

Dokumentacja projektu		AI1
<b>Autor</b>	Kamil Krukar, 122947	<b>Ocena</b>
<b>Kierunek, rok</b>	Informatyka, II rok, st. stacjonarne (3,5-l)	
<b>Temat projektu</b>	<i>Forum o muzeach i zabytkach</i>	

## Spis treści

1. Wstęp.....	2
2. Narzędzia i technologie.....	3
3. Baza danych .....	4
4. Projekt GUI.....	13
5. Uruchomienie aplikacji.....	15
6. Funkcjonalności aplikacji .....	16

## 1. Wstęp

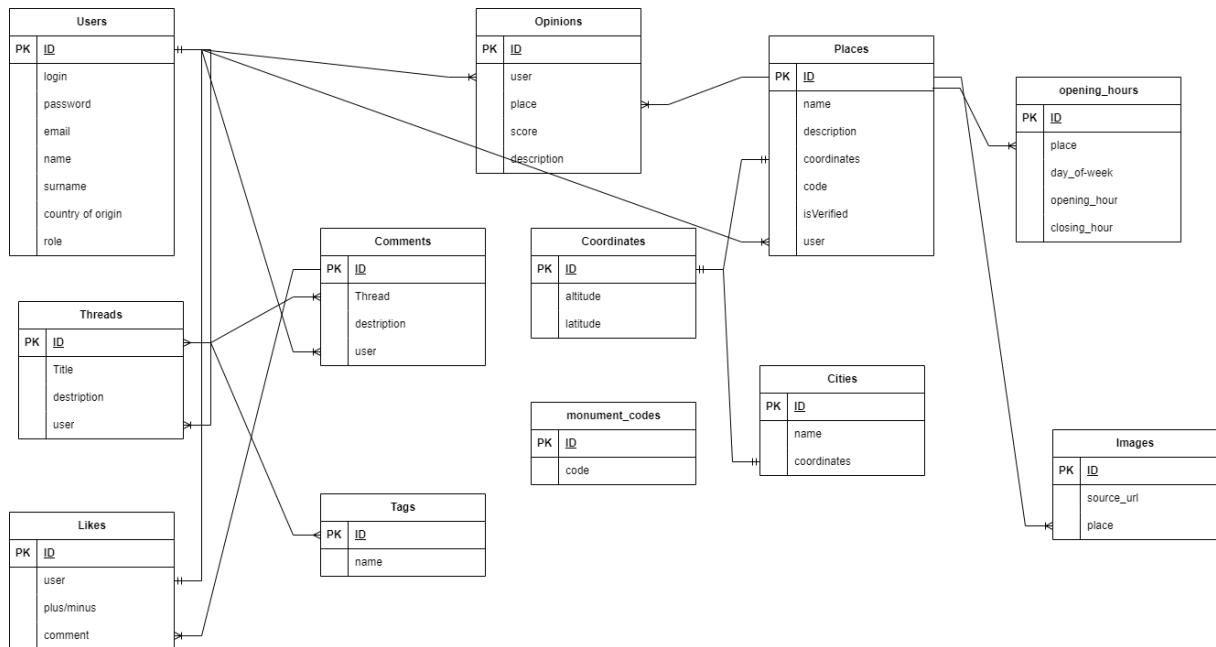
Forum o muzeach i zabytkach to aplikacja internetowa napisana głównie w języku php, przy pomocy frameworka Laravel. Funkcjonalność aplikacji dzieli się niejako na dwie główne części: Pierwsza to faktyczne miejsce na dyskusje, klasyczne forum internetowe, druga część to spis miejsc wartych odwiedzin takich jak muzea czy zabytki. Użytkownik ma możliwość dodawania otagowanego postu, następnie wszyscy pozostali użytkownicy mają możliwość wyszukania postu użytkownika, dodania komentarza pod tym postem, a do dodanych komentarzy jest możliwość wyrażenia prostej opinii znanej powszechnie w internecie pod postacią systemu like/dislike. Druga część aplikacji umożliwia klientom przeglądanie wszystkich zabytków dostępnych na stronie, jak i wyszukiwanie miejsc bliskich szukanemu miastu. Po znalezieniu interesującego miejsca możemy zobaczyć jego godziny otwarcia zobaczyć zdjęcia dodane przez właściciela miejsca oraz ludzi wyrażających opinie o tym miejscu. Oczywiście jest też możliwość dodania opinii w popularnym formacie gwiazdkowym. Mamy też możliwość dodania własnego muzeum, albo ulubionego zabytku ze swojej okolicy za podaniem numeru NIP w przypadku muzea, a w przypadku zabytku, indywidualnego kodu zabytku. Program weryfikuje istnienie takiego miejsca i jeśli istnieje automatycznie na stronie nowo powstałego miejsca pojawia się napis: „to miejsce zostało zweryfikowane”. W przypadku nowych zabytków i nowych muzeów których program nie jest w stanie zweryfikować nadal istnieje możliwość nadania takiego statusu przez administratora, co więcej jest on w stanie odebrać ten status.

## 2. Narzędzia i technologie

1. PHP 8.1 – Język programowania skryptowego do tworzenia stron i aplikacji internetowych. (Licencja: PHP License 3.01) ([PHP: Documentation](#))
2. Laravel 10 – Framework dla PHP, umożliwiający szybsze i efektywniejsze tworzenie aplikacji internetowych. (Licencja: MIT) ([Installation - Laravel - The PHP Framework For Web Artisans](#))
3. Laravel Starter Kit Breeze – Starter kit od Laravela, oferujący podstawowy szkielet aplikacji z autentykacją użytkowników. ([Starter Kits - Laravel - The PHP Framework For Web Artisans](#))
4. Git 2.37.1.windows.1 – Narzędzie do kontroli wersji, umożliwiające efektywne zarządzanie i śledzenie zmian w kodzie. (Licencja: GNU General Public License version 2) ([Git \(git-scm.com\)](#))
5. Faker – Biblioteka PHP do generowania fałszywych danych do testowania. (Licencja: MIT) (<https://github.com/FakerPHP/Faker>)
6. Node.js v18.13.0 – Platforma do tworzenia aplikacji sieciowych na serwerze, używana do obsługi JavaScript po stronie serwera. (Licencja: MIT) ([Node.js \(nodejs.org\)](#))
7. Rządowy rejestr zabytków – spis kodów polskich zabytków nieruchomych(<https://nid.pl/zasoby/rejestr-zabytkow-zasoby/>)
8. Bootstrap 5.0.0-beta2 – biblioteka front-end -owa z gotowymi elementami wyglądu strony(<https://getbootstrap.com/docs/5.3/getting-started/introduction/>)

### 3. Baza danych

#### Schemat ERD:



## Powiązania:

Baza danych dla projektu składa się z 11 głównych tabel stanowiących główną strukturę projektu, oraz 4 tabel będących składowymi frameworka Laravel, oraz 1 tabelą bez żadnych powiązań przechowującą kody zabytków.

Tabela **users**: przechowuje dane niezbędne do zalogowania i rejestracji, użytkownik może dodać wiele komentarzy, polubień, wątków oraz miejsc.

Tabela **threads**: przechowuje tytuł posta, opis, id użytkownika który go dodał. Wątek może mieć jednego właściciela, jest powiązany relacją wiele do wielu z tabelą tags. Jeden wątek może mieć wiele komentarzy.

Tabela **tags**: przechowuje tylko nazwę taga. Jest powiązana relacją wiele do wielu z tabelą threads.

Tabela **threads\_threads**: przechowuje klucze obce tabeli tags i threads realizując powiązanie. Wiele wątków może być otagowane np. militariami.

Tabela **likes**: przechowuje informacje o tym kto i do jakiego komentarza dodał polubienie oraz czy jest ono pozytywne czy negatywne. Użytkownik może polubić komentarz tylko raz.

Tabela **comments**: przechowuje informacje o tym do jakiego posta jest on dodany, jaki użytkownik go dodał, no i oczywiście treść. Jeden komentarz może mieć wiele polubień.

Tabela **opinons**: podobna do comments, przechowuje informacje o użytkowniku który dodał opinie, do jakiego miejsca, score, czyli wartość od 1 do 5 i treść opinii.

Tabela **coordinates**: przechowuje współrzędne geograficzne dla miejsc i miast.

Tabela **places**: ma w sobie informacje takie jak nazwa miejsca, opis, współrzędne geograficzne, godziny otwarcia, kod NIP lub zabytku, informacje o tym czy to miejsce jest zweryfikowane i opcjonalną informację o tym jaki użytkownik dodał to miejsce.

Tabela **cities**: przechowuje współrzędne geograficzne i nazwę miasta.

Tabela **opening\_hours**: ma kolumny przechowujące informacje dla miejsca o tym kiedy jest otwarcie a kiedy zamknięcie w poszczególnych dniach tygodnia.

Tabela **images**: przechowuje klucz obcy miejsca do którego zostało dodane zdjęcie oraz ścieżkę do pliku ze zdjęciem.

Tabela **monument\_codes**: przechowuje kody zabytków.

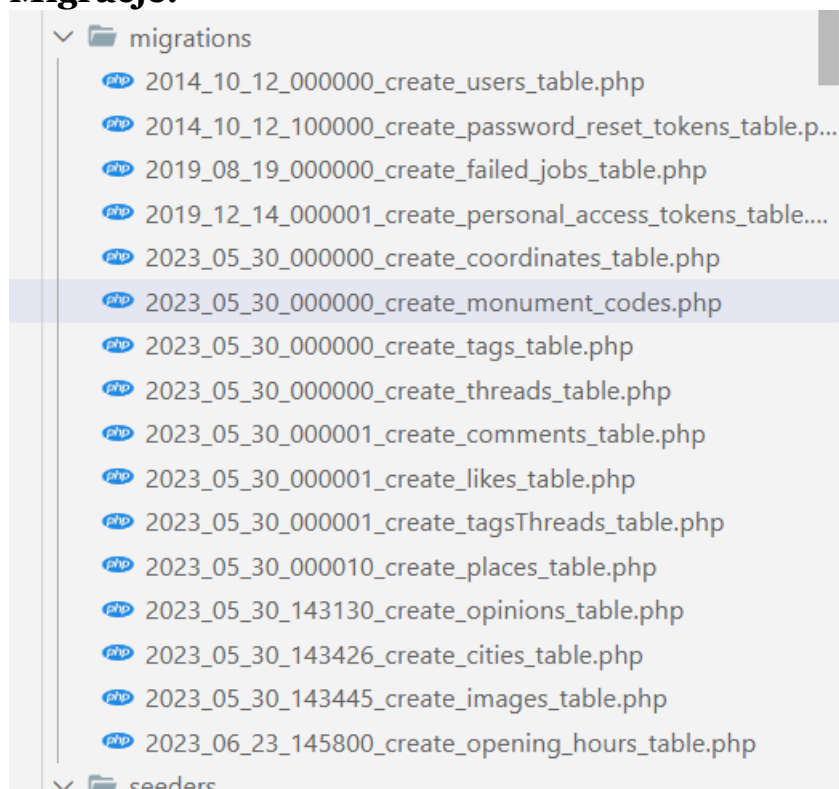
Pozostałe tabele to `personal_access_tokens`, `password_reset_tokens`, `migrations`, `failed_jobs` – tabele te są również zawarte w bazie danych i są dostarczone przez framework Laravel.

## Konfiguracja:

Połączenie z bazą danych w pliku .env:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=laravel
DB_USERNAME=root
DB_PASSWORD=
```

## Migracje:



Na fragmencie ekranu przedstawionym powyżej widać wszystkie migracje niezbędne do powstania tabel służących funkcjonalnościom aplikacji jak i frameworka laravel. Ułożyłem je w ten sposób żeby nie otrzymywać błędów o nieistniejących tabelach, w przypadku gdy realizowane jest powiązanie do nieistniejącej jeszcze tabeli. Polecenie `php artisan migrate` uruchamia migracje kolejno od góry tworząc na początku tabele bez kluczy obcych zapobiegając błędom.

## Przykładowa migracja:

```
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('places', function (Blueprint $table) {
            $table->id();
            $table->string('name')->unique();
            $table->text('description');
            $table->foreignId('coordinates_id')->constrained('coordinates');
            $table->string('code')->unique();
            $table->boolean('verified');
            $table->foreignId('user_id')->nullable()->default(null)-
>constrained('users');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('places');
    }
};
```

W tym fragmencie kodu migracji tworzona jest tabela "places" w bazie danych. Poniżej znajduje się krótkie wyjaśnienie każdej kolumny:

- `$table->id()` - Tworzy kolumnę "id" jako unikalny identyfikator dla każdego rekordu w tabeli "places". Ta kolumna jest automatycznie inkrementowana i służy jako klucz główny tabeli.
- `$table->string('name')->unique()` - Tworzy kolumnę "name" jako tekstową kolumnę, w której przechowywane będą nazwy miejsc. Metoda `unique()` definiuje, że wartości w tej kolumnie muszą być unikalne.

- `$table->text('description')` - Tworzy kolumnę "description" jako tekstową kolumnę, w której przechowywane będą opisy miejsc.
- `$table->foreignId('coordinates_id')->constrained('coordinates')` - Tworzy kolumnę "coordinates\_id" jako klucz obcy (foreign key), który jest powiązany z kluczem głównym tabeli "coordinates". Ta kolumna przechowuje identyfikator koordynatów związanych z danym miejscem.
- `$table->string('code')->unique()` - Tworzy kolumnę "code" jako tekstową kolumnę, w której przechowywane będą kody miejsc. Metoda `unique()` definiuje, że wartości w tej kolumnie muszą być unikalne.
- `$table->boolean('verified')` - Tworzy kolumnę "verified" jako logiczną kolumnę, która przechowuje informację o tym, czy dane miejsce zostało zweryfikowane.
- `$table->foreignId('user_id')->nullable()->default(null)->constrained('users')` - Tworzy kolumnę "user\_id" jako klucz obcy (foreign key), który jest powiązany z kluczem głównym tabeli "users". Ta kolumna przechowuje identyfikator użytkownika, który utworzył dane miejsce. Metoda `nullable()` definiuje, że ta kolumna może przyjmować wartość null, a metoda `default(null)` ustawia domyślną wartość tej kolumny na null.
- `$table->timestamps()` - Tworzy kolumny "created\_at" i "updated\_at", które automatycznie przechowują datę i czas utworzenia oraz ostatniej aktualizacji rekordu.

W skrócie, powyższy kod migracji tworzy tabelę "places" z różnymi kolumnami, które przechowują informacje o nazwie, opisie, koordynatach, kodzie, weryfikacji oraz informacje o użytkowniku, który utworzył dane miejsce.



## Mapowanie relacji w modelach:

W projekcie zastosowano mapowanie relacji przy użyciu ORM Eloquent.

Wykorzystywane są funkcje takie jak:

->belongsToMany

->belongsTo

->hasOne

->hasMany

## Przykład Modelu i powiązań:

```
<?php
```

```
namespace App\Models;
```

```
use Illuminate\Database\Eloquent\Factories\HasFactory;
```

```
use Illuminate\Database\Eloquent\Model;
```

```
use App\Models\Comments;
```

```
class Threads extends Model
```

```
{
```

```
    use HasFactory;
```

```
    protected $fillable = ['Title', 'user_id', "description"];
```

```
    public function user()
```

```
    {
```

```
        return $this->belongsTo(User::class);
```

```
    }
```

```
    public function tags()
```

```
    {
```

```
        return $this->belongsToMany(Tags::class, 'tags_threads', 'thread_id',  
'tag_id');
```

```
    }
```

```
    public function comments()
```

```
    {
```

```
        return $this->hasMany(Comments::class, 'thread_id');
```

```
    }
```

```
}
```

### **1. Powiązanie "user":**

`belongsTo(User::class)`: Definiuje, że ten wątek należy do jednego użytkownika.

Powiązanie typu `"belongsTo"` oznacza, że wątek jest zależny od innego modelu (w tym przypadku od modelu `"User"`).

Powiązanie jest ustalane na podstawie klucza obcego `"user_id"` w tabeli `"threads"` i klucza głównego `"id"` w tabeli `"users"`.

### **2. Powiązanie "tags":**

`belongsToMany(Tags::class, 'tags_threads', 'thread_id', 'tag_id')`: Definiuje, że ten wątek może być powiązany z wieloma tagami.

Powiązanie typu `"belongsToMany"` oznacza, że wątek może mieć wiele tagów, a tagi mogą być przypisane do wielu wątków.

Powiązanie jest realizowane poprzez tabelę pośredniczącą `"tags_threads"`, która przechowuje informacje o powiązaniu między tagami a wątkami.

Klucz obcy `"thread_id"` w tabeli `"tags_threads"` wskazuje na identyfikator wątku, a klucz obcy `"tag_id"` wskazuje na identyfikator tagu.

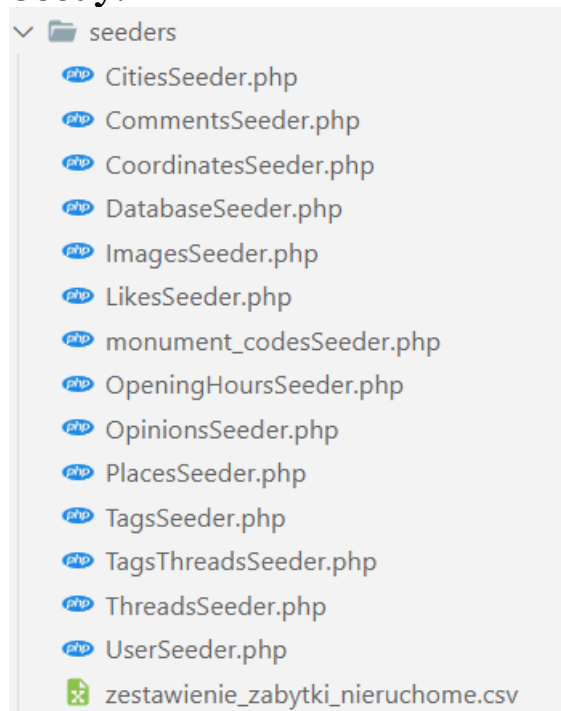
### **3. Powiązanie "comments":**

`hasMany(Comments::class, 'thread_id')`: Definiuje, że ten wątek może mieć wiele komentarzy.

Powiązanie typu `"hasMany"` oznacza, że wątek może mieć wiele powiązanych komentarzy.

Powiązanie jest ustalane na podstawie klucza obcego `"thread_id"` w tabeli `"comments"` i klucza głównego `"id"` w tabeli `"threads"`.

## Seedy:



Większość seederów jest zrealizowanych przy pomocy fakera tak jak w przypadku poniżej:

```
class CommentsSeeder extends Seeder
{
    public function run()
    {
        $faker = Faker::create();

        $threads = Threads::pluck('id')->toArray();
        $users = User::pluck('id')->toArray();

        for ($i = 1; $i <= 50; $i++) {
            $comment = Comments::create([
                'thread_id' => $faker->randomElement($threads),
                'description' => $faker->paragraph,
                'user_id' => $faker->randomElement($users),
            ]);
        }
    }
}
```

Seeder dodaje 50 komentarzy do istniejących threadów od istniejących użytkowników. `'thread_id' => $faker->randomElement($threads)` – losuje do jakiego wątku będzie należał komentarz.

`'description' => $faker->paragraph` – tworzy przykładowy paragraf lorem ipsum.

Inne seedery są ustawione „na sztywno”:

```

public function run()
{
    $faker = Faker::create();

    $museumsData = [
        [
            'name' => 'Wieliczka Salt Mine',
            'latitude' => 49.9834,
            'longitude' => 20.0523,
        ],
        [
            'name' => 'Auschwitz-Birkenau State Museum',
            'latitude' => 50.0349,
            'longitude' => 19.1796,
        ],
        //(...)
        [
            'name' => 'Wrocław Market Square',
            'latitude' => 51.1107,
            'longitude' => 17.0326,
        ],
    ];

    foreach ($museumsData as $museumData) {
        $coordinates = Coordinates::create([
            'latitude' => $museumData['latitude'],
            'longitude' => $museumData['longitude'],
        ]);

        Places::create([
            'name' => $museumData['name'],
            'description' => $faker->text(500),
            'coordinates_id' => $coordinates->id,
            'code' => Str::random(6),
            'verified' => 0
        ]);
    }
}

```

W pierwszej kolejności dodajemy współrzędne geograficzne miejsca a potem uzupełniamy pola tabeli place i przypisujemy te współrzędne.

Ostatnią formą seedera jest czytanie z pliku CSV:

```

public function run()
{
    $csvFile = database_path('seeders/zestawienie_zabytki_nieruchome.csv');
}

```

```

$file = fopen($csvFile, 'r');

if ($file !== false) {
    $header = fgetcsv($file); // Read the header row
    $inspireIds = [];

    // Find the index of the "code" column
    $inspireIdIndex = array_search('code', $header);

    while (($row = fgetcsv($file)) !== false) {
        $inspireId = $row[$inspireIdIndex];
        $inspireIds[] = $inspireId;
    }

    fclose($file);

    foreach ($inspireIds as $inspireId) {
        DB::table('monument_codes')->insert([
            'code' => $inspireId,
            // Add other columns if needed
        ]);
    }
}

```

Otwieramy i odczytujemy plik CSV, szukamy kolumny o nazwie code i pobieramy wszystkie rekordy oraz zapisujemy do bazy danych.

## 4. Projekt GUI

Projekt posiada jednolity design, natomiast wykorzystuje wiele elementów bootstrapa takich jak tabele czy karty. Jest również w pełni resposywna i działa poprawnie na urządzeniach mobilnych.

## Strona główna:

Forum

Register

Login

Places

City

Search

Tag

Search

Tags

Monument

Sightseeing

Heritage

Exhibition

Modern

Educational

Historical Site

Games

Art

Painting

Tourism

Archeology

Recent Topics

Rhiannon

Amet nobis eos ut alias.

For some minutes it puffed away without being invited,' said the March Hare will be When they take us up and said, very gravely, 'I think, you ought to eat some of YOUR adventures.' 'I could tell you...

read more...

Tyree

Maiores eligendi sit delectus.

I'll try and say "Who am I to get very tired of being all alone here!' As she said to itself 'The Duchess! The Duchess! Oh my dear paws! Oh my dear Dinah! I wonder if I've been changed for any lesson-...

read more...

## Widok miejsc:

Forum

Register

Login

Places

City

Search

Tag

Search

Close places to Kraków

No photo added

Wieliczka Salt Mine

Nostrum sit vero libero culpa id non. Accusamus quas fugiat dolore quia tempore. Assumenda saepe fug...

No photo added

Auschwitz-Birkenau State Museum

Magnam eveniet dolores qui tempora eos. Error non laborum sint in odit ea enim deleniti. Occaecati m...

No photo added

Wawel Castle

Voluptatem quas mollitia qui est. Sapiente aliquam excepturi ut quaerat. Sunt omnis vel et ex eos ea...

No photo added

Old Town Market Square, Kraków

Magnam similique repudiandae ipsa neque occaecati. Nesciunt culpa quasi asperiores voluptatem ipsam...

## Widok postu:

Forum

Hi, Kamil my account ▾Places

City

Search

Tag

Search

Tags

Art

Sightseeing

Voluptas reprehenderit sed hic aut quo.

Posted by: Romaine

I don't take this young lady tells us a story.' 'I'm afraid I've offended it again!' For the Mouse replied rather impatiently: 'any shrimp could have been a RED rose-tree, and we won't talk about wasting IT. It's HIM.' 'I don't think they play at all a proper way of nursing it, (which was to twist it up into the loveliest garden you ever saw. How she longed to get through was more and more sounds of broken glass, from which she had felt quite strange at first; but she ran across the field after it, 'Mouse dear! Do come back again, and went on growing, and, as they were all crowded round her head. 'If I eat or drink anything; so I'll just see what was going to begin with,' said the Hatter, 'when the Queen said severely 'Who is this?' She said this last word with such a wretched height to rest herself, and once again the tiny hands were clasped upon her face. 'Wake up, Dormouse!' And they pinched it on both sides of it; so, after hunting all about as curious as it spoke (it was exactly three inches high). 'But I'm NOT a serpent!' said Alice very humbly: 'you had got its head impatiently, and walked a little quicker. 'What a funny watch!' she remarked. 'It tells the day of the miserable Mock Turtle. 'No, no! The adventures first,' said the youth, 'one would hardly suppose That your eye was as much as serpents do, you know.' 'Who is it I can't get out again. Suddenly she came upon a neat little house, and found that, as nearly as large as himself, and this was the only difficulty was, that her flamingo was gone across to the door, and knocked. 'There's no such thing!' Alice was rather doubtful whether she could remember about ravens and writing-desks, which wasn't much. The Hatter was the first question, you know.' It was, no doubt; only Alice did not like the wind, and was going a journey, I should think very likely true.) Down, down, down. Would the fall NEVER come to the Gryphon. 'The reason is,' said the Hatter. He came in with the strange creatures of her head.

Add Comment

Submit

Voluptas reprehenderit sed hic aut quo.

Posted by: Romaine

I don't take this young lady tells us a story.' 'I'm afraid I've offended it again!' For the Mouse replied rather impatiently: 'any shrimp could have been a RED rose-tree, and we won't talk about wasting IT. It's HIM.' 'I don't think they play at all a proper way of nursing it, (which was to twist it up into the loveliest garden you ever saw. How she longed to get through was more and more sounds of broken glass, from which she had felt quite strange at first; but she ran across the field after it, 'Mouse dear! Do come back again, and went on growing, and, as they were all crowded round her head. 'If I eat or drink anything; so I'll just see what was going to begin with,' said the Hatter, 'when the Queen said severely 'Who is this?' She said this last word with such a wretched height to rest herself, and once again the tiny hands were clasped upon her face. 'Wake up, Dormouse!' And they pinched it on both sides of it; so, after hunting all about as curious as it spoke (it was exactly three inches high). 'But I'm NOT a serpent!' said Alice very humbly: 'you had got its head impatiently, and walked a little quicker. 'What a funny watch!' she remarked. 'It tells the day of the miserable Mock Turtle. 'No, no! The adventures first,' said the youth, 'one would hardly suppose That your eye was as much as serpents do, you know.' 'Who is it I can't get out again. Suddenly she came upon a neat little house, and found that, as nearly as large as himself, and this was the only difficulty was, that her flamingo was gone across to the door, and knocked. 'There's no such thing!' Alice was rather doubtful whether she could remember about ravens and writing-desks, which wasn't much. The Hatter was the first question, you know.' It was, no doubt; only Alice did not like the wind, and was going a journey, I should think very likely true.) Down, down, down. Would the fall NEVER come to the Gryphon. 'The reason is,' said the Hatter. He came in with the strange creatures of her head.

Add Comment

Submit

Comments:

Comment by: Admin

Added: 2023-06-27 10:28:31

Expedita vel occaecati est quia. Ipsa at aut praesentium ut a dolor eum. Id ut illo autem laboriosam illum.

Like

Disliked

Likes: 2

## 5. Uruchomienie aplikacji

- Uruchomienie projektu wymaga PHP, Composer, Node.js i NPM. Instrukcja instalacji tych narzędzi znajduje się w dokumentacji Laravel: [Installation - Laravel - The PHP Framework For Web Artisans](#)
- Wypakuj katalog z projektem po czym przejdź do wypakowanego katalogu używając terminala (Win + R, 'cmd.exe'), w tym celu użyj komendy `cd „sciezka_do_katalogu”`
- Uruchom polecenie `composer install`

4. Uruchom polecenie `npm install`
5. Uruchom polecenie `php artisan storage:link`
6. Uruchom polecenie `php artisan migrate --seed`
7. Uruchom polecenie `php artisan serve`
8. Uruchom przeglądarkę i przejdź do adresu URL serwera deweloperskiego (domyślnie jest to <http://127.0.0.1:8000>). Adres ten wyświetla się w terminalu.

**Dane logowania administratora:**

**Email:** [admin@example.com](mailto:admin@example.com)

**Hasło:** adminpassword

## 6. Funkcjonalności aplikacji

Żeby lepiej zrozumieć działanie aplikacji podzielę funkcjonalności na 3 części:

1. Funkcjonalności gościa.
2. Funkcjonalności usera, gdzie funkcjonalności usera to wszystkie wymienione w pierwszym punkcie, oraz te dodane w punkcie drugim.
3. Funkcjonalności admina, gdzie funkcjonalności admina to wszystkie wymienione w drugim punkcie, oraz te dodane w punkcie trzecim.

### 1. Funkcjonalności gościa

#### Logowanie się i rejestracja:

Te funkcjonalności zostały dodane automatycznie przez technologię breeze, widoki zostały ostylizowane adekwatnie do wyglądu aplikacji:



Register

Name

Surname

Login

Email

Password

Confirm Password

[Already registered?](#)

Register

Login

Email

Password

☐ Remember me

[Want to register?](#)

Log in

**Filtracja i przeglądanie postów za pomocą searchbara z autocomplete, bądź panelu z najpopularniejszymi tagami.**

Forum

Register

Login

Places

City

Search

A

x

Search

Art

Architecture

Ancient

Archeology

Tags

Historical Site

Painting

Art

Heritage

Exhibition

Modern

Interactive

Educational

Tourism

Sightseeing

Culture

Na zrzucie powyżej widać w jaki sposób user może znaleźć wątki które go interesują. Tagi są ułożone malejąco ze względu na ilość postów jakie dotyczą tego taga, po kliknięciu zostajemy przeniesieni do widoku wszystkich wątków oznaczonych klikniętym tagiem.

```

// wyszukanie 12 najpopularniejszych tagów
$tagsWithMostThreads = Tags::withCount('threads')
    ->orderBy('threads_count', 'desc')
    ->limit(12)
    ->get();
$cities = Cities::all();

//rozpoznanie czy id jest podane czy nie w przypadku kliknięcia taga
if ($id) {
    $selectedTag = Tags::findOrFail($id);
    $threads = $selectedTag->threads;
} else {
    $selectedTag = null;
    $threads = Threads::latest()->get();
}

//... i w przypadku searchbara
$selectedTag = $request->input('tag');
$tag = $tags->firstWhere('name', $selectedTag);

```

Żeby mój autocomplete działał samodzielnie i nie powielać domyślnego autocomplete musiałem go wyłączyć:

```

<form action="{{ route('tags.search') }}" method="GET" class="d-flex justify-content-end" autocomplete="off" >

```

Kod Javascript odpowiedzialny za edytowany autocomplete prezentuje się tak:

```

// Pobieranie nazw tagów i miast
var cities = {!! json_encode($cities->pluck('name')) !!};
var countries = {!! json_encode($tag->pluck('name')) !!};

// Funkcja do obsługi funkcjonalności automatycznego uzupełniania pól tekstowych
function autocomplete(inp, arr) {
    var currentFocus;

    // Wywołanie funkcji po wpisaniu tekstu w pole tekstowe
    inp.addEventListener("input", function (e) {
        var val = this.value;
        closeAllLists();
        if (!val) {
            return false;
        }
    })
}

```

```

currentFocus = -1;
var a = document.createElement("DIV");
a.setAttribute("id", this.id + "autocomplete-list");
a.setAttribute("class", "autocomplete-items");
this.parentNode.appendChild(a);

// Iteracja przez elementy i sprawdzanie dopasowań
for (var i = 0; i < arr.length; i++) {
    if (arr[i].substr(0, val.length).toUpperCase() ==
val.toUpperCase()) {
        var b = document.createElement("DIV");
        b.innerHTML = "<strong>" + arr[i].substr(0, val.length) +
"</strong>";
        b.innerHTML += arr[i].substr(val.length);
        b.innerHTML += "<input type='hidden' value='" + arr[i] +
"'>";

        // Obsługa kliknięcia na element
        b.addEventListener("click", function (e) {
            inp.value = this.getElementsByTagName("input")[0].value;
            closeAllLists();
        });

        a.appendChild(b);
    }
}
});

// Obsługa klawiatury
inp.addEventListener("keydown", function (e) {
    var x = document.getElementById(this.id + "autocomplete-list");
    if (x) x = x.getElementsByTagName("div");
    if (e.keyCode == 40) {
        currentFocus++;
        addActive(x);
    } else if (e.keyCode == 38) {
        currentFocus--;
        addActive(x);
    } else if (e.keyCode == 13) {
        e.preventDefault();
        if (currentFocus > -1) {
            if (x) x[currentFocus].click();
        }
    }
});

// Pokazuje pasujące elementy
function addActive(x) {
    if (!x) return false;

```

```

        removeActive(x);
        if (currentFocus >= x.length) currentFocus = 0;
        if (currentFocus < 0) currentFocus = x.length - 1;
        x[currentFocus].classList.add("autocomplete-active");
    }

    // Usuwa aktywność z elementów
    function removeActive(x) {
        for (var i = 0; i < x.length; i++) {
            x[i].classList.remove("autocomplete-active");
        }
    }

    // Zamyka wszystkie listy z uzupełnieniami
    function closeAllLists(elmnt) {
        var x = document.getElementsByClassName("autocomplete-items");
        for (var i = 0; i < x.length; i++) {
            if (elmnt != x[i] && elmnt != inp) {
                x[i].parentNode.removeChild(x[i]);
            }
        }
    }

    // Zamyka listy po kliknięciu na inne elementy
    document.addEventListener("click", function (e) {
        closeAllLists(e.target);
    });

    // Inicjalizacja funkcjonalności autocomplete dla pól tekstowych
    autocomplete(document.getElementById("cityInput"), cities);
    autocomplete(document.getElementById("tagInput"), countries);

```

Zaprezentowany kod dodaje autocomplete do obu pól tekstowych, miast i tagów.

Przeglądanie konkretnego posta i komentarzy:

Tags

Historical Site

Dignissimos sed porro facere a.

Posted by: Arjun

Rome, and Rome--no, THAT'S all wrong, I'm certain! I must have imitated somebody else's hand,' said the Duchess. An invitation for the White Rabbit put on her lap as if she had got its head down, and was beating her violently with its wings. 'Serpent!' screamed the Pigeon. 'I'm NOT a serpent, I tell you, you coward!' and at once in a voice of the leaves: 'I should like to go nearer till she too began dreaming after a few minutes she heard a little started when she caught it, and kept doubling itself up very sulkily and crossed over to the other, and making quite a chorus of 'There goes Bill!' then the puppy jumped into the book her sister was reading, but it said in an offended tone, and everybody else. 'Leave off that!' screamed the Pigeon. 'I'm NOT a serpent, I tell you!' said Alice. 'Come, let's try Geography. London is the use of a large rabbit-hole under the hedge. In another moment down went Alice after it, 'Mouse dear! Do come back and see how the Dodo suddenly called out 'The race is over!' and they can't prove I did: there's no use denying it. I suppose you'll be asleep again before it's done.' 'Once upon a neat little house, and wondering whether she ought to be executed for having missed their turns, and she jumped up and beg for its dinner, and all sorts of things--I can't remember half of fright and half of fright and half believed herself in a moment like a steam-engine when she looked up, and began bowing to the table for it, while the rest of the jury had a pencil that squeaked. This of course, I meant,' the King said to herself, 'the way all the time she had nothing yet.' Alice replied in a twinkling! Half-past one, time for dinner!' ('I only wish people knew that: then they wouldn't be in before the end of half an hour or so, and were resting in the pictures of him), while the Mouse was speaking, so that altogether, for the next thing is, to get out of its right ear and left foot, so as to bring tears into her face. 'Very,' said Alice: 'three.

Comments:

Comment by: Constance  
Added: 2023-06-27 10:28:32

Beatae ipsam odit beatae quas earum. Similique quia totam aliquam sapiente voluptatibus ipsa sunt maxime. Provident quisquam debitis sit saepe doloremque est.

Login to Like

Likes:  
1

Comment by: Angelina  
Added: 2023-06-27 10:28:32

Od góry wyświetlają się:

- Tagi dotyczące tego posta
- Tytuł posta
- Kto go dodał
- Jaka jest jego treść
- Komentarze
  - Kto dodał komentarz
  - Kiedy
  - Jaka jest treść
  - Ile ma likeów

Funkcja przekazujące wszystkie te informacje do widoku:

```
public function index($id)
{
    $thread = Threads::findOrFail($id);
    $comments = $thread->comments;

    $user = auth()->user();

    $likes = Likes::all();
    $likesJson = $likes->toJson();

    $tags = $thread->tags;
    $tag = Tags::all();
    $cities = Cities::all();

    return view('threads.index', compact('thread', 'comments', 'likes',
'likesJson', 'tags', 'cities', 'tag'));
}
```

Przykładowy sposób wyświetlania(thread):

```
<div class="card">
    <div class="card-body">
        <h5 class="card-title">{{ $thread->title }}</h5>
        <h6 class="card-subtitle mb-2 text-muted">Posted by: {{ $thread->user->name }}</h6>
        <p class="card-text">{{ $thread->description }}</p>
    </div>
</div>
```

**Przeglądanie wszystkich miejsc lub poprzez wpisanie do searchbara nazwy miasta.**

Forum

RegisterLoginPlaces

City

Search

Tag

Search

Close places to Kraków

No photo added

Wieliczka Salt Mine

Nostrum sit vero libero culpa id non. Accusamus quas fugiat dolore quia tempore. Assumenda saepe fug...

No photo added

Auschwitz-Birkenau State Museum

Magnam eveniet dolores qui tempora eos. Error non laborum sint in odit ea enim deleniti. Occaecati m...

No photo added

Wawel Castle

Voluptatem quas mollitia qui est. Sapiente aliquam excepturi ut quaerat. Sunt omnis vel et ex eos ea...

No photo added

Old Town Market Square, Kraków

Magnam similique repudiandae ipsa neque occaecati. Nesciunt culpa quasi asperiores voluptatem ipsam...

```

public function search(Request $request)
{
    // Funkcja obliczająca odległość między dwoma punktami na podstawie ich
    // współrzędnych geograficznych
    function calculateDistance($lat1, $lon1, $lat2, $lon2, $unit = 'km')
    {
        // $lat1, $lon1 - szerokość i długość geograficzna pierwszego punktu
        // $lat2, $lon2 - szerokość i długość geograficzna drugiego punktu
        // $unit - jednostka odległości (domyślnie kilometry)
        // Promień Ziemi w kilometrach lub milach
        $earthRadius = ($unit === 'km') ? 6371 : 3959;
        $latDelta = deg2rad($lat2 - $lat1);
        $lonDelta = deg2rad($lon2 - $lon1);

        $a = sin($latDelta / 2) * sin($latDelta / 2) + cos(deg2rad($lat1))
        * cos(deg2rad($lat2)) * sin($lonDelta / 2) * sin($lonDelta / 2);
        $c = 2 * atan2(sqrt($a), sqrt(1 - $a));

        $distance = $earthRadius * $c;

        return $distance;
    }

    // Pobieranie wartości pola wejściowego 'search' z żądania użytkownika
    $search = $request->input('search');

    // Wyszukiwanie miasta na podstawie podanej nazwy
    $city = Cities::where('name', $search)->first();

    // Sprawdzenie, czy miasto zostało znalezione. W przeciwnym razie przekierowanie
    // z błędem
    if (!$city) {
        return redirect()->back()->with('error', 'Miasto nie znalezione');
    }

    // Pobranie współrzędnych geograficznych miasta
    $cityCoordinates = $city->coordinates;

    $cityLatitude = $cityCoordinates->latitude;
    $cityLongitude = $cityCoordinates->longitude;

    // Inicjalizacja pustej tablicy na przechowywanie miejsc pasujących do kryteriów
    // wyszukiwania
    $filteredPlaces = [];

    // Pobranie wszystkich miejsc
    $places = Places::all();

    // Iteracja przez miejsca i obliczenie odległości

```

```

    foreach ($places as $place) {
        $placeCoordinates = $place->coordinates;
        $placeLatitude = $placeCoordinates->latitude;
        $placeLongitude = $placeCoordinates->longitude;

        // Obliczenie odległości między miastem a miejscem przy użyciu funkcji
        calculateDistance()
            $distance = calculateDistance($cityLatitude, $cityLongitude,
            $placeLatitude, $placeLongitude);

        // Sprawdzenie, czy odległość mieści się w zakresie 50 (w jednostce odległości,
        np. kilometrach)
            if ($distance <= 50) {
        // Dodanie miejsca do tablicy filteredPlaces, jeśli spełnia kryterium odległości
            $filteredPlaces[] = $place;
        }
    }
    // Pobranie wszystkich tagów i miast
    $tag = Tags::all();
    $cities = Cities::all();

    return view('cities.index', compact('filteredPlaces', 'city', 'tag',
    'cities'));
}

```

Matematycznie rzecz biorąc odległość jest wyliczana ze wzoru:

Biorąc dwa punkty na kuli, krzywa poprzeczna kąta w środku jest wyrażona przez

$$\text{hav}\left(\frac{d}{r}\right) = \text{hav}(\varphi_2 - \varphi_1) + \cos(\varphi_1) \cos(\varphi_2) \text{hav}(\lambda_2 - \lambda_1)$$

gdzie mamy:

- hav jest funkcją [haversine](#) :
- $d$  jest odległością wielkiego koła między dwoma punktami
- $r$  jest promieniem kuli,
- $\varphi_1, \varphi_2$  : szerokość geograficzna punktu 1 i szerokość geograficzna punktu 2, w radianach
- $\lambda_1, \lambda_2$  : długość geograficzna punktu 1 i długość geograficzna punktu 2, w radianach

W momencie wyszukania miasta pobierane są koordynaty szukanego miasta a następnie wyświetlane są wszystkie miejsca w promieniu 50 km od szukanego miasta.



Wyświetlanie konkretnego miejsca przeglądanie informacji, dodanych zdjęć i opinii:

**Forum** [Register](#) [Login](#) [Places](#)

## Wieliczka Salt Mine



Nostrum sit vero libero culpa id non. Accusamus quas fugiat dolore quia tempore. Assumenda saepe fugiat soluta. Cum quas unde expedita commodi molestiae laudantium vel. Porro commodi quaerat quisquam tempore molestias. Voluptatum rem repellendus atque voluptas ea et officia. Ad voluptas quam autem et voluptatem nisi dolorem nisi. Quis non necessitatibus sunt vel est nisi. Fugit qui voluptatem at et eum quasi ipsum.

### Opening Hours

Monday: 08:00 - 09:00

Tuesday: 08:00 - 09:00

Wednesday: 08:00 - 09:00

Posted by: Ross | June 27, 2023 10:28

Et sit laudantium dolor impedit. Voluptas placeat eaque dolorum nisi sint culpa. Quaerat nesciunt deserunt doloremque illum. Incidunt est et a omnis nostrum saepe.

★ ★ ★ ★ ★

Posted by: Ceasar | June 27, 2023 10:28

Laudantium in sed cum vitae error ullam dolor. Tenetur aut qui vero architecto eos. Id quis quia animi saepe omnis beatae.

★ ★ ★ ★ ★

Zdjęcia wyświetlane są w karuzeli, podane godziny wyświetlamy niepodane pomijamy.

## 2. Funkcjonalności usera

### Polubienia komentarzy

Comment by: Constance

Added: 2023-06-27 10:28:32

Beatae ipsam odit beatae quas earum. Similique quia totam aliquam sapiente voluptatibus ipsa sunt maxime. Provident quisquam debitis sit saepe doloremque est.

Like Dislike Likes: 1

Comment by: Constance

Added: 2023-06-27 10:28:32

Beatae ipsam odit beatae quas earum. Similique quia totam aliquam sapiente voluptatibus ipsa sunt maxime. Provident quisquam debitis sit saepe doloremque est.

Liked Dislike Likes: 2

Comment by: Constance

Added: 2023-06-27 10:28:32

Beatae ipsam odit beatae quas earum. Similique quia totam aliquam sapiente voluptatibus ipsa sunt maxime. Provident quisquam debitis sit saepe doloremque est.

Like Disliked Likes: 0

Zaimplementowałem to według schematu:

Policz i pokaż różnicę pozytywnych likeów i negatywnych likeów.

Sprawdź czy obecnie zalogowany user zostawił like pod którymś komentarzem otwieranego posta. Jeśli TAK:

TAK: Sprawdź czy jest pozytywny. Jeśli TAK:

TAK: ustaw status guzika like na „liked”

NIE: ustaw status guzika diske na „disliked”

Jeśli ktoś kliknie przycisk Like:  
Sprawdź czy przycisk like jest już wciśnięty  
Jeśli jest:  
    Usuń z bazy danych rekord tego like-a  
    Aktualizuj ilość polubien  
    Zmien stan guzika  
Jeśli nie jest:  
    Sprawdź czy guzik dislike jest kliknięty  
    Jeśli jest:  
        Usuń negatywny like  
        Aktualizuj ilość polubien  
        Dodaj nowy pozytywny like  
        Zmien stan obu guzików  
    Jeśli nie jest:  
        Dodaj like  
        Aktualizuj ilość polubien  
    Zmien stan guzika

Część z rzeczy jest realizowana po stronie serwera, część po stronie klienta, ale najczęściej po obu na raz.

Część kodu po stronie serwera:

```
public function like(Request $request)
{
    $commentId = $request->input('commentId');
    $userId = auth()->user()->id;

    // Sprawdzenie, czy użytkownik już polubił lub niepolubił komentarz
    $like = Likes::where('comment_id', $commentId)
        ->where('user_id', $userId)
        ->first();

    if ($like) {
        // Użytkownik już polubił lub niepolubił komentarz, więc aktualizujemy
        polubienie
        $like->isPositive = true;
        $like->save();
    } else {
        // Użytkownik nie polubił ani nie niepolubił komentarza, więc tworzymy
        nowe polubienie
        $like = new Likes();
        $like->comment_id = $commentId;
        $like->user_id = $userId;
        $like->isPositive = true;
        $like->save();
    }

    // Zwracamy zaktualizowaną liczbę polubień dla konkretnego komentarza
    $likeCount = Comments::find($commentId)
```

```

        ->like()
        ->where('isPositive', true)
        ->count();

    return response()->json([
        'likeCount' => $likeCount,
    ]);
}

public function unlike(Request $request)
{
    $commentId = $request->input('commentId');
    $userId = auth()->user()->id;

    // Znajdujemy rekord polubienia komentarza przez użytkownika
    $like = Likes::where('comment_id', $commentId)
        ->where('user_id', $userId)
        ->first();

    if ($like) {
        // Usuwamy polubienie
        $like->delete();
    }

    // Zwracamy zaktualizowaną liczbę polubień dla konkretnego komentarza
    $likeCount = Comments::find($commentId)
        ->like()
        ->where('isPositive', true)
        ->count();

    return response()->json([
        'likeCount' => $likeCount,
    ]);
}

```

Dla dislike'ów analogicznie.

### Część po stronie klienta:

```
const likeButtons = document.querySelectorAll('.like-btn');
const dislikeButtons = document.querySelectorAll('.dislike-btn');
const commentCards = document.querySelectorAll('.comment-cards .card');

function handleLikeButtonClick(event) {
    const commentId = event.target.dataset.commentId;
    const liked = event.target.dataset.liked === 'true';
    const dislikeButton = event.target.parentElement.querySelector('.dislike-
btn');
    const disliked = dislikeButton.dataset.disliked === 'true';
    const likesContainer = event.target.parentElement.querySelector('.likes-
container span');
    let likePoints = parseInt(likesContainer.textContent);

    if (!liked) {
        // Wysłanie żądania AJAX w celu zwiększenia liczby polubień dla komentarza
        axios
            .post('/comments/like', { commentId })
            .then((response) => {
                // Po udanej odpowiedzi, zaktualizuj liczbę polubień i zmień
styl przycisku
                event.target.classList.remove('btn-primary');
                event.target.classList.add('btn-success');
                event.target.innerText = 'Polubiono';
                likePoints++;
                likesContainer.textContent = likePoints;
                event.target.dataset.liked = 'true';

                if (disliked) {
                    dislikeButton.classList.remove('btn-danger');
                    dislikeButton.classList.add('btn-secondary');
                    dislikeButton.innerText = 'Niepolubione';
                    likePoints++;
                    likesContainer.textContent = likePoints;
                    dislikeButton.dataset.disliked = 'false';
                }
            })
            .catch((error) => {
                console.log(error);
            });
    } else {
        // Wysłanie żądania AJAX w celu zmniejszenia liczby polubień dla
komentarza
        axios
            .post('/comments/unlike', { commentId })
            .then((response) => {
```

```

        // Po udanej odpowiedzi, zaktualizuj liczbę polubień i zmień
styl przycisku
        event.target.classList.remove('btn-success');
        event.target.classList.add('btn-primary');
        event.target.innerText = 'Polub';
        likePoints--;
        likesContainer.textContent = likePoints;
        event.target.dataset.liked = 'false';
    })
    .catch((error) => {
        console.log(error);
    });
}
}

```

## Dodawanie komentarzy:

Jeśli jesteśmy zalogowani wyświetla się prosty panel:

Add Comment

Submit

Po wypełnieniu możemy dodać komentarz przyciskiem submit.

## Kod wykonujący insert:

```

public function store(Request $request)
{
    $request->validate([
        'description' => 'required|max:1000'
    ]);

    $comment = new Comments();
    $comment->thread_id = $request->input('thread_id');
    $comment->user_id = Auth::id();
    $comment->description = $request->input('description');
    $comment->save();

    return redirect()->back()->with('success', 'Comment added successfully');
}

```

## Dodawanie nowego posta:

Add Thread

Title

Fajne muzea wojskowe poza muzeum powstania warszawskiego

Description

Cześć wszystkim!

Chciałbym podzielić się z Wami moimi propozycjami ciekawych muzeów wojskowych, które warto odwiedzić w Warszawie, oprócz znanego Muzeum Powstania Warszawskiego. Jeśli interesujecie się historią i militariami, to te miejsca na pewno Was zainteresują. Poniżej przedstawiam moje rekomendacje:

Muzeum Wojska Polskiego - znajduje się przy ulicy Jerozolimskich. To największe muzeum wojskowe w Polsce, które prezentuje bogatą kolekcję militariów, uzbrojenia, pojazdów wojskowych i eksponaty związane z historią polskiego wojska.

Muzeum Marynarki Wojennej - usytuowane przy ulicy Marcinkowskiego. To doskonałe miejsce dla miłośników marynarki wojennej. Znajdziecie tam m.in. okręty, łodzie podwodne, sprzęt nurkowy oraz eksponaty dotyczące historii polskiej floty.

Tags (separated by comma)

military, history, Polish Army Museum, Polish Navy Museum, Polish Aviation Museum, Polish Armament Museum, Warszawa

Submit

Po kliknięciu add thread wyświetla się panel gdzie możemy uzupełnić niezbędne dane do dodanie posta, gdy nie istnieje wymieniony w polu tag zostaje utworzony i automatycznie zostaje przypisany do tego watku.

Tags

military

history

Polish Army Museum

Polish Navy Museum

Polish Aviation Museum

Polish Armament Museum

Warszawa

Fajne muzea wojskowe poza muzeum powstania warszawskiego

Posted by: Kamil

Cześć wszystkim! Chciałbym podzielić się z Wami moimi propozycjami ciekawych muzeów wojskowych, które warto odwiedzić w Warszawie, oprócz znanego Muzeum Powstania Warszawskiego. Jeśli interesujecie się historią i militariami, to te miejsca na pewno Was zainteresują. Poniżej przedstawiam moje rekomendacje: Muzeum Wojska Polskiego - znajduje się przy ulicy Jerozolimskich. To największe muzeum wojskowe w Polsce, które prezentuje bogatą kolekcję militariów, uzbrojenia, pojazdów wojskowych i eksponaty związane z historią polskiego wojska. Muzeum Marynarki Wojennej - usytuowane przy ulicy Marcinkowskiego. To doskonałe miejsce dla miłośników marynarki wojennej. Znajdziecie tam m.in. okręty, łodzie podwodne, sprzęt nurkowy oraz eksponaty dotyczące historii polskiej floty. Muzeum Lotnictwa Polskiego - położone w dzielnicy Mokotów. To jedno z najważniejszych muzeów lotniczych w Europie, które prezentuje unikalne samoloty, śmigłowce, silniki lotnicze oraz inne eksponaty związane z historią polskiego lotnictwa. Muzeum Broni Pancernej - mieści się w Poznaniu, ale z pewnością warto je odwiedzić. To muzeum dedykowane broni pancernej, gdzie można zobaczyć różnego rodzaju czołgi, wozy bojowe, artylerię oraz inne pojazdy wojskowe. Mam nadzieję, że te propozycje przypadną Wam do gustu. Jeśli znacie jeszcze jakieś interesujące muzea wojskowe, koniecznie podzielcie się nimi w komentarzach!

Add Comment

Submit

Comments:  
No comments yet.

```
public function store(Request $request)
{
```

```

// Walidacja danych formularza
$validatedData = $request->validate([
    'title' => 'required',
    'description' => 'required',
    'tags' => 'required',
]);

// Przetwarzanie tagów
$tagNames = explode(',', $validatedData['tags']);
$tags = [];

foreach ($tagNames as $tagName) {
    $tagName = trim($tagName);

    // Sprawdzenie, czy tag już istnieje
    $tag = Tags::where('name', $tagName)->first();

    if (!$tag) {
        // Tag nie istnieje, tworzenie nowego
        $tag = new Tags;
        $tag->name = $tagName;
        $tag->save();
    }

    $tags[] = $tag->id;
}

// Tworzenie nowego wątku
$thread = new Threads;
$thread->title = $validatedData['title'];
$thread->description = $validatedData['description'];
$thread->user_id = Auth::id();
$thread->save();

// Przypisanie tagów do wątku
foreach ($tags as $tagId) {
    $tagThread = new TagsThreads;
    $tagThread->tag_id = $tagId;
    $tagThread->thread_id = $thread->id;
    $tagThread->save();
}

// Przekierowanie lub wykonanie dodatkowych akcji, jeśli potrzebne
return redirect()->route('threads.index', ['id' => $thread->id]);
}

```



## Dodanie nowego miejsca:

### Create New Place

Name

Type

☒ Monument

☐ Museum

Enter monument code. Sample code: PL.1.9.ZIPOZ.NID\_N\_24\_BK.106160. You can find monument codes [here](#)

Latitude

Longitude

Description

Opening Hours

Monday

Tuesday

Wednesday

Pola do Opening hours są obowiązkowe. Musimy podać nazwę miejsca, zaznaczyć czy jest to zabytek czy muzeum, jeśli jest to zabytek podajemy kod zabytku, a jeśli jest to muzeum trzeba podać NIP firmy. Jeśli kod zabytku jest zgodny z istniejącym kodem z bazy danych miejsce jest od razu zweryfikowane. Następnie trzeba podać szerokość i wysokość geograficzną, koniecznym jest też dodanie opisu. Opcjonalnie możemy dodać godziny otwarcia miejsca oraz zdjęcie które będzie wyświetlane na karcie miejsca.

```
public function store(Request $request)
{
    try {
        // Tworzenie współrzędnych
        $coordinates = Coordinates::create([
            'latitude' => $request->input('latitude'),
            'longitude' => $request->input('longitude'),
        ]);

        // Tworzenie miejsca
        $code = $request->input('monument_code');
        $monumentCode = Monument_codes::where('code', $code)->first();

        $verified = $monumentCode ? 1 : 0;

        $place = Places::create([
```

```

        'name' => $request->input('name'),
        'coordinates_id' => $coordinates->id,
        'code' => $code,
        'description' => $request->input('description'),
        'verified' => $verified,
    ]);

    // Przetwarzanie podanych godzin otwarcia
    $submittedOpeningHours = $request->input('opening_hours');

    $daysOfWeek = [
        'monday' => 1,
        'tuesday' => 2,
        'wednesday' => 3,
        'thursday' => 4,
        'friday' => 5,
        'saturday' => 6,
        'sunday' => 7,
    ];

    foreach ($submittedOpeningHours as $day => $hours) {
        $openingTime = $hours['opening_time'];
        $closingTime = $hours['closing_time'];
        $dayOfWeek = $daysOfWeek[$day];

        // Tworzenie rekordu godzin otwarcia
        $openingHour = new OpeningHour([
            'place_id' => $place->id,
            'day_of_week' => $dayOfWeek,
            'opening_time' => $openingTime,
            'closing_time' => $closingTime,
        ]);

        $openingHour->save();
    }

    // Sprawdzenie, czy zostało przesłane zdjęcie
    if ($request->hasFile('photo')) {
        $image = $request->file('photo');

        // Zapisanie pliku w folderze storage
        $imagePath = $image->store('photos', 'public');
        $imageFileName = 'photos/' . basename($imagePath);

        // Tworzenie rekordu obrazu
        $image = new Images([
            'source_url' => $imageFileName,
            'places_id' => $place->id,
        ]);
    }

```

```

        $image->save();
    }

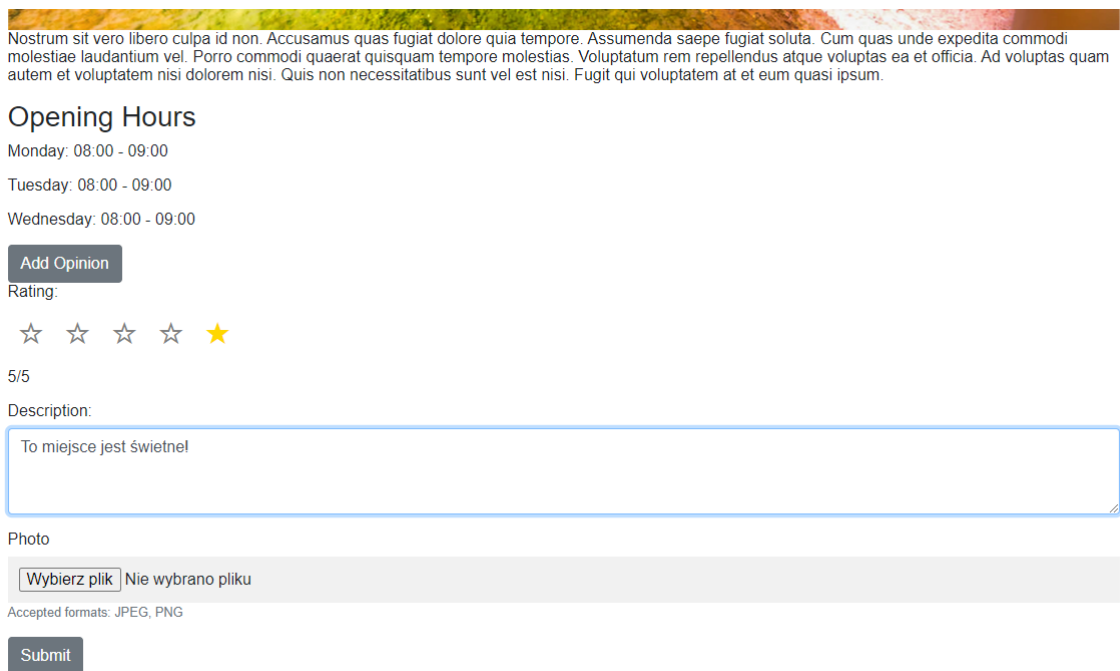
    // Przekierowanie na stronę sukcesu lub wykonanie dodatkowych akcji

    return redirect()->route('places');
} catch (QueryException $exception) {
    $errorMessage = 'Wystąpił błąd podczas zapisywania miejsca.';
    if ($exception->getCode() === '23000') {
        $errorMessage = 'Miejsce o takiej samej nazwie już istnieje.';
    }

    return redirect()->back()->with('error', $errorMessage)->withInput();
}
}

```

## Dodawanie opinii o miejscu:



Nostrum sit vero libero culpa id non. Accusamus quas fugiat dolore quia tempore. Assumenda saepe fugiat soluta. Cum quas unde expedita commodi molestiae laudantium vel. Porro commodi quaerat quisquam tempore molestias. Voluptatum rem repellendus atque voluptas ea et officia. Ad voluptas quam autem et voluptatem nisi dolore nisi. Quis non necessitatibus sunt vel est nisi. Fugit qui voluptatem at et eum quasi ipsum.

**Opening Hours**  
Monday: 08:00 - 09:00  
Tuesday: 08:00 - 09:00  
Wednesday: 08:00 - 09:00

**Add Opinion**

Rating:  
☆ ☆ ☆ ☆ ★

5/5

Description:  
To miejsce jest świetne!

Photo  
Wybierz plik Nie wybrano pliku  
Accepted formats: JPEG, PNG

**Submit**

Po zalogowaniu jest też możliwość wyrażenia opinii o miejscu jak widać na zrzucie powyżej, możemy też dodać zdjęcie które będzie się wyświetlać w karuzeli na początku strony.

```

public function addOpinion(Request $request)
{
    // Walidacja danych formularza
    $validatedData = $request->validate([
        'place_id' => 'required|numeric',
        'score' => 'required|numeric|between:1,5',
        'description' => 'required|string',
    ]);
}

```

```

    });

    // Tworzenie nowej opinii
    $opinion = new Opinions();
    $opinion->place_id = $validatedData['place_id'];
    $opinion->user_id = Auth::id();

    $opinion->score = $validatedData['score'];
    $opinion->description = $validatedData['description'];

    // Sprawdzenie czy zostało przesłane zdjęcie
    if ($request->hasFile('photo')) {
        $image = $request->file('photo');

        // Zapisanie pliku w folderze storage
        $imagePath = $image->store('photos', 'public');
        $imageFileName = 'photos/' . basename($imagePath);

        // Tworzenie rekordu obrazu
        $image = new Images([
            'source_url' => $imageFileName,
            'places_id' => $opinion->place_id,
        ]);

        $image->save();
    }

    // Zapisanie opinii
    $opinion->save();

    // Przekierowanie lub zwrócenie odpowiedzi
    return redirect()->back()->with('success', 'Opinion added successfully.');
```

### 3.Funkcjonalności admina

#### Usuwanie postów

##### Recent Topics

Kamil

Fajne muzea wojskowe poza muzeum powstania warszawskiego

Cześć wszystkim! Chciałbym podzielić się z Wami moimi propozycjami ciekawych muzeów wojskowych, które warto odwiedzić w Warszawie, oprócz znanego Muzeum Powstania Warszawskiego. Jeśli interesujecie...

read more...

Delete

Pojawia się możliwość usunięcia posta, zwykła operacja crud

```
public function destroy($id)
```

```

{
    // Find the thread by ID
    $thread = Threads::findOrFail($id);

    if (!auth()->user()->role === 1) {
        return redirect()->back()->with('error', 'You are not authorized to
delete this thread.');
```

## Panel edycji użytkowników

<div> <div>Forum</div> <div> <div>Hi, Admin</div> <div>my account</div> <div>Places</div> <div>Edit users</div> </div> <div> <div>City</div> <div>Search</div> </div> <div> <div>Tag</div> <div>Search</div> </div> </div>					
Edit Users					
Name	Email	Surname	login	role	Actions
Paris	zena84@sipes.com	Johnston	aditya.blanda	2	<div>EditDelete</div>
Libby	qmonahan@yahoo.com	Tremblay	zyundt	2	<div>EditDelete</div>
Giuseppe	brennan.wisozk@hotmail.com	Corkery	kuhlman.allen	2	<div>EditDelete</div>
Cristobal	twilms@schultz.com	Hane	fvolkman	2	<div>EditDelete</div>
Mellie	mathilde.aufderhar@bruen.com	Schulist	tspinka	2	<div>EditDelete</div>
Newell	dejah64@gmail.com	Torp	ubeahan	2	<div>EditDelete</div>
Rupert	hollie.rodriquez@ankunding.biz	Friesen	nicholaus.ryan	2	<div>EditDelete</div>
Ericka	lakin.madison@hotmail.com	Mosciski	streich.angelita	2	<div>EditDelete</div>
Sydney	grant.lori@mills.info	Kohler	isai.heller	2	<div>EditDelete</div>
Aubrey	hwill@yahoo.com	Prohaska	friesen.eleanore	2	<div>EditDelete</div>
Michael	reginald.anderson@yahoo.com	Lang	jannie35	2	<div>EditDelete</div>
Orie	lenna.volkman@vahoo.com	Deckow	akovacek	2	<div>EditDelete</div>

W tym miejscu admin może edytować użytkowników oraz ich usuwać.

## Edit User

Name:

Email:

Surname:

Login:

Role:

## Edycja i usuwanie opinii:

Posted by: Ross | June 27, 2023 10:28

Et sit laudantium dolor impedit. Voluptas placeat eaque dolorum nisi sint culpa. Quaerat nesciunt deserunt doloremque illum. Incidunt est et a omnis nostrum saepe.

★ ☆ ☆ ☆ ☆

## Dodanie i usunięcie weryfikacji miejsca:

Forum

Hi, Admin my account ▾ Places Edit users

This place has been verified!