користувача важливо, щоб система була простою у використанні. Інтерфейс має бути логічним, зрозумілим, без зайвих елементів і складної навігації. Чим менше кроків потрібно для оформлення бронювання, тим зручніше користувачеві.

2. Забезпечення цілісності даних та відсутність дублювань.

Система повинна гарантувати коректність усіх даних і не допускати випадків, коли два користувачі можуть забронювати те саме місце одночасно. Для цього потрібні механізми транзакцій, блокування та перевірки актуальності інформації.

3. Захист персональної інформації користувачів.

Оскільки система працює з особистими даними, а інколи й з платіжною інформацією, необхідно забезпечити високий рівень безпеки. Це включає шифрування, захищені протоколи зв’язку, надійне зберігання даних та обмеження доступу.

4. Підтримка роботи на різних пристроях.

У сучасних умовах багато бронювань здійснюється зі смартфонів або планшетів. Система повинна коректно відображатися на різних екранах, бути адаптивною та забезпечувати однаково якісний досвід як на комп’ютері, так і на мобільних пристроях.

5. Висока швидкість доступу до даних.

Система має працювати швидко, навіть у періоди пікового навантаження. Низька швидкість роботи або часті «підвисання» можуть призвести до втрати клієнтів. Оптимізація бази даних, кешування та ефективні алгоритми обробки

запитів є необхідною умовою продуктивності.

6. Масштабованість і стабільність роботи.

Хоч це не завжди включають у базові вимоги, але сучасні системи повинні бути готовими до розширення. Зі збільшенням кількості користувачів, залів чи маршрутів система має залишатися стабільною та не вимагати повної перебудови.

Процес створення клієнтської частини веб-ресурсу можна умовно поділити на кілька послідовних етапів:

- **Проєктування макета.** На початковій стадії формується графічний макет майбутнього сайту за допомогою спеціалізованих редакторів (Photoshop, Fireworks та ін.). Результатом є зображення інтерфейсу у форматах *.jpg або *.psd, що відображає загальну концепцію дизайну.
- **Верстка інтерфейсу.** Це базовий етап клієнтської розробки, який включає:
  - розділення макета на окремі шари та виділення графічних елементів;
  - створення необхідних файлів проєкту у форматах *.html та *.css;
  - написання коду мовами HTML та CSS для реалізації структури та стилістики сторінки.
- **Програмування інтерактивних можливостей.** На додатковому етапі застосовується JavaScript для забезпечення динамічної взаємодії користувача з веб-ресурсом. Це дозволяє реалізувати випадаючі списки, анімаційні ефекти, роботу кнопок та інші функції, що виконують конкретні дії (наприклад, відправлення повідомлень чи виконання розрахунків).
- **Тестування та перевірка.** Завершальна стадія передбачає комплексне тестування усіх елементів веб-сайту та оцінку їх відповідності початковому макету й технічним вимогам [5].

Архітектура онлайн-сервісу

"Контентні силос" (content silos) - це спосіб структурування контенту вебсайту в тематичні, логічні секції або "силоси", які полегшують навігацію як для користувачів, так і для пошукових систем. Кожен силос складається з основної, загальної сторінки та пов'язаних з нею сторінок, що глибше розкривають підтеми, створюючи таким чином тематичну ієрархію та демонструючи експертизу сайту в певній галузі.

Як це працює

- **Ієрархічна структура**: Контент організовано за принципом "від загального до конкретного". На вершині знаходиться головна тема, яка розділяється на кілька підтем, а кожна з них, у свою чергу, на ще більш специфічніші теми.
- **Взаємопов'язаність**: Сторінки всередині одного силосу пов'язані між собою внутрішніми посиланнями, що створює зрозумілий шлях для користувачів і допомагає пошуковим роботам краще зрозуміти зв'язки між контентом.
- **Тематична авторитетність**: Завдяки чіткій структурі, сайт може отримати вищий авторитет у пошукових системах, оскільки він демонструє глибоке розуміння певної теми, а не просто набір розрізнених статей.

Приклад

- **Головна тема (силос)**: "SEO".
- **Другий рівень**: "На сторінці SEO" та "Поза сторінкою SEO".
- **Третій рівень**:

- **"На сторінці SEO"**: Сторінки про "дослідження ключових слів" та "alt теги".
- **"Поза сторінкою SEO"**: Сторінки про "інфлюенсер-маркетинг" та "побудову зворотних посилань".

Архітектура веб-ресурсу визначає спосіб організації сторінок, доступу до них та навігації між елементами системи. До її складових належать: навігаційні меню та гіперпосилання, URL-адреси, «хлібні крихти», сторінки категорій та файл Sitemap [6].

Коректно побудована архітектура забезпечує зручність пошуку інформації як для користувачів, так і для пошукових систем. Вона формує уявлення про значущість і релевантність контенту, спрямовує відвідувачів та пошукових роботів до ключових сторінок і допомагає зрозуміти структуру ресурсу [39–50].

Основний принцип архітектури – відповідність намірам користувача.

1. Спрощення структури

Якісна архітектура дозволяє швидко знаходити потрібну інформацію. «Плоска» структура передбачає, що важливі сторінки розташовані максимально близько до головної, що зменшує кількість кліків для доступу до них. Оптимізатори та дизайнери часто дотримуються правила «трьох кліків», згідно з яким будь-яка значуща сторінка має бути доступною не більш ніж за два переходи від головної або іншої авторитетної сторінки.

2. Сторінки-концентратори

Це оглядові сторінки, що охоплюють широку тему або категорію та пов’язують її з більш конкретними підрозділами. Їхнє призначення:

- надання стислого огляду теми чи розділу;
- відповідь на найпоширеніші запитання користувачів;

- посилання на ключові підтеми та популярні продукти;
- створення більш дружнього інтерфейсу порівняно зі стандартними сторінками категорій;
- акцентування уваги на предметі дослідження чи діяльності.

3. Формування тематичних «воронок»

Сторінки-концентратори ефективно агрегують однотипну інформацію. Використання структури **SILO** дозволяє підвищити якість архітектури. SILO передбачає ієрархічну організацію контенту за тематичними блоками: концентратор виступає агрегованим вузлом, а SILO – його структурованим ієрархічним представленням.

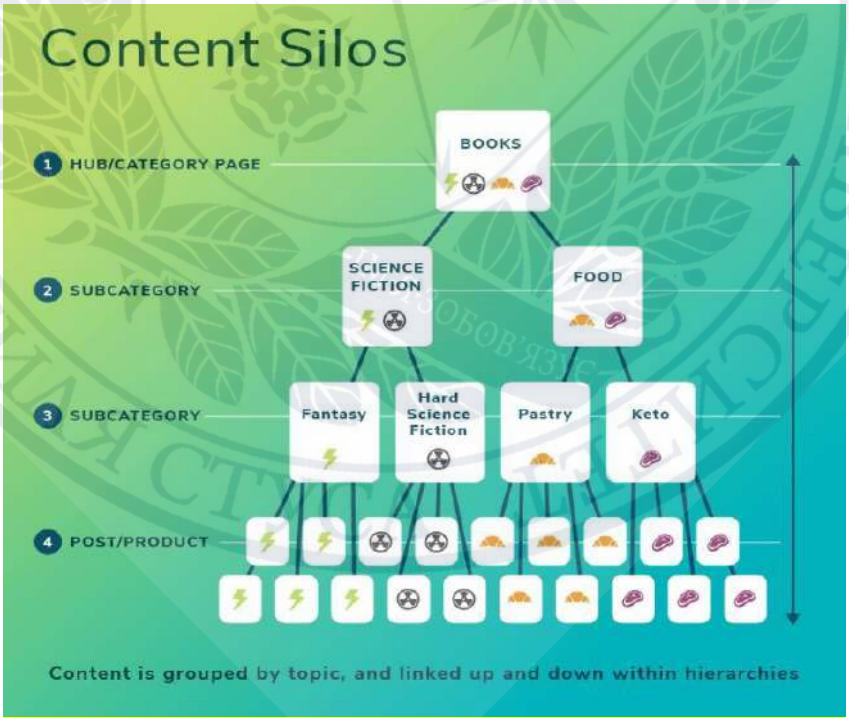


Рисунок. 2.1 – Ієрархія Silo

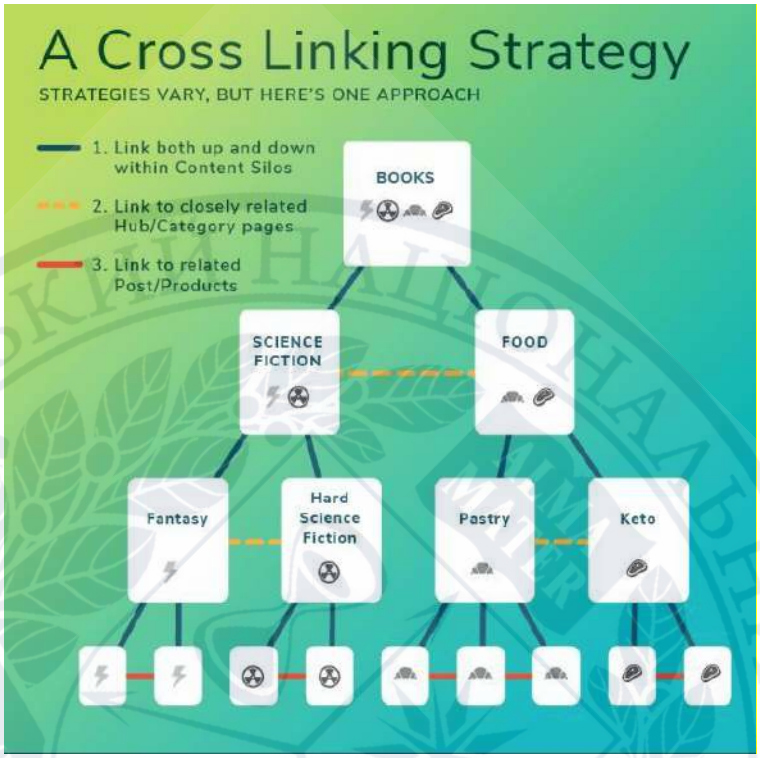


Рисунок 2.2 – Структура перехресних посилань

У межах ієрархічної структури кожен елемент має зв’язок із вищим та нижчим рівнем. Така взаємопов’язаність забезпечує користувачам зручну навігацію по сайту, а пошуковим системам — кращу інтерпретацію його змісту.

Тематична «воронка» зазвичай включає ключові елементи: навігаційні меню, «хлібні крихти», контекстні посилання та структуру URL. Саме вони формують основу для організації контенту за принципом **SILO**.

Основні принципи побудови:

1. Використання перехресних посилань та контекстно-залежних сторінок.

2. Перенаправлення трафіку з авторитетних сторінок на менш відомі, але важливі.

Після створення розділів та визначення їхньої ієрархії налаштовуються внутрішні посилання. Додатковим інструментом оптимізації є **цільові сторінки**, які орієнтовані на конкретних відвідувачів. Приклади:

- сторінка продажу товарів, доступна з головної;
- сторінка з високою конверсією всередині SILO-структури;
- сторінка з меншою значущістю, яка має слабкі зв’язки з іншими елементами.

Авторитетні сторінки — це ті, що добре ранжуються у пошукових системах та отримують значний обсяг трафіку. Оптимізація полягає у правильному розміщенні внутрішніх посилань, які спрямовують користувачів із таких сторінок на інші важливі ресурси. При цьому посилання варто використовувати не лише у навігаційних панелях, а й у контенті.

Методи визначення потреби у нових посиланнях:

Для аналізу внутрішньої структури посилань застосовується **Google Search Console** - безкоштовний сервіс для власників сайтів. Він надає дані про ефективність ресурсу у пошуковій видачі, допомагає відстежувати індексацію сторінок, знаходити та виправляти технічні помилки, а також оптимізувати контент для підвищення видимості у Google [22 - 38].

Ключові функції та можливості:

- **Оптимізація та аналіз:**
  - Оцінка трафіку з пошуку Google, включаючи показ, кліки та позиції сторінок за різними запитами.

- Аналіз того, за якими запитами користувачі знаходять ваш сайт.
- Відстеження ефективності сторінок та виявлення найбільш трафікових сторінок.

- **Технічна підтримка та індексація:**
- Перевірка, чи може Google знайти та просканувати ваш сайт.
- Виявлення та усунення помилок індексації та інших технічних проблем.
- Відправка файлів Sitemap та окремих URL для індексації, а також запит на повторне сканування.

- **Моніторинг та безпека:**
- Отримання сповіщень про проблеми, такі як спам або помилки безпеки на сайті.
- Аналіз мобільної версії сайту та її адаптивності.
- Перегляд зовнішніх посилань, які вказують на ваш ресурс.

Для аналізу внутрішньої структури сайту зазвичай доступна інформація про першу тисячу URL-адрес. Якщо ресурс містить понад 1000 сторінок, доцільно застосовувати сегментацію за каталогами для отримання детальніших даних. Інструменти **Moz**, **Ahrefs** та **SEMRush** надають відомості про кожне посилання, дозволяючи відфільтровувати звіти та виділяти як найбільш авторитетні, так і рідкісні посилання. Система **Google Analytics** показує, які сторінки отримують найбільший обсяг трафіку та які характеризуються високим рівнем конверсії. Оптимізація полягає у перенаправленні користувачів саме на сторінки з високою конверсійною ефективністю.

Методи згладжування архітектури категорійних сторінок

Для сторінок, що містять сотні або тисячі елементів, застосовуються три основні підходи:

- **Пагінація.** Найбільш поширений метод, що передбачає нумерацію сторінок та розбиття великого списку на частини. При коректній реалізації структура стає зрозумілішою для пошукових систем, які

інтерпретують усі об’єкти як складові одного розділу. Такий підхід зручний як для користувачів, так і для пошукових роботів.

- **Режим «показати все».** У цьому випадку всі елементи відображаються на одній сторінці. Оптимізатори вважають, що такий формат полегшує індексацію контенту пошуковими системами. Метод ефективний для великих каталогів, проте має недолік — значне уповільнення завантаження сторінки при великій кількості записів.
- **Нескінченний скролінг.** Комбінований підхід, коли дані підвантажуються поступово під час прокручування сторінки. Попри інтерактивність, контент залишається розділеним на сторінки, що сприяє швидшій індексації пошуковими системами.

Фасетна навігація

Фасетна навігація дозволяє користувачам застосовувати фільтри, сортувати об’єкти та звужувати область пошуку. Найчастіше вона використовується в інтернет-магазинах. Для користувачів це зручний спосіб знайти потрібний товар, однак для пошукових систем така навігація створює сотні дубльованих URL-адрес. Це може ускладнювати індексацію та призводити до технічних проблем. Рекомендоване рішення — стимулювати індексацію унікальних сторінок із високим трафіком та обмежувати доступ до низькорівневих дублікатів.

Додаткові елементи оптимізації:

1. **Помітні посилання на новий контент.** Вони допомагають швидше привернути увагу користувачів та пошукових систем до актуальних матеріалів.
2. **«Breadcrumbs».** Забезпечують зрозумілу навігацію, дозволяючи користувачам легко орієнтуватися у структурі сайту та водночас покращують індексацію.

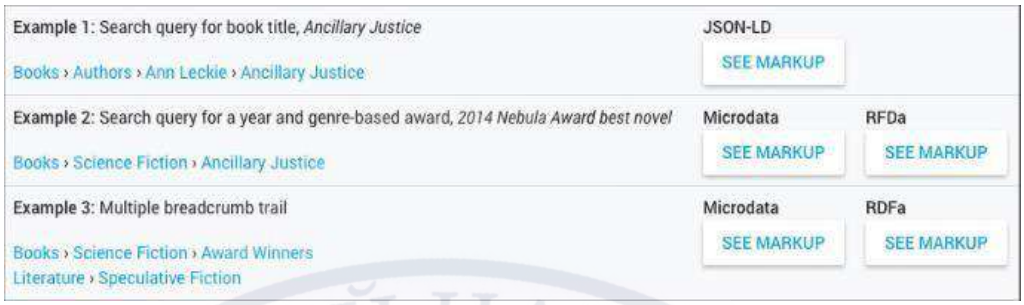


Рисунок 2.3 – Структура “**breadcrumbs**” на мапі сайту

3. Ієрархічна структура URL.

Деякі розробники намагаються штучно створити «плоску» структуру сайту, мінімізуючи кількість вкладених папок або розміщуючи всі сторінки безпосередньо в кореневому каталозі. Хоча подібний підхід інколи може давати результат, для пошукових систем ключовим чинником є не кількість символів «/» у URL-адресі, а реальна глибина доступу до контенту — тобто, скільки кліків необхідно зробити користувачеві, щоб потрапити на потрібну сторінку [7].

2.2. Концептуальне моделювання архітектури програмного забезпечення

Під час створення моделі функціонування системи необхідно враховувати вимоги, що забезпечують її масштабованість та простоту управління. Такий підхід дозволяє підключати кількох клієнтів одночасно та гарантує гнучкість у подальшій експлуатації.

Архітектура системи включає чотири ключові компоненти:

1. **База даних (БД).** Виконує роль сховища інформації про ресторани, доступні столики, їхні адреси, тип кухні та особливості дизайну.

2. **Сервер.** Забезпечує взаємодію між базою даних та клієнтським сервісом. Для обміну інформацією використовується спеціально розроблений API, що працює з форматом JSON.

3. **Клієнтська частина.** Це веб-сервіс, призначений для онлайн-бронювання столиків у ресторанах, який є основним інтерфейсом взаємодії користувача із системою.

4. **Кеш.** Містить дані користувача, отримані після авторизації. Вони дублюються на сервері та у базі даних. Використання кешу дозволяє відображати інформацію навіть за відсутності інтернет-з'єднання.

Онлайн-сервіс має бути побудований таким чином, щоб його архітектура підтримувала масштабування, внесення змін без модифікації основного коду та відповідала вимогам гнучкості й надійності. Схематичне зображення архітектури системи наведено на рисунку 2.4.

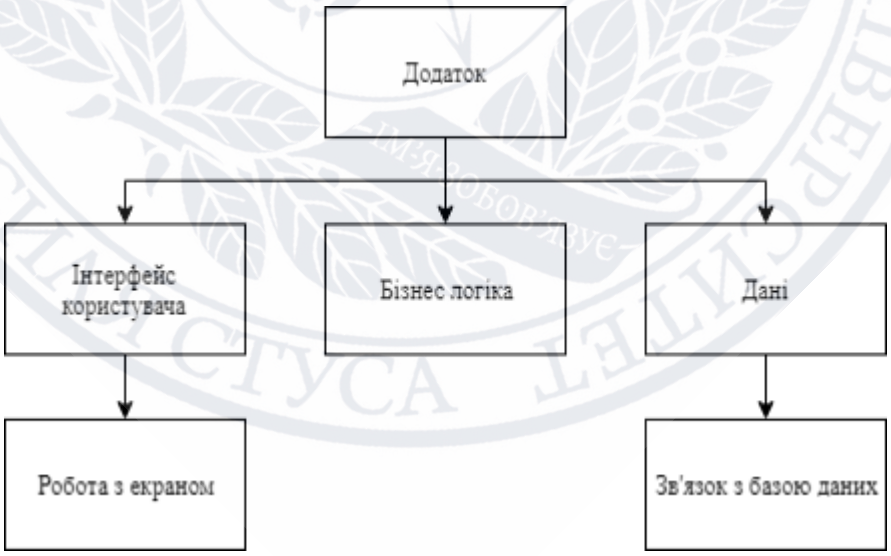


Рисунок 2.4 - Структурна схема додатку для бронювання місць

Архітектура онлайн-додатку побудована за деревоподібним принципом, що забезпечує можливість незалежного контролю та розробки кожного модуля окремо. Такий підхід дозволяє реалізувати логіку обробки даних без необхідності змінювати користувацький інтерфейс.

Система складається з трьох основних рівнів:

1. **Користувацький інтерфейс.** Містить класи для пошуку ресторанів за заданими критеріями та здійснення бронювання столиків. Додатково включає набір допоміжних класів із часто використовуваними функціями, що спрощують роботу з інтерфейсом.
2. **Бізнес-логіка.** Виконує роль посередника між інтерфейсом та даними. Завдяки цьому користувач отримує уніфіковане відображення інформації незалежно від джерела її надходження.
3. **Модуль роботи з даними.** Забезпечує управління різними джерелами даних. У даному випадку він включає серверну частину, яка здійснює API-запити, а також механізм кешування. За потреби до цього рівня можуть бути додані локальна база даних чи зовнішні сервіси, наприклад PostgreeSQL, MySQL, MongoDB, Hostinger, Kamatera, Viltur і т.д.

Запропонована архітектура клієнтської частини та системи загалом гарантує ефективність роботи, підтримує масштабованість і дозволяє вносити зміни окремо як у веб-сервіс, так і в мобільний додаток без модифікації базового коду.

2.3 Формування концептуальних засад Progressive Web Application

Progressive Web Application - це сучасна технологія веб-розробки, яка поєднує переваги класичних веб-ресурсів та мобільних застосунків. Сам термін «прогресивний» підкреслює його вдосконалений характер порівняно зі звичайними

веб-додатками. **Progressive Web Application** забезпечує повноцінний функціонал, але водночас візуально та логічно нагадує мобільний застосунок.

Традиційно встановлення програм на смартфон здійснюється через офіційні магазини — **AppStore** для iOS та **Google Play Market** для Android. У випадку з PWA цей етап спрощується: користувач може інсталювати додаток безпосередньо із сайту, оминаючи стандартні сервіси поширення.

На рисунку 2.5 наведено приклад процесу завантаження прогресивного веб-додатку.

Рисунок 2.6 – Додаток що був завантажений за допомогою технології **Progressive Web Application**

Додаткові переваги Progressive Web Application:

- **Стабільність роботи.** Програма відкривається миттєво та відображає інтерфейс незалежно від якості чи наявності мережевого з’єднання.
- **Висока продуктивність.** Обмін даними здійснюється швидко, а користувацький інтерфейс залишається плавним і чутливим до дій користувача.

- **Зручність та естетичність.** Progressive Web Application забезпечує комфортний та приємний досвід взаємодії, створюючи відчуття роботи з повноцінним мобільним застосунком.

Згідно з позицією **Google**, саме ці характеристики сьогодні дозволяють відрізнити сучасні веб-додатки, що побудовані за принципами Progressive Web Application, від традиційних сайтів, наближаючи їх за виглядом та функціональністю до мобільних застосунків.

2.4 Висновки

Таким чином, у другому розділі здійснено детальний аналіз ключових стадій проєктування онлайн-сервісу. У процесі дослідження було визначено та обґрунтовано переваги технологій, застосованих при створенні системи. Окрему увагу приділено розгляду її архітектурних особливостей.

3 РОЗРОБКА АЛГОРИТМІВ ОСНОВНИХ МОДУЛІВ СИСТЕМИ

3.1. Розробка проекту інформаційної системи бронювання місць

Проєктування інформаційної системи бронювання місць є одним із найбільш відповідальних етапів розробки, оскільки саме на цьому етапі формується логічна структура, принципи взаємодії між компонентами, модель даних та процеси, які забезпечуватимуть роботу всієї системи. Грамотне проєктування дозволяє уникнути помилок на стадії реалізації, забезпечує стабільність роботи та можливість подальшого розширення системи без суттєвих змін у її основній архітектурі.

Проєктування включає кілька ключових кроків: аналіз структури даних, побудову моделі бази даних, проєктування взаємодії між користувачем і системою, визначення програмних модулів та їхніх ролей у внутрішній логіці. Також важливо враховувати майбутню масштабованість, можливість інтеграції з іншими сервісами (платіжними системами, SMS/E-mail сервісами), а також забезпечення безпеки та продуктивності [1-26].

База даних

База даних — це ядро всієї інформаційної системи. Саме тут зберігаються всі відомості про користувачів, місця, зали, бронювання, події та платежі. Модель бази даних має бути логічною, цілісною, оптимізованою та такою, що легко масштабується під час збільшення кількості даних і навантаження.

Основна складність у проєктуванні бази даних ІС бронювання полягає в тому, що система повинна працювати з ресурсами, які швидко змінюють статус (вільне → заброньоване → продане). Тому таблиці та зв’язки між ними повинні бути побудовані таким чином, щоб унеможливити дублювання інформації та забезпечити точність і актуальність даних.

Основні таблиці та їх призначення:

- Users — зберігає інформацію про всіх користувачів: логін, пароль, контактні дані, роль у системі (звичайний користувач, адміністратор, менеджер).

- Halls/Rooms — описує зали, конференц-приміщення, автобуси, літаки або готельні корпуси, у яких здійснюється бронювання.

- Seats — містить інформацію про конкретні місця: ряд, номер, тип місця, координати на схемі.

- Sessions/Trips — описує події, рейси, сеанси або дати заїздів, у межах яких можна забронювати місце.

- Reservations — фіксує всі бронювання, їх статус, дату створення та прив’язку до користувача.

- Payments (за потреби) — містить інформацію про транзакції, платіжні методи, перевірку оплати та стан платежів.

Між таблицями встановлюються зв’язки типу «один-до-багатьох» (наприклад, один зал → багато місць) та «багато-до-багатьох» (наприклад, один користувач може мати багато бронювань, і одна подія може містити багато місць).

Структурна схема взаємозв’язків основних таблиць бази даних ІС бронювання місць наведена на рисунку 3.1.

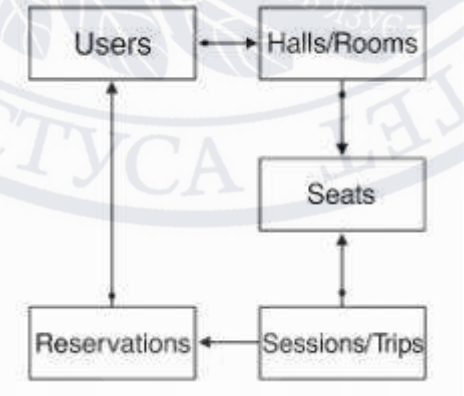


Рисунок 3.1 - ER-модель бази даних

Архітектура системи

Архітектура визначає загальну структуру програмного забезпечення, взаємодію між компонентами та розподілення відповідальності. Найпоширенішим і найбільш ефективним підходом для подібних систем є трирівнева архітектура, або three-tier architecture, яка розділяє систему на три незалежні модулі. Це забезпечує гнучкість, модульність і можливість оновлювати кожен з рівнів без шкоди для інших.

1. Рівень представлення (Frontend)

Це інтерфейс, з яким взаємодіє користувач. Він відображає:

- схеми залів або номерів,
- доступні місця,
- календар бронювань,
- форми авторизації,
- панель керування для адміністратора.

Інтерфейс має бути адаптивним (працювати на телефоні, планшеті, ПК) та інтуїтивно зрозумілим.

2. Рівень логіки (Backend)

Backend відповідає за:

- обробку запитів користувачів,
- керування бронюваннями,
- перевірку зайнятості місць,
- зміну статусів,
- роботу з платежами,
- перевірку прав доступу.

Саме тут реалізуються правила бізнес-логіки: наприклад, блокування місця на час оформлення бронювання або перевірка валідності даних.

3. Рівень даних (Database)

Це база даних, яка забезпечує:

- надійне зберігання інформації,
- цілісність і стабільність даних,
- унеможливлення дублювань.

Логіка взаємодії між компонентами

Щоб забезпечити плавну роботу системи, необхідно продумати механізм обміну даними між усіма рівнями архітектури. Наприклад, коли користувач натискає на місце, система повинна:

1. Передати запит через інтерфейс (Frontend).

2. Обробити запит у Backend: перевірити статус місця, можливість бронювання.

3. Звернутися до бази даних та отримати актуальну інформацію.

4. Повернути результат користувачеві.

Функціональна схема трирівневої архітектури системи бронювання: рівень представлення, рівень логіки та рівень даних наведена на рисунку 3.2.

Масштабованість та перспективи розвитку

Під час проєктування потрібно враховувати, що система може розширюватися: виникнуть нові зали, маршрути, події, збільшиться кількість користувачів. Тому архітектура має дозволяти:

- додавання нових модулів,

- інтеграцію з іншими сервісами,
- збільшення продуктивності,
- розміщення системи на хмарних серверах.

Це створює можливість довготривалого розвитку системи без необхідності її повної переробки.

Блок-схема послідовного процесу взаємодії користувача з системою під час бронювання місця наведена на рисунку 3.3.

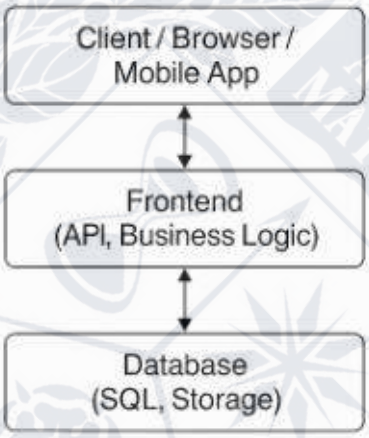


Рисунок 3.2 - Трирівнева архітектура системи

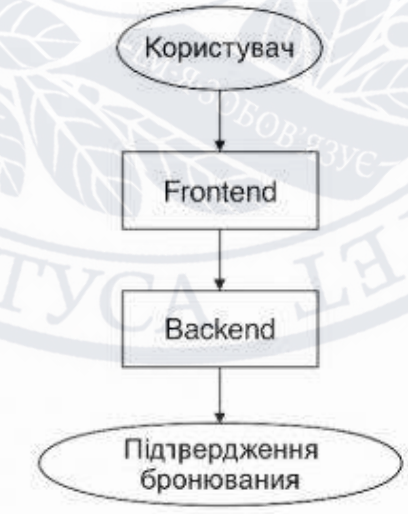


Рисунок 3.3 - Блок-схема процесу бронювання

Етап реалізації інформаційної системи бронювання місць є ключовим у всьому процесі розробки, оскільки саме в цей момент концептуальні моделі, схеми, бізнес-логіка та вимоги перетворюються на робочий програмний продукт. Реалізація включає створення інтерфейсів, написання серверної логіки, організацію взаємодії з базою даних, налаштування механізмів безпеки, тестування та оптимізацію. Від того, наскільки якісно виконаний цей етап, залежить швидкість, стабільність, зручність і безпека всієї системи.

Реалізація інтерфейсу користувача

Одним із перших компонентів, що реалізується, є інтерфейс користувача (Frontend). Він відповідає за візуальне представлення даних і взаємодію з користувачем. На цьому етапі створюється:

- Головна сторінка, де доступний список подій, рейсів, сеансів або номерів.
- Сторінка вибору місць, на якій користувач бачить інтерактивну схему залу або список доступних місць.
- Форма бронювання, що дозволяє ввести персональні дані або увійти в акаунт.
- Особистий кабінет, де зберігається історія бронювань і доступні інструменти управління ними.

Основна складність реалізації інтерфейсу полягає у тому, щоб зробити його інтуїтивно зрозумілим, швидким у роботі та адаптивним. Система повинна коректно відображатися на різних пристроях — від великих моніторів до смартфонів.

Важливо також передбачити кольорові індикатори статусів місць:

- зелений — доступне,

- жовтий — тимчасово заблоковане,

- червоний — заброньоване або продане.

Такі позначення полегшують навігацію та пришвидшують процес бронювання.

Реалізація адміністративної панелі

Другим ключовим компонентом є адміністраторська панель, яка дозволяє співробітникам організації керувати структурою системи та обробляти дані. У ній реалізуються:

- інструменти додавання та редагування залів, номерів, рейсів, сеансів;
- налаштування розкладів, тарифів, категорій місць;
- перегляд активних і завершених бронювань;
- модуль управління користувачами;
- механізми формування звітів.

Адмін-панель має бути зручною й безпечною, оскільки саме через неї здійснюється управління всією системою. Потрібно реалізувати чітку систему ролей та дозволів: адміністратор з повними правами, менеджер із обмеженими правами, оператор, касир тощо.

Алгоритми перевірки доступності місць

Для запобігання конфліктних ситуацій надзвичайно важливо реалізувати правильні алгоритми перевірки доступності місця.

Система повинна:

1. Перевірити статус місця в базі даних.

2. Тимчасово заблокувати місце (hold), щоб інший користувач не міг його взяти.

3. Дозволити користувачу завершити оформлення бронювання протягом визначеного часу (наприклад, 5–10 хвилин).

4. Автоматично звільнити місце, якщо бронювання не завершено.

Такий механізм часто називають транзакційним резервуванням. Він критично важливий у періоди пікового навантаження — під час популярних концертів, розпродажів або святкових рейсів.

Обробка одночасних запитів

Ще одне важливе завдання — забезпечити стабільну роботу системи при великій кількості одночасних запитів.

Для цього реалізуються:

- механізми блокування записів у базі даних (Row Lock / Table Lock);
- черги запитів (Queue Processing) через Redis або RabbitMQ;
- оптимізація SQL-запитів;
- кешування часто використовуваних даних;
- балансування навантаження між кількома серверами.

Правильна реалізація цього механізму гарантує, що система не «впаде» у моменти масового бронювання, а користувачі отримають коректні дані.

Перевірка валідності введених даних

Усі дані, які вводить користувач, повинні проходити подвійну перевірку:

1. На клієнтському рівні (Frontend) — базову перевірку правильності формату (телефон, email, кількість місць, дата).

2. На серверному рівні (Backend) — повну перевірку достовірності, щоб уникнути спроб некоректного або шкідливого введення.

Завдяки цьому система стає більш безпечною та стабільною.

Механізми оновлення інформації

Бронювання — динамічний процес, тому система повинна уміти оновлювати інформацію в режимі реального часу.

Для цього застосовують:

- WebSocket-з’єднання;
- AJAX-запити;
- SignalR або інші push-технології;
- автоматичне оновлення статусів без перезавантаження сторінки.

Користувач повинен миттєво бачити, що місце стало недоступним або доступним.

Мінімізація помилок при збереженні бронювання

Збереження нового бронювання — одна з найважливіших операцій. Система повинна виконувати:

- контроль цілісності транзакції;
- автоматичне відновлення у разі збоїв;
- логування всіх дій;
- збереження історії оновлень;
- перевірку конфліктів (випередження або дублювання).

Також необхідно передбачити механізм резервного копіювання даних, щоб у разі аварії система могла швидко відновити інформацію.

Інтеграція додаткових сервісів

На практиці система бронювання часто інтегрується з зовнішніми сервісами:

- платіжними платформами (LiqPay, WayForPay, Stripe);
- SMS/E-Mail сервісами (Twilio, SendGrid);
- CRM або ERP системами;
- мобільними застосунками.

Це розширює функціональність і робить систему універсальною.

3.2. Проектування внутрішньої організації системи

Структурна схема функціонування онлайн-сервісу наведена на рисунку 3.4. У межах цієї системи користувач взаємодіє з графічним інтерфейсом, здійснюючи пошук закладів громадського харчування за визначеними критеріями та виконуючи процедуру бронювання. Модуль бронювання реалізовано за допомогою спеціалізованих класів, які відповідають за перевірку доступності столиків та формування результатів для відображення користувачеві.

Рисунок 3.4 - Архітектура додатку для вирішення задачі резервування місць у закладах харчування

3.3. Розробка структурної схеми бронювання місць в закладах громадського харчування

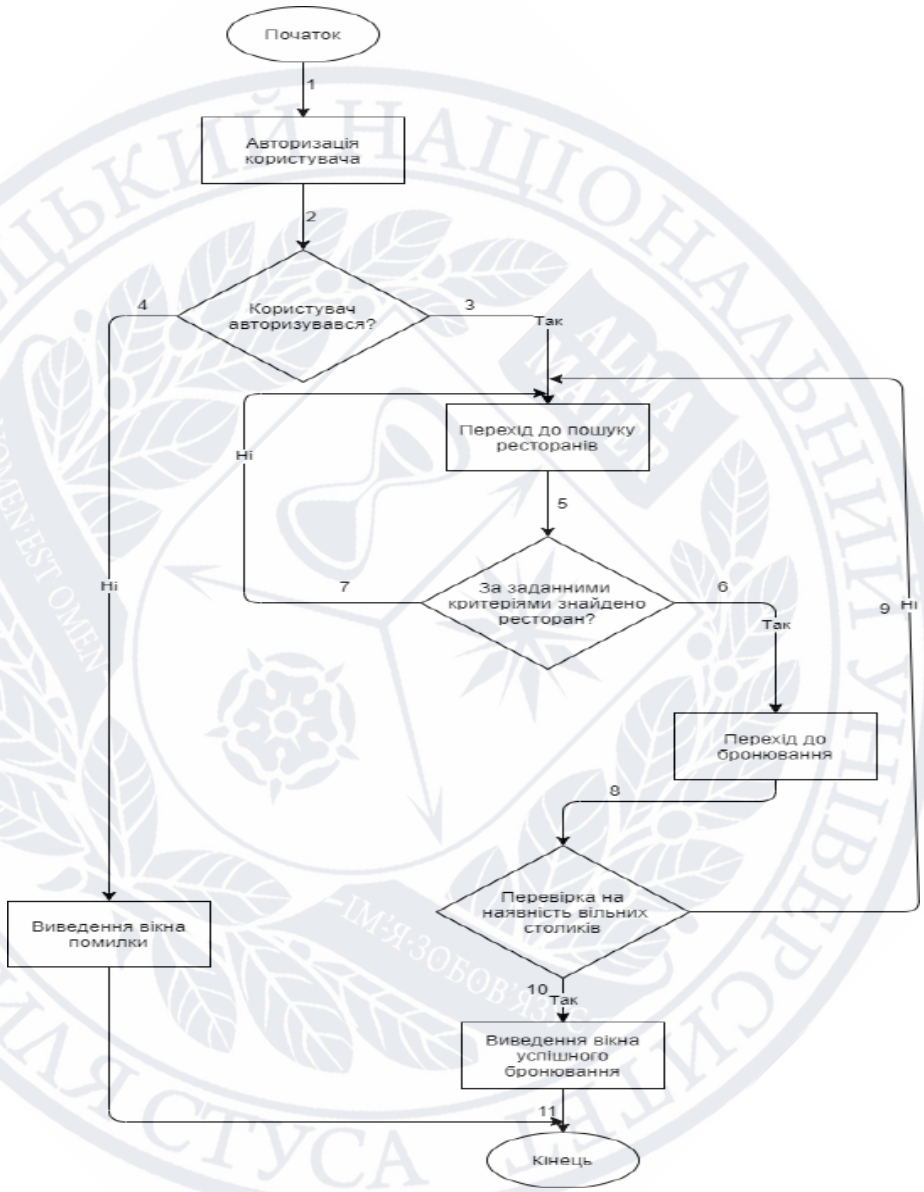


Рисунок 3.5 – Блок схема роботи програмного забезпечення для вирішення задачі бронювання місця у закладі харчування споживачем

На рисунку 3.5 представлено блок-схему роботи програмного забезпечення

для вирішення задачі бронювання місця у закладі харчування споживачем, яка відображає послідовність дій користувача під час роботи з онлайн-сервісом бронювання столиків у закладах громадського харчування.

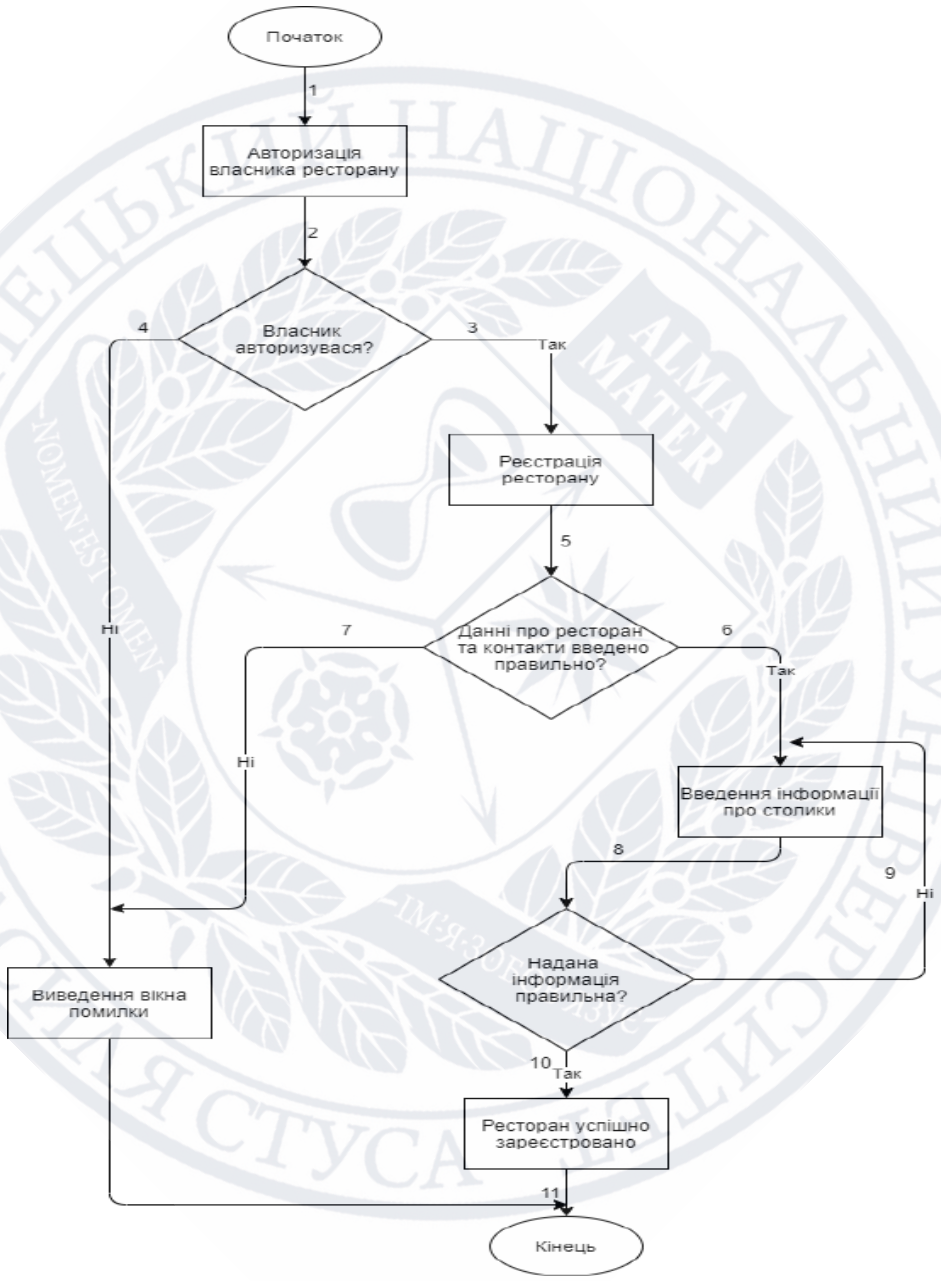


Рисунок 3.6 – Блок схема реєстрації закладу харчування у програмному забезпеченні для вирішення задачі бронювання місця у закладі харчування споживачем

На рисунку 3.6 наведено блок схему, яка відображає послідовність дій власника закладу громадського харчування під час взаємодії з онлайн-сервісом бронювання столиків, включно з процесом реєстрації його ресторану в системі.

Для виконання даної процедури, необхідно, щоб власник зареєструвався у системі та заповнив необхідні поля реєстраційної форми закладу.

3.4. Висновки

У третьому розділі було представлено концепцію та здійснено проєктування інформаційної системи, орієнтованої на онлайн-бронювання столиків у закладах громадського харчування. Розроблено алгоритми функціонування ключових модулів, що забезпечують взаємодію як користувачів, так і власників закладів із сервісом. Окремо продемонстровано схему роботи графічного інтерфейсу та механізм запису даних до бази даних.

Процес бронювання передбачає кілька послідовних етапів:

- реєстрацію та авторизацію користувача;
- вибір закладу зі списку із застосуванням сортування за визначеними критеріями;
- формування запиту щодо наявності вільних столиків;
- підтвердження бронювання у випадку позитивної відповіді системи.

Для власників закладів передбачено окремий процес реєстрації у базі даних сервісу. Він потребує верифікації введених даних, що здійснюється у режимі онлайн за участю технічного персоналу системи.

У подальшому на платформі формується модель розташування столиків у закладах громадського харчування із зазначенням усіх релевантних критеріїв, які можуть бути важливими для користувачів. Це забезпечує можливість здійснювати бронювання відповідно до індивідуальних потреб клієнтів.

4. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Аргументація вибору мови програмування для реалізації системи

Для проєктування архітектури та реалізації клієнтської частини системи обрано мову програмування **JavaScript** у поєднанні з бібліотекою **React**, що забезпечує створення інтерактивних інтерфейсів. Для організації роботи з даними на стороні клієнта використовується бібліотека **Redux**, яка дозволяє ефективно керувати станом додатку.

JavaScript є об’єктно-орієнтованою мовою сценаріїв, призначеною для розробки інтегрованих застосунків, що можуть виконуватися як на клієнтській, так і на серверній стороні. Найчастіше вона застосовується у вигляді компонентів, інтегрованих у веб-браузер, що дає змогу створювати динамічні веб-сторінки та вдосконалені інтерфейси.

Мова використовує можливості середовища виконання та відповідає стандарту **ECMAScript**, будучи його діалектом. JavaScript характеризується як динамічна мова сценаріїв [1 - 37].

Сфери застосування JavaScript включають:

- створення інтерактивних сценаріїв для веб-сторінок;
- розробку **Single Page Application (SPA)** за допомогою бібліотек і фреймворків (React, AngularJS, Vue.js);
- серверне програмування (Node.js);
- створення настільних застосунків (Electron, NW.js);
- розробку мобільних застосунків (React Native, Cordova);

- написання сценаріїв для прикладного програмного забезпечення (наприклад, Adobe Creative Suite, Apache JMeter) [1–37].

Таблиця 4.1 – Порівняння мов програмування

Властивість	JavaScript	C#
Швидкодія	+	-
Ручне управління пам’яттю	+	-
Перевантаження функцій	+	+
ООП	+	+
Шаблони	+	+
Значення параметрів за замовчуванням	+	+
Багатовимірні масиви	+	+
Динамічні масиви	+	+
Множинне наслідування	+	+

3.2 Критерії та обґрунтування застосування бібліотеки в системі

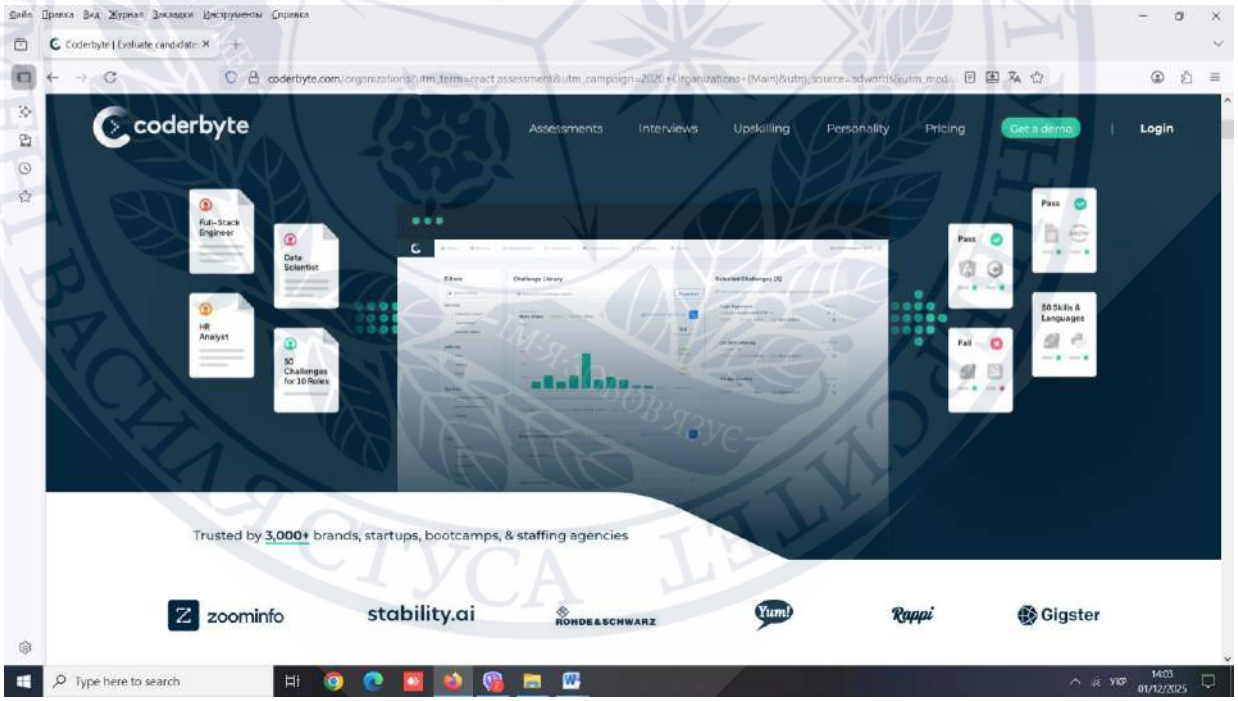
Сучасне середовище веб-програмування характеризується динамічним розвитком та постійним оновленням інструментарію. Для мови **JavaScript** створено значну кількість бібліотек і фреймворків, що розширюють її можливості та спрощують процес розробки. У межах даного проєкту було обрано бібліотеку **React**, яка забезпечує ефективну реалізацію клієнтської частини системи.

React - це спеціалізована бібліотека для JavaScript, призначена для побудови **Single Page Application (SPA)**, тобто односторінкових веб-додатків. Основна перевага таких застосунків полягає в тому, що при переході між сторінками браузер

не виконує повного перезавантаження, що істотно підвищує швидкодію та забезпечує плавність роботи інтерфейсу.

Ключові особливості React включають:

- компонентний підхід до розробки інтерфейсу;
- повторне використання елементів коду;
- високу продуктивність завдяки віртуальному DOM;
- гнучку інтеграцію з іншими бібліотеками та фреймворками;
- активну підтримку спільноти та регулярні оновлення.



Лінійна схема поширення даних.

У середовищі **React** властивості передаються від батьківських компонентів до дочірніх. Останні отримують їх у вигляді набору незмінних значень (*Immutable*), що унеможливлює їх пряме редагування всередині компонента. Зміни можуть бути ініційовані лише через виклик спеціальних *callback*-функцій. Такий принцип взаємодії описується формулою: **«дані рухаються вниз, а події — вгору»**.

1. **Віртуальний DOM.** React застосовує концепцію *Virtual DOM*, створюючи в пам'яті кешовану структуру інтерфейсу. Це дозволяє визначати різницю між попереднім і поточним станами та оптимізувати процес оновлення реального DOM у браузері. Таким чином розробник працює з уявленням сторінки як цілісного об'єкта, тоді як бібліотека самостійно вирішує, які саме елементи потребують оновлення.

2. **JSX**

JavaScript XML (JSX) — це синтаксичне розширення JavaScript, що дає змогу описувати структуру інтерфейсу у формі, подібній до HTML. Хоча більшість компонентів створюється саме з використанням JSX, можливе й застосування чистого JavaScript. Історично JSX має схожість із мовою XHP, розробленою компанією Facebook для розширення PHP [3].

Для побудови дизайну інтерфейсу використано **HTML**, **CSS** та бібліотеку **Material UI**, яка інтегрується з React і містить набір готових компонентів для повторного використання.

Redux виступає менеджером стану додатку. Хоча найчастіше він застосовується разом із React, його функціонал не обмежується лише цією бібліотекою. Redux реалізує концепцію централізованого управління станом, що

ґрунтується на кількох базових принципах. Їхнє розуміння дозволяє ефективно вирішувати проблеми, пов’язані з керуванням даними у складних застосунках.

Лого **Redux** представлено на рис. 4.2.

Рисунок 4.2 - Redux logo

Однією з ключових особливостей бібліотеки **React** є підтримка синтаксичного розширення **JSX**, яке за своєю структурою нагадує HTML, але компілюється у JavaScript. Це забезпечує зручність опису інтерфейсу та підвищує зрозумілість коду.

Для оптимізації продуктивності React застосовує механізм **Virtual DOM**, що дозволяє ефективно порівнювати попередній і поточний стан інтерфейсу та оновлювати лише ті елементи, які зазнали змін. Завдяки цьому користувач отримує швидкий відгук системи без тривалого очікування завантаження даних чи відображення сторінки.

Ще однією перевагою є можливість створення **ізоморфних додатків**, які поєднують клієнтську та серверну логіку, забезпечуючи швидке відображення контенту та зменшення часу очікування користувача.

Компонентний підхід у React сприяє легкому модифікуванню та повторному використанню елементів у нових проєктах. Високий рівень перевикористання коду

підвищує ефективність тестування, що, у свою чергу, забезпечує кращий контроль якості програмного продукту.

Окрім веб-розробки, технологія **React Native** дозволяє створювати мобільні застосунки для платформ **Android** та **iOS**, використовуючи вже наявний досвід роботи з JavaScript та React [37–49].

Ізоморфні додатки

Поняття ізоморфних додатків або ізоморфного JavaScript означає можливість використання одного й того ж програмного коду як на стороні сервера, так і на стороні клієнта. У традиційних SPA-додатках (Single Page Application) користувач під час відкриття сайту змушений чекати завантаження даних із сервера, що часто супроводжується відображенням порожньої сторінки чи анімації завантаження. З огляду на сучасні вимоги до швидкодії, затримка понад дві секунди сприймається як суттєвий недолік. Додатковою проблемою є недостатня індексація таких сторінок пошуковими системами. Виконання JavaScript-коду на сервері дозволяє усунути ці обмеження, забезпечуючи попередній рендеринг сторінки та покращуючи користувацький досвід. Такий підхід сприяє швидшому відображенню контенту, кращій індексації та скороченню часу розробки. У випадку React розробники мають змогу створювати універсальні компоненти, що функціонують одночасно на клієнтській і серверній стороні.

Virtual DOM

Document Object Model (DOM) є ієрархічним представленням структури HTML-, XHTML- та XML-документів у вигляді дерева елементів. Для динамічних веб-сторінок важливо забезпечити швидке оновлення DOM після зміни стану елементів. У деяких фреймворках застосовується метод «dirty checking», який передбачає регулярне опитування стану документа, що може бути надмірно

ресурсомістким у високонавантажених системах. React використовує концепцію **Virtual DOM**, що зберігається у пам’яті. Зміни спочатку вносяться у віртуальну модель, після чого бібліотека порівнює її з реальним DOM і оновлює лише ті компоненти, які змінилися. Це забезпечує значне підвищення продуктивності та ефективності роботи додатків. Крім того, ізоморфна природа React дозволяє виконувати рендеринг як на сервері, так і на клієнті.

Повторне використання коду

Мобільні додатки мають низку переваг порівняно з веб-сайтами: вони можуть працювати без підключення до Інтернету, мають доступ до функцій пристрою (наприклад, push-сповіщень) і забезпечують постійну взаємодію з користувачем. **React Native** — це фреймворк, що дозволяє створювати мобільні застосунки для Android та iOS, використовуючи JavaScript та досвід веб-розробки з React. Компоненти, створені для веб-середовища, можуть бути адаптовані для мобільних платформ, що значно скорочує час і витрати на розробку.

Порівняння підходів показує:

- нативні додатки забезпечують високу продуктивність, але потребують значних фінансових витрат;
- фреймворки на основі HTML5, CSS3 та JavaScript (наприклад, PhoneGap) знижують вартість, але поступаються у швидкодії;
- використання React дозволяє досягти продуктивності, близької до нативних рішень, при цьому витрати залишаються помірними.

Ключовою перевагою є **перевикористання коду**: добре спроєктовані компоненти можна застосовувати повторно, що зменшує кількість нового коду, скорочує ризик виникнення помилок і підвищує якість тестування. Оскільки такі

компоненти вже були апробовані у реальних проєктах, розробники краще розуміють їхню поведінку та можуть швидше локалізувати можливі проблеми.

3.3 Специфіка середовища програмування і причини його використання

Visual Studio Code - це багатоплатформовий редактор вихідного коду, створений компанією *Microsoft* для операційних систем **Windows**, **Linux** та **macOS**. Він позиціонується як «легкий» інструмент для кросплатформної розробки веб- та хмарних застосунків. Серед його основних можливостей — інтегрований відлагоджувач, підтримка системи контролю версій **Git**, підсвічування синтаксису, технологія **IntelliSense** та засоби для рефакторингу коду. Редактор вирізняється високим рівнем кастомізації: користувач може налаштовувати теми, комбінації клавіш та конфігураційні файли. Продукт поширюється безкоштовно, розробляється як проєкт із відкритим вихідним кодом, проте готові збірки мають пропрієтарну ліцензію [10].

Архітектурно Visual Studio Code базується на фреймворку **Electron**, який дозволяє створювати настільні застосунки з використанням **Node.js** та рушія **Blink**. Водночас редактор не є похідним від Atom, а використовує власний веб-редактор **Monaco**, розроблений для Visual Studio Online.

Редактор підтримує широкий спектр мов програмування та забезпечує функції навігації по коду, налагодження, рефакторингу й інтеграцію з Git. Значна частина можливостей реалізується не через графічний інтерфейс, а за допомогою **палітри команд** або конфігураційних JSON-файлів. Палітра команд функціонує як аналог командного рядка, що викликається спеціальними комбінаціями клавіш.

Крім того, Visual Studio Code дозволяє змінювати кодування файлів при збереженні, налаштовувати символи переносу рядків та визначати мову програмування для поточного документа.

Починаючи з 2018 року для **Visual Studio Code** було створено розширення **Python** з відкритим вихідним кодом, яке значно розширює можливості розробників у сфері редагування, налагодження та тестування програмного коду.

Станом на березень 2019 року через інтегрований користувацький інтерфейс редактора можна завантажити та встановити тисячі додаткових модулів, лише у категорії «мови програмування». Це забезпечує гнучкість середовища та його адаптацію до потреб різних проєктів.

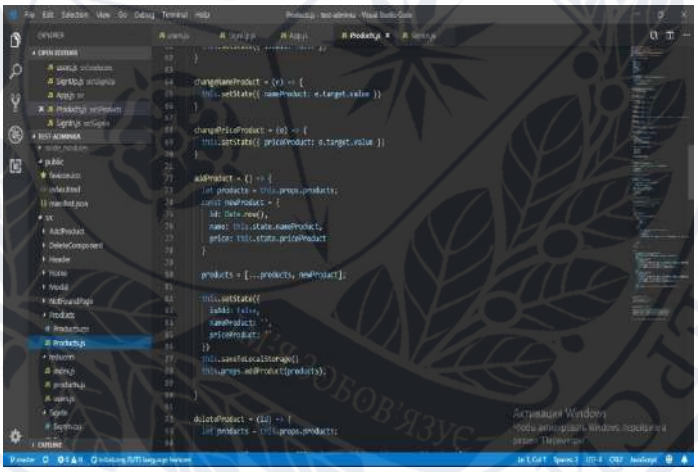


Рисунок 4.3 – Робоче вікно Microsoft VSC.

Brackets — це текстовий редактор, розроблений компанією *Adobe* для роботи з мовами **JavaScript**, **HTML** та **CSS**. Його вихідний код створено із застосуванням веб-технологій і поширюється за ліцензією **MIT**. Редактор реалізовано як окремий

настільний застосунок, для інсталяції якого доступні пакети **deb**, **dmg** та **msi** для операційних систем Linux, macOS та Windows.

Однією з ключових функцій Brackets є режим **Live Development**, що забезпечує миттєве відображення змін у браузері під час редагування коду. Це дозволяє розробнику одразу бачити результат внесених правок. Налагодження також може виконуватися синхронно з браузером: є можливість встановлювати точки зупину та переглядати попередні стани.

Редактор має вбудовану підтримку препроцесорів **LESS** та **SCSS**, а також систему контекстних інструментів, які з'являються у робочому вікні залежно від потреб користувача. Для розширення функціоналу передбачено систему доповнень. Серед найбільш корисних плагінів можна виділити:

- **Emmet** — для швидкого введення шаблонних конструкцій;
- **Beautifier** — для автоматичного форматування коду;
- **FTP-Sync** — для синхронізації з віддаленим сервером;
- **ColorHints** — для зручного вибору кольорів.

Brackets сумісний із різними версіями операційних систем Windows (Vista, 7–10).

Приклад роботи редактора **Brackets** наведено на рис. 4.4.

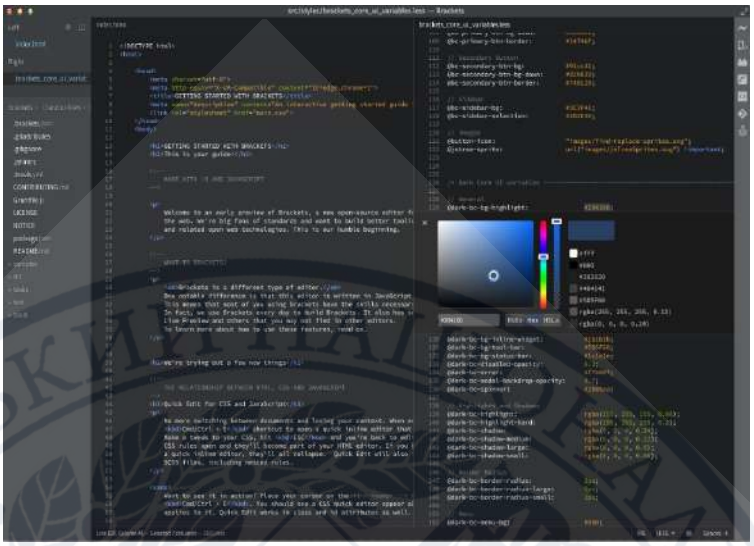


Рисунок 4.4 – Робоче вікно програми «Brackets»

Sublime Text — це високопродуктивний кросплатформний редактор тексту, орієнтований на швидку роботу та гнучкість у налаштуванні. Він підтримує систему плагінів, що створюються мовою програмування **Python**, завдяки чому функціонал редактора може бути значно розширений.

Програмне забезпечення не належить до категорії відкритого чи вільного, однак частина плагінів поширюється за відкритими ліцензіями та активно підтримується спільнотою розробників.

Редактор має широкий вибір візуальних тем, а також можливість завантаження додаткових оформлень. Для зручності користувачів у правій частині інтерфейсу відображається **міні-карта коду**, яка дозволяє швидко орієнтуватися у файлі та здійснювати навігацію кліком.

Sublime Text підтримує кілька режимів відображення:

- багатопанельний режим (від 1 до 4 панелей), що дає змогу працювати одночасно з кількома файлами;

- «повноекранний» режим (*free modes*), у якому відкритий лише один файл без додаткових меню та інтерфейсних елементів.

Приклад роботи редактора **Sublime Text 3** наведено на рисунку 4.5.

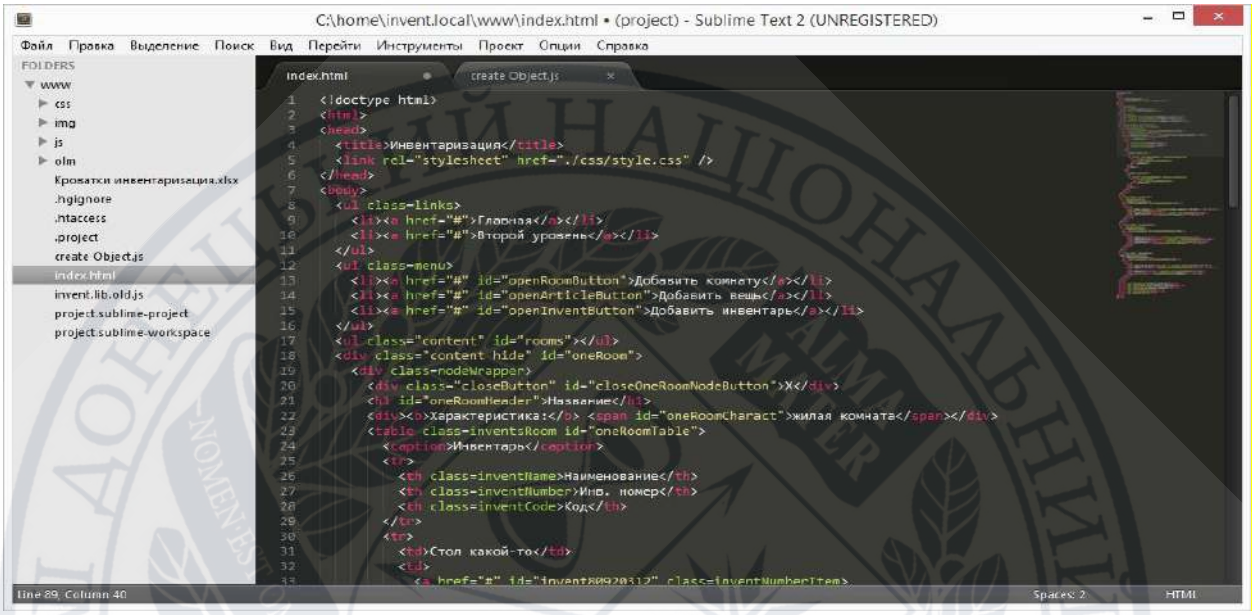


Рисунок 4.5 – Робоче вікно програми Sublime Text 3

Узагальнені результати зіставлення характеристик програмних середовищ подано в Таблиці 4.2.

Таблиця 4.2 – Основні властивості IDE

Функції	Середовище програмування		
	Brackets	Sublime Text 3	Microsoft Visual Studio
Режим відлагодження (0,9)	0.5	0.5	1
Кросплатформеність (0,6)	0	0	1
Функція авто заповнення (0,4)	0.5	0.5	1
Загальний коефіцієнт	0,65	0,5	1.9

3.4 Дослідження підходів та інструментарію для тестування

Тестування визначається як цілеспрямований процес перевірки програмного продукту шляхом виконання спеціально розроблених тестів, що мають на меті виявлення помилок та недоліків, допущених під час його створення. Сутність цього процесу полягає у запуску програмного забезпечення, спостереженні за його функціонуванням та аналізі отриманих результатів. У випадку виникнення збоїв формується звіт, який підлягає детальному розгляду для встановлення джерела та характеру помилки.

Ефективність тестування залежить від обраної стратегії та послідовності дій, адже саме вони визначають якість і повноту перевірки. У сучасній практиці тестування веб - ресурсів застосовуються такі основні види:

1. **Функціональне тестування.** Спрямоване на перевірку коректності реалізації функцій та логіки роботи системи. Наприклад, для інтернет-магазину важливо врахувати всі можливі сценарії: наявність знижок, різні статуси замовлень, кількість товарів тощо. Якщо функціонал некоректно працює хоча б в одному браузері, це свідчить про загальну несправність системи.
2. **Тестування зручності користування.** Орієнтоване на оцінку практичності та комфортності взаємодії користувача із сайтом, забезпечення інтуїтивності та простоти навігації.
3. **Тестування продуктивності.** Виконується переважно у вигляді навантажувальних перевірок із застосуванням спеціалізованих програмних засобів. Мета полягає у визначенні здатності системи витримувати одночасні запити великої кількості користувачів. Наприклад, перевірка роботи

сайту при 100 паралельних діях (купівля товарів, авторизація) дозволяє оцінити його стійкість до навантаження.

4. **Тестування інтерфейсу користувача.** Передбачає перевірку графічних елементів сайту на відповідність вимогам до дизайну та зручності відображення.

5. **Тестування безпеки.** Є ключовим етапом забезпечення надійності веб-ресурсів. Основна увага приділяється пошуку вразливостей до атак, зокрема SQL-ін’єкцій, які дозволяють впроваджувати шкідливий код у запити до бази даних і становлять серйозну загрозу для користувачів.

Окрім зазначених видів, важливими критеріями сучасних веб - сайтів є **кросбраузерність** та **адаптивність**. Кросбраузерність означає коректне відображення та функціонування елементів у різних браузерах. Перевірка проведена в Google Chrome, Firefox та Opera показала відсутність помилок: усі елементи сайту відображаються однаково та працюють стабільно.

3.5 Оцінювання роботи основних структурних елементів системи

У результаті виконання проєкту було створено веб-сервіс, що забезпечує можливість електронного бронювання місць у закладах ресторанного типу. Архітектура системи побудована на модульному принципі та включає такі основні складові:

- Модуль автентифікації користувачів
- Модуль пошуку закладів
- Модуль оформлення бронювання

Модуль автентифікації

Для отримання повного доступу до функціональних можливостей сервісу користувач має пройти процедуру автентифікації. Першим етапом є реєстрація, яка передбачає введення персональних даних: ім’я, електронна адреса, номер телефону, інтереси та пароль.

Після завершення реєстрації користувач переходить до форми входу, де здійснює авторизацію для подальшої роботи із системою.

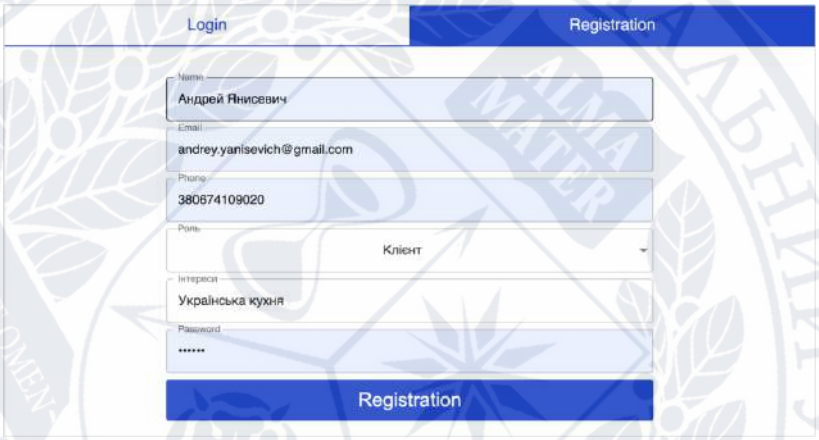


Рисунок 4.6 – Реєстраційна інформація

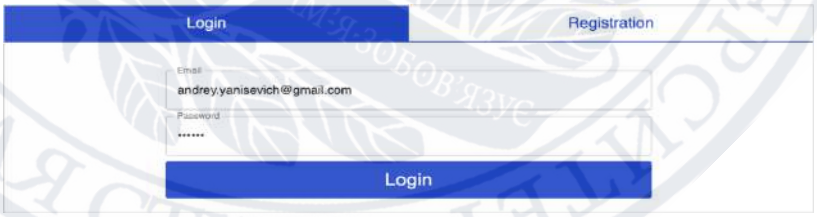


Рисунок 4.7 – Робоче вікно модуля автентифікації

Інтерфейс основної сторінки (рис. 4.8) містить такі складові: навігаційну шапку, форму для пошуку ресторанів та відображення їхнього переліку.

Після завершення етапу програмної реалізації та інтеграції всіх модулів

інформаційної системи настає один із найважливіших періодів — тестування та оцінка ефективності роботи системи. Саме на цьому етапі визначається якість розробленого рішення, його стабільність, безпека, працездатність та здатність витримувати реальні сценарії використання. Незалежно від того, наскільки добре була спроєктована система, лише комплексне тестування дозволяє виявити недоліки, усунути помилки та оптимізувати роботу.

Завершальний етап реалізації перед тестуванням. Після створення інтерфейсів, налаштування логіки бронювання, реалізації адміністраторської панелі, механізмів синхронізації та алгоритмів обробки даних, проводяться підготовчі роботи до тестування. До них належать:

- Збір та об’єднання всіх модулів у цілісну систему.
- Попереднє технічне тестування розробниками.
- Оптимізація критичних ділянок.
- Підготовка технічної та користувацької документації.
- Налаштування серверного середовища.

Сюди входить конфігурація веб-сервера, розгортання бази даних, налаштування системи логування, підключення сервісів безпеки та можливих інтеграцій з API. Лише після виконання всіх цих кроків система переходить на наступний рівень — до повного циклу тестувань.

Функціональне тестування.

Функціональне тестування перевіряє, наскільки система відповідає встановленим вимогам та чи працюють усі процеси так, як заплановано. На цьому етапі оцінюються:

- коректність відображення місць у режимі реального часу;

- правильність роботи алгоритмів бронювання;
- точність зміни статусів місць («вільне», «забронюване», «продане»);
- робота особистого кабінету користувача;
- можливість оформлення, редагування та скасування бронювання;
- взаємодія адміністратора з даними, оновлення сеансів, залів та розкладів.

Важливо перевірити поведінку системи у випадках, коли:

- користувач вводить некоректні або неповні дані,
- бронювання оформлюється під час зміни статусу місця,
- одночасно кілька людей намагаються обрати те саме місце.

Такі сценарії дозволяють побачити реальні слабкі місця системи.

Навантажувальне тестування

Оскільки система бронювання зазвичай переживає пікові навантаження (наприклад, під час відкриття продажів на концерт або святкові рейси), обов’язково проводиться тестування продуктивності. Це включає:

- моделювання великої кількості одночасних звернень до сервера;
- перевірку здатності бази даних обробляти десятки або сотні паралельних транзакцій;
- оцінку часу реакції системи;
- перевірку роботи механізмів блокування місць під час масового бронювання;
- симуляцію довготривалого навантаження.

Таке тестування допомагає виявити критичні затримки, «вузькі місця» в логіці, неоптимальні запити до бази даних та недоліки архітектури. Після аналізу результатів проводиться необхідна оптимізація: кешування даних, індексація таблиць, підсилення серверних ресурсів.

Тестування безпеки

Безпека — один із найважливіших аспектів будь-якої сучасної системи. Під час тестування перевіряють:

- захищеність персональних даних користувачів;
- наявність шифрування важливих даних;
- захист від SQL-ін’єкцій, XSS, CSRF та інших типових атак;
- правильність роботи системи аутентифікації та авторизації;
- обмеження доступу адміністративних функцій для неавторизованих осіб;
- поведінку системи при спробі навмисного порушення логіки (наприклад, подвійне бронювання або обхід бізнес-правил).

Результатом цього етапу є не лише виправлення знайдених проблем, а й удосконалення всієї системи захисту [37-50].

Перевірка адміністративних функцій

Адміністратори та менеджери відіграють важливу роль у підтримці системи, тому тестування охоплює:

- додавання й редагування залів і сеансів;
- зміну статусів бронювань;
- перевірку формування звітів;

- обробку помилок;
- перевірку ролей і прав доступу.

Тут важливо не лише перевірити правильність виконання дій, а й оцінити зручність роботи адміністратора з системою.

Загальна оцінка ефективності системи

Після завершення тестування проводиться комплексний аналіз, який визначає, наскільки система відповідає поставленим цілям. Оцінюється:

- прискорення процесів бронювання — чи стала процедура швидшою порівняно з попередніми методами;

- зменшення кількості помилок — чи вдалося системі запобігти подвійним бронюванням, плутанині та неточностям;

- спрощення керування розкладом та місцями — чи отримав адміністратор зручніше та ефективніше робоче середовище;

- підвищення зручності для користувачів — чи є інтерфейс інтуїтивно зрозумілим і швидким;- стабільність роботи — як система поводиться під навантаженням;- безпека — чи немає критичних ризиків;- масштабованість — можливість подальшого розвитку.

Лише після проходження всіх етапів тестування та підтвердження ефективності система може бути передана замовнику і введена в експлуатацію. Завдяки цьому гарантовано, що користувачі отримають стабільний, надійний та зручний сервіс.

ВИСНОВКИ

Розробка інформаційної системи бронювання місць є важливим і необхідним етапом у процесі цифрової трансформації сучасних сервісів. Упровадження такої системи дозволяє автоматизувати рутинні операції, які раніше виконувалися вручну та вимагали значних трудових і часових витрат. Завдяки автоматизації процесів резервування підвищується загальна ефективність роботи організацій, покращується якість обслуговування клієнтів і зменшується кількість помилок, пов’язаних із людським фактором. Створена система надасть користувачам можливість швидко і зручно переглядати доступні місця, обирати конкретні позиції та оформлювати бронювання без необхідності звернення до кас чи операторів. Крім того, інформаційна система бронювання забезпечить високий рівень точності та надійності даних. Важливим є й те, що система здатна автоматично формувати статистичні та управлінські звіти, які дають керівництву можливість глибше аналізувати діяльність підприємства, планувати подальший розвиток і вчасно приймати стратегічні рішення.

Упровадження подібних систем значно підвищує конкурентоспроможність організацій, оскільки сучасні клієнти очікують швидкого, зручного й технологічного сервісу. Наявність інтуїтивної системи онлайн-бронювання часто є вирішальним фактором під час вибору закладу чи послуги. Така система відповідає актуальним тенденціям цифровізації, мобільності та автоматизації, що робить її важливою складовою успішної діяльності в умовах сучасного ринкового середовища. Отже, розробка і впровадження інформаційної системи бронювання місць є не лише технічним оновленням, а й стратегічним інструментом удосконалення бізнес-процесів, покращення взаємодії з клієнтами та підвищення загальної ефективності підприємства. Це інвестиція в майбутнє, яка окупається завдяки стабільності, точності та можливості подальшого розвитку системи.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Андрущенко, Т. В. Інформаційні системи в сервісних підприємствах: сучасні рішення та тренди. – Київ: КНЕУ, 2024. – 312 с.
2. Бодненко, Д. М. Архітектура програмних систем: навч. посіб. – Київ: Ліра-К, 2025. – 280 с.
3. Гуменюк, Ю. О. Бази даних та інформаційні сховища: моделювання і оптимізація. – Львів: ЛНУ ім. Франка, 2024. – 364 с.
4. ДСТУ ISO/IEC 25010:2023. Інженерія систем та програмного забезпечення. Вимоги до якості. – Київ: УкрНДНЦ, 2024.
5. ДСТУ ISO/IEC 27001:2024. Системи управління інформаційною безпекою. Вимоги. – Київ: Мінекономіки, 2024.
6. Євтушенко, К. О. Проєктування інформаційних систем: сучасні методи та практики. – Харків: ХНЕУ, 2025. – 248 с.
7. Захарченко, Л. П. Web-розробка та інтерактивні інтерфейси. – Київ: Видавничий дім «Освіта», 2024. – 290 с.
8. Кравченко, І. В. Методологія створення корпоративних ІС. – Дніпро: ДНУ, 2025. – 308 с.
9. Козак, С. І. Основи тестування програмного забезпечення: навч. посіб. – Тернопіль: ТНТУ, 2024. – 212 с.
10. Литвин, В. М., Шаховська, Н. Б. Інтелектуальні ІС: моделі, алгоритми, застосування. – Львів: ЛП, 2024. – 420 с.
11. Мазаракі, А. А. Цифровізація сервісних послуг: виклики та рішення. – Київ: КНТЕУ, 2025. – 280 с.
12. Мельник, О. В. Програмна інженерія: від вимог до впровадження. – Львів: Сполом, 2024. – 356 с.

13. Олійник, В. Ю. Хмарні технології та веб-сервіси: навч. посіб. – Київ: КНУ ім. Шевченка, 2025. – 260 с.
14. Петренко, Д. В. Технології кібербезпеки в сучасних ІС. – Київ: ДУТ, 2024. – 300 с.
15. Pichler, M. Modern Web-Based Booking Systems: Architecture and Performance. – Springer, 2024. – 215 p.
16. Johnson, A. Designing Reservation Systems: User Experience and Data Flows. – O'Reilly Media, 2025. – 198 p.
17. IEEE Standard 29148-2024. Systems and Software Engineering — Requirements Engineering. – IEEE, 2024.
18. ISO/IEC 42001:2024. Artificial Intelligence Management Systems — Requirements. – Geneva: ISO, 2024.
19. European Union Agency for Cybersecurity (ENISA). Cybersecurity Guidelines for Online Service Platforms. – Brussels, 2025.
20. MDN Web Docs. Web Technologies Documentation. – URL: https://developer.mozilla.org/
21. W3C. Web Application Security and Architecture Guidelines. – URL: https://www.w3.org/
22. Sharma, K. Load Testing Techniques for High-Traffic Web Systems. – ACM Journal, 2024. – Vol. 19(3), P. 44–58.
23. Nguyen, T. Advanced Database Transactions in Multi-User Systems. – IEEE Computing, 2025. – №1, P. 18–33.
24. Li, J. Scalable Backend Architecture for Real-Time Booking Platforms. – MIT Press, 2024. – 276 p.
25. European Data Protection Board. Guidelines on Personal Data Processing in Online Platforms. – Brussels, 2024.

26. Journal of Information Systems Engineering. Trends in Reservation System Automation. – 2025. – Vol. 12(2), P. 5–29.
27. Що таке On-line сервіси? [Електронний ресурс] – Режим доступу: https://avada-media.ua/services/on-line-servisy/.
28. Онлайн сервіси і їх види [Електронний ресурс] – Режим доступу: http://book-science.ru/applied/it/onlajn-servisy-i-ih-vidy.html.
29. За і проти: Навіщо ресторанам сервіси бронювання столиків [Електронний ресурс] – Режим доступу: https://habr.com/ru/company/jowi/blog/369115/.
30. Клієнтська оптимізація і етапи розробки [Електронний ресурс] – режим доступа: http://tods-blog.com.ua/ezarabotok/klientskaya-chast/.
31. Купер А.Б. Про інтерфейс. Основи проектування взаємодії. – Лондон, 2012. – 365 с.
32. Раскин Д.М. Інтерфейс: нові напрямки в проектуванні комп'ютерних систем. – Париж, 2009. – 144 с.
33. Поради до розробки архітектури сайту [Електронний ресурс] – Режим доступу: https://semantica.in/blog/polnoe-rukovodstvo-po-planirovaniyu- arkhitektury-sajta-15-sovetov-dlya-maksimalnogo-seo.html#toc1Eclipse.
34. React [Електронний ресурс] – режим доступа: https://ru.wikipedia.org/wiki/React.
35. React Native [Електронний ресурс] – режим доступа: https://en.wikipedia.org/wiki/React_Native.
36. Переваги використання React Js [Електронний ресурс] – режим доступа: https://xbsoftware.ru/blog/pochemu-stoit-ispolzovat-react-js-razrabotke- prilozhenij/
37. Документація Python [Онлайн]. Доступно: https://docs.python.org/3/

38. Документація Django. [Онлайн]. Доступно: https://docs.djangoproject.com/en/3.2/

39. Документація TextBlob. [Онлайн]. Доступно: https://textblob.readthedocs.io/en/dev/

40. HTML. Доступно: https://developer.mozilla.org/en-US/docs/Web/HTML

41. CSS:. [Онлайн]. Доступно: https://developer.mozilla.org/en-US/docs/Web/CSS

42. Deep Learning Book by Ian Goodfellow, Yoshua Bengio, and Aaron Courville [Онлайн]. Доступно: http://www.deeplearningbook.org/

43. Downey A. Think Python: How to Think Like a Computer Scientist, 2nd edition. Needham: Green Tea Press, 2015. 36 с

44. James A., Edward R. Talking Nets An Oral History of Neural Networks. Cambridge: The MIT Press, 2020. 10 с

45. Christian B. The Alignment Problem: Machine Learning and Human Values. New York: W. W. Norton & Company, 2020. 25 с.

46. Bradski G., Kaehler A. Learning OpenCV: Computer Vision with the OpenCV Library. Sebastopol: O'Reilly Media, Inc, 2011. 53 с.

47. Raschka S., Mirjalili V. Python Machine Learning. Livery Place: Packt Publishing Ltd, 2019. 296 с.

48. Goldberg, Y. (2017). Neural Network Methods for Natural Language Processing. Morgan & Claypool Publishers.

49. Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing. Pearson. https://colah.github.io/posts/2015-08-Understanding-LSTMs/

50. Тези доповідей: Розробка інформаційної технології для системи бронювання місць в закладах громадського харчування. VI Всеукраїнська науково-практична конференція «Комп’ютерні технології обробки даних» (КТОД–2025),м.Вінниця, ДонНУ імені Василя Стуса, 5 грудня 2025.

Маркуш Юрій Васильович
(Прізвище, ім’я, по-батькові)

Інформаційних і прикладних технологій
(Факультет)

122 Комп’ютерні науки
(Шифр і назва спеціальності)

Комп’ютерні технології обробки даних (Data Science)
(Освітня програма)

ДЕКЛАРАЦІЯ АКАДЕМІЧНОЇ ДОБРОЧЕСНОСТІ

Усвідомлюючи свою відповідальність за надання неправдивої інформації, стверджую, що подана кваліфікаційна (магістерська) робота на тему: «Інформаційна технологія бронювання місць в закладах громадського харчування» є написаною мною особисто.

Одночасно заявляю, що ця робота:

- не передавалась іншим особам і подається до захисту вперше;
- не порушує авторських та суміжних прав, закріплених статтями 21-25 Закону України «Про авторське право та суміжні права»;
- не отримувалась іншими особами, а також дані та інформація не отримувалась у недозволений спосіб.

Я усвідомлюю, що у разі порушення цього порядку моя кваліфікаційна робота буде відхилена без права її захисту, або під час захисту за неї буде поставлена оцінка «незадовільно».

_____ _____

(дата) (підпис здобувача освіти)

ДОДАТОК

Тези доповідей: Розробка інформаційної технології для системи бронювання місць в закладах громадського харчування. VI Всеукраїнська науково-практична конференція «Комп'ютерні технології обробки даних» (КТОД–2025),м.Вінниця, ДонНУ імені Василя Стуса, 5 грудня 2025.

УДК 004

Маркуш Ю.В.

РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ДЛЯ СИСТЕМИ БРОНЮВАННЯ МІСЦЬ В ЗАКЛАДАХ ГРОМАДСЬКОГО ХАРЧУВАННЯ

У сучасному світі процеси цифровізації охоплюють практично всі сфери діяльності людини. Те, що ще десять-п'ятнадцять років тому виконувалося вручну, вимагало паперових журналів, тривалих телефонних розмов або особистої присутності, сьогодні поступово переходить у формат зручних електронних сервісів. Інформаційні технології стали не просто допоміжним інструментом, а необхідною умовою швидкої комунікації, ефективної організації роботи та якісного надання послуг.

Одним із напрямів, де потреба в автоматизації виявилася особливо гострою, є процес бронювання місць в закладах громадського харчування, тому що на сучасному етапі— люди прагнуть отримати можливість забронювати потрібний ресурс швидко, у зручний для себе час і без додаткових клопотів. Сучасне життя надто динамічне, тому очікування в телефонній черзі або необхідність їхати особисто вже виглядають дуже застарілими та незручними.

Традиційні способи бронювання, які десятиліттями здавалися нормою, — телефонний дзвінок, запис у журналі, спілкування з адміністратором — сьогодні не можуть забезпечити ані достатньої точності, ані оперативності. Вони створюють ризики людських помилок, дублювання резервувань та непорозумінь. Крім того, вони вимагають додаткових витрат часу як від клієнтів, так і від працівників установи. Саме тому розробка інформаційної системи бронювання місць у закладах громадського харчування стала логічним кроком еволюції таких процесів. Сучасна ІС дозволяє автоматизувати більшість дій, зробити їх прозорими, передбачуваними та зручними.

Автоматизована система дає змогу користувачам самостійно обирати час, місце, тип послуги, бачити актуальну доступність та одразу отримувати підтвердження бронювання. Для організацій це означає зменшення навантаження на персонал, оптимізацію робочих процесів, точніший облік зайнятості ресурсів та можливість аналізувати попит на послуги.

Метою даної роботи є детальний аналіз особливостей створення інформаційної системи бронювання місць, визначення її ключових структурних компонентів, основних функціональних можливостей, а також виявлення переваг, які отримують організації та користувачі після впровадження подібної системи. Розуміння цих аспектів дозволяє побудувати технічно грамотну, зручну та ефективну ІС, що відповідатиме вимогам сучасного цифрового середовища.