B1: Chuẩn bị dữ liệu

- Tải GADM khu vực Việt Nam
- Với dữ liệu cho các khu vực giải trí
    - Vào trang https://overpass-turbo.eu/#
    - Dán đoạn mã này và truy vấn:

```
[out:json][timeout:900];
// Relation ID for Vietnam is 49915 → area ID = 3600049915
area(3600049915)->.searchArea;

(
node["leisure"](area.searchArea);
way["leisure"](area.searchArea);
relation["leisure"](area.searchArea);
);
out center tags;
>;
out skel qt;
```

- Kết quả ra rất nhiều điểm vì nhiều cột và nhiều địa điểm trên toàn Việt Nam

---

B2: Tiền xử lí dữ liệu (Mục đích chính: Loại bỏ các thông tin dư thừa, chỉ cần giữ lại các thông tin chính)

- Công cụ : Python (Google Colab)
- Link: https://colab.research.google.com/drive/1u7SNyXcMERbwIqULQYlqE1pvdGqUbEis#scrollTo=FkiCdjrCWY1j
- Drive: https://drive.google.com/drive/folders/1HH52bPWVfDSXPnk35Hop2YxqrUg3wNfr?usp=sharing

a. file gadm (level 3 để gán đơn vị địa phương cho điểm)

```
```python
import geopandas as gpd

overlay = gpd.read_file("vietnam.geojson")
overlay = overlay.rename(columns = {
    "NAME_1" : "Tỉnh/thành",
    "NAME_2" : "Quận/huyện",
    "NAME_3" : "Phường/xã/thị trấn",
    "TYPE_3" : "Đơn vị hành chính"
```
```

````
})
overlay.to_file("vietnam_renamed.geojson", driver='GeoJSON')
```
````

b. file gadm (level 0 để bản đồ Leaflet intersect với Vietnam)

````
```py
import geopandas as gpd

vietnam = gpd.read_file("gadm41_VNM_0.shp").to_crs(4326)
vietnam.to_file("vietnam_full.geojson", driver="GeoJSON")
```
````

c. file từ overpass.turbo

- Loại bỏ trước các đối tượng không phù hợp

```
df = gpd.read_file("export.geojson")
df.head(5)

# 1. Phân nhóm leisure mới
valid_leisure = {
    # Thể thao
    "pitch": "sports",
    "track": "sports",
    "playground": "sports",
    "golf_course": "sports",
    "sports_centre": "sports",
    "sports_hall": "sports",
    "miniature_golf": "sports",
    "dance": "sports",
    "sport": "sports",
    "stadium;pitch": "sports",
    "bowling_alley": "sports",
    "horse_riding": "sports",
    "ice_rink": "sports",

    # Hồ bơi
    "swimming_pool": "pool",
    "bathing_place": "pool",

    # Phòng gym
    "fitness_centre": "gym",
    "fitness_station": "gym",

    # Trường học
    "schoolyard": "school",

    # Sân vận động (riêng biệt nếu muốn)
```
```

```
    "stadium": "stadium",

    # Bãi biển
    "beach_resort": "beach",

    # Công viên
    "park": "park",
    "garden": "park",
    "nature_reserve": "park",

    # Công viên nước
    "water": "water",

    # Resort
    "resort": "resort",

    # Nơi dã ngoại
    "picnic_table": "picnic",
    "recreation_ground": "picnic",
    "outdoor_seating": "picnic",
    "firepit": "picnic",
}

# 2. Nhãn tiếng Việt tương ứng
vi_labels = {
    "sports": "Thể thao",
    "pool": "Hồ bơi",
    "gym": "Phòng gym / thể hình",
    "school": "Trường học",
    "stadium": "Sân vận động",
    "beach": "Bãi biển",
    "park": "Công viên",
    "water_park": "Công viên nước",
    "resort": "Resort / Khu nghỉ dưỡng",
    "picnic": "Khu dã ngoại"
}


# 2. Chọn và lọc dữ liệu
columns_to_keep = ["@id", "name", "leisure", "description", "opening_hours",
"geometry"]
leisure_df = (
    df[df["leisure"].isin(valid_leisure.keys())]   # giữ các leisure có
trong nhóm
    .copy()
    .reset_index(drop=True)
    [columns_to_keep]
)

# 3. Gom nhóm leisure
leisure_df["leisure_grouped"] = leisure_df["leisure"].map(valid_leisure)

# 4. Gán tên tiếng Việt cho nhóm leisure
```

```
leisure_df["leisure_vi"] = leisure_df["leisure_grouped"].map(vi_labels)
```

- Cắt ghép theo lãnh thổ Việt Nam (và gán thêm tên đơn vị hành chính)

```
import geopandas as gpd

overlay = gpd.read_file("gadm41_VNM_3.shp").to_crs(4326)
overlay = overlay.rename(columns = {
    "NAME_1" : "Tỉnh/thành",
    "NAME_2" : "Quận/huyện",
    "NAME_3" : "Phường/xã/thị trấn",
    "TYPE_3" : "Đơn vị hành chính"
})

# Lớp này cắt từ leisure_df với overlay là lãnh thổ Việt Nam
leisure_with_location = gpd.sjoin(
    leisure_df,
    overlay,
    how="left",
    predicate="intersects"
)

leisure_with_location.to_file("leisure_with_location.geojson",
driver="GeoJSON")
```

B3: Chuẩn bị dữ liệu

- Chú ý rất kỹ: ở data/admin.ini, thay đổi toàn bộ thông tin về postgres bằng thông tin bạn có
- Tạo database mới (create_db.sql)

```
---- Đoạn này chỉ uncomment khi database entertainment đã tồn tại
---- Kết nối vào database khác (thường là 'postgres')
-- \c postgres
--
---- Xóa database cũ nếu tồn tại (nếu bạn chắc chắn)
-- DROP DATABASE IF EXISTS entertainment;

-- Tạo CSDL mới tên là "entertainment"
CREATE DATABASE entertainment;

-- Kết nối vào database vừa tạo (nếu đang chạy từ psql hoặc pgAdmin)
\c entertainment

-- Kích hoạt extension PostGIS để làm việc với dữ liệu không gian
CREATE EXTENSION postgis;
```

- Tạo file script.bat

```
@echo off
echo Đang import GeoJSON vào PostgreSQL...

ogr2ogr -f "PostgreSQL" PG:"host=localhost dbname=entertainment
user=postgres password=1"
"C:\xampp\htdocs\webgis_project\data\leisure_with_location.geojson" -nln
leisure -nlt PROMOTE_TO_MULTI -lco GEOMETRY_NAME=geom -lco FID=id -lco
PRECISION=NO -overwrite

echo Hoàn tất import.
pause
```

- Chạy file script.bat trong OSGeo4W Shell (hoặc QGIS Shell)

B4: Cấu trúc thư mục

```
webgis_project/
├── api/                  # Tập tin PHP dùng để truy vấn dữ liệu từ database
(query.php)
├── data/                 # Dữ liệu không gian và tệp cấu hình (GeoJSON, .ini,
.json)
│   ├── leisure_with_location.geojson
│   ├── normalization.json
│   └── vietnam/          # Bộ dữ liệu ranh giới hành chính (SHP, DBF, PRJ...)
├── docs/                 # Tài liệu hướng dẫn (Leaflet, Geolocation, Bootstrap)
├── public/               # Mã nguồn chính hiển thị bản đồ và giao diện người dùng
│   ├── index.html
│   ├── style.css
│   ├── showingMap.js
│   ├── extractingToJson.js
│   └── images/
│       ├── marker-icon.png, marker-shadow.png
│       └── icon/         # Icon đại diện cho các loại hình giải trí (sports.png,
resort.png, ...)
├── sql/                  # Script tạo CSDL và import dữ liệu
│   ├── create_db.sql
│   └── script.bat
├── template/             # Các thư viện frontend (Bootstrap, Leaflet)
│   ├── css/
│   └── js/
└── .vscode/              # Cấu hình cho VS Code (tuỳ chọn)
```

B5: Build Frontend

1. Công nghệ sử dụng

- HTML + CSS: tạo cấu trúc và giao diện chính.
- JavaScript: xử lý logic hiển thị bản đồ, gọi dữ liệu, vẽ marker.
- Leaflet.js: thư viện bản đồ mã nguồn mở, hỗ trợ tile map, marker, popup,...
- Bootstrap 5 (cài thủ công): hỗ trợ layout responsive và các thành phần giao diện.

2. Cấu trúc mã nguồn frontend

Toàn bộ mã frontend được lưu trong thư mục public/:

- index.html: giao diện chính và phần tử chứa bản đồ (`<div id="map">`)
- showingMap.js: script chính để hiển thị bản đồ, tile layer, marker và popup.
- extractingToJson.js: đọc và xử lý dữ liệu từ normalization.json để liên kết icon tương ứng.
- style.css: tùy chỉnh giao diện (màu sắc, kích thước bản đồ, responsive,...)

images/: chứa icon các loại hình giải trí, và icon marker mặc định của Leaflet.

3. Chức năng đã triển khai

- Hiển thị bản đồ Leaflet, sử dụng tile từ OpenStreetMap.
- Thêm marker theo loại hình giải trí, với icon đại diện riêng biệt từ normalization.json.
- Hiển thị thông tin địa điểm trong popup khi người dùng click vào marker.
- Tự động đọc dữ liệu từ backend (query.php), trả về dưới dạng JSON.

4. Kết nối với backend

JavaScript gọi tới API query.php thông qua fetch (Dòng 74-80 của public/extractingToJson.js)

```
```js
fetch("/webgis_project/api/query.php")
    .then(response => response.json())
    .then(result => {
        // thêm marker vào bản đồ
    });
```
```

B6: Build Backend

1. Công nghệ sử dụng

- PHP: ngôn ngữ chính để viết API truy vấn.

- PostgreSQL + PostGIS: cơ sở dữ liệu lưu trữ dữ liệu địa lý và thuộc tính.
- ogr2ogr: dùng để import dữ liệu không gian từ GeoJSON vào PostgreSQL.

2. Cấu trúc backend

- Mã nguồn backend nằm trong thư mục api/, hiện tại có tệp chính là:
  - query.php: script xử lý truy vấn dữ liệu từ bảng leisure, trả về danh sách địa điểm có thông tin không gian (lat/lng) và thuộc tính (name, type, v.v.).

3. Nội dung chính của query.php

- Kết nối đến cơ sở dữ liệu bằng pg_connect.
- Thực hiện câu lệnh SQL truy vấn các địa điểm từ bảng leisure.
- Chuyển đổi tọa độ không gian (từ geometry sang lat/lng).
- Trả về kết quả dạng JSON, dùng cho Leaflet hiển thị marker.

```
// Kết nối CSDL
$config = parse_ini_file("../data/admin.ini", true);
$server = $config["SERVER"];
$host = $server["host"];
$user = $server["user"];
$password = $server["password"];
$database = "entertainment";

$conn = pg_connect("host=$host dbname=$database user=$user password=$password");
if (!$conn) {
    echo json_encode(["error" => "Không kết nối được đến PostgreSQL"]);
    exit;
}

// Đảm bảo encoding
pg_query($conn, "SET client_encoding = 'UTF8'");

// Truy vấn địa điểm trong bán kính
$sql = "
    SELECT
        id,
        COALESCE(name, 'Không có tên') AS name,
        ST_X(ST_GeometryN(geom, 1)) AS lng,
        ST_Y(ST_GeometryN(geom, 1)) AS lat,
        ST_Distance(
            ST_GeometryN(geom, 1)::geography,
            ST_SetSRID(ST_MakePoint($2, $3), 4326)::geography
        ) AS distance
    FROM leisure
    WHERE LOWER(leisure_grouped) = LOWER($1)
    AND ST_DWithin(
        ST_GeometryN(geom, 1)::geography,
        ST_SetSRID(ST_MakePoint($2, $3), 4326)::geography,
        $4
    )
    ORDER BY distance ASC
```

```
    LIMIT $5;
";



$result = pg_query_params($conn, $sql, [$type, $lng, $lat, $radius, $quantile]);
if (!$result) {
    echo json_encode(["error" => "Lỗi khi truy vấn dữ liệu"]);
    exit;
}

$rows = [];
while ($row = pg_fetch_assoc($result)) {
    $rows[] = $row;
}

echo json_encode([
    "status" => "ok",
    "results" => $rows
]);
```

B7: Mô tả sản phẩm

1. Giao diện người dùng

- Bản đồ nền sử dụng Leaflet, hỗ trợ pan và zoom mượt mà.
- Hiển thị các điểm giải trí (leisure places) từ dữ liệu PostgreSQL, với biểu tượng (icon) tương ứng từng loại hình như:
    - Thể thao (sports.png)
    - Phòng gym (dumbbell.png)
    - Khu nghỉ dưỡng (resort.png)
    - Công viên (park.png) v.v.
- Popup thông tin hiển thị tên địa điểm khi người dùng click vào marker.

2. Dữ liệu hiển thị

- Dữ liệu được lấy từ bảng leisure trong PostgreSQL, đã được chuẩn hóa và xử lý trước.
- Mỗi điểm có thông tin:
    - Tên địa điểm (name)
    - Loại hình (leisure_grouped)
    - Vị trí (tọa độ lat, lng)

3. Tự động hóa hiển thị icon

- Mỗi loại hình leisure được ánh xạ đến một icon tương ứng thông qua file normalization.json.
- JS tự động chọn đúng biểu tượng để hiển thị trên bản đồ, giúp giao diện trực quan hơn.

4. Hiệu năng và tính năng

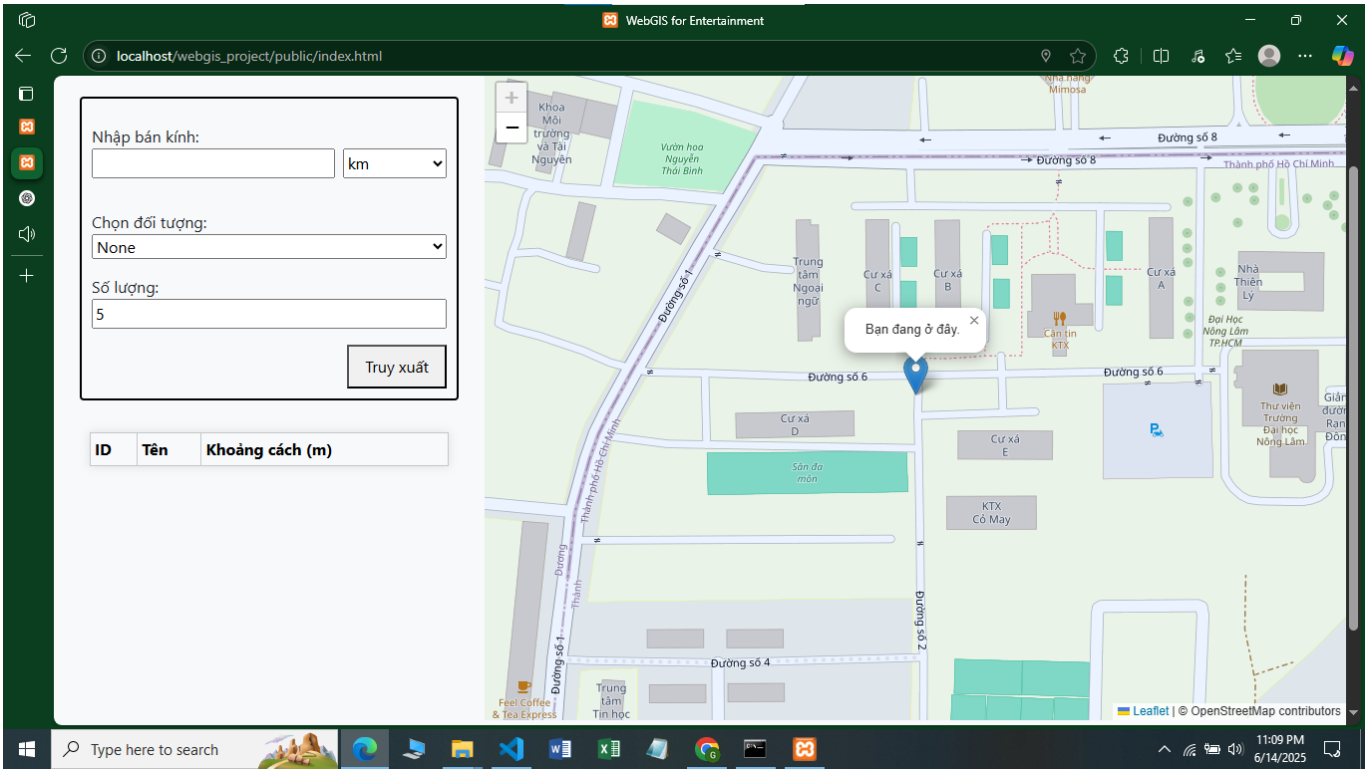
- Dữ liệu được truy vấn từ backend PHP thông qua API và hiển thị động trên bản đồ.
- Hệ thống có thể mở rộng để lọc dữ liệu, tìm kiếm, phân loại, hoặc hiển thị chi tiết địa điểm.

5. Cấu trúc thư mục rõ ràng

- Dữ liệu, mã nguồn frontend/backend và tài liệu được phân chia theo thư mục (/data, /api, /public, /docs,...).

B8: Cách hoạt động

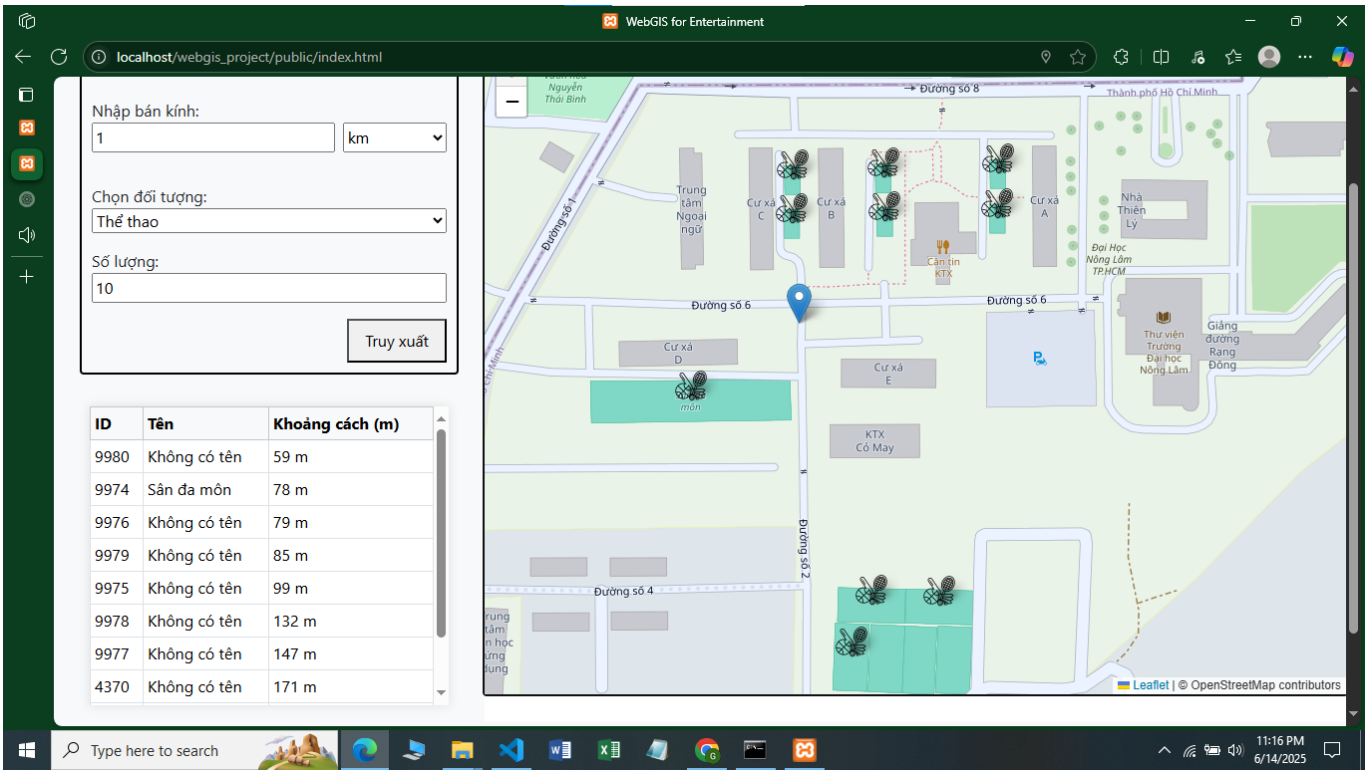
1. Tại trang chính:



- Cột 1, phía trên: nhập các thông tin:
    - Bán kính phạm vi:
        - Không được là số âm hoặc 0 (Nhưng mặc định là số 0)
        - Có thể ở đơn vị mét hoặc kilomet (mặc định)
    - Thể loại địa điểm
    - Số lượng địa điểm tối đa có thể truy xuất gần nhất (mặc định là 5)
    - Nút 'Truy xuất' dùng để xuất thông tin từ các thông tin truy vấn
- Cột 1, phía dưới: Trích xuất thông tin:
    - ID: Số thứ tự của các điểm đó
    - Name: Tên của địa điểm; nếu không có, tên của nó là 'Không có tên'
    - Khoảng cách (m): Khoảng cách từ người dùng tới các vị trí đó
- Toàn bộ cột 2 là bản đồ nền:
    - Lãnh thổ Việt Nam: màu sáng
    - Ngoài lãnh thổ Việt Nam: màu tối (không xét hải đảo)

- Vị trí hiện tại của người dùng: đánh dấu bằng một marker (tạm thời không kiểm tra fake GPS)

2. Sau khi truy vấn



- Ở cột 1, phía dưới: Hiển thị các thông tin sau khi truy vấn
- Ở cột 2: Hiển thị kết quả trên bản đồ sau khi truy vấn