МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет комп'ютерних наук і технологій
Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ
Завідувач кафедри
Аліна САВЧЕНКО
"_____"_____2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ «БАКАЛАВР»
ЗА СПЕЦІАЛЬНІСТЮ 122 «КОМП'ЮТЕРНІ НАУКИ»

Тема: «WEB-сервіс надання послуг масажу»

Виконавець: Брагінець Андрій Олександрович

Керівник: к.т.н., доцент Климова Асія Сабирівна

Нормоконтролер: Олександр ШЕВЧЕНКО
(підпис)

Київ 2023

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук і технологій

Кафедра комп'ютерних інформаційних технологій

Освітній ступінь: «Бакалавр»

Галузь знань, спеціальність, освітньо-професійна програма:

12 «Інформаційні технології», 122 «Комп'ютерні науки», «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Аліна САВЧЕНКО

"_____"_____2023 р.

ЗАВДАННЯ

на виконання кваліфікаційної роботи

Брагінця Андрія Олександровича

(прізвище, ім'я, по батькові випускника в родовому відмінку)

1. **Тема кваліфікаційної роботи:** «WEB-сервіс надання послуг масажу» затверджена наказом ректора №623/ст від 01.05.2023 р.
2. **Термін виконання роботи**: 15.05.2023 – 25.06.2023
3. **Вихідні дані до роботи**: мова розмітки гіпертексту HTML, каскадні таблиці стилів CSS, мови програмування JavaScript та Java з її бібліотекою розробки Spring, середовище розробки VS Code, реляційна система керування базами даних PostgreSQL.
4. **Зміст пояснювальної записки**: аналіз предметної області та існуючих підходів для створення веб-системи, засоби розробки веб-сервісу, розробка веб-сервісу.
5. **Перелік обов'язкового графічного (ілюстративного) матеріалу:** принцип роботи веб-сервісів, архітектура веб-застосунку, логічна модель бази даних, дизайн форми реєстрації користувача, дизайн форми авторизації користувача, дизайн форми бронювання масажних сесій.

6. Календарний план-графік

№ пор.	Завдання	Термін виконання	Відмітка про виконання
1	Аналіз та дослідження предметної області. Огляд та аналіз доступних технологій та фреймворків.	15.05.2023-19.05.2023	Виконано
2	Розробка деталізованого змісту розділів кваліфікаційної роботи. Написання вступу.	22.05.2023-23.05.2023	Виконано
3	Розробка структури веб-сервісу. Написання 1 розділу кваліфікаційної роботи.	24.05.2023-26.05.2023	Виконано
4	Створення веб-сервісу з використанням досліджених технологій. Написання 2 розділу кваліфікаційної роботи.	29.05.2023-02.06.2023	Виконано
5	Написання 3 розділу кваліфікаційної роботи. Усунення недоліків та завершення створення пояснювальної записки кваліфікаційної роботи.	05.06.2023-09.06.2023	Виконано
6	Оформлення та друк пояснювальної записки кваліфікаційної роботи.	12.06.2023-13.06.2023	Виконано
7	Підготовка презентації та доповіді для виступу.	14.06.2023-16.06.2023	Виконано
8	Захист кваліфікаційної роботи.	19.06.2023-23.06.2023	Виконано

7. Дата видачі завдання: 15.05.2023 р.

Керівник кваліфікаційної роботи _____ Асія КЛИМОВА
(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Андрій БРАГІНЕЦЬ
(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Кваліфікаційна робота «WEB-сервіс надання послуг масажу» містить 59 сторінок, 27 рисунків, 1 таблицю, 16 використаних джерел.

Об’єкт дослідження: веб-сервіс надання послуг масажу

Предмет дослідження: розробка веб-сайту та його функціоналу.

Мета кваліфікаційної роботи: розробка веб-сервісу з використанням різних технологій для надання послуг масажу; аналіз предметної області, визначення вимог, огляд доступних технологій.

Метод дослідження: аналіз літературних джерел, дослідження та аналіз уже існуючих систем, теоретичне ознайомлення з існуючими технологіями, які використовують при розробці веб-сервісів з використанням баз даних, мови програмування JavaScript та Java, біблеотека Spring, реляційна система управління базами даних PostgreSQL.

Результат проекту: розроблений веб-сервіс, який надає функціонал для сфери послуг.

ВЕБ-СЕРВІС, МАСАЖ, HTML, CSS, JAVASCRIPT, JAVA, SPRING FRAMEWORK.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СКОРОЧЕНЬ, ТЕРМІНІВ

Framework (Software framework) – це набір зумовлених інструментів, бібліотек, шаблонів та правил, що полегшує процес розробки додатків

Фрейм (Frame) – це структура даних, що використовується для організації інформації в комп'ютерних програмних системах

RESTful (Representational State Transfer architecture) – це архітектурний стиль, який використовується для розробки мережевих програм, в основному в розробці веб-служб і інтерфейсів прикладного програмування.

API (Application Programming Interface) – це набір правил і протоколів, які дозволяють різним програмам програмного забезпечення спілкуватися та взаємодіяти один з одним.

DAO (Data Access Object) – це шаблон проектування, який використовується для абстрагування та інкапсуляції взаємодії з системою зберігання даних, такою як база даних або файлова система.

HTTP (Hypertext Transfer Protocol) – це прикладний протокол, який служить основою для спілкування у Всесвітній павутині.

JVM (Java Virtual Machine) – це важливий компонент платформи Java і служить середовищем виконання програм Java.

ВСТУП

Сьогоднішній світ переживає швидкий розвиток інформаційних технологій, що відкриває безліч можливостей для спілкування, отримання інформації та надання різноманітних послуг через Інтернет. Інтернет став справжнім простором, де люди можуть знаходити і споживати різноманітні послуги, незалежно від місцезнаходження і часу.

У сфері послуг, інтернет забезпечує появу веб-сервісів – спеціалізованих онлайн-платформ, які надають доступ до різних послуг через Інтернет. Веб-сервіси стали невід'ємною складовою частиною життя сучасного суспільства, оскільки вони пропонують зручний та ефективний спосіб отримання різноманітних послуг, які раніше були доступні тільки в офлайн-режимі.

Однією зі сфер, де веб-сервіси набули особливого значення, є сфера надання послуг масажу. Масаж – це давнє мистецтво, яке сприяє поліпшенню фізичного та психологічного самопочуття людини. Завдяки розширенню Інтернету, клієнти тепер можуть зручно та швидко знайти та забронювати сеанс масажу через спеціалізований веб-сервіс.

Інтернет і веб-сервіси зробили революцію в тому, як ми взаємодіємо з навколишнім світом. Вони зробили спілкування швидшим та ефективнішим, а також відкрили нові можливості для комерції, освіти та розваг. Оскільки Інтернет продовжує розвиватися, ми можемо очікувати на появу ще більш інноваційних додатків і послуг, які змінять наш спосіб життя і роботи.

РОЗДІЛ 1
АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ІСНУЮЧИХ ПІДХОДІВ ДЛЯ СТВОРЕННЯ ВЕБ-СИСТЕМИ

1.1. Що таке веб-система?

Веб-система – це програмне забезпечення, яке працює через мережу Інтернет та надає користувачам доступ до різноманітних функцій та сервісів через веб-браузер. Веб-система може складатися з веб-сторінок, додатків та інших компонентів, що забезпечують функціональність взаємодії з користувачем.

Веб-системи можуть бути розроблені для різних цілей, таких як збір інформації, автоматизація бізнес-процесів, надання доступу до ресурсів, забезпечення взаємодії зі сторонніми сервісами, тощо.

Веб-системи можуть бути розгорнуті на веб-серверах, які забезпечують доступ до ресурсів через інтернет. Це дозволяє користувачам з будь-якої точки світу звертатися до системи та взаємодіяти з нею. Веб-системи можуть бути розроблені для використання в різних галузях, таких як фінанси, освіта, медицина, туризм, електронна комерція та багато інших.

1.2. Роль веб-систем у сфері послуг

Веб-системи мають важливу роль у сфері послуг, так як вони забезпечують зручний доступ до інформації про послуги, а також дають можливість користувачам замовляти та оплачувати послуги онлайн. Завдяки веб-системам клієнти можуть швидко та зручно знаходити інформацію про послуги, порівнювати ціни та відгуки інших користувачів, а також здійснювати покупки в будь-який зручний час.

Крім того, веб-системи також забезпечують можливість зберігання та обробки даних про клієнтів для подальшого аналізу та роботи з ними. Веб-системи також можуть забезпечувати можливість розсилки спеціальних пропозицій та акцій для постійних клієнтів.

В цілому, веб-системи для послуг можуть бути корисними як для клієнтів, так і для масажистів, допомагаючи їм ефективно взаємодіяти один з одним та забезпечуючи швидкий і зручний доступ до послуг.

1.3. Технології для створення веб-систем

Веб-система складається з двох основних складових: клієнтської частини (frontend) та серверної частини (backend).

Клієнтська частина відповідає за взаємодію з користувачем та відображення інформації. На цій стороні зазвичай використовуються мови програмування, які підтримуються браузерами, такі як HTML, CSS та JavaScript. Клієнтська частина забезпечує інтерактивність та користувацький досвід, який відображається в браузері користувача.

Серверна частина відповідає за обробку запитів, які надійшли від клієнтської частини та виконання необхідних дій. На цій стороні використовуються різні мови програмування та технології, такі як PHP, Ruby, Python, Node.js, Java, .NET та інші. Серверна частина відповідає за збереження та обробку даних, що збираються на клієнтській стороні, та надання користувачам доступу до різноманітних послуг та інформації.

Згадуючи про WEB-системи, не можна не згадати про основні стандарти веб-сервісів, такі як:

- XML;
- WSDL;
- UDDI;
- SOAP;
- REST.

1.3.1. XML (Extensible Markup Language)

XML (Extensible Markup Language) – це мова розмітки, що використовується для опису даних у вигляді структурованого тексту. XML є універсальним форматом даних, який може бути використаний в різних додатках та платформах, оскільки він не залежить від будь-якої конкретної програми чи мови програмування.

У форматі XML дані представлені у вигляді дерева, що складається з вузлів (елементів) та їх атрибутів. Кожен елемент може мати декілька дочірніх елементів, які у свою чергу можуть мати свої власні дочірні елементи та атрибути. В кінцевому результаті, дані у форматі XML можуть бути зручно і структуровано представлені та зчитані різними програмами та пристроями.

XML використовується для зберігання та обміну даними між різними програмами та платформами, таких як веб-браузери, бази даних, мови програмування та інші. XML також використовується для створення та розробки веб-сторінок та веб-служб, які можуть бути доступні з різних пристроїв та платформ.

Для обробки даних у форматі XML використовуються спеціальні програми та бібліотеки, які здатні читати та записувати дані у форматі XML, а також здійснювати обробку та аналіз цих даних.

1.3.2. WSDL (Web Services Description Language)

WSDL (Web Services Description Language) – це мова опису веб-сервісів, яка використовується для опису функцій та параметрів, які доступні через веб-сервіс. WSDL забезпечує формальний спосіб опису інтерфейсу веб-сервісу та дозволяє клієнтам з легкістю знаходити та використовувати функції веб-сервісу.

WSDL використовується для опису того, як клієнти можуть звертатися до веб-сервісу та які параметри вони можуть передавати. WSDL описує структуру повідомлень, що обмінюються між клієнтом та веб-сервісом, типи даних, які використовуються для передачі параметрів та результатів, а також адресу веб-сервісу, до якого клієнти можуть звертатися для отримання послуг.

WSDL використовується в SOAP-веб-сервісах, які передають дані за допомогою SOAP-протоколу. WSDL може бути написаний у XML-форматі, тому його можна легко інтерпретувати програмами та іншими технологіями, що підтримують XML.

Щоб скористатися веб-сервісом, клієнт може завантажити WSDL-файл та використовувати його для створення запитів до веб-сервісу. WSDL дозволяє клієнтам генерувати клієнтський код, який використовується для взаємодії з веб-сервісом, що спрощує процес розробки веб-додатків та забезпечує більшу надійність та цілісність взаємодії між клієнтами та веб-сервісом.

1.3.3. UDDI (Universal Description Discovery, and Integration)

UDDI (Universal Description, Discovery, and Integration) – це протокол для пошуку та вибору веб-сервісів в мережі Інтернет. UDDI дозволяє компаніям публікувати описи своїх веб-сервісів, що дозволяє іншим компаніям знаходити та використовувати ці послуги.

UDDI складається з трьох основних компонентів:

1. UDDI реєстр – це централізована база даних, в якій компанії можуть публікувати описи своїх веб-сервісів. Кожен запис у реєстрі містить інформацію про веб-сервіс, включаючи адресу, тип, параметри та інші деталі.
2. UDDI сховище – це компонент, який зберігає копії записів реєстру та дозволяє шукати веб-сервіси на основі різних параметрів, таких як назва, ключові слова та інші.
3. UDDI клієнт – це програмне забезпечення, яке дозволяє користувачам шукати та вибирати веб-сервіси на основі їх описів. Клієнт може бути інтегрований з іншими програмними продуктами, такими як розробник веб-сервісів, що дозволяє знаходити та використовувати доступні веб-сервіси.

UDDI забезпечує стандартизований спосіб опису та пошуку веб-сервісів, що знижує складність процесу їх розробки та використання. До переваг UDDI відно-

ситься можливість знаходження веб-сервісів з різних джерел та використання їх у різних програмних продуктах та сервісах. У той же час, на практиці використання UDDI було менш значущим порівняно з іншими протоколами, такими як SOAP та WSDL, в розробці веб-сервісів. Основна причина полягає в тому, що UDDI має об-межений функціонал, інтерфейс та зручність використання порівняно з іншими про-токолами.

У цілому, UDDI не є обов'язковим компонентом для розробки веб-сервісів, але він може бути корисним для організації пошуку та використання веб-сервісів у великих корпоративних системах. Також UDDI може бути використаний як додатковий інструмент для публікації та пошуку веб-сервісів у мережі Інтернет.

1.3.4. SOAP (Simple Object Access Protocol)

SOAP (або Simple Object Access Protocol) – це протокол для обміну повідомленнями у веб-сервісах. SOAP був створений з метою стандартизації способу взаємодії між різними платформами та мовами програмування, що дозволяє веб-сервісам працювати разом.

SOAP базується на XML та використовується для передачі повідомлень між веб-серверами та клієнтами. Повідомлення SOAP складається з трьох частин: заголовку, тіла та опційного блоку даних. Заголовок містить додаткову інформацію про повідомлення, наприклад, про версію SOAP або тип повідомлення. Тіло містить фактичну інформацію, яка передається між клієнтом та сервером. Опційний блок даних містить додаткові параметри, які можуть бути використані для визначення різноманітних аспектів повідомлення.

SOAP може використовуватися з різними протоколами передачі даних, такими як HTTP, SMTP, TCP і т.д. SOAP є дуже гнучким і стійким до помилок протоколом, оскільки містить механізми перевірки цілісності та надійності даних.

SOAP підтримує використання різних форматів даних, таких як XML, JSON та інші, що робить його універсальним для використання у різних середовищах та з різними технологіями. SOAP також підтримує використання різних механізмів автен-

тифікації та шифрування, що робить його безпечним для передачі конфіденційної інформації.

У загальному, SOAP є одним з ключових стандартів у веб-сервісах, і використовується для передачі повідомлень у різноманітних інформаційних системах.

Тобто, перед тим, як створити веб-сервіс, розробники створюють його опис у форматі документа WSDL, який містить інформацію про те, де знаходиться сервіс в Інтернеті і які функції він може виконувати. Цю інформацію можна додати до реєстру UDDI, щоб користувачі могли шукати та знаходити необхідні їм сервіси. Розробники клієнтських веб-сервісів використовують інструкції з WSDL, спираючись на інформацію з реєстру UDDI, щоб створювати SOAP повідомлення для обміну даними з сервісом через протокол HTTP.

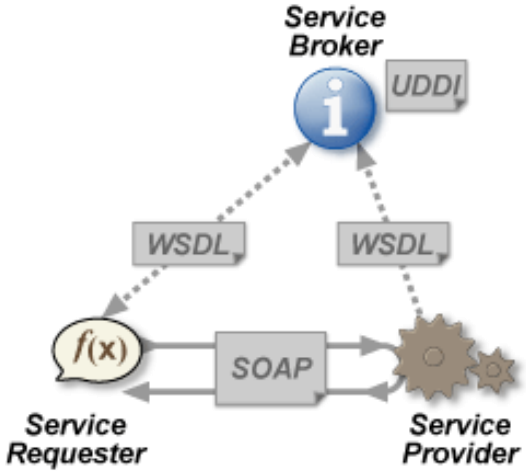


Рис. 1.1. Принцип роботи вебслужб

1.3.5. REST (або Representational State Transfer)

REST (або Representational State Transfer) – це архітектурний стиль для розробки інтернет-програм, який забезпечує доступ до веб-ресурсів. REST базується на протоколі HTTP і використовується для створення веб-сервісів.

У REST існує низка основних принципів, які дозволяють розробникам створювати надійні та ефективні веб-сервіси. Основні принципи REST наведені нижче в таблиці 1.1.

Таблиця 1.1.

Основні принципи REST

Принцип	Опис	Результат
Клієнт-серверна архітектура	Система повинна бути розділена на дві частини: клієнт та сервер.	Забезпечує зменшення залежності між клієн-том та сервером і дозволяє розробникам розробляти та вдосконалювати кожну частину системи окремо.
Безстійність	Кожен запит від клієнта повинен містити всю необхідну інформацію, щоб сервер міг зрозуміти і відповісти на запит. Сервер не повинен зберігати жодних інформаційних даних про клієнта між запитами.	Забезпечує більшу масштабованість і зменшує навантаження на сервер.
Кешування	Сервер повинен надавати можливість клієнтам кешувати отримані відповіді на запити, щоб зменшити кількість запитів, що потрібно робити на сервер.	Зменшує час відпові-ді та зменшує наванта-ження на сервер.

Продовження таблиці 1.1.

Стандартні методи HTTP	REST використовує стандартні методи HTTP, такі як GET, POST, PUT та DELETE, для доступу до ресурсів на сервері.	Забезпечує однаковий підхід до доступу до ресурсів і дозволяє розробникам створювати універсальні та незалежні від технологій веб-сервіси.
Представлення ресурсів	REST використовує представлення ресурсів відповідно до формату, який вказує клієнт. Зазвичай використовуються формати JSON або XML.	Забезпечує більшу гнучкість та можливість адаптуватися до різних клієнтських додатків.
Ідентифікатор ресурсів	Кожен ресурс на сервері повинен мати унікальний ідентифікатор, який дозволяє клієнту доступатися до цього ресурсу. Зазвичай ідентифікатори використовуються в URL-адресах.	Забезпечує більш простий доступ до конкретних ресурсів.

REST є одним з найбільш популярних підходів до розробки веб-сервісів через його гнучкість, масштабованість та простоту використання. Він використовується для створення API для різноманітних систем, таких як соціальні мережі, онлайн-магазини, мобільні додатки та багато іншого.

1.4. Переваги та недоліки веб-систем

Веб-системи є досить поширеними інструментами для надання послуг в Інтернеті. Їх переваги та недоліки залежать від багатьох факторів, таких як зручність та доступність, ефективність та швидкість, надійність та масштабованість, приватність та безпека.

1.4.1. Зручність та доступність

Однією з головних переваг веб-систем є їх зручність та доступність для користувачів. Веб-системи зазвичай прості у використанні та не потребують спеціальних знань або навичок, що робить їх зручними для широкої аудиторії. Крім того, веб-системи дозволяють користувачам отримувати доступ до необхідної інформації та послуг з будь-якого місця та в будь-який час.

Особливо важливою перевагою зручності та доступності веб-систем є їх можливість працювати на різних платформах та пристроях. Більшість веб-систем можуть бути використані на різних операційних системах та пристроях, таких як комп'ютери, планшети та смартфони. Це дозволяє користувачам отримувати доступ до веб-сервісів з будь-якого пристрою з Інтернет-підключенням.

Ще однією важливою перевагою зручності та доступності є можливість отримання інформації про послуги та продукти з будь-якого місця та в будь-який час. Користувачі можуть отримувати необхідну інформацію про товари та послуги веб-сервісу, а також здійснювати замовлення, не виходячи з дому або офісу. Це забезпечує зручність та швидкість процесу, що є дуже важливим у сучасному світі, де люди дедалі більше зайняті та мають менше часу на відвідування магазинів або офісів.

Крім того, веб-системи також дозволяють знизити витрати на оренду та утримання офісного простору, оскільки деякі бізнеси можуть працювати повністю в онлайн режимі, не потребуючи фізичного присутності клієнтів або співробітників у офісі. Це дозволяє збільшити ефективність роботи, зменшити витрати на зарплати та інші витрати на утримання фізичного простору.

Загалом, зручність та доступність веб-системи надають користувачам широкий доступ до інформації та послуг з будь-якого місця та в будь-який час. Вони дозволяють економити час та гроші, а також підвищувати ефективність роботи та знижувати витрати на утримання фізичного простору. Проте, варто зазначити, що деякі користувачі можуть відчувати незручність у використанні веб-систем та бажати спілкуватися зі співробітниками компанії особисто, що є однією з недоліків веб-

систем. Також до недоліків можна віднести те, що веб-системи можуть бути недоступні у випадку відсутності Інтернету або проблем зі з'єднанням.

1.4.2. Ефективність та швидкість

Ефективність та швидкість є важливими аспектами веб-систем, оскільки вони впливають на задоволеність користувачів та продуктивність бізнесу. Швидкість завантаження веб-сторінок та відповідь сервера на запити користувачів є ключовими факторами, які впливають на ефективність веб-системи.

Для забезпечення ефективної роботи веб-систем, необхідно мати оптимізовану архітектуру та кодування, яке дозволяє максимально використовувати ресурси сервера та мережі. Також важливим фактором є використання швидкісного та надійного хостингу, який забезпечує максимальну доступність та швидкість роботи системи.

При проектуванні веб-системи необхідно пам'ятати, що час відповіді веб-сторінок впливає на користувацький досвід. Більшість користувачів очікують, що сторінки будуть завантажуватися швидко та без перерв. Якщо час відповіді веб-сторінок буде надто великим, то це може призвести до того, що користувачі відмовляться від використання системи та перейдуть до конкурентів.

Також ефективність веб-системи визначається її функціональністю та можливостями. Наприклад, якщо веб-система пропонує послуги масажу, то вона повинна мати широкий вибір різних масажів, можливість запису на прийом, зручну систему оплати та інші функціональні можливості. Це дозволить забезпечити високу ефективність роботи веб-системи та задовольнення користувачів.

Однак, варто зазначити, що неправильно розроблена архітектура веб-системи може призвести до її низької ефективності та повільної роботи. Тому розробники повинні звертати увагу на оптимізацію даних та запитів, використовувати кешування та інші методи покращення продуктивності веб-системи.

Для забезпечення швидкої роботи веб-системи також важливо правильно налаштувати її серверне обладнання та програмне забезпечення. Наприклад, можна використовувати CDN (Content Delivery Network), який дозволяє зменшити час

завантаження веб-сторінок за рахунок розподілення контенту по різних серверах у різних країнах.

До інших методів покращення ефективності веб-системи можна віднести використання компресії даних, мінімізацію HTTP-запитів, використання кешування на клієнтській та серверній стороні та інші. Такі заходи допомагають зменшити час завантаження сторінок та зробити веб-систему більш продуктивною.

1.4.3. Надійність та масштабованість

Надійність та масштабованість – це ще дві важливі характеристики веб-систем. Надійність означає, що веб-система повинна бути стійкою до можливих помилок та збоїв, які можуть виникнути в процесі роботи. Для забезпечення надійності веб-системи важливо розробити правильну стратегію збереження даних, резервне копіювання, а також налагодити систему моніторингу, яка допомагає оперативно виявляти та усувати можливі проблеми.

Щодо масштабованості, то вона означає, що веб-система повинна бути здатною працювати ефективно незалежно від кількості користувачів та обсягу оброблюваних даних. Для забезпечення масштабованості веб-системи, розробники повинні використовувати горизонтальне та вертикальне масштабування. Горизонтальне масштабування передбачає збільшення кількості серверів, які оброблюють запити веб-системи, в той час, як вертикальне масштабування передбачає збільшення ресурсів (CPU, RAM, HDD) одного сервера.

Для досягнення максимальної надійності та масштабованості веб-системи також важливо використовувати розподілені системи збереження даних та балансування навантаження, що дозволяє рівномірно розподіляти запити між серверами та забезпечити високу доступність веб-системи.

1.4.4. Привітність та безпека

Приватність та безпека є одними з найважливіших аспектів веб-систем, особливо в тих, які збирають та обробляють особисту інформацію користувачів. Відповідальність за збереження цієї інформації лежить на розробниках веб-систем та власниках бізнесу, які використовують ці системи.

Однією з переваг веб-систем є можливість шифрування даних, що зменшує ризик несанкціонованого доступу до них. Більшість веб-систем мають вбудовані заходи безпеки, які допомагають у запобіганні атакам на сервери та інші форми вторгнень. Проте, деякі веб-системи можуть мати недостатньо ефективні заходи безпеки, що може призвести до витоку конфіденційної інформації.

Іншим аспектом є збір та використання особистої інформації користувачів, що може викликати питання щодо приватності. Деякі користувачі можуть бути обурені, якщо вони відчувають, що їх особисті дані використовуються без їхньої згоди. Тому розробники веб-систем повинні забезпечити можливість вибору користувачами, що-до збору та використання їхніх даних.

Окрім цього, веб-системи можуть стати об'єктом кібератак та зломів. Це може призвести до втрати конфіденційної інформації, виходу систем з ладу та інших проблем. Тому важливо, щоб веб-системи мали високий рівень безпеки та забезпечували захист від кібератак.

Незважаючи на різні заходи забезпечення безпеки, що можна вжити, веб-системи не можуть бути повністю захищені від зловмисних дій. Ризик атак на веб-системи зростає залежно від того, які дані вони зберігають, хто має до них доступ і як вони передаються між користувачами та серверами.

Однією з основних проблем веб-систем є можливість атак з використанням збірки належностей, де зловмисник вводить шкідливий код в поля введення на веб-сторінках, що дозволяє їм отримати доступ до баз даних або зламати систему. Інші типи атак включають перехоплення даних, переповнення буферу, злам аутентифікації та авторизації, а також атаки на програмне забезпечення.

Тому для забезпечення безпеки веб-систем важливо дотримуватись кращих практик, включаючи захист від збірки належностей, використання сильних паролів та механізмів аутентифікації, регулярне оновлення програмного забезпечення та використання захисту від DDos-атак.

Також, для забезпечення приватності користувачів веб-систем, необхідно використовувати шифрування для передачі конфіденційних даних між серверами та користувачами, дотримуватись нормативно-правових вимог щодо захисту персональних даних та забезпечення конфіденційності відповідно до вимог GDPR, HIPAA, або інших відповідних стандартів залежно від того, в якій країні веб-система діє.

1.5. Вимоги до веб-систем

При створенні веб-системи для надання послуг масажу необхідно враховувати різноманітні вимоги, які впливають на якість та ефективність її функціонування. Основні вимоги можуть бути розділені на такі групи:

Функціональність:

- Реєстрація та авторизація користувачів – система повинна мати можливість реєстрації нових користувачів та авторизації вже зареєстрованих з метою забезпечення доступу до відповідних функцій.
- Управління замовленнями – система повинна мати можливість приймати та обробляти замовлення на масаж, зберігати дані про клієнтів та їхній історії звернень, а також відправляти повідомлення про підтвердження замовлення.
- Адміністрування системи – система повинна мати можливість налаштування та керування різними параметрами, такими як: доступність типів масажу, розклад масажних сеансів тощо.
- Звітність – система повинна мати можливість зберігати дані про клієнтів та їхній історії звернень.

Ергономіка:

- Простота та зручність у використанні – система повинна бути зрозумілою та зручною у використанні для користувачів, що включає в себе легку навігацію по сайту.
- Адаптивність та респонсивність – система повинна бути адаптивною до різних розмірів екранів та пристроїв, що використовуються користувачами, а також бути респонсивною, тобто швидко реагувати на дії користувачів.

Безпека:

- Захист від несанкціонованого доступу – система повинна мати надійну систему авторизації та аутентифікації, що забезпечує захист від несанкціонованого доступу до даних користувачів та інформації про їхні замовлення.
- Захист від атак – система повинна мати надійний захист від різноманітних атак, таких як SQL-ін'єкції, кросс-сайтові скрипти, атаки типу "брутфорс" тощо.
- Захист даних – система повинна забезпечувати надійний захист даних користувачів, що включає в себе шифрування даних, забезпечення їхньої цілісності та конфіденційності.

Сумісність:

- Сумісність з різними браузерами та операційними системами – система повинна бути сумісною з різними браузерами та операційними системами, що використовуються користувачами.
- Сумісність зі стандартами та протоколами – система повинна відповідати різним стандартам та протоколам, що використовуються у веб-розробці, таким як HTML, CSS, JavaScript, HTTP, HTTPS тощо.

1.6. Що таке масаж та його користь

Масаж є процедурою, під час якої використовуються різні техніки терапевтичного впливу на м'язову та кісткову систему людини з метою поліпшення їхнього функціонування та забезпечення загального покращення самопочуття. Ця

процедура має безліч корисних властивостей та може використовуватися як для попередження захворювань, так і для лікування різноманітних хвороб.

Масаж може мати багато корисних впливів на організм людини. Зокрема, він може знімати напругу та біль в м'язах, поліпшувати кровообіг та лімфоток, знижувати рівень стресу та тривожності, підвищувати імунітет та загальну витривалість організму. Крім того, масаж може сприяти зниженню рівня шкідливого холестерину, покращувати рухливість суглобів та зменшувати ймовірність розвитку хвороб серцевосудинної системи.

У залежності від цілей та потреб клієнта, існує безліч різних видів масажу, кожен з яких має свої особливості та корисні властивості.

1.7. Типи масажу та його особливості

Вибір певного виду масажу залежить від мети його застосування, фізіологічних особливостей організму, наявності певних захворювань, а також від персональних вподобань клієнта.

Основні типи масажу:

1. Класичний масаж. Це найпоширеніший тип масажу, який включає в себе різні техніки відшарування м'язових напружень, покращення кровообігу та лікування болів у м'язах і суглобах.
2. Спортивний масаж. Цей тип масажу розроблений спеціально для спортсменів та людей, які займаються активним спортом. Він має на меті збільшення рухливості та еластичності м'язів, покращення кровообігу та прискорення відновлювальних процесів в тілі.
3. Точковий масаж. Цей тип масажу спрямований на роботу з конкретними точками на тілі, де зосереджуються м'язові напруження та болі. Точковий масаж може допомогти покращити кровообіг, зняти біль та покращити рухливість.
4. Тайський масаж. Цей вид масажу виконується за допомогою рук, ліктів та ніг, та включає в себе елементи йоги та акупресури. Він має на меті зняття

напруги та стресу, покращення кровообігу та зняття болів у м'язах та суглобах.

1.8. Висновки до розділу

Отже, у даному розділі було проаналізовано поняття веб-системи та її роль у сфері послуг, описано технології для створення веб-систем та їх переваги та недоліки. Було розглянуто вимоги до веб-систем, а також детально описано поняття масажу, його користь та основні типи масажу.

З метою розробки веб-сервісу надання послуг масажу було виявлено, що такий сервіс повинен мати простий та зручний інтерфейс для користувачів, швидку та ефективну роботу, надійний та масштабований механізм, а також високий рівень безпеки та приватності.

Підсумовуючи, можемо сказати, що застосування веб-системи для надання послуг масажу є доцільним та має багато переваг, таких як зручність та доступність, ефективність та швидкість, надійність та масштабованість, а також приватність та безпека. Продовження дослідження у цьому напрямку може допомогти у поліпшенні якості надання послуг масажу та покращенні задоволення клієнтів.

РОЗДІЛ 2
ЗАСОБИ РОЗРОБКИ ВЕБ-СЕРВІСУ

2.1. Мова розмітки гіпертексту HTML

Мови програмування є невід'ємними частинами розробки програмного забезпечення. Вони вже давно спростили життя людей в усіх сферах, в тому числі і в сфері послуг. Так як мов програмування існує безліч, то перед початком реалізації потрібно ознайомитися з доступними засобами розробки і вибрати ті, які найбільше нам підходять.

HTML (Hypertext Markup Language) є стандартною мовою розмітки гіпертексту, яка використовується для створення та представлення веб-сторінок. Вона визначає структуру та семантику контенту на веб-сторінці за допомогою різних елементів та тегів.

Роль HTML у розробці сервісів та систем полягає в тому, що вона є основною мовою для створення веб-сторінок. HTML дозволяє визначити заголовки, абзаци, списки, таблиці, посилання, зображення та інші елементи, що становлять контент сторінки. За допомогою тегів HTML можна визначити структуру документа та відношення між його різними частинами.

HTML також забезпечує можливість вбудовування різних мультимедійних елементів, таких як відео або аудіо, у веб-сторінки. Вона також підтримує форми, які дозволяють користувачам взаємодіяти зі сторінкою шляхом заповнення полів, надсилання даних на сервер тощо.

Крім того, HTML використовується для створення структури документів, що має важливе значення для пошукових систем і доступності веб-сторінок. Правильна розмітка та використання семантичних тегів допомагають пошуковим системам краще індексувати контент сторінки та покращують його доступність для людей з обмеженими можливостями.

Рис. 2.1. Логотип HTML5

2.2. Каскадні таблиці стилів CSS

Каскадні таблиці стилів (CSS) є мовою, що використовується для опису зовнішнього вигляду документів HTML та XML. CSS визначає, як елементи на веб-сторінці повинні бути відображені, включаючи кольори, шрифти, розташування, розміри і фонові зображення.

Одним з важливих понять у CSS є каскадність. Каскадність означає, що стилі можуть бути визначені на різних рівнях і їх застосовуватиметься відповідно до певного порядку. В CSS існує три рівня каскадності:

1. Рівень елементу: Стилі можуть бути прямо визначені на самому елементі. Наприклад, ви можете встановити стиль для конкретного елемента безпосередньо у тезі HTML.
2. Рівень класу: CSS дозволяє використовувати класи для групування елементів і застосовувати до них спільні стилі. Ви можете створити клас у CSS і використовувати його для будь-якого елемента в HTML.

3. Рівень ідентифікатора: Ідентифікатори в CSS є унікальними для конкретного елемента. Ви можете використовувати ідентифікатор для встановлення стилів, які будуть застосовуватися тільки до одного елемента.

Рис. 2.2. Логотип CSS

Роль каскадних таблиць стилів у розробці сервісів та систем полягає в тому, що вони забезпечують розділення вмісту веб-сторінки від її представлення. Це означає, що ви можете мати окремі файли CSS, які визначають зовнішній вигляд вашого веб-сайту, тим самим полегшуючи підтримку і зміну стилю в майбутньому.

Застосування каскадних таблиць стилів дозволяє також створювати однорідний вигляд для всього веб-сайту або системи, використовуючи однакові стилі для різних елементів на всіх сторінках. Вони також дозволяють забезпечити адаптивність та респонсивний дизайн, що дозволяє вашому веб-сайту адаптуватися до різних розмірів екранів та пристроїв.

Загальне поняття каскадних таблиць стилів та їх роль в розробці сервісів і систем полягає в тому, щоб забезпечити контроль над зовнішнім виглядом вашого веб-сайту і забезпечити його консистентність, легкість підтримки та гнучкість у зміні стилів.

2.3. Мова програмування JavaScript

JavaScript є однією з найпоширеніших мов програмування, використовуваних у веб-розробці. Вона забезпечує можливість створювати динамічні та інтерактивні веб-сторінки, а також виконувати різноманітні завдання на боковому (клієнтському) боці.

Однією з основних особливостей даної мови програмування є те, що JavaScript виконується безпосередньо в браузері користувача, що дозволяє здійснювати маніпуляції з HTML-кодом та CSS стилями, змінювати вміст сторінки під час її завантаження або після натискання користувачем на елементи.

Не менш важливою особливістю є кросплатформеність, яка полягає в тому, що JavaScript підтримується більшістю сучасних веб-браузерів на різних операційних системах, таких як Windows, macOS, Linux, Android та iOS. Це дозволяє розробникам створювати веб-додатки, які можуть працювати на різних пристроях.

Також JavaScript підтримує асинхронне програмування, що дозволяє виконувати різні завдання паралельно без блокування основного потоку виконання. Це особливо корисно для обробки мережевих запитів, роботи з базами даних та інших операцій, які можуть займати час.

Згадуючи про JavaScript, не можна не згадати про її багатофункціональність. JavaScript має широкий набір вбудованих функцій та можливостей. Це включає роботу зі змінними, рядками, масивами, об'єктами, динамічне створення HTML-елементів, маніпуляції з подіями, анімації, роботу з кукісами та локальним сховищем.

Роль JavaScript у розробці сервісів та систем:

JavaScript відіграє велику роль в веб-розробці. Вона використовується для створення динамічних веб-сторінок, додавання ефектів, валідації форм, маніпуляцій з DOM-структурою сторінки та інших клієнтських операцій. Він дозволяє розробникам створювати багатофункціональні, інтерактивні і користувацьки-орієнтовані інтерфейси.

А ще JavaScript може бути використаний для валідації та перевірки даних на клієнтському та серверному боці. Розробники можуть перевіряти коректність

введених даних, формат рядків, числові значення, масиви та об'єкти перед збереженням або обробкою.

Узагальнюючи, JavaScript грає важливу роль у веб-розробці, дозволяючи створювати динамічні, інтерактивні та масштабовані веб-сервіси та системи як на клієнтському, так і на серверному боці. Його широкі можливості та популярність серед розробників зробили його одним з основних інструментів для створення сучасних веб-додатків.

Рис. 2.3. Логотип JavaScript

2.4. Мова програмування Java

Java – це потужна, об'єктно-орієнтована мова програмування, яка була створена компанією Sun Microsystems у 1995 році. Вона була спеціально розроблена для того, щоб бути "платформонезалежною", що означає, що програми, написані на Java, можуть працювати на будь-якій операційній системі, яка підтримує віртуальну машину Java (JVM).

Java має багатий набір функціональних можливостей і широко використовується для розробки різних видів програмного забезпечення, включаючи сервіси та системи. Саме через її багатофункціональність, можна виділити велику кількість ключових рис.

Найважливішою рисою Java, як і багатьох інших поупулярних мов програмування, є її об'єктно-орієнтованість. Java побудована на об'єктно-орієнтованому підході до програмування, що дозволяє розбити складність системи на

менші, самостійні компоненти (об'єкти). Це сприяє більшій модульності, повторному використанню коду та полегшує розробку та підтримку сервісів та систем.

Другою ключовою рисою даної мови програмування можна вважати платформонезалежність. Java програми компілюються в байт-код, який виконується на віртуальній машині Java (JVM). Це означає, що один і той же вихідний код може працювати на будь-якій платформі, яка підтримує JVM, незалежно від операційної системи або апаратного забезпечення. Це дає розробникам більшу гнучкість та можливість написати одну програму, яка працює на різних платформах.

Java поставляється з великою стандартною бібліотекою, яка містить різноманітні класи та інструменти для розробки різних видів програм. Це включає підтримку мережевого програмування, обробки рядків, криптографії, роботи з базами даних та багато іншого. Ця бібліотека допомагає розробникам зосередитися на бізнес-логіці своїх проектів, прискорюючи процес розробки.

Java пропонує високу продуктивність та масштабованість, що особливо важливо для розробки сервісів та систем. Вона підтримує багатопоточність (здатність виконувати декілька операцій одночасно) та має ефективну збірку сміття (garbage collection), що допомагає уникнути витрат пам'яті та підтримує стабільну роботу системи при великому обсязі даних.

Ще Java має велику спільноту розробників, яка надає велику кількість ресурсів, бібліотек, фреймворків та інструментів для підтримки розробки. Це дозволяє розробникам швидше вирішувати проблеми, отримувати підказки та сприяє швидкому розвитку проектів.

Усі ці функції та можливості роблять Java популярним вибором для розробки сервісів та систем. Вона використовується в різних сферах, включаючи розробку веб-додатків, мобільних додатків, корпоративних систем, фінансових додатків та багато іншого.

Рис. 2.4. Логотип Java

Вибір мною саме мов JavaScript та Java для моєї роботи зумовлені тим, що я їх непогано знаю, адже вивчав до цього, а також вони є одними з найопопулярніших мов програмування в світі та Україні. Не дарма в рейтингу на кінець 2022 року найбільшої спільноти розробників України DOU.ua ці мови посідаються 1 і 3 місця відповідно.

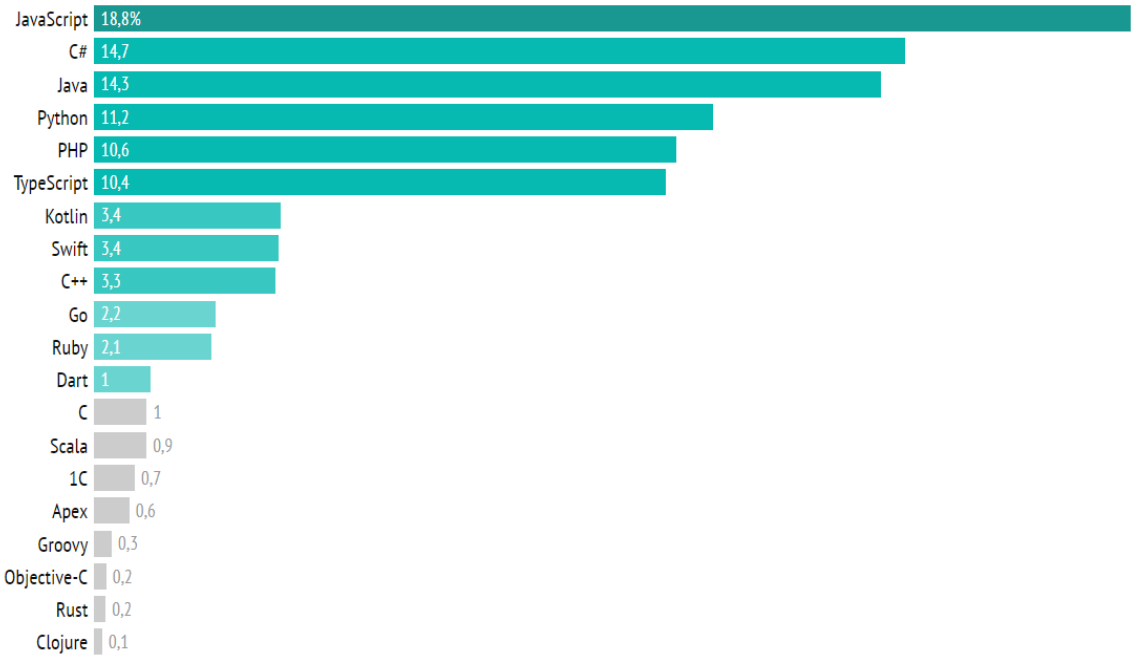


Рис. 2.5. Рейтинг популярних мов програмування в Україні на кінець 2022 року

2.5. Середовище розробки Visual Studio Code

Visual Studio Code (VS Code) – це безкоштовне середовище розробки, створене компанією Microsoft. Воно надає зручний та потужний інструментарій для програмістів, який дозволяє розробляти різноманітні програми та веб-додатки. Visual Studio Code підтримує операційні системи Windows, macOS і Linux, що дозволяє розробникам працювати на своєму зручнішій для них ОС.

VS Code має масив розширень, які дозволяють розширити його функціональність для ваших потреб. Можна встановити розширення для підтримки різних мов програмування, засобів роботи з Git, відладки, форматування коду та багато іншого.

Це середовище розробки надає потужний редактор коду з можливістю підсвітки синтаксису для багатьох мов програмування. Це допомагає покращити читабельність коду та полегшує редагування.

Також можна налаштовувати точки зупину та крокувати через код для відладки своїх програм. Visual Studio Code надає інструменти для відлагодження коду на рівні рядка, що допомагає виявити й усунути помилки.

Дане середовище розробки вбудовується з популярними системами керування версіями, такими як Git. Це дозволяє легко відстежувати зміни в коді та працювати з репозиторіями безпосередньо з середовища розробки.

Вбудовані в даний програмний продукт пошук та заміна дозволяють знаходити й змінювати текст у файлах проекту. Можна просто використовувати регулярні вирази та інші параметри для точного налаштування пошуку.

Visual Studio Code підтримує можливість підключатися до віддалених серверів чи контейнерів для розробки на віддалених системах або в хмарних середовищах. Також він має вбудований термінал, що дозволя виконувати команди з проекту, не виходячи з середовища розробки.

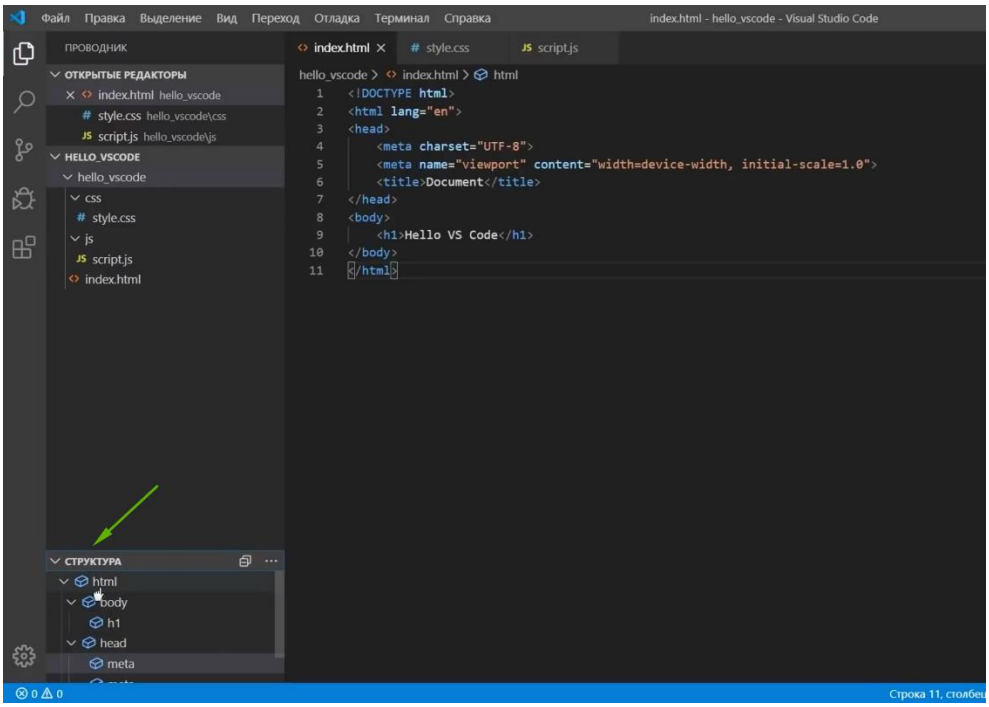


Рис. 2.6. UI вигляд середовища розробки Visual Studio Code

Всі ці риси роблять Visual Studio Code популярним вибором для розробки програмного забезпечення та веб-додатків. Він надає широкі можливості та персоналізацію для підтримки різних мов програмування та стеків технологій.

2.6. СУБД PostgreSQL

PostgreSQL (або просто Postgres) – це потужна реляційна система керування базами даних (СКБД), яка забезпечує надійне зберігання та керування даними. Вона є однією з найпопулярніших відкритих джерел даних, розроблених з використанням SQL (Structured Query Language).

PostgreSQL заснований на реляційній моделі, що дозволяє зберігати дані в таблицях зі зв'язками між ними. Це спрощує структурування даних та виконання складних запитів.

Дана СУБД підтримує стандартний SQL-синтаксис, а також надає додаткові можливості, такі як вкладені запити, функції та процедури, вирази, тригери, відкладені перевірки, віконні функції тощо.

PostgreSQL забезпечує високу надійність шляхом використання механізмів транзакцій та журналування. Він також підтримує цілісність даних за допомогою обмежень, перевірок, індексів та зовнішніх ключів.

СУБД може бути розширена за допомогою додаткових модулів, що надають можливості, які не входять до основної установки. Це дозволяє розробникам використовувати PostgreSQL для різноманітних завдань, включаючи географічні, текстові та інші типи даних.

PostgreSQL має вбудовану підтримку реплікації, що дозволяє створювати копії бази даних для забезпечення високої доступності та масштабованості. Він також підтримує горизонтальне шарування даних для розподілу навантаження між багатьма серверами.

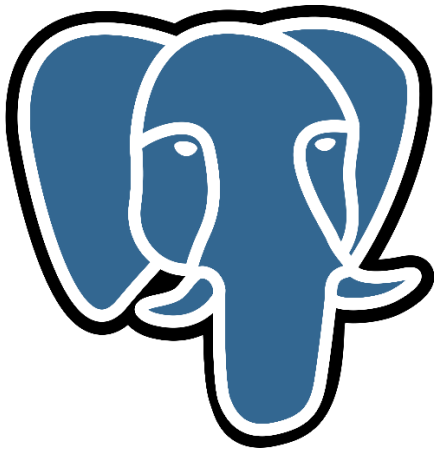


Рис. 2.7. Логотип PostgreSQL

Даний програмний продукт використовується для зберігання даних у багатьох типах додатків, включаючи веб-додатки, системи керування вмістом та інші. Він дозволяє ефективно організувати та управляти даними в базі даних, а також надає потужні можливості для виконання складних запитів та аналітики даних. А ще він підтримує розширені функції SQL, що дозволяють виконувати розрахунки, агрегування, фільтрацію, сортування та інші операції з даними.

PostgreSQL забезпечує надійність та цілісність даних, що є критичними аспектами для багатьох систем. Він має механізми транзакцій та журналування, які

дозволяють відновлювати базу даних в разі відмови. Також PostgreSQL має різні рівні безпеки, включаючи автентифікацію, авторизацію та шифрування даних.

PostgreSQL може бути використаний для побудови масштабованих систем за допомогою реплікації та шарування даних. Це дозволяє забезпечити високу доступність та розподілену обробку навантаження.

В цілому, PostgreSQL є потужним інструментом для розробки різноманітних сервісів та систем, які вимагають надійного зберігання, керування та обробки даних. Він надає розширені можливості SQL, масштабованість та надійність, що робить його популярним вибором для багатьох розробників.

2.7. Spring Framework

Spring Framework є одним з найпопулярніших фреймворків для розробки програмного забезпечення на мові Java. Він надає розробникам широкий набір інструментів і бібліотек для створення різноманітних додатків, включаючи веб-сервіси та системи.

Основні поняття, на яких ґрунтується Spring Framework, включають інверсію керування (Inversion of Control, IoC) та аспектно-орієнтоване програмування (Aspect-Oriented Programming, AOP).

IoC дозволяє зменшити залежність між компонентами додатку шляхом зсуву відповідальності за управління залежностями з клієнта на контейнер Spring. Контейнер Spring керує створенням та внедренням залежностей між компонентами, що робить додаток більш модульним, легким для тестування та розширення.

AOP дозволяє відокремити побічні аспекти додатку (наприклад, журналювання, транзакції, безпеку) від його основної бізнес-логіки. Завдяки цьому можна застосовувати ці аспекти повсюдно в додатку, без необхідності їх включення в кожну окрему функцію або клас.

Spring Framework також надає багато інших можливостей, таких як обробка HTTP-запитів, робота з базами даних, кешування, безпека, планування завдань та багато іншого. Він підтримує різні модулі, такі як Spring MVC для розробки веб-

додатків, Spring Data для роботи з базами даних, Spring Security для забезпечення безпеки та багато інших.

Роль Spring Framework в розробці сервісів та систем полягає в спрощенні процесу розробки, поліпшенні модульності, підвищенні ефективності та забезпеченні більшої контролюваності додатків. Він дозволяє розробникам концентруватись на бізнес-логіці додатку, а не на рутинних завданнях, таких як керування залежностями та інфраструктурою. Крім того, Spring Framework має велику спільноту користувачів, що сприяє обміну знаннями та підтримці.

У загальному розумінні, Spring Framework допомагає розробникам створювати розширювані, гнучкі та надійні сервіси та системи на основі мови Java.

2.8. Висновки до розділу

У другому розділі я проаналізував і обрав технології, які буду використовувати при розробці веб-сервісу. Всі обрані мною технології використовуються в розробці сучасних веб-сервісів та веб-систем.

Для створення веб-сторінок я використовуватиму HTML5, для надання привабливого вигляду буде використано CSS, а для надання інтерактивності на сторінці буде використано JavaScript та її бібліотеки.

В серверній частині веб-сервісу буде використовуватися мова програмування Java та найпопулярніший її фреймворк – Spring Framework, який містить багато бібліотек, з якими буде зручніше розроблювати систему.

Базою даних було обрано PostgreSQL, яка має високу надійність, а також механізм журналювання та можливість адміністрування у самому редакторі.

РОЗДІЛ 3
РОЗРОБКА ВЕБ-СЕРВІСУ

3.1. Основні функції та система авторизації користувачів системи

Веб-сервісом надання послуг масажу є веб-сайт, який допомагає спростити та автоматизувати роботу масажного салону, за допомогою технологій, які використані при розробці.

До функцій, які будуть на сайті, потрібно віднести реєстрацію нового користувача, авторизацію користувача, можливість переглядати інформацію про масажний салон, можливість переглядати профіль користувача, можливість бронювати місця на масаж. Також потрібно поділити всіх осіб, які будуть користуватися даним сервісом на ролі, тому поділимо їх на такі ролі: гість, користувач, працівник та адміністратор. Саме так подідяють ролі на сучасних сайтах.

Гостем вважається особа, яка не авторизована на сайті. Вона може переглядати інформацію про масажний салон та має можливість зареєструватися і авторизуватися на сайті.

Користувачем є особа, яка авторизувалася на сайті, та має доступ до всіх базових функцій. Він може переглядати інформацію про масажний салон, має можливість переглядати власний профіль та змінювати його, а також може бронювати місця.

Працівником вважається користувач, який має всі базові функції, а також може переглядати дані про замовлення, щоб мати дані про свій робочий графік, але не може змінювати їх.

Адміністратором вважається користувач, який має доступ до всіх даних сервісу, які він може змінювати, а також він може налаштовувати сервіс та його базу даних.

3.2. Архітектура веб-сервісу

Архітектура веб-системи допомагає організувати структуру та взаємозв’язки компонентів для того, щоб досягти її функціональних вимог.

Даний веб-сервіс використовує архітектуру Model View Controller (MVC). Вона дозволяє розділити компоненти додатку на три основні частини: модель (Model), представлення (View) та контролер (Controller). Кожна з цих частин виконує свої функції та має відповідальність за конкретний аспект програми.

Модель представляє бізнес-логіку системи та дані, з якими вона працює. Вона має знати, як маніпулювати цими даними та забезпечувати їхню консистентність. Модель не залежить від представлення або контролера і може бути використана в різних частинах системи

Представлення відповідає за візуалізацію даних моделі та їх відображення користувачеві. В нашому випадку воно є веб-сторінкою або будь-яким іншим способом відображення інформації. Представлення отримує дані з моделі і не має прямого зв'язку з контролером. У деяких реалізаціях представлення може надсилати повідомлення контролеру при дії користувача, але не має жорсткої залежності.

Контролер приймає вхідні дані від користувача через представлення та виконує відповідні дії. Він має знати, як взаємодіяти з моделлю, щоб оновити її стан, і відправляти необхідні дані до представлення для відображення змін. Контролер також може обробляти різні події та взаємодіяти з іншими контролерами.

MVC дозволяє розділити логіку додатку на окремі компоненти, що полегшує його розробку та тестування. Він також забезпечує більшу модульність, оскільки зміни в одному компоненті не впливають на інші.

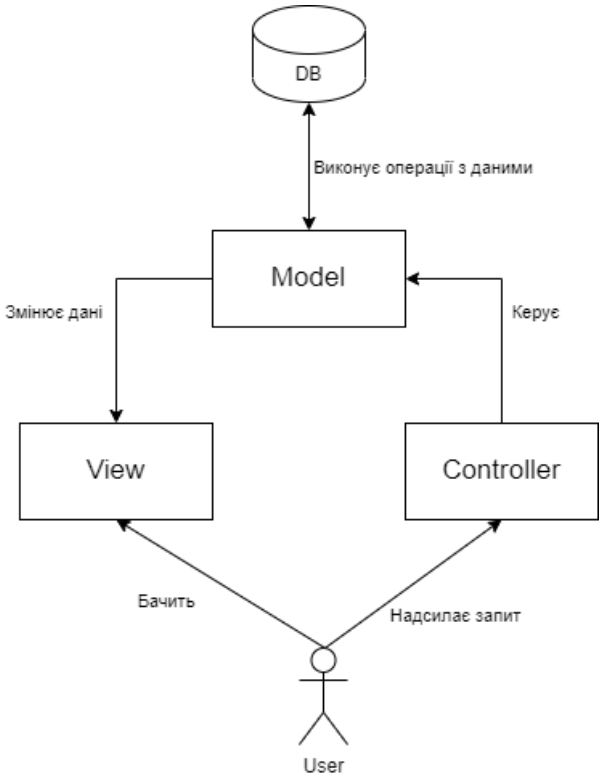


Рис. 3.1. Архітектура системи

3.3. Проектування та створення бази даних

Перед створення бази даних необхідно ретельно проаналізувати вимоги і потреби системи. Це означає, що потрібно визначити типи і обсяг даних, які будуть зберігатися в базі даних, а також функціональні вимог до системи.

Після того як ми визначилися з типами і обсягом даних, потрібно описати структру даних, включаючи таблиці, в яких містяться атрибути та їх сутності, та зв’язки між ними, а потім потрібно побудвати логічну модель. Але при описі структури даних потрібно дотримуватися деяких правил, щоб правильно все описати, зокрема потрібно дотримуватися трьох нормальних форм.

Головною ідеєю першої нормальної форми є те, що кожне значення в базі даних повинно бути неділеним на більш прості компоненти, а також кожен рядок цієї таблиці повинен містити лише одне значення для кожного атрибуту, тобто не можна зберігати кілька однотипних елементів у одному полі.

За другою нормальною формою, кожен неключовий атрибут повинен залежати від всього первинного ключа, а не лише від його підрядкової частини. Якщо який-небудь неключовий атрибут залежить тільки від частини первинного ключа, він має бути виокремлений у власну таблицю.

Суть третьої нормальної форми полягає в тому, щоб усунути транзитивні функціональні залежності між атрибутами таблиці. Іншими словами, кожен атрибут повинен залежати тільки від первинного ключа таблиці або від атрибутів, які безпосередньо залежать від первинного ключа, і не повинен залежати від інших атрибутів таблиці.

Після проведення нормалізації, можна виділити такі сутності:

- Користувач (users) з атрибутами: id, username, password, email, first_name, last_name, sex;
- Замовлення (orders) з атрибутами: id, order_date, user_id, description;
- Послуга (service) з атрибутами: id, massage_type, price, description;
- Час сесій (session_time) з атрибутами: id, day, time_start, time_end, employee_id, order_id;
- Роль (role) з атрибутом name.

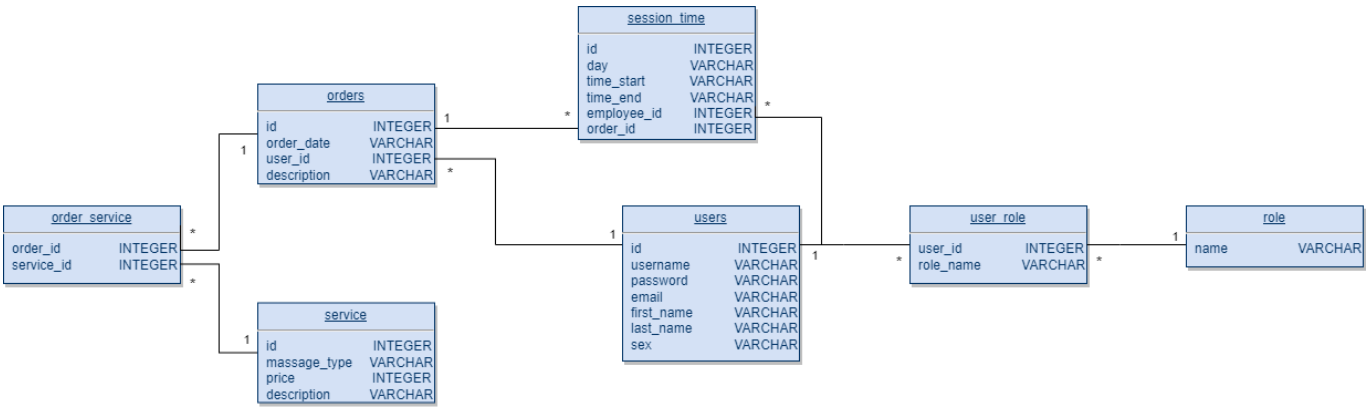


Рис. 3.2. Логічна модель бази даних веб-системи

3.4. Реалізація функціоналу та користувацього інтерфейсу

Для того, щоб розуміти як працює система, потрібно ознайомитися з принципами її роботи. Розглянемо на прикладі того, як система буде працювати для нових користувачів.

Щоб отримати доступ до функцій користувача, гостеві для початку потрібно зареєструватися. При реєстрації дані надсилваються на сервер, де проходять валідацію даних, і якщо все добре, то дані про нового користувача записуються до бази даних.

Далі користувачеві потрібно авторизуватися в свій особистий профіль користувача, щоб отримати доступ до функцій користувача. Тут дані з форми, також надсилаються до серверу, де він перевіряє правильність введених даних в формі з даними в базі даних.

Після авторизації користувач може створити бронювання масажної сесії, переглянути свій особистий профіль та інформацію щодо масажного салону.

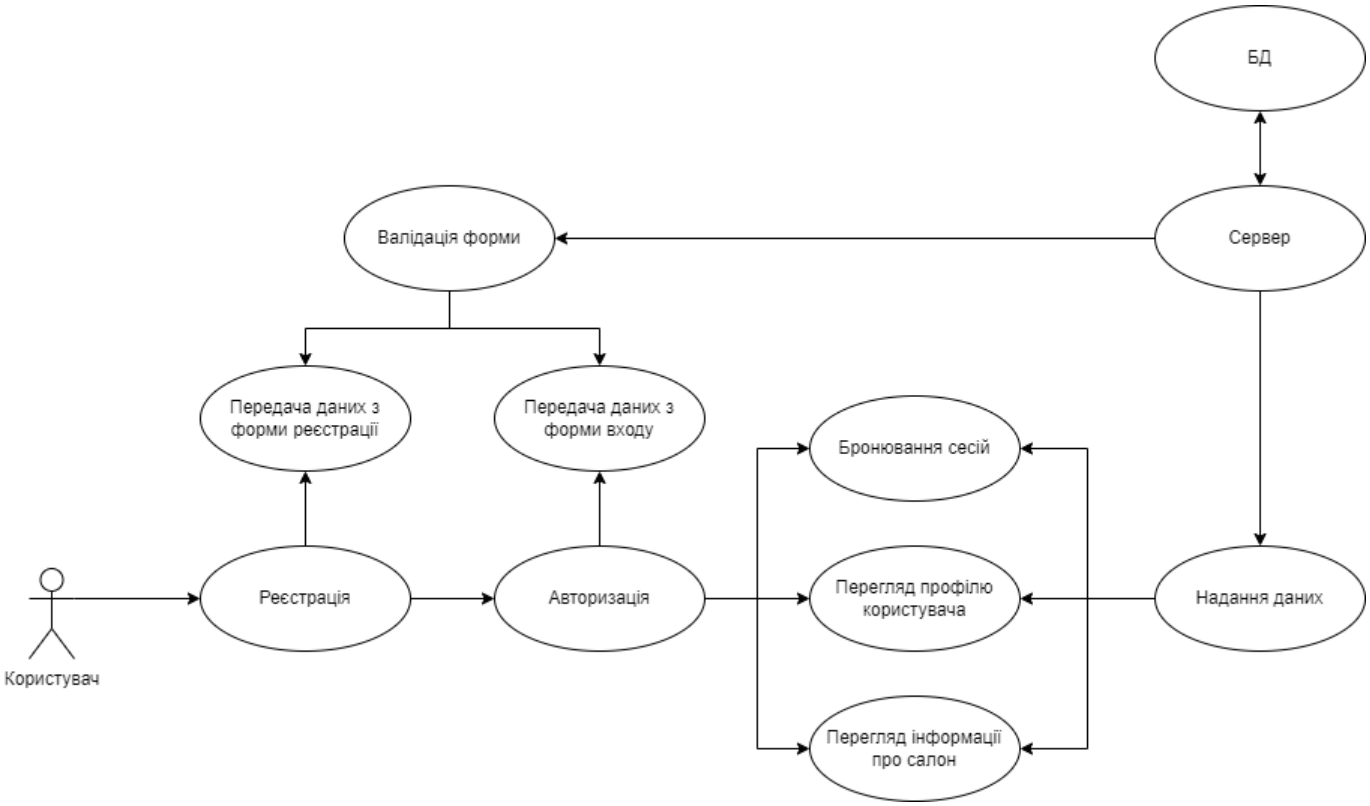


Рис. 3.3. Принцип роботи системи для нового користувача

Як уже було сказано раніше, в даній системі використовується Model-View-Controller архітектура. Тому для кращого розуміння того, як вона працює в даній системі, далі буде більш детальний її опис.

Модель представляє собою об'єкти або класи, які відображають структуру та логіку даних системи. У випадку використання бази даних PostgreSQL, модель включає класи, що відображають таблиці бази даних, а також класи для доступу до даних (Data Access Objects – DAO). Spring допомагає забезпечити зручний доступ до бази даних через Jdbc.

Представлення відображає дані користувачеві та забезпечує взаємодію з ним. У нашому випадку представленням є HTML, JavaScript та CSS, які забезпечують взаємодію з користувачем.

Контролер приймає HTTP-запити від користувача та взаємодіє з моделлю та представленням. У Spring Framework контролери визначаються за допомогою анотацій, таких як @Controller або @RestController. Контролери обробляють HTTP-запити, виконують необхідну бізнес-логіку та передають дані до відповідного представлення.

У цьому підході контролери отримують дані з моделі та передають їх в представлення, яке генерує відповідь для користувача. При отриманні запиту від користувача, контролер обробляє його, виконує необхідну логіку, взаємодіє з базою даних за допомогою DAO та, якщо потрібно, передає дані до представлення для відображення результату.

Як уже стало зрозуміло, то контролери, є однією з найважливіших частин даної архітектури, тому для початку потрібно їх створити. Основні контролери, які повинні бути створенні, відносяться до функціоналу реєстрації, авторизації, бронювання місць на масажні сесії та профілю користувача. Також потрібно буде створити контролери інших сторінок сервісу.

```java
@Controller
public class RegistrationController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("/register")
    public String showRegistrationForm(Model model) {
        model.addAttribute("user", new User());
        return "registration";
    }

    @PostMapping("/register")
    public String registerUser(@ModelAttribute("user") User user) {
        User existingUser = userRepository.findByUsername(user.getUsername());
        User existingEmail = userRepository.findByUsername(user.getEmail());
        if (existingUser != null || existingEmail != null) {
            return "redirect:/register?error";
        }

        userRepository.save(user);

        return "redirect:/sucreg";
    }
}
```

Рис. 3.4. Загальний вигляд контролера реєстрації

```java
@Controller
public class LoginController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("/login")
    public String showLoginForm() {
        return "login";
    }

    @PostMapping("/login")
    public String loginUser(@RequestParam String username, @RequestParam String password) {
        User user = userRepository.findByUsername(username);
        if (user == null || !user.getPassword().equals(password)) {
            return "redirect:/login?error";
        }

        return "redirect:/home";
    }
}
```

Рис. 3.5. Загальний вигляд контролера авторизації

Після створення контролерів потрібно створити представлення сайту, що представляє собою сторінки сайту, а також потрібно реалізувати модулі сервісу та всі функції, які буде виконувати система. Наприклад, можна створити сторінку, яка буде містити інформацію про сам масажний салон, його працівників, послуги, які він надає, адресу салону, електронну пошту або контактний телефон салону тощо.

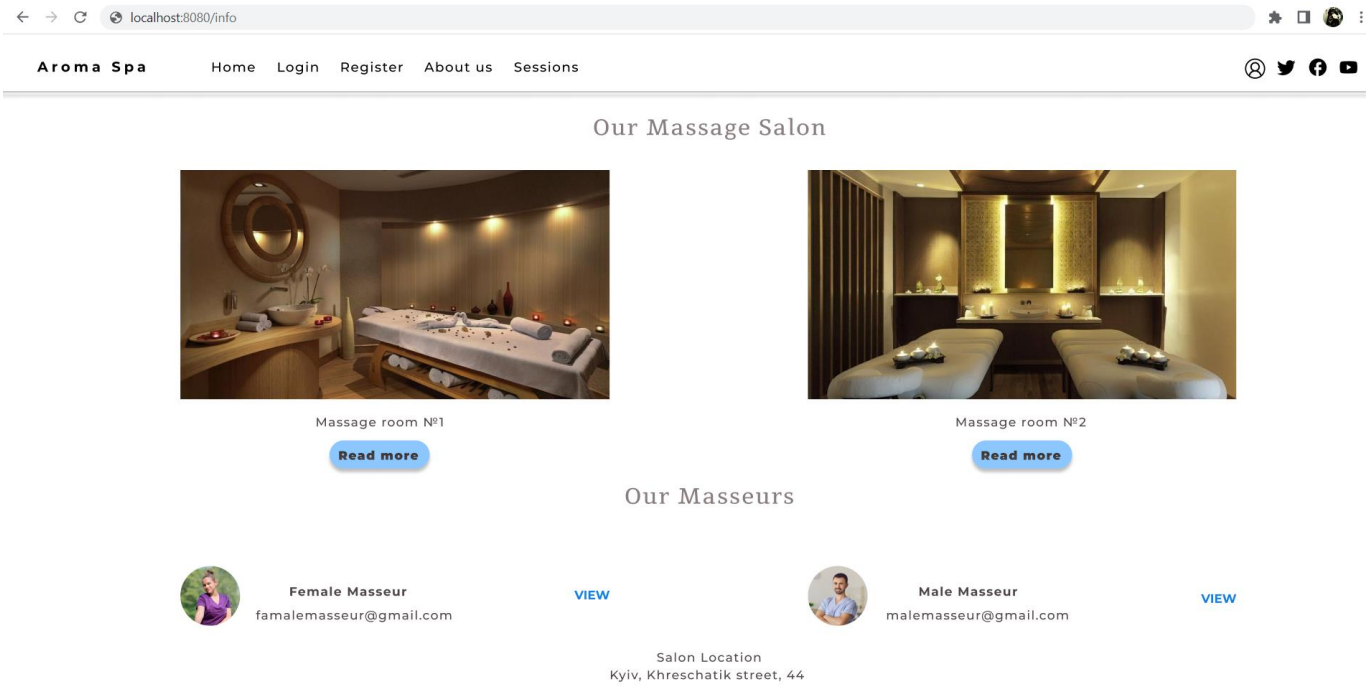


Рис. 3.6. Сторінка «info»

Загалом було вирішено, що сайт повинен мати не складний, приваблий та зрозумілий інтерфейс, щоб користувачі сайту могли розуміти, що де знаходиться, і що потрібно заповнювати чи нажимати в тому чи іншому випадку.

3.4.1. Реєстрація користувачів системи

Реєстрація є складовою частиною веб-системи, яка дозволяє зареєструватися та створити особистий профіль гостям сайту. Вона представляє собою веб-сторінку, яка складається з таких полів для заповнення:

- Email: поле для введення електронної пошти користувача, яке може використовуватися для зв’язку з користовучем.
- Ім’я користувача: поле для введення унікального імені користувача, яке буде використовуватися в подальшому для входу в систему.
- Пароль: поле для введення паролю користувача, який потрібний для захисту облікового запису користувача, а також потрібне буде для авторизації.
- Ім’я та прізвище: поля для введення особистих даних користувача.
- Стать: поле, в якому користувач може обрати свою стать.

Також на формі є 3 кнопки:

- «Реєстрація»: кнопка, яка проводить валідацію форми та перевіряє правильність введених даних, та зберігає дані користувача в базі даних, при правильному введенні, а також створює його особистий акаунт.
- «Умови надання послуг»: кнопка, при натисканні якої користувачу виводяться умови надання послуг, які містять угоду між користувачем та особою надання послуг, після ознайомлення з якою користувач може натиснути на прапорець в разі згоди.
- «Авторизуйтеся тут»: кнопка, яка переправляє користувача на сторінку авторизації, якщо він уже зареєстрований.

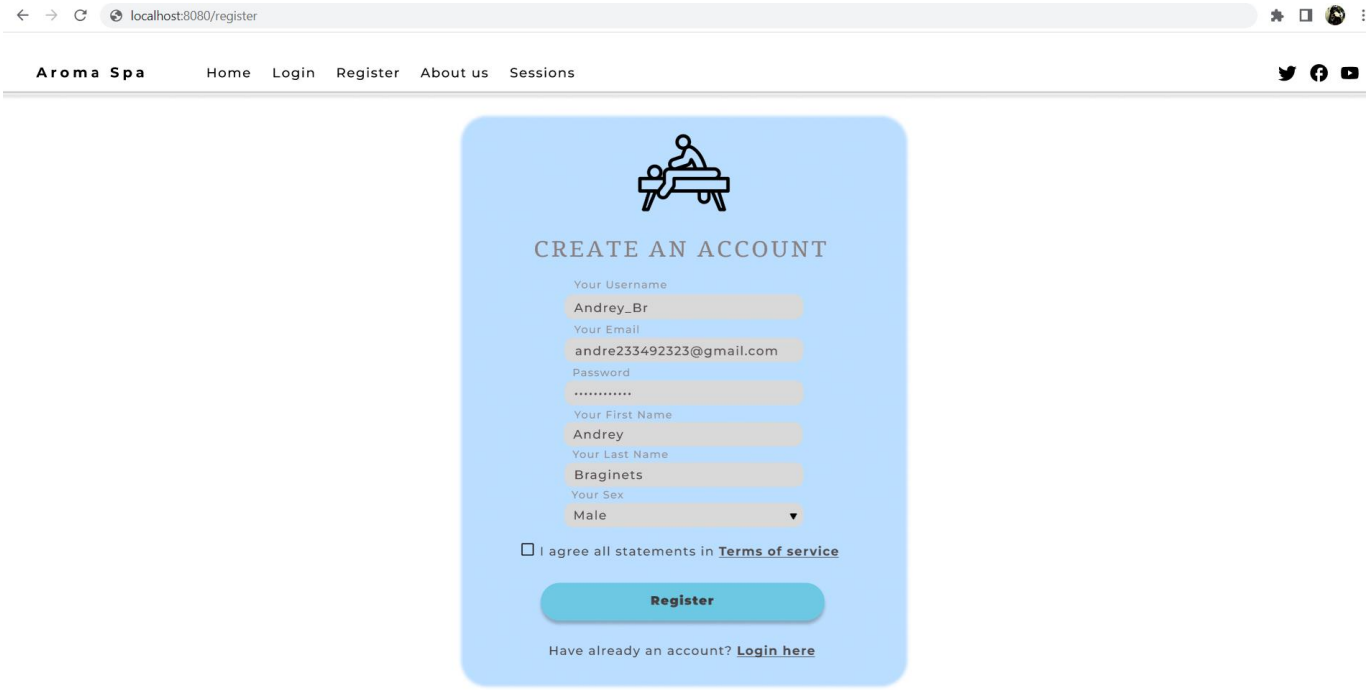


Рис. 3.7. Сторінка реєстрації

При реєстрації користувач відкриває сторінку реєстрації та вводить свої дані, такі як: ім’я користувача, електронну пошту, пароль, ім’я, прізвище та вибирає стать. Далі він ознайомлюється з умовами надання послуг та, в разі згоди, натискає на прапорець. Потім він натискає на кнопку реєстрації і дані відправляються на сервер.

Контролер реєстрації обробляє цей запит та перевіряє чи існує користувач з таким же іменем та електронною поштою в базі даних, і якщо ні, то зберігає нового користувача в базі даних.

```java
@Entity
@Table(name = "users")
public class User {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String username;
    private String email;
    private String password;
    private String first_name;
    private String last_name;
    private String sex;

    public void setlog() {
        this.username = inp.username;
        this.email = inp.email;
        this.password = inp.password;
        this.first_name = inp.first_name;
        this.last_name = inp.last_name;
        this.sex = inp.sex;
    }
}

@Repository
public interface UserRepository extends JpaRepository<User, Long> {
    User findByUsername(String username);
}
```

Рис. 3.8. Частина коду моделі реєстрації

Також при реєстрації використовується метод createUser та функція createAccount для створення нового облікового запису користувача за допомогою введених імені користувача та пароля.

```javascript
async function createAccount(user_info){
    startTransition(() => {setReg(true)})
    localStorage.setItem('user', JSON.stringify({...user_info}))

    try{
        await createUser(auth, user_info.email, user_info.password)
    }
    catch(error){
        localStorage.removeItem('user')
        console.log(error.massage)
        setError(error.massage)
    }

}
```

Рис. 3.9. Код функції createAccount

Після успішної реєстрації, користувача переводить на сторінку успішної реєстрації, де йому повідомляють, що обліковий запис був створений і він може авторизуватися в нього.

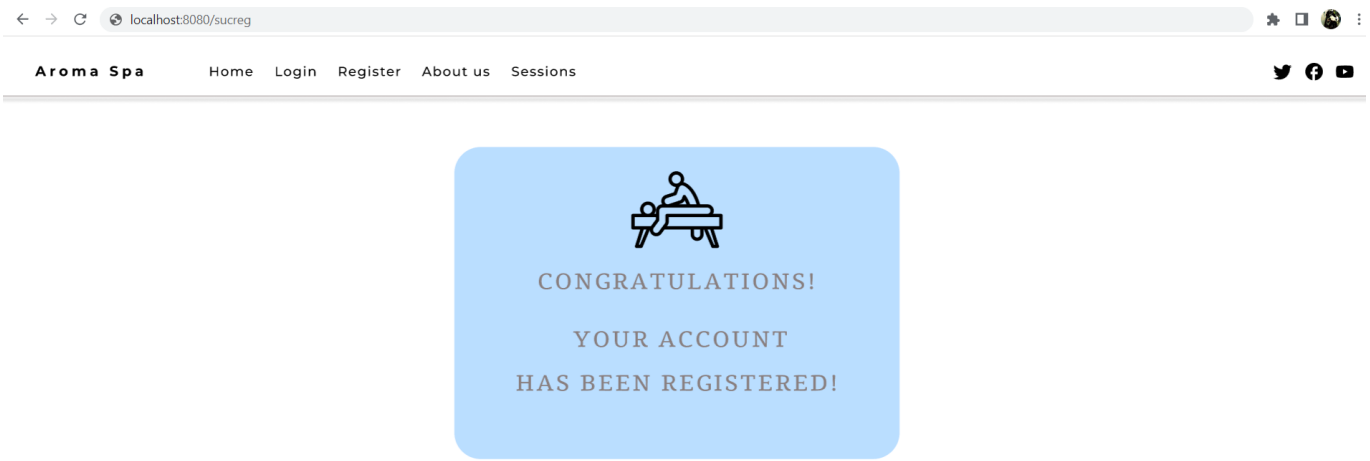


Рис. 3.10. Сторінка повідомлення про успішну реєстрацію

3.4.2. Авторизація користувачів системи

Авторизація – це важлива частина веб-сервісу, так як вона відповідає за процес ідентифікації користувача і надання йому доступу до його облікового запису. Форма атворизації складається всього лише з 2 полів:

- Ім’я користувача: поле для введення унікального імені користувача.
- Пароль: поле для введення пароля користувача, який використовується для перевірки відповідності до збереженого пароля в базі даних.

Також дана сторінка містить в собі дві кнопки:

- «Авторизуватися»: кнопка, при натисканні на яку проходить перевірка введених даних на відповідні з бази даних.
- «Створіть новий»: кнопка, яка перенаправляє користувача на сторінку реєстрації.

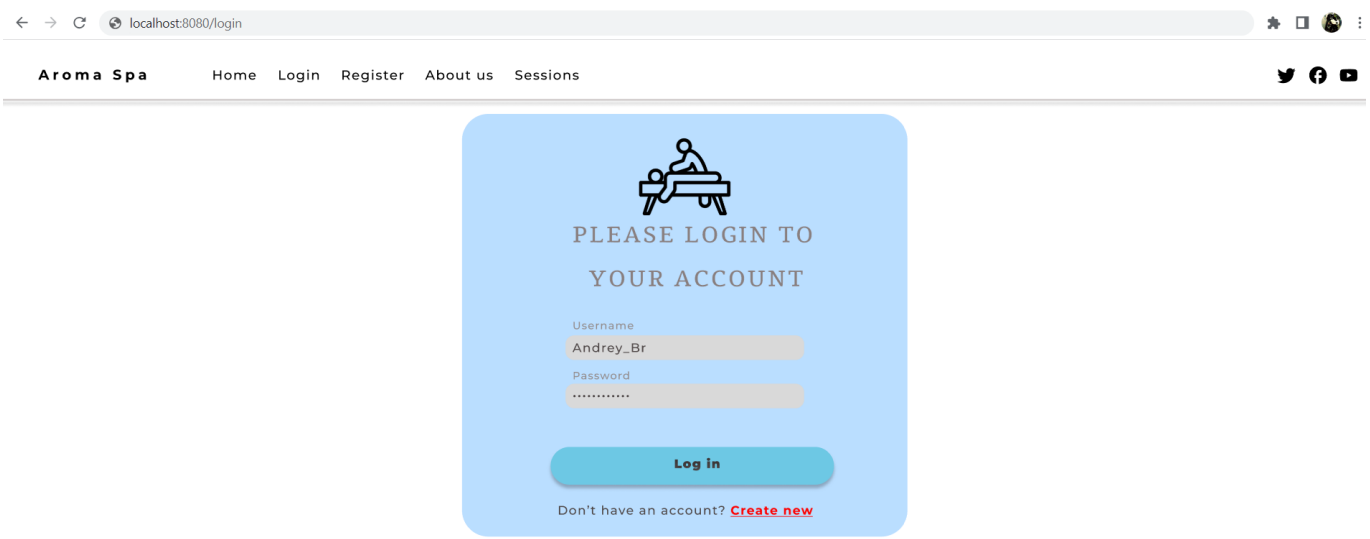


Рис. 3.11. Сторінка авторизації

Під час авторизації, користувач відкриває сторінку входу і вводить свої дані авторизації, такі як ім'я користувача та пароль. При натисканні на кнопку авторизації, дані відправляються на сервер.

Контролер авторизації перевіряє, чи існує користувач з таким ім'ям і паролем в базі даних. Якщо авторизація пройшла успішно, то користувач отримує доступ до функцій користувача, таких як можливість бронювання, переглядання та налаштування власного профілю.

Також при вході в обліковий запис використовуються метод loginAcc та функція authAcc.

```
async function authAcc(user_info){
    try{
        await loginAcc(auth, user_info.email, user_info.password)
    }
    catch(error){
        console.log(error.massage)
        setError(error.massage)
    }
}
```

Рис. 3.12. Код функції authAcc

3.4.3. Бронювання масажних сесій

Бронювання сесій є, напевно, найважливішою функцією користувача, так як саме завдяки їй користувач може записатися на масажну сесію в день та час, коли йому це зручніше.

Сторінка масажних сесій складається з фреймів, на яких знаходяться кнопки з часом, які відповідають за бронювання того часу і дня, які зображені на кнопці та зверху фрейму відповідно. Також було вирішено, що вихідні дні будуть пропускатися, так як буде вважатися, що салон не працює по вихідним.

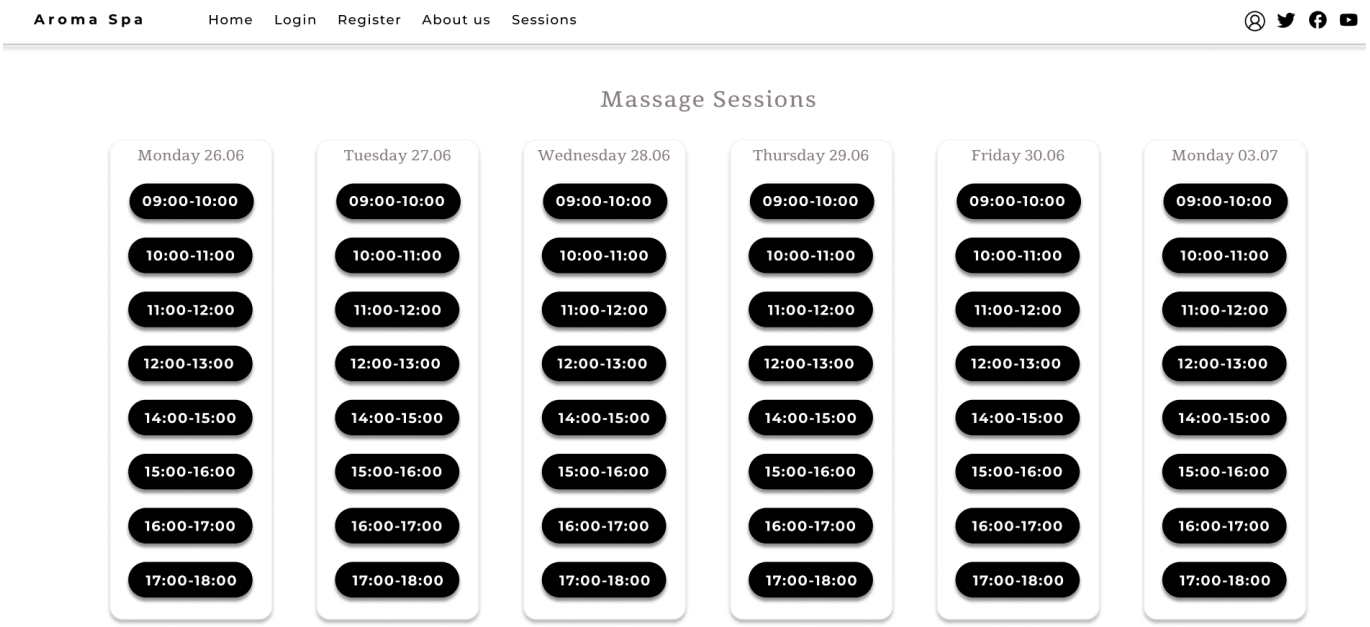


Рис. 3.13. Сторінка масажних сесій

Після авторизації користувач може відкрити сторінку бронювання масажних сесій, де він може вибрати доступну сесію для бронювання. При виборі сесії на сторінці з’являється спливаюче вікно підтвердження бронювання, на якому в користувача запитують, чи він хоче забронювати дану сесію масажу, а нижче зображено дата і час, які були зображені на панелі так кнопкі.

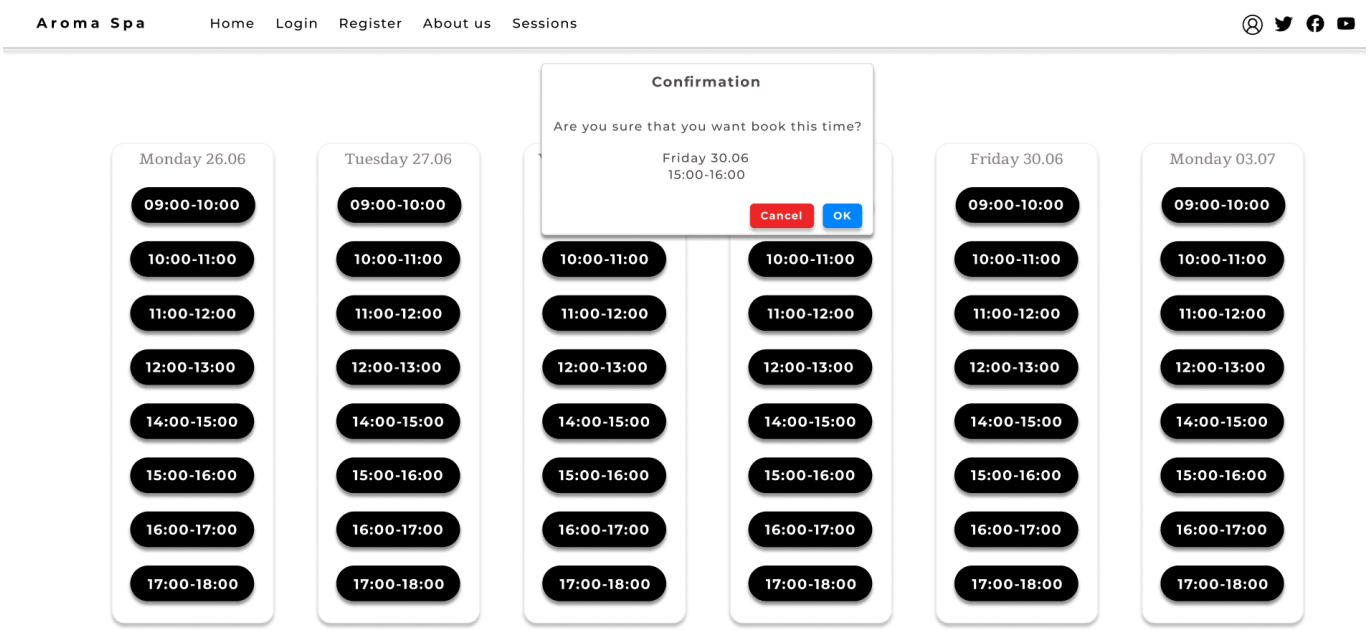


Рис. 3.14. Вікно підтвердження бронювання

Якщо користувач нажав кнопку «*OK*», то дані відправляються на сервер. Контролер бронювання масажних сесій перевіряє дані та зберігає нове бронювання сесії в базі даних за даним користувачем. Після успішного бронювання, дані про нього додаються в спеціальне поле на сторінці користувача.

```java
@Controller
public class BookingController {
    @Autowired
    private MassageSessionRepository massageSessionRepository;

    @GetMapping("/book")
    public String showBookingForm(Model model) {
        model.addAttribute("massageSession", new MassageSession());
        return "booking";
    }

    @PostMapping("/book")
    public String bookSession(@ModelAttribute("massageSession") MassageSession massageSession) {
        List<MassageSession> existingSessions = massageSessionRepository.findByDateTimeBetween(
                massageSession.getDateTime().minusHours(1),
                massageSession.getDateTime().plusHours(1)
        );
        if (!existingSessions.isEmpty()) {
            return "redirect:/book?error";
        }

        massageSessionRepository.save(massageSession);

        return "redirect:/home";
    }
}
```

Рис. 3.15. Загальний вигляд контролера бронювання

Якщо ж користувач натиснув кнопку «Cancel», то вспливаюче вікно закривається і користувач залишається на цій сторінці і далі.

3.4.4. Профіль користувача

Не менш важливою частиною веб-сервісу є профіль користувача. У кожного користувача він особистий і не стосується інших профілей. Він в основному зберігає дані про користувача. Профіль користувача складається з 3 фреймів, на яких зображені різні дані про користувача.

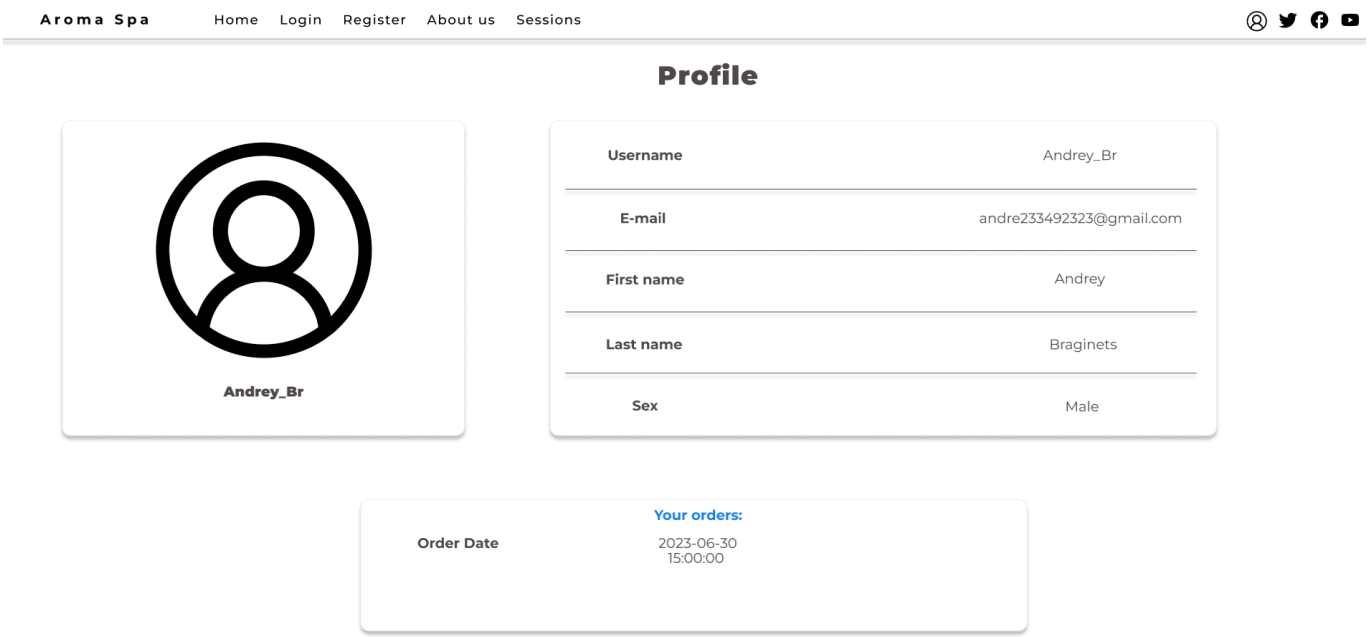
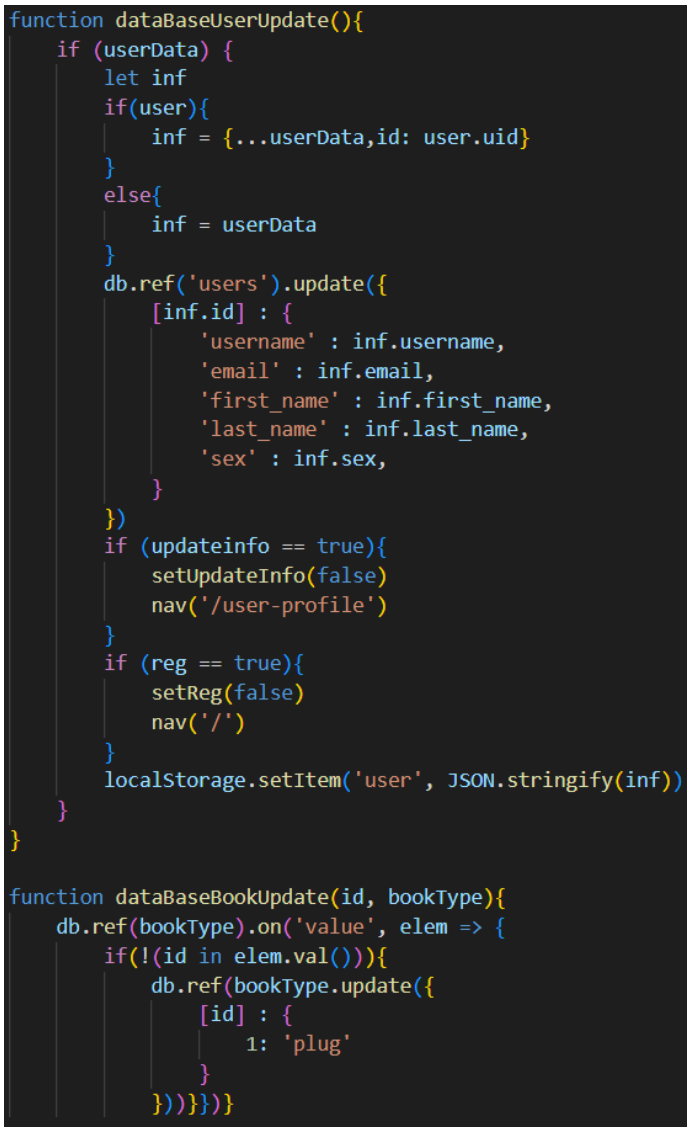


Рис. 3.16. Сторінка профілю користувача

На першому фреймі зображений аватар профіля користувача та ім’я користувача під ним. Користувач може змінити стандартний аватар користувача на той, який він бажає.

На другому фреймі зображені дані, які користувач ввів при реєстрації, окрім пароля. Тобто на ньому записані ім’я користувача, електронна пошта, ім’я та прізвище, а також стать.

На третьому фреймі зображено заброньовані масажні сесії користувача, де вказується дата і час початку заброньованих сесій.



```
function dataBaseUserUpdate(){
    if (userData) {
        let inf
        if(user){
            inf = {...userData,id: user.uid}
        }
        else{
            inf = userData
        }
        db.ref('users').update({
            [inf.id] : {
                'username' : inf.username,
                'email' : inf.email,
                'first_name' : inf.first_name,
                'last_name' : inf.last_name,
                'sex' : inf.sex,
            }
        })
        if (updateinfo == true){
            setUpdateInfo(false)
            nav('/user-profile')
        }
        if (reg == true){
            setReg(false)
            nav('/')
        }
        localStorage.setItem('user', JSON.stringify(inf))
    }
}

function dataBaseBookUpdate(id, bookType){
    db.ref(bookType).on('value', elem => {
        if(!(id in elem.val())){
            db.ref(bookType.update({
                [id] : {
                    1: 'plug'
                }
            }))}})}
```

Рис. 3.17. Код функцій зв’язку з базою даних

Для отримання даних про користувача з бази даних контролер використовує ідетифікатор, який міститься в принципалі.

```java
@Controller
public class UserProfileController {
    @Autowired
    private UserRepository userRepository;

    @GetMapping("/profile")
    public String showUserProfile(Model model, Principal principal) {
        String username = principal.getName(info.username);
        User user = userRepository.findByUsername(username);
        model.addAttribute("username", info.username);
        model.addAttribute("email", info.email);
        model.addAttribute("first_name", info.first_name);
        model.addAttribute("last_name", info.last_name);
        model.addAttribute("sex", info.sex);

        return "profile";
    }

    @PostMapping("/profile")
    public String updateProfile(@ModelAttribute("user") User updatedUser) {
        userRepository.save(updatedUser);

        return "redirect:/profile";
    }
}
```

Рис 3.18. Загальний вигляд контролера профіля

3.5. Сховище об'єктів

Так як в PosrgreSQL, як і в будь-якій іншій реляційній базі даних, можна зберігати інформацію лише в текстовому вигляді, виникає проблема в збереженні об’єктів, таких як: файли різних типів, відео, зображення тощо. В даній системі потрібно десь зберігати аватари користувачів, які представлені у вигляді зображень. Саме для цього потрібне локальне сховище дане. Як локальне сховище об’єктів буде використовуватися Azure.

Azure – це хмарна платформа та набір послуг, розроблених компанією Microsoft. Вона надає широкий спектр хмарних послуг, включаючи обчислення, зберігання даних, аналітику та багато іншого. Azure пропонує гнучкість та масштабованість для розробників у будь-якій галузі. Вона дозволяє створювати, розгортати та керувати додатками в хмарі з використанням різних мов програмування, платформ та інфраструктур.

У своєму проекті я обрав Azure як хмарне середовище через його гнучкість, масштабованість та надійний захист даних. Дана служба також надає один сервер для

зберігання об’єктів. Як уже було сказано раніше, в даній системі будуть використовуватися функції додавання і видаляння даних, а також отримування файлів. Всі ці функції будуть відбуватися через відправляння запиту до контролера.

```
@override
public byte[] getFile(String fileName, String bucketName){
    if (!isBucketExist(bucketName)){
        throw new IllArgExc("bucket with name " + bucketName + " don't exist");
    }
    try {
        GetObjResp object = AzureClient.getObj(GetObjArgs.builder()
        .bucket(bucketName)
        .object(fileName)
        .build());
        return object.readAllBytes();
    } catch (Exception e) {
        throw new IllArgExc(e.getMassage());
    }
}
```

Рис. 3.19. Приклад реалізації одного з методу сервісу Azure

Microsoft Azure Blob Storage використовує використовує контрейнери та папки для збереження та організації об’єктів. Контрейнери та папки можуть містити довільнку кількість об’єктів.

3.6. Висновки до розділу

В даному розділі було розроблено веб-сервіс за допогою використання сучасних мов програмування та фреймворків.

Розробка відбувалась на основі архітектури MVC, яка використовувала підхід клієнт-серверної архітектури, з використанням JavaScript на клієнтській стороні та Spring на серверній стороні.

Було створено реляційну базу даних масажного салону на основі бази даних PosgreSQL для зберігання текстових даних, а також хмарне середовище Azure для зберігання об’єктів системи.

При розробці було розроблено сторінки веб-сайту, зокрема: сторінки реєстрації, авторизації, бронювання сесій, профілю користувача тощо.

Отже, можемо сказати, що розглянуті технології та інструменти дозволяють створювати веб-системи з високою функціональністю та динамічним інтерфейсом. Використання обраних бази даних та хмарного середовища дозволяє забезпечити зручне зберігання та швидке отримання даних, а використання бібліотек мов програмування та фреймворків значно полегшує процес розробки.

ВИСНОВКИ

У рамках даної кваліфікаційної роботи роботи було розроблено веб-сервіс надання послуг масажу, який спрямований на полегшення доступу до масажних послуг для користувачів і забезпечення зручного та ефективного способу бронювання масажних сеансів.

Під час виконання роботи було досліджено поняття веб-сервісу і його роль у сфері послуг, та його популярні технології, а також поняття масажу і його види. Можемо сказати, що веб-служби є невід'ємною частиною сучасного Інтернету і виконують важливу роль у сприянні взаємодії між різними програмними системами та користувачами. Однією з основних переваг веб-сервісів є їх стандартизований протокол взаємодії, який базується на веб-стандартах, таких як HTTP і XML. Це дозволяє різним системам і платформам зв'язуватися між собою незалежно від їх технологічного стеку, що робить веб-сервіси універсальними і сумісними з багатьма різними системами. Веб-сервіси також забезпечують масштабованість і розширюваність, оскільки вони можуть бути доступні для великої кількості користувачів одночасно і взаємодіяти з іншими сервісами, що робить їх корисними для побудови складних та інтегрованих систем.

Під час розробки веб-сервісів в основному використовують RESTful-сервіси, які використовують архітектурний стиль REST. Вони підтримують безстанну комунікацію між клієнтом і сервером за допомогою стандартних HTTP-запитів та надають простоту використання, ефективність та гнучкість.

У процесі розробки веб-сервісу було використано сучасні технології програмування, включаючи HTML, CSS, JavaScript для користувацьої частини, а також Spring Framework на основі мови програмування Java для серверної частини. Користувацький інтерфейс був створений з використанням зручних інструментів для розробки веб-додатків з метою забезпечення зручності інтерфейсу для користувачів, а під час створення серверної частини було використано багато бібліотек з безліччю

функціональних можливостей, які працюють за модульним принципом та надають розробнику можливість використовувати ті функції, які йому дійсно потрібні.

В останньому розділі, було представлено реалізацію розробленої веб-системи. У процесі розробки було проведено дослідження сучасних підходів до розробки веб-сервісів та визначено ключові вимоги до функціоналу сервісу. Була виконана аналітична робота для визначення основних акторів системи, їхніх потреб та взаємодій.

На основі проведених досліджень була розроблена архітектура веб-сервісу, включаючи базу даних для зберігання інформації про користувачів, масажні сесії, ролі, замовлення та послуги. Розроблено функціонал для реєстрації та авторизації користувачів, перегляду доступних масажних сеансів, бронювання сесії.

Веб-сервіс надання послуг масажу успішно пройшов тестування та демонструє задовільні результати в роботі. Він може бути використаний як основний інструмент для бронювання та отримання масажних послуг для користувачів.

Загалом, розроблений веб-сервіс надання послуг масажу має потенціал для подальшого розширення та покращення. Додаткові функції, такі як огляди клієнтів, можуть бути додані, щоб поліпшити досвід користувачів та збільшити довіру до сервісу. Крім того, можливості інтеграції з іншими сервісами, такими як календарі, можуть бути досліджені для полегшення управління часом та планування масажних сеансів.

У результаті даної кваліфікаційної роботи було успішно розроблено та впроваджено веб-сервіс надання послуг масажу, що допомагає зробити процес бронювання та отримання масажних послуг зручним та ефективним для користувачів. Цей проект може бути використаний в якості початкової точки для подальшого розвитку та вдосконалення подібних веб-сервісів у майбутньому.

СПИСОК БІБЛІОГРАФІЧНИХ ПОСИЛАНЬ ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Johnson R. Introduction to Web Services Architecture / R. Johnson. – Indianapolis, IN: Wiley Publishing, 2016. – P. 32-35.

2. Smith J. Building RESTful Web Services / J. Smith. – Sebastopol, CA: O'Reilly Media, 2017. – P. 112-115.

3. Sandoval J. RESTful Java web services: Master core REST concepts and create RESTful web services in Java. / J. Sandoval. – Birmingham, U.K : Packt Pub., 2009. – 256 p.

4. Харрісон С. Масаж як засіб стрес-релієфу: дослідження і практика / С. Харрісон. – Одеса: Видавництво "Наука і здоров'я", 2016. – 180 с.

5. Дакетт Д. HTML і CSS. Розробка та дизайн веб-сайтів / Д. Дакетт, Д. Марквардт. – Київ: Видавництво "ДУХ І ЛІТЕРА", 2018. – 480 с.

6. Макфарланд Д. HTML і CSS. Захоплююча веб-розробка / Д. Макфарланд. – Київ: Видавництво "ДУХ І ЛІТЕРА", 2019. – 512 с.

7. Макаренко Є. Каскадні таблиці стилів CSS: Посібник для початківців / Євгеній Макаренко. – Київ: Видавництво "Правильна книга", 2017. – 128 с.

8. Flanagan D. JavaScript: The Definitive Guide. 7th Edition. / D. Flanagan. – O'Reilly Media, 2012. – 259 p.

9. Еккель Б. Філософія Java / Брюс Еккель – 4-е вид. – Київ: Видавництво "Пітер Прес", 2015. – 1168 с.

10. Smith J. Java Web Services: Concepts, Architectures, and Applications / John Smith. – Addison-Wesley, 2019. – 245p.

11. Schiller F. Developing Java Web Services: Architecting and Developing Secure Web Services Using Java / Frank Schiller. – Apress, Inc., 2017. – 357 p.

12. Walls C. Spring in action / Craig Walls. – 2-ге вид. – Greenwich, CT : Manning Publications Co., 2008. – 730 с.

13. Spring Framework Documentation [Електронний ресурс] – режим доступу:

https://docs.spring.io/spring-framework/docs/current/reference/html/ (дата звернення: 30.05.2023). – Назва з екрану

14. Spring Framework – Overview [Електронний ресурс] – Режим доступу: https://www.tutorialspoint.com/spring/spring_overview.htm (дата звернення: 30.05.2023). – Назва з екрану.

15. PostgreSQL About [Електронний ресурс] – Режим доступу: https://www.postgresql.org/about/ (дата звернення: 31.05.2023). – Назва з екрану.

16. Документація по службі сховища Azure. [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/ua-ua/azure/storage/ (дата звернення: 05.06.2023). – Назва з екрану.