

Міністерство освіти і науки України
Національний університет «Києво-Могилянська Академія»
Кафедра мультимедійних систем

КУРСОВА РОБОТА

за спеціальністю «Інженерія програмного забезпечення»
на тему: «Дослідження можливостей фреймворку Spring на прикладі
розробки веб-платформи для пошуку екскурсій»

Науковий керівник:

ст. викладач Борозенний С. О.

Виконала:

студентка 3-го курсу Чернова Т. А.

Київ – 2019

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА
АКАДЕМІЯ»

Кафедра мультимедійних систем

ЗАТВЕРДЖУЮ

Зав. кафедри мультимедійних
систем, доцент, к.ф.-м.н.

_____ О.П. Жежерун

“ ____ ” _____ 2020 р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу

студентці _____ Черновій Тетяні _____ 3-го курсу факультету інформатики

ТЕМА Дослідження можливостей фреймворку Spring на прикладі розробки
веб-платформи для пошуку екскурсій

Зміст ТЧ до курсової роботи

Анотація

Вступ

1. Дослідження та аналіз предметної області
2. Spring Framework
3. Опис засобів програмування
4. Структура застосунку

Висновки

Список використаної літератури

Дата видачі “15” жовтня 2020 р. Керівник Борозенний С. О.

Завдання отримала _____

Календарний план виконання курсової роботи

Тема: Дослідження можливостей фреймворку Spring на прикладі розробки веб-платформи для пошуку екскурсій

№ п/п	Назва етапу дипломного проекту (роботи)	Термін виконання етапу	Примітка
1.	Отримання завдання на курсову роботу.	жовтень 2019р.	
2.	Огляд технічної літератури за темою роботи.	листопад – грудень 2019р.	
3.	Аналіз архітектури програм, розроблених на основі Spring Framework.	грудень – січень 2019р	
4.	Проектування архітектури застосунку.	січень 2020р.	
5.	Розробка застосунку.	лютий-березень 2020 р.	
6.	Написання текстової частини роботи.	квітень 2020р.	
7.	Створення доповіді.	квітень 2020р.	
8.	Захист курсової роботи	травень 2020р.	

Студент Чернова Т. А.

Керівник Борозенний С. О.

“ ” р.

Зміст

<i>Анотація</i>	<i>5</i>
<i>Вступ.....</i>	<i>6</i>
<i>1 Дослідження та аналіз предметної області</i>	<i>7</i>
<i>1.1 Опис предметної області</i>	<i>7</i>
<i>1.2 Опис груп користувачів.....</i>	<i>7</i>
<i>1.3 Функціональні вимоги до додатку</i>	<i>8</i>
<i>2 Spring Framework</i>	<i>10</i>
<i>2.1 Загальні відомості.....</i>	<i>10</i>
<i>2.2 Використання Spring Security.....</i>	<i>12</i>
<i>2.3 Використання Spring Boot</i>	<i>12</i>
<i>2.4 Використання Spring MVC</i>	<i>13</i>
<i>2.5 Використання Spring Data.....</i>	<i>15</i>
<i>3 Опис засобів програмування</i>	<i>17</i>
<i>3.1 Обрана СКБД.....</i>	<i>17</i>
<i>3.2 Обрані технології, середовища, мови</i>	<i>18</i>
<i>4 Структура застосунку</i>	<i>21</i>
<i>4.1 Інтерфейс користувача</i>	<i>21</i>
<i>Висновок.....</i>	<i>34</i>
<i>Список використаної літератури.....</i>	<i>35</i>

Анотація

Курсова робота присвячена дослідженню можливостей фреймворку Spring на основі розробки веб-платформи для пошуку екскурсій. В ході розробки веб-платформи було досліджено особливості побудови веб-застосунків з використанням наступних технологій: Java, Spring, Spring Boot, Spring Security, Hibernate, FreeMarker, Spring MVC та СКБД PostgreSQL.

Основні можливості застосунку: пошук вигідних екскурсій, перегляд профілів екскурсовода та туриста, бронювання екскурсій, підтвердження або скасування бронювань, зміна профілю, створення нових екскурсій, редагування вже існуючих авторських екскурсій.

Ключові слова: Java, Web, Spring, Spring Boot, Spring MVC, Spring Security, Hibernate, Spring Data, FreeMarker, Maven, MVC, Spring MVC, СКБД PostgreSQL.

Вступ

На сьогоднішній день розробка веб-застосунків стає все більше популярною. Вона має своє місце майже у всіх сферах людського життя. Веб-застосунок покращує та автоматизує бізнес-процеси, допомагає компаніям збільшити свою аудиторію та налагодити комунікацію з клієнтами. Інколи веб-застосунки можуть виступати у ролі систем для організації та налагодження внутрішньої роботи підприємства.

Для покращеної взаємодії користувача та системи, розроблений застосунок мусить відповідати ряду вимог: дружність, доступність, простота та зрозумілість в користуванні, швидкість роботи, візуальна привабливість та можливості, що він може надати своєму користувачу. Зрозуміло, що не всі застосунки відповідають даним вимогам. Тому при першому знайомстві користувача з додатком, компанія може втратити свого потенційного клієнта.

Варто також зазначити про інформативність застосунку. Він повинен бути максимально спрямованим на запити користувача та спроектованим так, щоб користувач за максимально короткий проміжок часу міг віднайти всю потрібну інформацію в одній платформі, без використання допоміжних ресурсів.

Я обрала тему, що пов'язана з дослідженням сучасного фреймворку Spring для побудови прогресивного та зручного веб-застосунку.

Метою моєї курсової роботи є аналіз можливостей, що надає Spring Framework разом з набором інших допоміжних технологій, які я використала при розробці веб-платформи для пошуку екскурсій. В процесі дослідження були також розглянуті основні аспекти розробки повноцінного веб-застосунку.

1 Дослідження та аналіз предметної області

1.1 Опис предметної області

Представлена платформа призначена для швидкого пошуку екскурсій, їх бронювання та полегшення комунікації між екскурсоводом і туристом.

Метою даного проекту є об'єднання екскурсоводів та туристів. Для перших це зручна платформа для розміщення авторських екскурсій, для других – можливість максимально швидко знайти екскурсію, що відповідає бажанням та потребам самого туриста.

1.2 Опис груп користувачів

Користувачами даного застосунку є турист та екскурсовод.

- Турист

Турист має змогу здійснювати пошук екскурсій. Він має можливість знайти екскурсію за містом, перейти на сторінку екскурсії, що його зацікавила та отримати всю потрібну інформацію, що стосується даної екскурсії. Також турист може переглянути профіль екскурсовода та отримати його персональну інформацію. Якщо туриста зацікавила якась певна екскурсія, він може забронювати її, після чого отримує лист на пошту про подальші дії.

- Екскурсовод

Екскурсовод має змогу розміщувати свої авторські екскурсії на платформі, змінювати інформацію про екскурсію за потребою, переглядати свої бронювання та підтверджувати або ж скасовувати їх, надавши при цьому короткий коментар щодо своїх дій. Також екскурсовод має можливість переглядати профіль туриста, який здійснив бронювання екскурсії. У випадку підтвердження екскурсії,

екскурсовод може зв'язатись з туристом через електронну пошту та обговорити з ним деталі бронювання.

1.3 Функціональні вимоги до додатку

Спільні:

- a) створення профілю в системі використовуючи електронну пошту та пароль;
- b) здійснення реєстрації з надсиланням активаційного посилання на пошту;
- c) авторизація в системі;
- d) можливість редагувати свій профіль;
- e) вихід з системи.

Для туриста:

- a) перегляд профілю екскурсовода та отримання детальної інформації про нього: електронна пошта, дата народження, телефон, кількість заброньованих екскурсій, національність та місце проживання екскурсовода, актуальні екскурсії, персональні екскурсії та найбільш відвідувані екскурсії;
- b) пошук екскурсій за містом;
- c) перегляд екскурсії та отримання всієї інформації про дану екскурсію: інформація про екскурсовода, тривалість та місце проведення екскурсії, детальний опис екскурсії, максимальна кількість відвідувачів, вартість екскурсії для однієї особи;
- d) можливість бронювання екскурсії з доданням бажаного часу її проведення та кількості осіб, що планують відвідати дану екскурсію.

Для екскурсовода:

- a) створення авторської екскурсії;

- b) можливість редагувати авторські екскурсії;
- c) можливість видалити авторську екскурсію;
- d) перегляд заявок на бронювання екскурсій;
- e) перегляд профілів туристів, що здійснили бронювання та отримання їх контактних даних: електронна пошта, дата народження, телефон, кількість заброньованих екскурсій, національність та місце проживання;
- f) можливість переглядати не підтвердженні бронювання, минулі бронювання, майбутні бронювання та пропущені бронювання;
- g) можливість підтвердити або скасувати бронювання з доданням коментаря, що буде відправлений туристу на електронну пошту.

2 Spring Framework

2.1 Загальні відомості

Даний застосунок був реалізований на основі сучасного фреймворку Spring. Spring framework на сьогодні є чи не найпопулярнішим засобом для розробки Java застосунків. Він дозволяє спростити створення та керування програмним забезпеченням, зокрема автоматизувати більшість стандартних процесів пов'язаних з розробкою [1]. Але водночас Spring framework надає розробнику широку функціональність: можливість обробляти HTTP запити, взаємодіяти з базою даних, впровадити security (засоби авторизації та автентифікації) та виконувати певну бізнес-логіку, передбачену корпоративними вимогами системи.

Для того щоб досягти простоти розробки, Spring використовує основні концепції:

- Plain Old Java Objects (POJOs) – «Старі добрі об'єкти Java»;
- Dependency Injection/Inversion of Control (IoC) – Впровадження залежностей та інверсія управлінням;
- Convention over configuration – принцип програмування «угода головніша за конфігурацію»;
- Modular Architecture – модульна архітектура.

Plain Old Java Objects (POJOs) – це звичайні об'єкти Java, які не містять в собі ніяких залежностей, не імплементують жодного інтерфейсу та не наслідуються від жодного іншого класу. Перевага такого підходу полягає у тому, що при зміні якоїсь частини програми, на інші частини ці зміни не будуть поширюватись. Spring дбає про з'єднання всіх частин програми,

проте окремі компоненти не знають про роботу інших компонентів застосунку та є максимально ізольованими від них. Саме для зв'язку всіх компонентів Spring використовує підходи Dependency Injection/Inversion of Control (IoC). Підхід полягає у тому, щоб передати управління об'єктами або ж окремими частинами програми в контейнер або середовищу. Це значно полегшує роботу застосунку, дозволяє легко управляти потоком, відділяти задачі від реалізації та досягати ізольованості окремих компонентів програми [1].

У Spring існує контейнер Spring IoC, що реалізує принцип інверсії управлінням. В середовищі Spring контейнер Spring IoC представлений у вигляді інтерфейсу `ApplicationContext`. Цей контейнер відповідає за з'єднання, створення, налаштування та збірку об'єктів – в Spring відомих як `beans` – а також за управління життєвим циклом програми. Імплементацією IoC в Spring є `Dependency Injection`. За допомогою цього патерну, фреймворк самостійно створює об'єкт та впроваджує залежності з цим об'єктом. Для збірки компонентів Spring контейнер використовує метадані конфігурації, що можуть бути у вигляді конфігурації XML файлу або ж анотацій. Я використовувала анотації, оскільки це більше просунутий та простіший спосіб досягти конфігурації метаданих для подальшого впровадження принципу інверсії управління. До прикладу замість звичайного синтаксису створення нового об'єкту: `new Object()`, Spring дозволяє використати відповідну анотацію `@Resource`, це означатиме, що фреймворк самостійно впровадить залежність даного об'єкту у вказаному класі.

Для того щоб додати будь-який клас до Spring контексту, достатньо вказати над ним відповідну анотацію. Ось декілька прикладів поширених анотацій, що я використовувала при розробці свого застосунку:

- `@Component` — це стандартна анотація, що може бути додана для будь-якого класу.
- `@Repository` — використовується для класів, що відповідають за доступ до бази даних – DAO – Data Access Object.
- `@Service` — використовується для класів рівня сервісу, що відповідають за бізнес логіку та зв'язок між UI та DAO.
- `@Controller` — використовується для класів, що обробляють клієнтські запити.

Основна ідея Spring полягає в її модульній архітектурі. Spring намагається модулізувати весь фреймворк у менші шматочки – модулі – кожен з яких відповідає за свій аспект в реалізації застосунку. Завдяки чому це досить зручний засіб для розробки великомасштабних застосунків.

2.2 Використання Spring Security

Spring Security - це потужний високоефективний фреймворк, що забезпечує механізм побудови системи автентифікації та авторизації для застосунків, створених за допомогою Spring Framework[2].

Spring Security може бути легко доповнений допоміжним функціоналом. Окрім комплексної підтримки автентифікації та авторизації, він захищає від поширених атак таких як фіксація сесії, клікджекінг, міжсайтова підробка запиту.

2.3 Використання Spring Boot

Для полегшеного налаштування Spring-проекту я використовувала фреймворк Spring Boot. Він дозволяє пришвидшити процеси створення та розгортання проекту розробленого за допомогою на Spring Framework. За

допомогою нього можна найбільш простим способом створювати web-застосування без додаткових зусиль для налаштування самого проєкту.

Spring Boot надає ряд можливостей для полегшення процесів налаштування створення та розвиток проєкту. Його найбільш значимими можливостями є управління залежностями, автоматична конфігурація та вбудовані контейнери сервлетів [3].

Для того щоб полегшити розробнику процес додаткових залежностей в проєкт, Spring Boot неявно упаковує всі зовнішні залежності в так звані starter-пакети, що являють собою дескриптори залежностей та які легко підключаються до проєкту. Це дозволяє зробити процес підключення додаткових залежностей до проєкту універсальним та позбавити програміста від пошуку та підключення окремих дескрипторів [3].

Після вибору потрібних starter-пакетів, Spring Boot автоматично налаштовує застосунок на основі добавлених jar-залежностей. Незважаючи на це, автоматична конфігурація може бути змінена в будь-який момент часу за допомогою користувацьких налаштувань.

2.4 Використання Spring MVC

Розробляючи даний web-застосунок, я використовувала патерн програмування MVC. При використанні даного шаблону застосунок розділяється на три рівні: модель даних (Model), користувацький інтерфейс (View) та рівень логіки задля взаємодії користувача з системою (Controller). Завдяки такому розділенню модифікація одного з трьох компонентів лише мінімально або зовсім не впливає на інші два рівні.

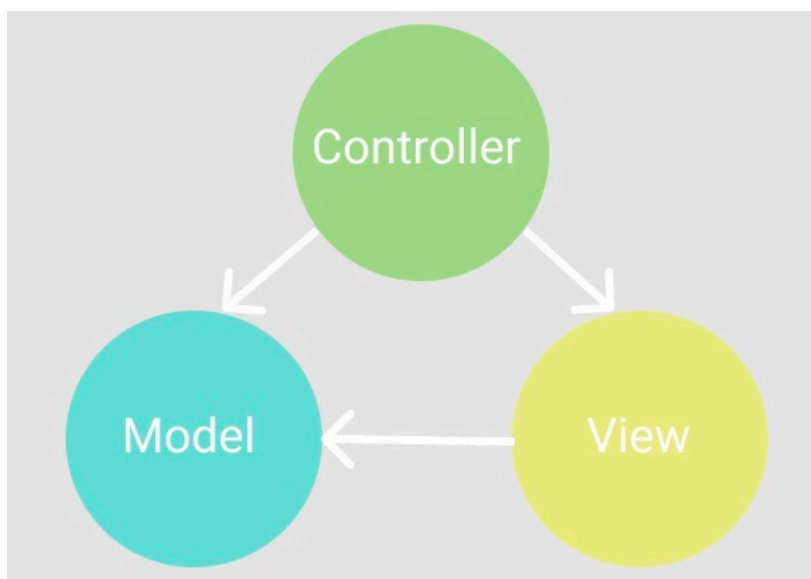


Рисунок 1- Патерн «Model-View-Controller»

Фреймворк Spring MVC забезпечує архітектуру даного патерна Model - View – Controller за допомогою слабо зв'язаних готових компонентів.

- Model об'єднує дані застосунку, що складається з Java-об'єктів.
- View відповідає за відображення даних: зазвичай генеруючи HTML.
- Controller обробляє запити користувача, створює відповідну Model та передає її для відображення у View [4].

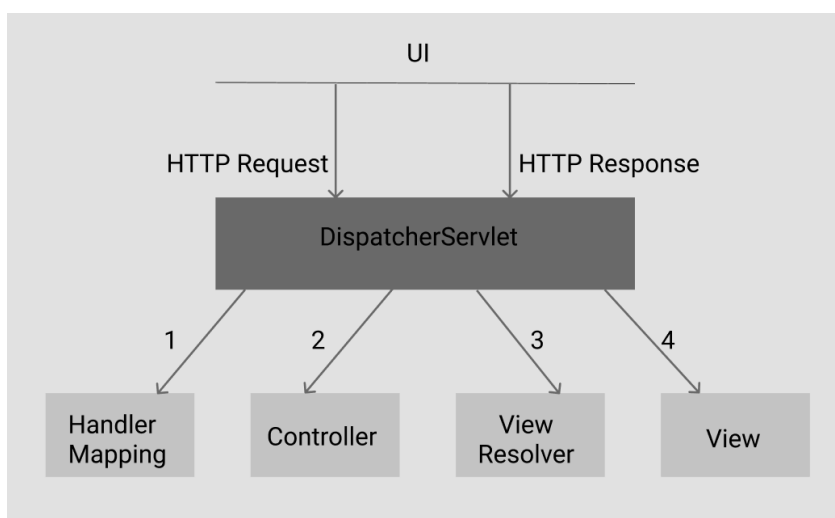


Рисунок 2- Робота Spring MVC

Принцип роботи Spring MVC:

Вся робота Spring MVC зосереджена навколо одного компоненту – DispatcherServlet. Він отримує HTTP запити від клієнта, обробляє їх та передає результат.

Після отримання HTTP запиту від клієнта, DispatcherServlet звертається до інтерфейсу HandlerMapping. Даний інтерфейс визначає Controller, якому буде переданий вхідний запит. У свою чергу Controller приймає запит та викликає відповідний метод для обробки запиту. В ході обробки запиту контроллером, визначаються дані Моделі (приймають вигляд ключ-значення), що будуть передані у відповідь та ім'я View, що повернеться контроллером. Після цього контроллер звертається до інтерфейсу View Resolver. View Resolver визначає за іменем отриманим від контроллера, який сам View повинен бути переданий на клієнтську частину. Після створення відповідного View, DispatcherServlet поміщає дані Моделі у вигляді атрибутів у View та передає його на клієнтську частину для відображення у браузері. Всі вищеперераховані компоненти є частинами WebApplicationContext з деякими особливостями для створення веб-додатків.

2.5 Використання Spring Data

Звернення до бази даних у моєму застосунку відбувається за допомогою Spring Data JPA та Hibernate. Spring Data Jpa – частина Spring Framework, що спрощує та реалізує рівень доступу до даних. Реалізація шаблонного коду для написання рівня доступу до даних може бути надто громіздким. Натомість Spring Data Jpa значно спрощує та покращує реалізацію даного шару.

Ніibernate – це популярний фреймворк, що забезпечує зв'язність та відображення даних, що лежать у базі даних, доменним java- об'єктам та навпаки.

Розробнику достатньо написати інтерфейс репозиторію, що буде містити методи пошуку, читання та модифікації об'єктів, що містяться в базі даних, а Spring забезпечить їх автоматичну реалізацію.

3 Опис засобів програмування

3.1 Обрана СКБД

Для розробки АІС була обрана СКБД PostgreSQL. Такий вибір був зумовлений тим, що PostgreSQL є некомерційною, гарно документованою та широко функціональною системою керування реляційними базами даних. PostgreSQL повністю задовольняє потреби розробленого застосунку на даному етапі і водночас надає великий запас можливостей для росту і розвитку проекту в цілому та архітектури бази даних зокрема.

Серед використаних переваг обраної СКБД можна навести наступне:

- підтримка найпопулярніших операційних систем (Windows, Mac, Linux);
- високий рівень продуктивності (performance);
- широкий набір стандартних типів;
- наявний механізм підтримки цілісності посилань;
- підтримка SQL sequence;
- можливість створювати рекурсивні запити.

Серед переваг PostgreSQL, що можуть бути використані в майбутньому є:

- об'єктна-реляційність. Тобто в PostgreSQL розробник може створювати нестандартні типи даних, зберігати дані у вигляді масиву, організовувати ієрархію наслідування на рівні таблиць тощо;
- необмежений максимальний розмір пам'яті, що може займати база даних;
- використання функцій.

За допомогою СКБД PostgreSQL була розроблена така схема даних:

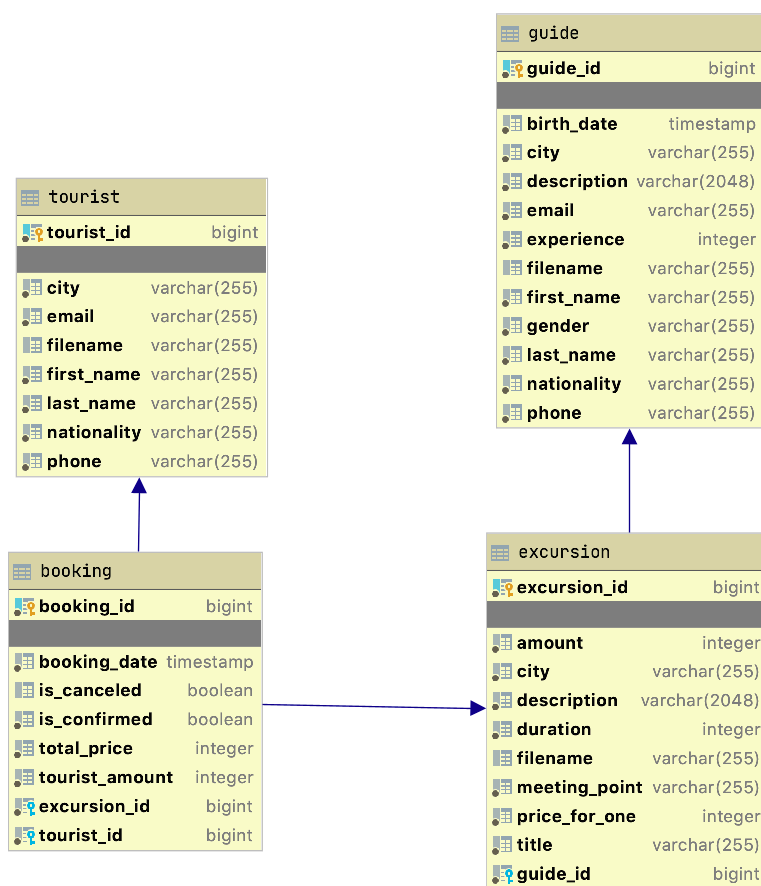


Рисунок 3 - Схема БД

3.2 Обрані технології, середовища, мови

- Серверна частина

Даний застосунок був розроблений за допомогою об'єктно-орієнтованої мови програмування Java та потужного фреймворку для розробки великомасштабних мережевих застосунків на Java – Spring. Такий вибір мови для розробки застосунку можна обґрунтувати тим, що Java є строго об'єктно-орієнтованою мовою програмування зі зручним та очевидним способом представлення моделей об'єктів реального світу в програмі. Більше того, Java дозволяє створювати крос-платформенні

застосунки, тому програму можна легко запустити як локально (необхідно лише встановити JVM – віртуальну машину Java), так і на віддаленому сервері. Java дозволяє у досить легкий спосіб представляти предмети реального світу у вигляді об'єктів в програмі та надавати їм властивості та поведінку.

В свою чергу Spring дозволяє спростити створення та керування програмним забезпеченням заснованим на мові програмування Java. За рахунок DI – Dependency Injection, що лежить в основі Spring Core, можна створювати менше коду для зв'язності окремих компонентів. Spring надає можливість легкого конфігурування застосунку: для впровадження залежностей між різними компонентами достатньо лише налаштувати XML або ж додати відповідні анотації. Це також дозволяє у дуже зручний спосіб керувати конфігурацією застосунку, оскільки вся інформація про зовнішні залежності містяться в одному репозиторії.

Даний застосунок має три-рівневу архітектуру:

Controller -> Service -> Repository.

На рівні контролерів запрограмовані ендпоінти (точки входу HTTP запитів для взаємодії з ресурсами). Controller приймає дані та запити користувача, віддає їх на обробку на рівень сервісів, в результаті отримує оброблені дані та відправляє їх на клієнтську частину.

Рівень сервісів призначений для взаємодії рівнів Controller та Repository. Він обробляє дані згідно з прописаною бізнес-логікою. Сервіс звертається до рівня Repository, щоб отримати дані з бази або модифікувати їх певним чином та передає їх як результат певного клієнтського запиту контроллеру.

Рівень Repository призначений виключно для роботи з базою даних, зокрема для виконання певних SQL запитів, отримання результатів,

перетворення результатів запиту в Java-об'єкт та передачі його на рівень сервісу.

- Клієнтська частина

На рівні View (View Layer) Web MVC застосунку я використовувала потужний та популярний шаблонізатор (Template engine) FreeMarker. Він досить зручний у використанні, оскільки дозволяє читати шаблонні файли, додавати в них певну логіку, поєднувати з Java об'єктами та генерувати готовий HTML код. Прості вирази шаблонізатора підтримують різноманітні обчислення, операції порівняння, умови, списки, цикли, внутрішньо влаштовані функції та макроси.

Freemarker підтримує макроси. Макрос – це фрагмент шаблону, визначений користувачем, що зазвичай використовується для уникнення повторюваності коду [5].

Варто зазначити, що FreeMarker підтримується фреймворком Spring, оскільки в Spring є вже підготовлений компонент конфігурації для даного Template Engine.

4 Структура застосунку

4.1 Інтерфейс користувача

Інтерфейс прикладної програми було створено за допомогою мови розмітки HTML, каскадних таблиць стилів (CSS) та шаблонізатора FreeMarker. Для більш зрозумілості інтерфейсу та його дружності з користувачем застосунку була обрана приємна кольорова схема та максимально зручний перехід між сторінками. Кожна сторінка виконує свою основну функцію та надає всю можливу інформацію, що потрібна користувачу. Задля утримання уваги користувача, розроблені максимально зрозумілі вказівки та переходи на кожній сторінці. Розглянемо деякі з основних сторінок застосунку:

Головна сторінка веб-застосунку виглядає наступним чином:

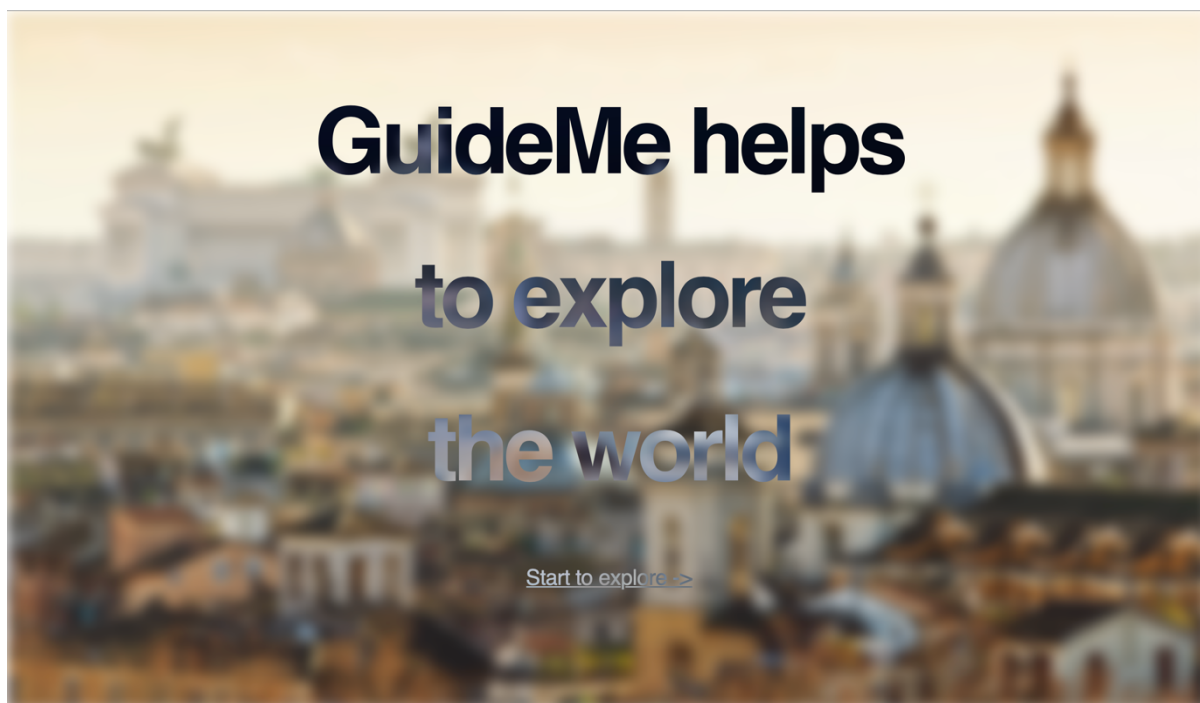


Рисунок 4.1.1 – Головна сторінка застосунку

При переході за «Start to explore», користувач потрапляє на сторінку «Login».

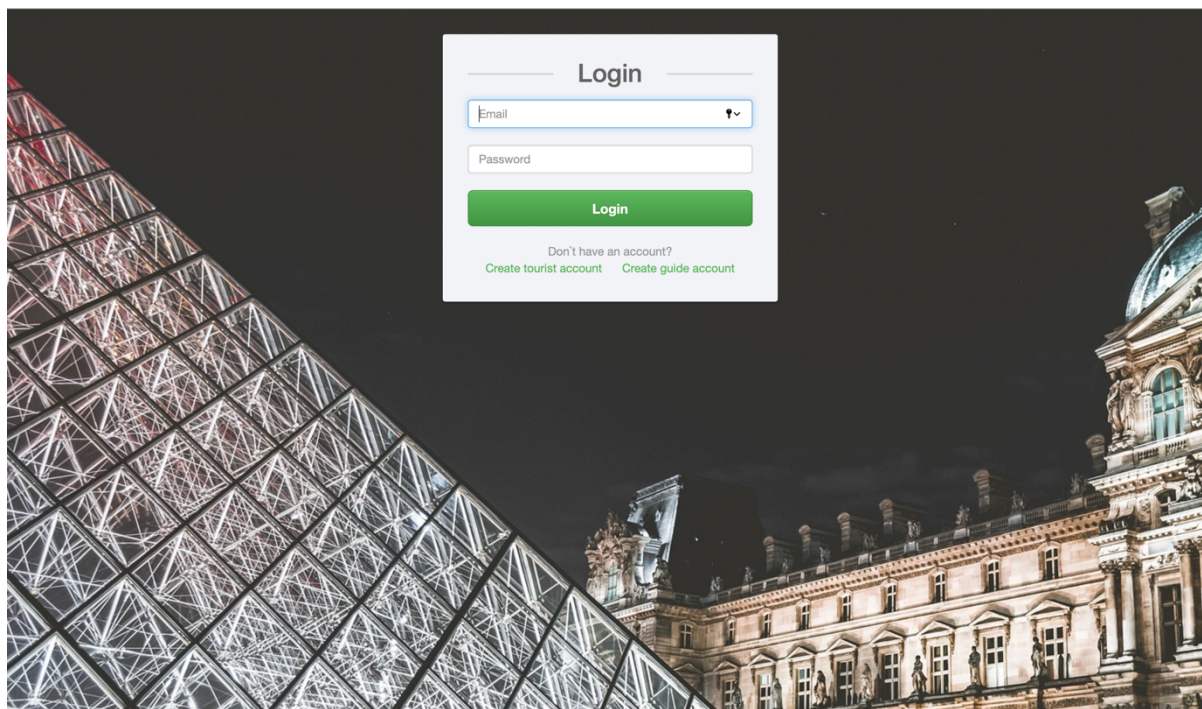
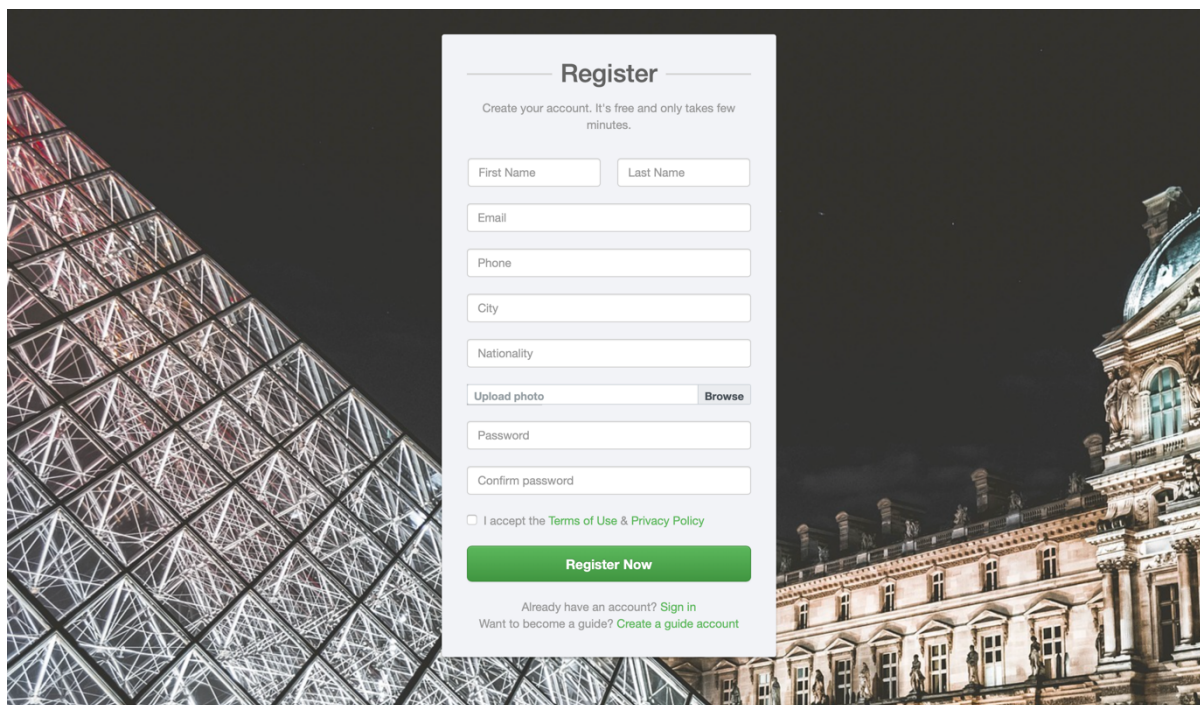


Рисунок 4.1.2 – Сторінка «Login»

Зі попередньої сторінки не зареєстрованим користувачам є можливість перейти на реєстраційну сторінку відповідно до ролі користувача в системі: турист або екскурсовод.



Register

Create your account. It's free and only takes few minutes.

First Name Last Name

Email

Phone

City

Nationality

Upload photo [Browse](#)

Password

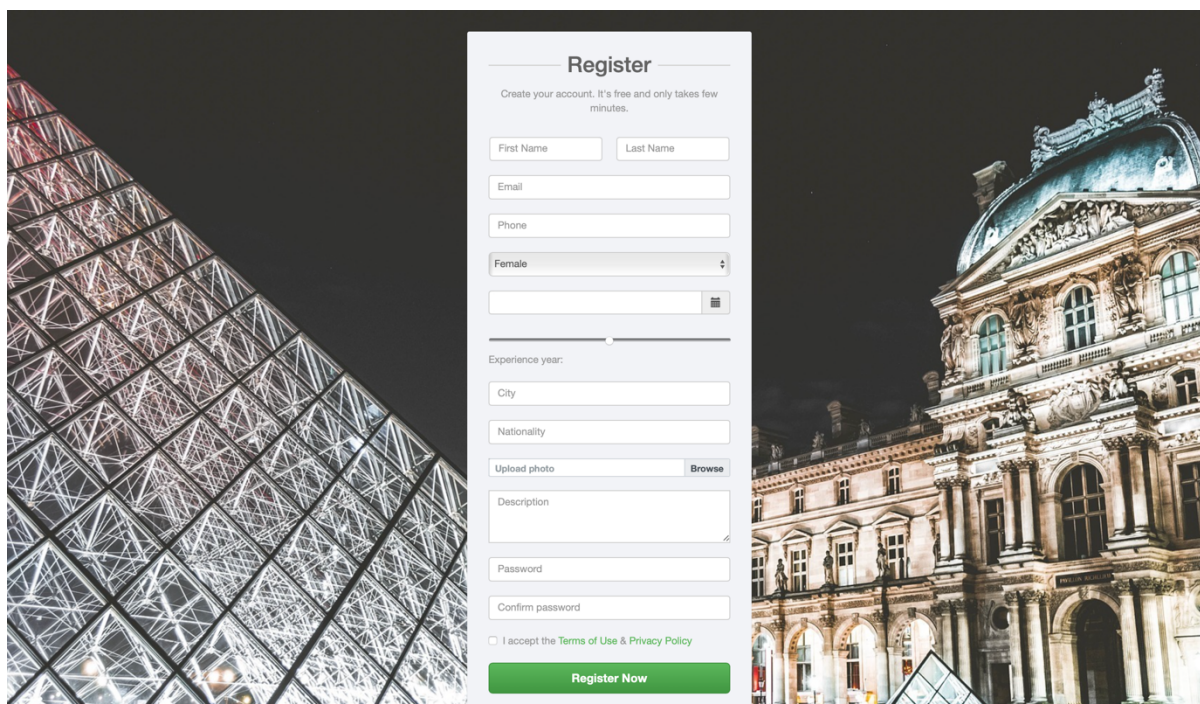
Confirm password

☐ I accept the [Terms of Use & Privacy Policy](#)

[Register Now](#)

Already have an account? [Sign in](#)
Want to become a guide? [Create a guide account](#)

Рисунок 4.1.3 – Реєстрація туриста



Register

Create your account. It's free and only takes few minutes.

First Name Last Name

Email

Phone

Female

Experience year:

City

Nationality

Upload photo [Browse](#)

Description

Password

Confirm password

☐ I accept the [Terms of Use & Privacy Policy](#)

[Register Now](#)

Рисунок 4.1.4 – Реєстрація екскурсовода

Варто зазначити, що реєстрація користувача системи здійснюється з надсиланням коду підтвердження на пошту. Якщо користувач перейде за

надісланим посиланням, та увійде у систему, він автоматично стане активним користувачем застосунку.

Після реєстрації користувач потрапляє на домашню сторінку:

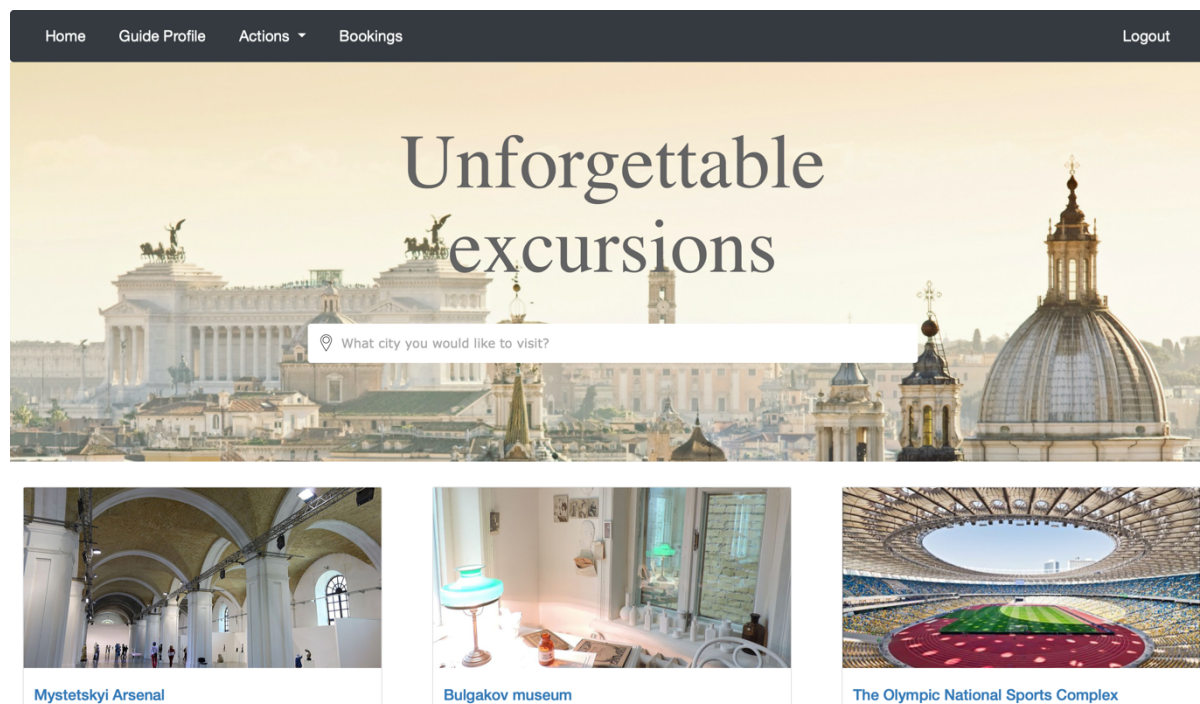


Рисунок 4.1.5 – Домашня сторінка

На даній сторінці представлені усі екскурсії та можливість фільтрації екскурсій за містом. У верхній частині сторінки знаходиться меню, що надає можливість перейти на свій профіль, виконати дії відповідно до ролі користувача (турист може змінити свій профіль, екскурсовод може змінити свій профіль та створити нову авторську екскурсію). Також користувачам надається можливість вийти з системи.

Головна сторінка містить так звані карточки екскурсій, що надають користувачу мінімальну важливу інформацію про кожну екскурсію. При натисканні на екскурсійну картку, користувач має змогу перейти на сторінку екскурсії та отримати розгорнуту інформацію про неї. Також у розпорядженні користувача можливість переглянути сторінку

екскурсовода при натисканні на фото екскурсовода, що також знаходиться на екскурсійній картці.



Рисунок 4.1.6 – Екскурсійна картка

Сторінка екскурсії виглядає наступним чином:

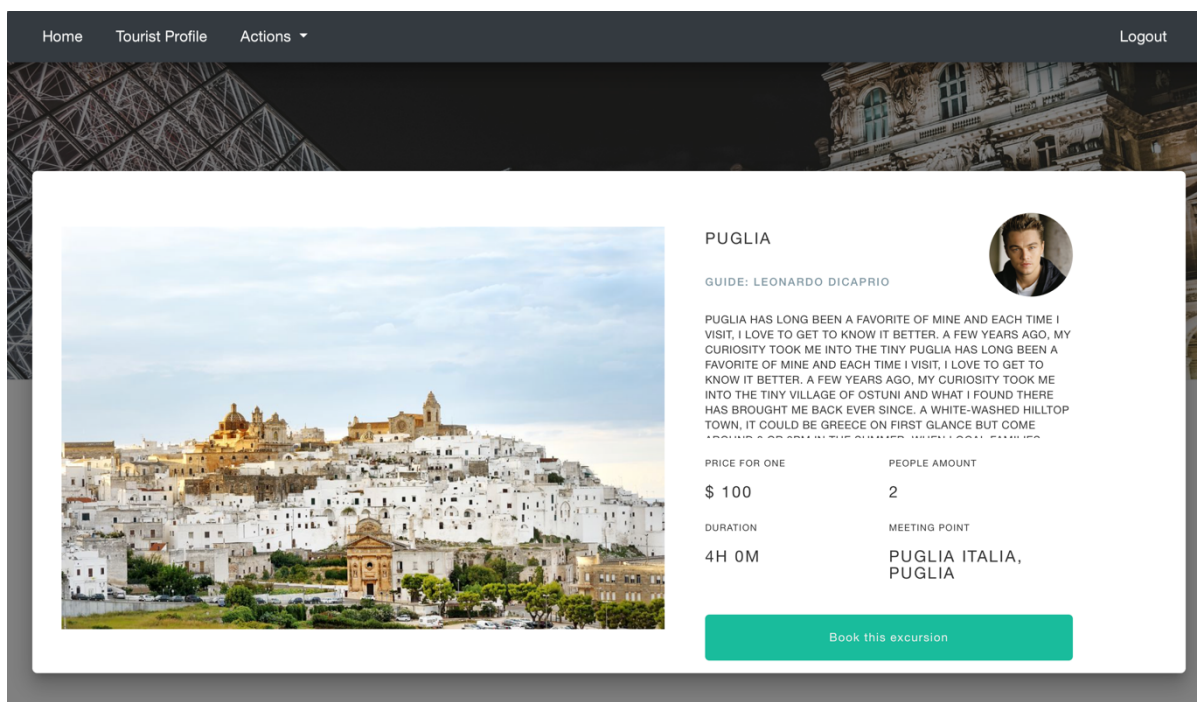


Рисунок 4.1.7 – Сторінка екскурсії (для туриста)

Користувачу надається розгорнута інформація про екскурсію: час та місце проведення, ціна за одну особу, тривалість, детальний та розгорнутий опис екскурсії, екскурсійне фото, прізвище та ім'я екскурсовода. Користувач також має змогу перейти на сторінку екскурсовода при натисканні на фото. У нижній частині сторінки розташована кнопка бронювання екскурсії. Вона переводить користувача на сторінку бронювання:

Book the excursion

People amount

05/03/2020

May 2020

Su	Mo	Tu	We	Th	Fr	Sa
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Close Confirm

GUIDE: LEONARDO DICAPRIO

PUGLIA HAS LONG BEEN A FAVORITE OF MINE AND EACH TIME I VISIT, I LOVE TO GET TO KNOW IT BETTER. A FEW YEARS AGO, MY CURIOSITY TOOK ME INTO THE TINY PUGLIA HAS LONG BEEN A FAVORITE OF MINE AND EACH TIME I VISIT, I LOVE TO GET TO KNOW IT BETTER. A FEW YEARS AGO, MY CURIOSITY TOOK ME INTO THE TINY VILLAGE OF OSTUNI AND WHAT I FOUND THERE HAS BROUGHT ME BACK EVER SINCE. A WHITE-WASHED HILLTOP TOWN, IT COULD BE GREECE ON FIRST GLANCE BUT COME

Рисунок 4.1.8 – Бронювання екскурсії

Користувач має змогу забронювати екскурсію вказавши кількість осіб, що бажають відвідати екскурсію та зазначити дату проведення екскурсії. Дата екскурсії можлива починаючи з поточного числа.

Home Guide Profile Actions Bookings Logout

PUGLIA

GUIDE: LEONARDO DICAPRIO

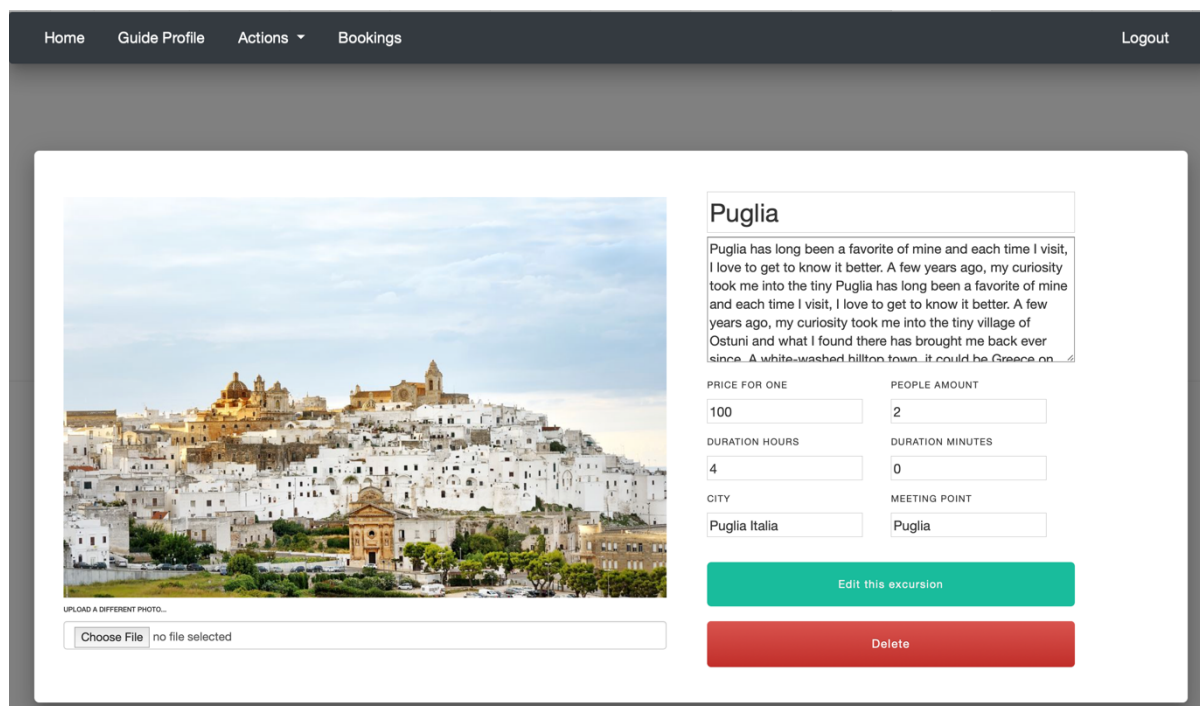
PUGLIA HAS LONG BEEN A FAVORITE OF MINE AND EACH TIME I VISIT, I LOVE TO GET TO KNOW IT BETTER. A FEW YEARS AGO, MY CURIOSITY TOOK ME INTO THE TINY PUGLIA HAS LONG BEEN A FAVORITE OF MINE AND EACH TIME I VISIT, I LOVE TO GET TO KNOW IT BETTER. A FEW YEARS AGO, MY CURIOSITY TOOK ME INTO THE TINY VILLAGE OF OSTUNI AND WHAT I FOUND THERE HAS BROUGHT ME BACK EVER SINCE. A WHITE-WASHED HILLTOP TOWN, IT COULD BE GREECE ON FIRST GLANCE BUT COME

PRICE FOR ONE	PEOPLE AMOUNT
\$ 100	2
DURATION	MEETING POINT
4H 0M	PUGLIA ITALIA, PUGLIA

Edit this excursion

Рисунок 4.1.9 – Сторінка екскурсії (для екскурсовода)

Екскурсовод має змогу змінити екскурсію (тільки у випадку, якщо ця екскурсія його авторська). При натисканні на кнопку «Edit this excursion», екскурсовод переходить на сторінку редагування екскурсії:



The screenshot shows a web interface for editing an excursion. At the top, there is a navigation bar with links: Home, Guide Profile, Actions (dropdown), Bookings, and Logout. The main content area is titled 'Puglia'. It features a large photo of a white-washed hilltop town with a church. Below the photo is a text area with a description of Puglia. To the right of the photo and text are several input fields for editing the excursion details:

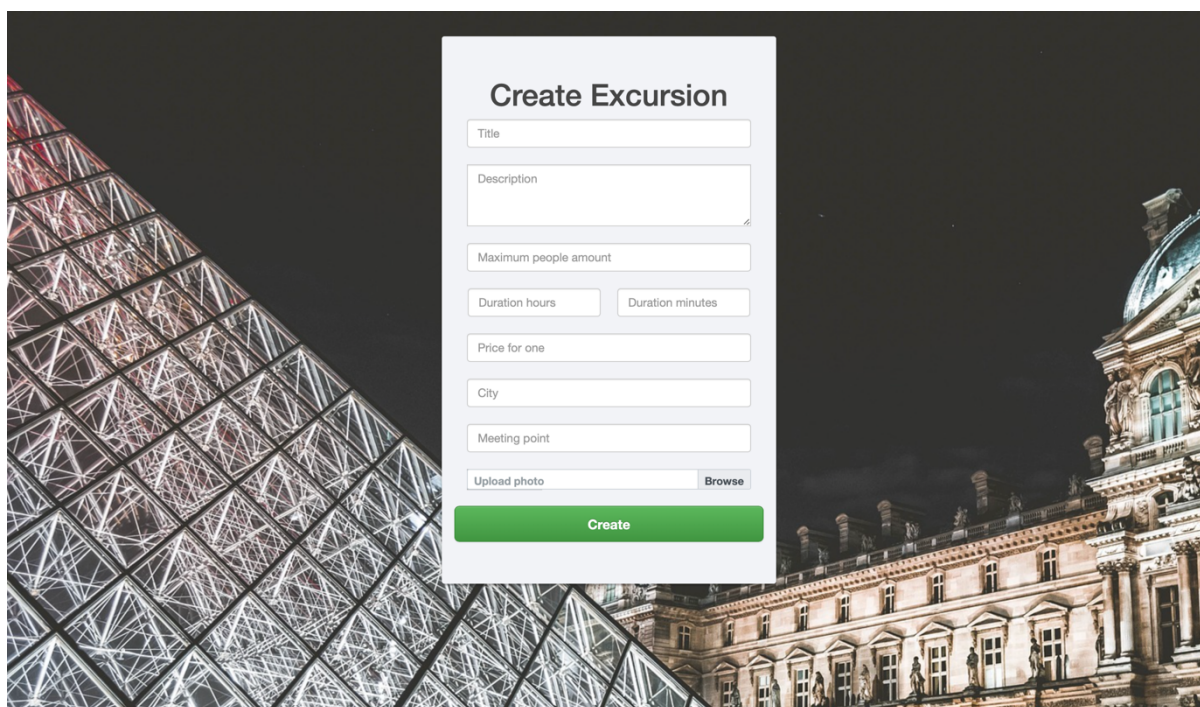
- PRICE FOR ONE: 100
- PEOPLE AMOUNT: 2
- DURATION HOURS: 4
- DURATION MINUTES: 0
- CITY: Puglia Italia
- MEETING POINT: Puglia

At the bottom of the form, there are two buttons: 'Edit this excursion' (green) and 'Delete' (red). Below the photo, there is a section for uploading a different photo, with a 'Choose File' button and the text 'no file selected'.

Рисунок 4.1.10 – Сторінка редагування екскурсії (для екскурсовода)

Екскурсовод може вносити зміни до поточної екскурсії. У випадку успішного редагування, користувач системи потрапляє на сторінку екскурсії, де будуть відображені щойно змінені дані. Якщо ж при внесенні змін екскурсовод допустив помилки у формі – екскурсія не буде редагована, при цьому екскурсовод отримає повідомлення про помилки у заповненні форми редагування та всі дані екскурсії залишаться без змін.

Сторінка створення нової авторської екскурсії виглядає наступним чином:

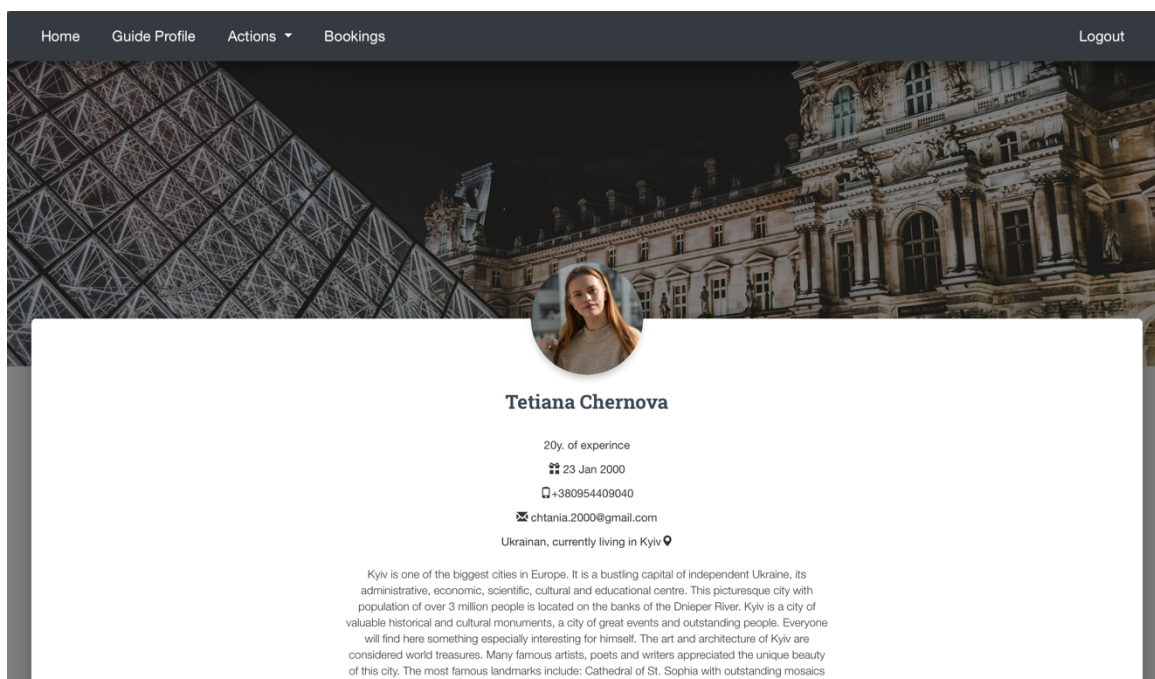


Create Excursion


Рисунок 4.1.11 – Сторінка створення екскурсії (для екскурсовода)

Після створення екскурсії, екскурсовод потрапляє на власний профіль, де буде відображена щойно створена екскурсія.

Профіль екскурсовода містить наступну інформацію:



[Home](#)
[Guide Profile](#)
[Actions](#)
[Bookings](#)
[Logout](#)



Tetiana Chernova

20y. of experience

📅 23 Jan 2000

📞 +380954409040

✉️ chtania.2000@gmail.com

Ukrainian, currently living in Kyiv 📍

Kyiv is one of the biggest cities in Europe. It is a bustling capital of independent Ukraine, its administrative, economic, scientific, cultural and educational centre. This picturesque city with population of over 3 million people is located on the banks of the Dnieper River. Kyiv is a city of valuable historical and cultural monuments, a city of great events and outstanding people. Everyone will find here something especially interesting for himself. The art and architecture of Kyiv are considered world treasures. Many famous artists, poets and writers appreciated the unique beauty of this city. The most famous landmarks include: Cathedral of St. Sophia with outstanding mosaics

Рисунок 4.1.12 – Профіль екскурсовода

Tetiana Chernova

20y. of experience

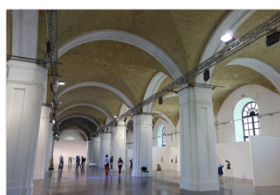
📅 23 Jan 2000

📞 +380954409040

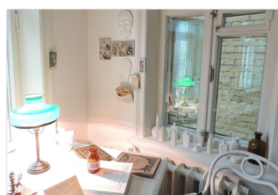
✉️ chtania.2000@gmail.com

🇺🇦 Ukrainian, currently living in Kyiv 📍

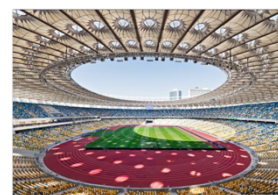
Kyiv is one of the biggest cities in Europe. It is a bustling capital of independent Ukraine, its administrative, economic, scientific, cultural and educational centre. This picturesque city with population of over 3 million people is located on the banks of the Dnieper River. Kyiv is a city of valuable historical and cultural monuments, a city of great events and outstanding people. Everyone will find here something especially interesting for himself. The art and architecture of Kyiv are considered world treasures. Many famous artists, poets and writers appreciated the unique beauty of this city. The most famous landmarks include: Cathedral of St. Sophia with outstanding mosaics and frescoes dating back to the 11th century, Kyievo-Pechersk Lavra featuring several churches and cathedrals, Golden Gate of Kyiv (1037), Ukrainian Baroque Church of St. Andrew, the magnificent 19th-century Cathedral of St. Vladimir and many other attractions.



Mystetskyi Arsenal



Bulgakov museum



The Olympic National Sports Complex

Рисунок 4.1.12 – Профіль екскурсовода(продовження)

На сторінці екскурсовода міститься детальна інформація про нього та список усіх екскурсій, персональних екскурсій та найбільш відвідуваних авторських екскурсій цього екскурсовода.

Профіль туриста є доступним лише для самого туриста, або для екскурсовода, чиї екскурсії даний турист забронював. Окрім контактних даних, профіль містить інформацію про кількість здійснених туристом бронювань.

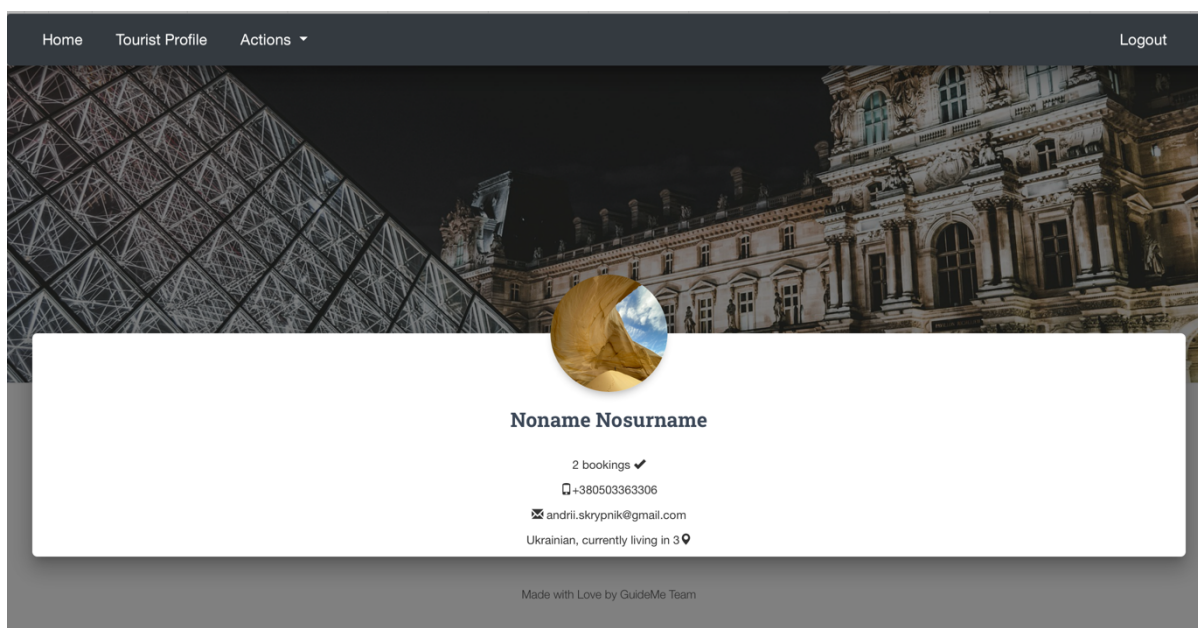


Рисунок 4.1.13 – Профіль туриста

Сторінки редагування профілів екскурсовода та туриста виглядають наступним чином:

Рисунок 4.1.14 – Редагування профілю екскурсовода

Рисунок 4.1.15 – Редагування профілю туриста

У екскурсовода також є можливість переглядати не підтверджені бронювання екскурсії, заплановані бронювання екскурсії, минулі та пропущені бронювання екскурсії. Якщо бронювання не є підтвердженим – екскурсовод може або ж підтвердити, або скасувати його. Після чого турист, що здійснив бронювання, отримає лист на електронну пошту з відповідними деталями.

Рисунок 4.1.16 – Сторінка бронювань екскурсій (для екскурсовода)

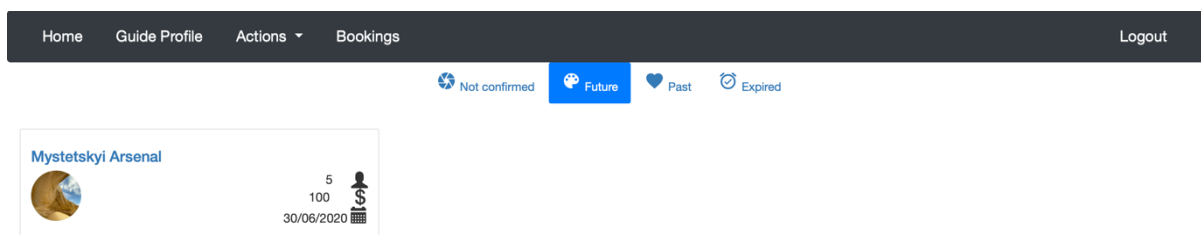


Рисунок 4.1.16 – Сторінка бронювань екскурсій (продовження)

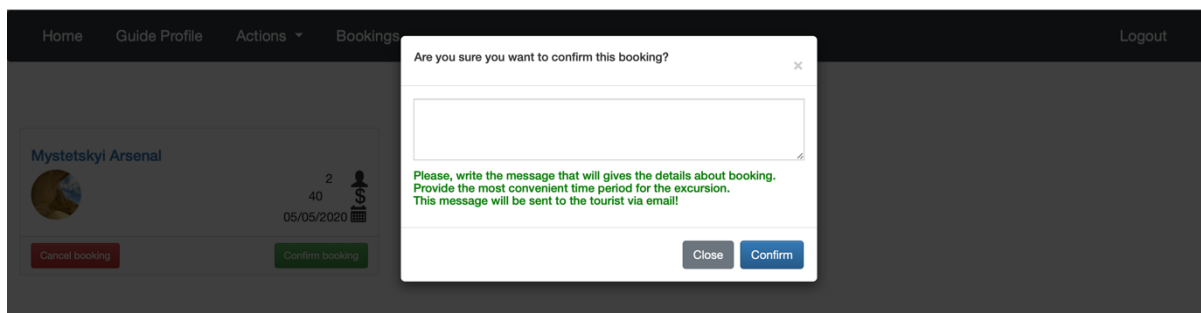


Рисунок 4.1.17 – Підтвердження бронювання

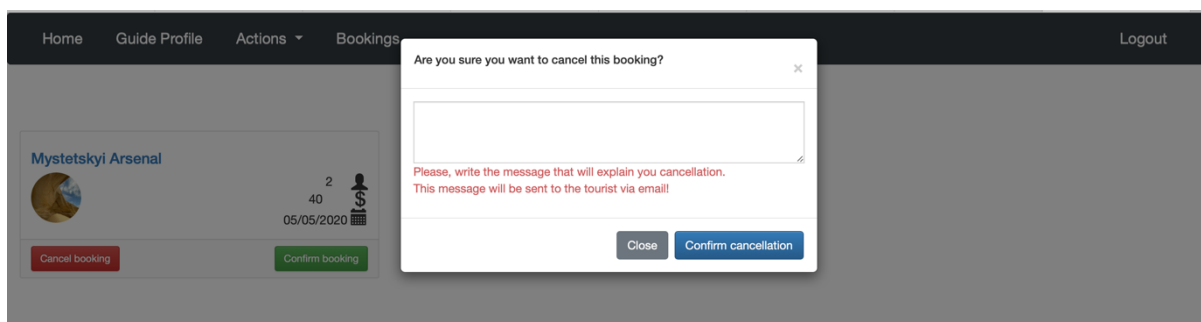


Рисунок 4.1.18 – Скасування бронювання

Висновок

У ході розробки веб-платформи для пошуку екскурсій, було досліджено основні аспекти розробки веб-застосунків на основі сучасного фреймворку Spring Framework для створення веб-застосунків мовою Java.

Spring Framework надає безліч можливостей для розробки сучасних веб-застосунків. Застосунок, створений на основі Spring досить легко налаштовується, розробляється та тестується. Spring полегшує роботу розробника, надаючи йому усі можливості для створення великомасштабного сучасного веб-додатку. Разом з Spring Security застосунок захищений від поширених атак, комплексно підтримує автентифікацію та авторизацію. Spring Boot дозволяє пришвидшити процеси конфігурування, створення та розгортання додатку. Застосунок, розроблений за допомогою технології Spring Boot автоматично налаштовується. Використання Spring MVC дозволяє зменшити зв'язність коду та відділити наслідки модифікації одного з компонентів на інші. Використання Spring Data та Hibernate дозволяє значно спростити та покращити реалізацію рівня доступу до даних, у легких та безпечний спосіб створювати та модифікувати дані, що містяться в базі даних.

Як висновок, Spring пропонує розробнику досить зручні та модернізовані засоби для побудови веб-застосунку мовою Java, дозволяє розробляти легко масштабовану архітектуру серверної частини та у зручний спосіб взаємодіяти з клієнтською частиною. На даному етапі Spring Framework повністю задовольняє потреби розробленої веб-платформи і водночас надає великий запас можливостей для росту і розвитку проекту в цілому.

Список використаної літератури

1. <https://medium.com/@wkrzywiec/why-spring-framework-is-so-cool-8472ceabaab1>
2. <https://spring.io/projects/spring-security>
3. <https://habr.com/ru/post/435144/>
4. <https://habr.com/ru/post/336816/>
5. https://freemarker.apache.org/docs/dgui_misc_userdefdir.html