Trabalho Prático Final de Banco de Dados

Grupo: Alexsander Antunes, Américo Almeida, Pedro Chitarra, Túlio Araújo

Tema: Pokédex

A proposta é criar um banco de dados sobre a Pokédex, o registro geral de todos os Pokemon existentes nos jogos lançados pela empresa Nintendo. Além disso, manter o registro das regiões e gerações relacionadas a cada jogo lançado.

Entidades:

1. Pokemon: Id, Weight, Height, Name, BaseXP, Image, Generation, Type1Id, Type2Id;
2. Move: Id, Name, Class, Power, Accuracy, PP, Ailment, Description;
3. Type: Id, Name, Generation;
4. PokemonMove: PokeID, MoveID, Level;
5. PokemonGame:
6. Region: Id, Name, Generation;
7. Game: Id, Name, Year, Generation, Platform;
8. Ability: Id, Name, Description;
9. DamageMultiplier: AttackingType, DefendingType, Multiplier;

Relacionamentos:

1. Pokemon N ---- 1 Type (Type1),
2. Pokemon N ---- 1 Type (Type2),
3. Pokemon N ---- M Game,
4. Pokemon N ---- M Ability,
5. Pokemon N -- PokemonMove -- M Move,
6. Move N ---- 1 Type,
7. Region 1 ---- 1 Game
8. Type N -- DamageMultiplier -- M Type

Possíveis Consultas:

1. Tabela de multiplicador de dano para determinado tipo,
2. Tipo mais comum,
3. Região com mais Pokemon,
4. Plataforma com mais jogos,
5. Golpes de um determinado tipo,
6. Pokemon que possuem uma certa combinação de tipos,
7. Todos os golpes de um Pokemon,

1 - ESQUEMA RELACIONAL

pokemon (id NOT NULL, weight NOT NULL, height NOT NULL, name NOT NULL, baseXp NOT NULL, image, generation NOT NULL, type1Id NOT NULL, type2Id)
    type1Id REFERENCIA type
    type2Id REFERENCIA type

move (id NOT NULL, name NOT NULL, class NOT NULL, power NOT NULL, accuracy NOT NULL, pp NOT NULL, ailment NOT NULL, description NOT NULL, typeId NOT NULL)
    typeId REFERENCIA type

pokeType (id NOT NULL, name NOT NULL, generation NOT NULL)

pokémonMove (pokeId NOT NULL, moveId NOT NULL, level NOT NULL)
    pokeId REFERENCIA pokémon
    moveId REFERENCIA move

region (id NOT NULL, name NOT NULL, generation NOT NULL)

game (id NOT NULL, name NOT NULL, year NOT NULL, generation NOT NULL, platform NOT NULL)

ability (id NOT NULL, name NOT NULL, description NOT NULL)

damageMultiplier (id NOT NULL, attackingType NOT NULL, defendingType NOT NULL, multiplier NOT NULL)
    attackingType REFERENCIA Type
    defendingType REFERENCIA Type

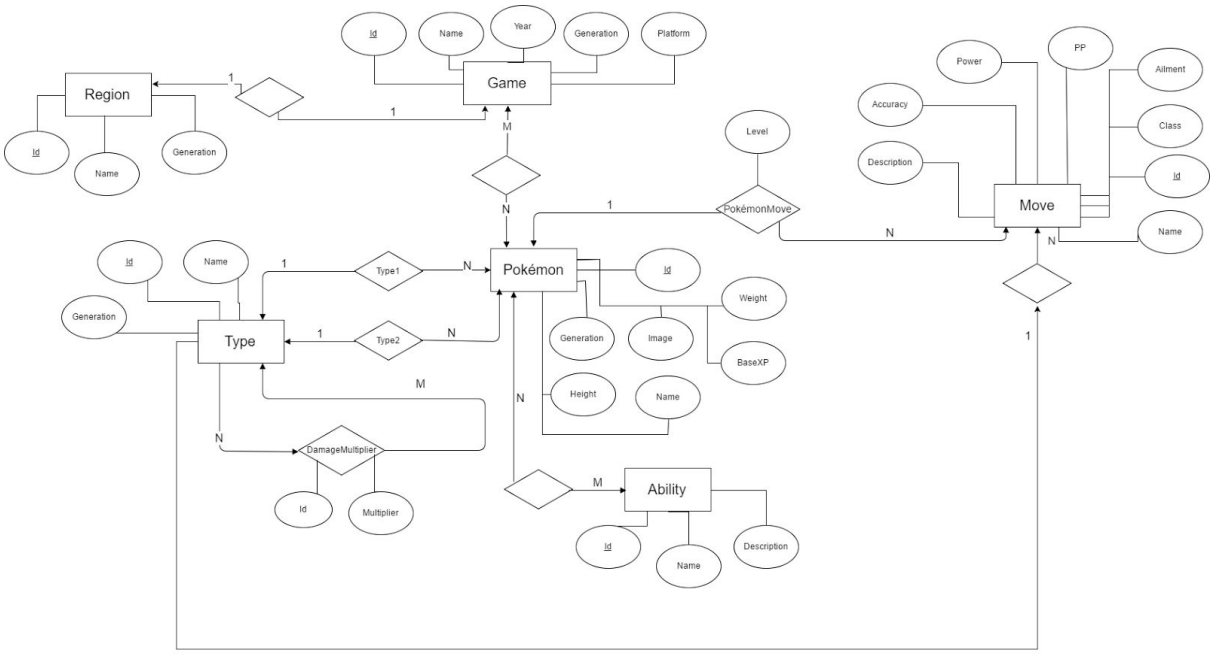
gameHasPokemon (gameId NOT NULL, pokeId NOT NULL)
    gameId REFERENCIA game
    pokémonId REFERENCIA pokémon

pokemonHasAbility (pokeId NOT NULL, abilityId NOT NULL)

pokeId REFERENCIA pokemon

abilityId REFERENCIA ability

2 - DIAGRAMA ER



3 - CRIAÇÃO DATABASE SQL

```
CREATE DATABASE IF NOT EXISTS pokedex;

CREATE TABLE IF NOT EXISTS pokemon (
      id int NOT NULL,
      weight float NOT NULL,
      height float NOT NULL,
      name varchar(255) NOT NULL,
      baseXP float NOT NULL,
      image LONGBLOB,
      generation int NOT NULL,
    type1Id int NOT NULL,
      type2Id int,
      PRIMARY KEY (id)
);

CREATE TABLE IF NOT EXISTS pokeType (
      id int NOT NULL,
```

```
      name varchar(255) NOT NULL,
      generation int NOT NULL,
      PRIMARY KEY (id)
);

CREATE TABLE IF NOT EXISTS move(
      id int NOT NULL,
    typeId int NOT NULL,
      name varchar(255) NOT NULL,
      class varchar(255) NOT NULL,
      power int NOT NULL,
      accuracy int NOT NULL,
      pp int NOT NULL,
      ailment varchar(255) NOT NULL,
      description varchar(255) NOT NULL,
      PRIMARY KEY (id),
    FOREIGN KEY (typeId) REFERENCES pokeType(id)
);

CREATE TABLE IF NOT EXISTS pokemonMove (
      pokeId int NOT NULL,
      moveId int NOT NULL,
      level int,
      PRIMARY KEY(pokeId,moveId),
      FOREIGN KEY (pokeId) REFERENCES pokemon(id),
      FOREIGN KEY (moveId) REFERENCES move(id)
);

CREATE TABLE IF NOT EXISTS region(
      id int NOT NULL,
      name varchar(255) NOT NULL,
      generation int NOT NULL,
      PRIMARY KEY(id)
);

CREATE TABLE IF NOT EXISTS  game (
      id int NOT NULL,
    regionId int NOT NULL,
      name varchar(255) NOT NULL,
      year int NOT NULL,
      generation int NOT NULL,
      platform varchar(255) NOT NULL,
      PRIMARY KEY (id),
    FOREIGN KEY (regionId) REFERENCES region (id)
);
```

```sql
CREATE TABLE IF NOT EXISTS ability (
	id int NOT NULL,
	name varchar(255) NOT NULL,
	description varchar(255) NOT NULL,
	PRIMARY KEY(id)
);

CREATE TABLE IF NOT EXISTS damageMultiplier (
	attackingTypeId int NOT NULL,
	defendingTypeId int NOT NULL,
	multiplier float NOT NULL,
	PRIMARY KEY (attackingTypeId, defendingTypeId),
	FOREIGN KEY (attackingTypeId) REFERENCES pokeType(id),
	FOREIGN KEY (defendingTypeId) REFERENCES pokeType(id)
);

CREATE TABLE IF NOT EXISTS gameHasPokemon (
	gameId int NOT NULL,
	pokeId int NOT NULL,
	PRIMARY KEY(gameId,pokeId),
	FOREIGN KEY (gameId) REFERENCES game (id),
	FOREIGN KEY (pokeId) REFERENCES pokemon (id)
);

CREATE TABLE IF NOT EXISTS pokemonHasAbility (
	pokeId int NOT NULL,
	abilityId int NOT NULL,
	PRIMARY KEY (pokeId,abilityId),
	FOREIGN KEY (pokeId) REFERENCES pokemon (id),
	FOREIGN KEY (abilityId) REFERENCES ability (id)
);

ALTER TABLE pokemon
ADD FOREIGN KEY (type1Id) REFERENCES pokeType(id),
ADD FOREIGN KEY (type2Id) REFERENCES pokeType(id);
```

4 - CONSULTAS AO BANCO DE DADOS SQL

1. Tabela de multiplicador de dano para tipo FOGO (id = 10)

SELECT * FROM damageMultiplier WHERE attackingTypeId = 10;

attackingTypeId	defendingTypeId	multiplier
10	6	5
10	7	2
10	9	2
10	10	5
10	11	5
10	12	2
10	15	2
10	16	5

2. Tipo mais comum de Pokemon

SELECT type.name, max (qtde) FROM pokémonHasType P1 JOIN type ON P1.type1Id = type.Id AND pokémonHasType P2 JOIN type ON P2.type2Id = type.Id GROUP BY type.name, count (pokeId) AS qtde

name	qtde
water	118
normal	97
flying	90
grass	84
psychic	74
bug	66
ground	60
poison	59
fire	56
rock	55
dark	44
fighting	44
electric	42
steel	41

dragon	38
fairy	35
ghost	35
ice	33

3. Ordem descendente do número de Pokemon em cada jogo

SELECT game.name, count(gameHasPokemon.pokeId) as qtde FROM game JOIN gameHasPokemon ON game.id = gameHasPokemon.gameId GROUP BY game.id ORDER BY qtde DESC

name	qtde
black and white	156
black and 2-white-2	156
red and blue	151
yellow	151
ruby and sapphire	135
emerald	135
firered and leafgreen	135
colosseum	135
xd	135
diamond and pearl	107
platinum	107
heartgold and soulsilver	107
gold and silver	100
crystal	100
x and y	72
omega ruby and alpha sapphire	72

4. Plataforma com mais jogos

SELECT platform, count(id) AS qtde FROM gameGROUP BY platform ORDER BY qtde DESC

platform	qtde
Nintendo DS	5
Game Boy Advance	3
Game Boy	2
Game Boy Color	2
Nintendo Game Cube	2
Nintendo 3DS	2

5. Pokemon que possuem uma certa combinação de tipos (Flying e Poison, por exemplo)

SELECT pokemon.name, pt1.name as tipo1, pt2.name as tipo2
FROM pokemon JOIN pokeType pt1 ON pokemon.type1Id = pt1.id JOIN pokeType pt2 ON pokemon.type2Id = pt2.id
WHERE (pokemon.type1Id = 3 AND pokemon.type2Id = 4)
OR (pokemon.type1Id = 4 AND pokemon.type2Id = 3)

name	tipo1	tipo2
zubat	poison	flying
golbat	poison	flying
crobat	poison	flying

6. O nome de todos os golpes de um Pokémon (Metang, por exemplo, de id = 375) (*)

SELECT move.name FROM move JOIN pokemonMove ON move.id = pokemonMove.moveId WHERE pokemonMove.pokeId = 375;

name
ice-punch
thunder-punch
cut
headbutt

body-slam
take-down
double-edge
hyper-beam
strength
earthquake
toxic
confusion
psychic
agility

7. Tabela de informações de Pokémons associando o nome de seus tipos, o jogo no qual foram incluídos e a região na qual estão presentes (*)

SELECT p.*, pt1.name AS type1, pt2.name AS type2, game.name AS game, region.name AS region FROM pokemon p JOIN pokeType pt1 ON p.type1Id = pt1.id LEFT JOIN pokeType pt2 ON p.type2Id = pt2.id JOIN gameHasPokemon ON p.id = gameHasPokemon.pokeId JOIN game ON gameHasPokemon.gameId = game.id JOIN region ON region.id = game.regionId WHERE game.id IN (SELECT min(game.id) FROM game JOIN gameHasPokemon ON game.id = gameHasPokemon.gameId WHERE gameHasPokemon.pokeId = p.id);

8. Tabela de golpes de um determinado Pokemon (selecionado pelo Id, 63 por exemplo), incluindo o level em que aprende cada habilidade e o tipo do golpe (*)

SELECT move.*, pokemonMove.level, pokeType.name AS type FROM pokemon JOIN pokemonMove ON pokemon.id = pokemonMove.pokeId JOIN move ON pokemonMove.moveId = move.id JOIN pokeType ON move.typeId = pokeType.id WHERE pokemon.id = 63 ORDER BY pokemonMove.level;

9. Move mais comum no jogo “Red and Blue”

SELECT M.id AS moveId, M.name AS moveName, G.name AS game, COUNT(M.id) AS quantity FROM pokemon P JOIN pokemonMove PM ON P.Id = PM.pokeId JOIN move M ON PM.moveId = M.Id JOIN gameHasPokemon GHP ON PM.pokeId = GHP.pokeId JOIN game G ON GHP.gameId = G.Id WHERE G.id = 1 GROUP BY M.id ORDER BY quantity DESC LIMIT 1;

moveId	moveName	game	quantity
173	snore	red and blue	146

10. Tabela de habilidades do grupo de Pokemons capazes de utilizar o golpe Karate Chop (id = 2)

SELECT A.* FROM pokemon P JOIN pokemonMove PM ON P.Id = PM.pokeId JOIN move M ON PM.moveId = M.Id JOIN pokemonHasAbility PHA ON P.id = PHA.pokeId JOIN ability A ON PHA.abilityId = A.id WHERE M.id = 2 GROUP BY A.id ORDER BY A.id;

id	name	description
5	sturdy	Prevents being KOed from full HP, leaving 1 HP instead. Protects against the one-hit KO moves regardless of HP.
7	limber	Prevents paralysis.
9	static	Has a 30% chance of paralyzing attacking Pokémon on contact.
39	inner-focus	Prevents flinching.
49	flame-body	Has a 30% chance of burning attacking Pokémon on contact.
62	guts	Increases Attack to 1.5× with a major status ailment.
72	vital-spirit	Prevents sleep.
80	steadfast	Raises Speed one stage upon flinching.
83	anger-point	Raises Attack to the maximum of six stages upon receiving a critical hit.
84	unburden	Doubles Speed upon using or losing a held item.
89	iron-fist	Strengthens punch-based moves to 1.2× their power.

99	no-guard	Ensures all moves used by and against the Pokémon hit.
104	mold-breaker	Bypasses targets' abilities if they could hinder or prevent a move.
113	scrappy	Lets the Pokémon's normal and fighting moves hit ghost Pokémon.
128	defiant	Raises Attack two stages upon having any stat lowered.

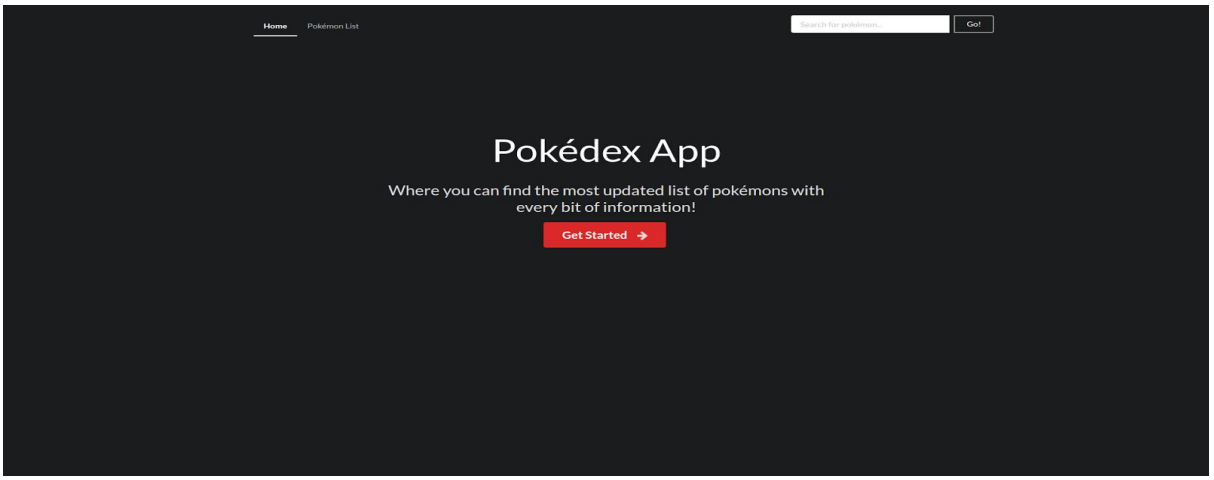
(*) O resultado das consultas é muito extenso e, portanto, foi omitido ou apresentada somente uma parcela do resultado

5 - INTERFACE

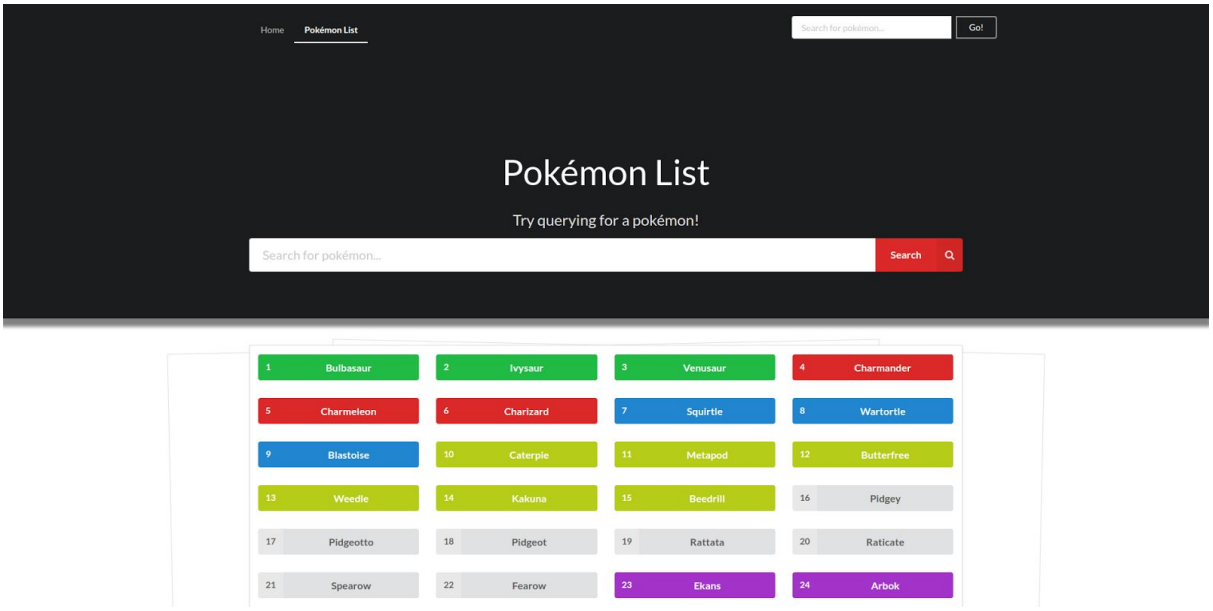
Foi desenvolvida uma interface para o banco de dados na forma de um sistema WEB programado em NodeJS em associação a um banco de dados MySQL populado por meio de scripts de requisição para a base de dados da PokéAPI (encontrada no endereço https://pokeapi.co/).

O sistema WEB conta com o framework *front-end* Semantic UI e a biblioteca Express para NodeJS na seção *back-end*. Sua função é representar um Índice de Pokémons contendo informações específicas de cada pokémon, bem como a capacidade de buscas simples parametrizadas pelo nome.

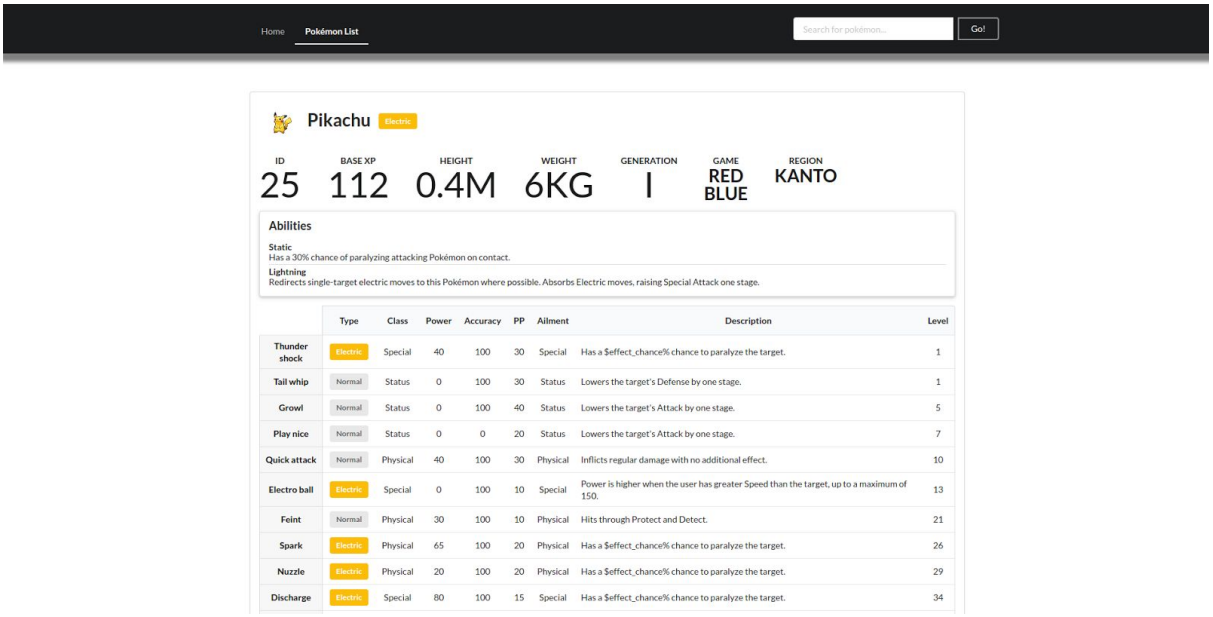
Abaixo estão algumas imagens da interface desenvolvida.



Tela inicial da Interface



Tela de busca



Tela de exibição de dados