Short Description

Pokemon Zoo backend API that enables the logic necessary to process and manage different activities. We can have different users that can log in. The users can look around at different pokemon and can even look for specific ones by name or species. Just the same the user can look at habitats where pokemon live and can donate to help them and assist their development.
Apart from what the user can do, there is some admin functionality to manage the addition of new pokemon or habitats.

Business Requirments

Authentication:

1. A user can register and log-in into the application

User management:

2. A user can get or delete his information
3. A user can create a list of favorite pokemon and manage it

Pokemon management:

4. A user can query information about pokemon in general or by name or species
5. A user can add the information of a new pokemon

Habitat management:

6. A user can look at the list of all habitats or details about one
7. A user can add a new habitat
8. A user can add a pokemon to a habitat

Donation management:

9. A user can look at all donations
10. A user can make a new donation

At least 5 endpoints
Tests for all endpoints and services
At least 6 entities and 4 relationships between them
One service and repository per entity

MVP

1. Users can register and login
2. Users can manage their account, look at their detailed information or delete it
3. Users can browse pokemon, filter by name or species and add them to favorites
4. Users can browse habitats and add pokemon to that habitat
5. Users can look at all donations with detailed information and can make a new donation

Entities

AppClient:

- id (UUID, PK)
- username (string, unique, not blank)
- password (string, not blank)
- role (string, default="ROLE_USER")
- favoritePokemon (Set<ClientFavoritePokemon>, One to Many)
- donations (Set<Donation>, One to Many)

ClientFavoritePokemon:

- id (UUID, PK)
- client_id (UUID, FK -> AppClient, not null, Many to One)
- pokemon_id (UUID, FK -> Pokemon, not null, Many to One)

Donation:

- id (UUID, PK)
- client_id (UUID, FK -> AppClient, not null, Many to One)
- habitatDonations (Set<DonationHabitat>, One to Many)
- type (string)
- amount (double)

DonationHabitat:

- id (UUID, PK)
- donation_id (UUID, FK -> Donation, not null, Many to One)
- habitat_id (UUID, FK -> Habitat, not null, Many to One)
- allocationAmount (double)

Habitat:

- id (UUID, PK)
- name (string, unique)
- description (string)
- foodSupply (double)
- pokemonInHabitat (Set<Pokemon>, One to Many)
- habitatDonations (Set<DonationHabitat>, One to Many)

Pokemon:

- id (UUID, PK)
- name (string)
- species (string)
- age (integer)
- weight (integer)
- height (integer)
- favoritedByClients (Set<ClientFavoritePokemon>, One to Many)
- habitat_id (UUID, FK -> Habitat, Many to One)

Relationships:

1. AppClient-ClientFavoritePokemon: One to Many (One client can have many favorite Pokemon)
2. AppClient-Donation: One to Many (One client can make many donations)
3. Pokemon-ClientFavoritePokemon: One to Many (One Pokemon can be favorited by many clients)
4. Donation-DonationHabitat: One to Many (One donation can be allocated to many habitats)
5. Habitat-DonationHabitat: One to Many (One habitat can receive many donations)
6. Habitat-Pokemon: One to Many (One habitat can contain many Pokemon)

Endpoints

/api

Auth Controller `/api/auth`
- Login - POST `/login`
  json
  Body: {
  "username": "string",
  "password": "string"
  }

  Response:
  - 200: `{ "token": "Bearer jwt-token" }`
  - 400: `{ "token": "Error message" }`
  - 500: `{ "token": "An error occurred" }`

- Register - POST `/register`
  json
  Body: {
  "username": "string",
  "password": "string"
  }

  Response:
  - 200: "User registered successfully"
  - 400: Error message
  - 500: "An error occurred"

AppClient Controller `/api/users`
- Get User Details - POST `/get_user`
  json
  Body: "Bearer token"

  Response:
  - 200: AppClientDetailsDto
  - 400: Bad Request
  - 403: Unauthorized

- Delete User - DELETE `/`
  json
  Body: UUID

  Response:
  - 200: "User deleted successfully"
  - 400: Error message

- Add Favorite Pokemon - POST `/favorites`
  json
  Body: {
  "clientId": "UUID",
  "pokemonIds": ["UUID"]
  }

  Response:
  - 200: "Favorite Pokemons added successfully"
  - 400: Bad Request

- Remove Favorite Pokemon - DELETE `/favorites`
  json
  Body: {
  "clientId": "UUID",
  "pokemonIds": ["UUID"]
  }

  Response:
  - 200: "Favorite Pokemons removed successfully"
  - 400: Bad Request

Donation Controller `/api/donations`
- Get All Donations - GET `/all`
  Response:
  - 200: List of DonationDetailsDto
  - 500: Internal Server Error

- Make Donation - POST `/add`
  json
  Body: {
  "clientName": "string",

"type": "string",
"amount": number,
"habitatNames": ["string"]
}

Response:
- 200: "Donation made successfully"
- 400: Bad Request
- 500: Internal Server Error

Habitat Controller `/api/habitats`
- Get All Habitats - GET `/all`
Response:
- 200: List of HabitatDetailsDto
- 500: Internal Server Error

- Add Habitat - POST `/add`
json
Body: {
"name": "string",
"description": "string",
"foodSupply": number
}

Response:
- 200: "Habitat added successfully"
- 400: Bad Request
- 500: Internal Server Error

- Get Habitat Details - POST `/details`
json
Body: "habitatName"

Response:
- 200: HabitatDetailsDto
- 400: Bad Request

- Add Pokemon to Habitat - POST `/add-pokemon`
json
Body: {

"habitatId": "UUID",
"pokemonIds": ["UUID"]
}

Response:
- 200: "Pokemon added to habitat successfully"
- 400: Bad Request

Pokemon Controller `/api/pokemon`
- Get All Pokemon - GET `/all`
Parameters:
- page: Integer
Response:
- 200: List of PokemonDetailsDto
- 500: Internal Server Error

- Search Pokemon - POST `/search`
json
Body: {
"searchType": "enum",
"searchTerm": "string",
"page": number
}

Response:
- 200: List of PokemonDetailsDto
- 400: Bad Request
- 500: Internal Server Error

- Count Pokemon - POST `/count`
json
Body: {
"searchType": "enum",
"searchTerm": "string"
}

Response:
- 200: Integer
- 400: Bad Request
- 500: Internal Server Error

- Add Pokemon - POST `/add`
  json
  Body: {
        "name": "string",
        "species": "string",
        "age": number,
        "weight": number,
        "height": number
  }

  Response:
  - 200: "Pokemon added successfully"
  - 400: Bad Request
  - 500: Internal Server Error