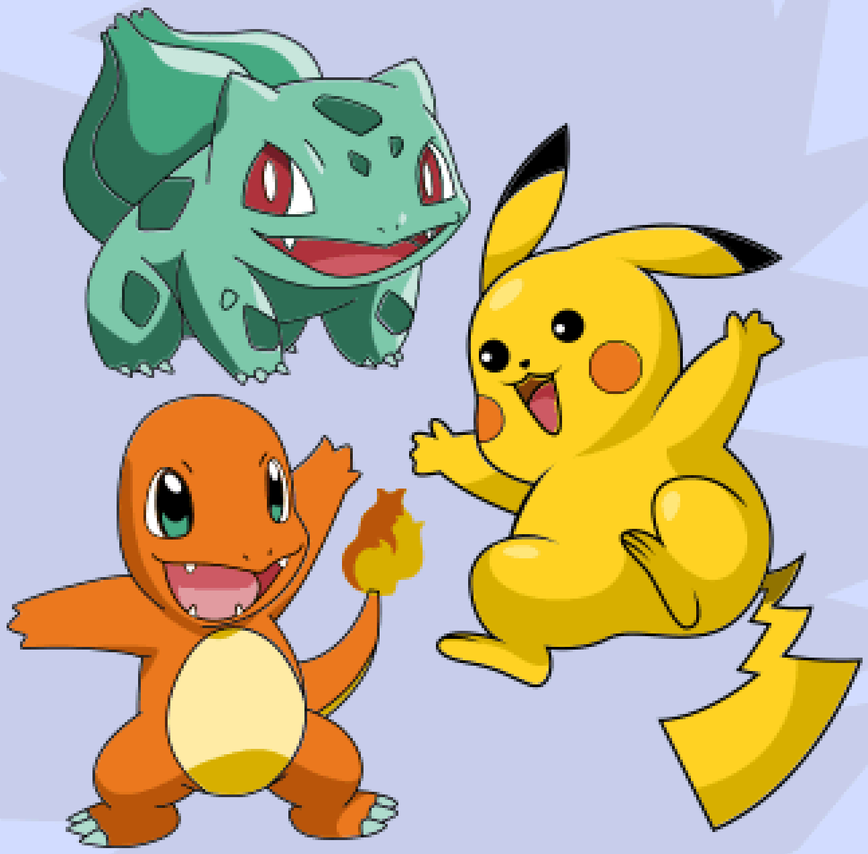


Projeto Pokedex

Foundations IOS

Kemuel de Matos

Luiz Augusto

Theo Resende

Tela Inicial

```swift
import SwiftUI


struct PokemonsView: View {
    func filterPokemons(search: String) -> [Pokemon] {
        if search.isEmpty {
            return pokemonsAPI
        }
        return pokemonsAPI.filter({
            return $0.name.localizedCaseInsensitiveContains(search)
        })
    }

    @State var pokemonsAPI: [Pokemon] = []

    @State var text: String = ""
    var body: some View {
        NavigationStack {
            ScrollView {
                ForEach(filterPokemons(search: text), id: \.id) { pokemon in
                    NavigationLink(destination: StatisticsView(pokemon: pokemon),

                                   label: {
                        HStack {
                            AsyncImage(url:URL(string: "https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/\(pokemon.id).png")) { image in
                                image.image?.resizable()
                            }
                            .frame(width: 50, height: 50)
                            .padding(.leading)
                            Spacer()
                            Text(pokemon.name.capitalized)
                                .foregroundColor(.black)
                            Spacer()
                            Text("#\(pokemon.id)")
                                .padding(.trailing)
                                .foregroundColor(.black)
                        }
                        .frame(width: 370, height: 80)
                        .foregroundColor(.white)
                        .background(.blue)
                        .cornerRadius(10)
                    })
                }
            }
            .scrollClipDisabled()
            .searchable(text: $text)
```

Tela Inicial

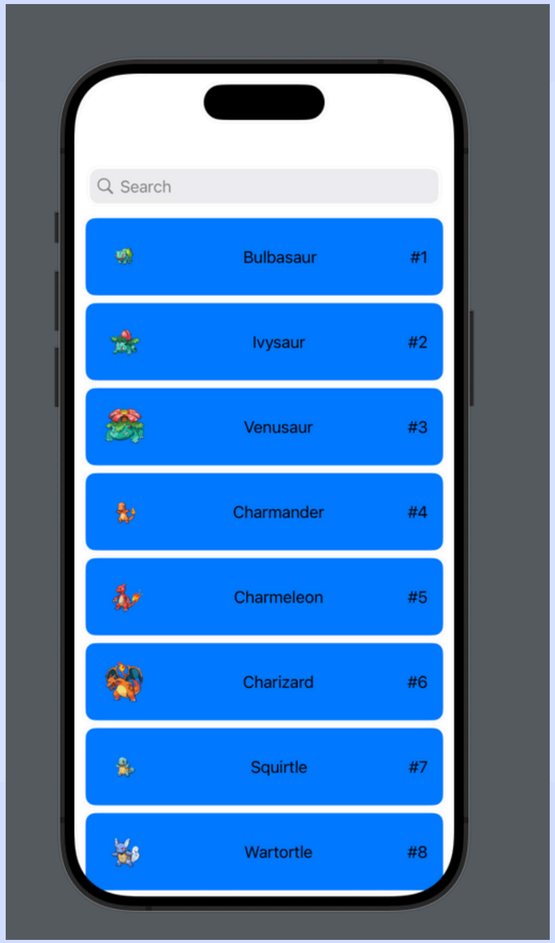
```
        }
        .onAppear {
            Task {
                for i in 0...151 {
                    var pokemon: Pokemon?
                    do {
                        pokemon = try await getPokemon(id: i)
                    } catch PokeError.invalidURL {
                        print("Invalid URL")
                    } catch PokeError.invalidResponse {
                        print("Invald Response")
                    } catch PokeError.invalidData {
                        print("Invalid Data")
                    } catch {
                        print("Unexpected error")
                    }

                    if pokemon != nil {
                        pokemonsAPI.append(pokemon!)
                    }
                }
            }
        }
    }

}

#Preview {
    PokemonsView()
}
```

Tela Inicial



Tela de Estatísticas

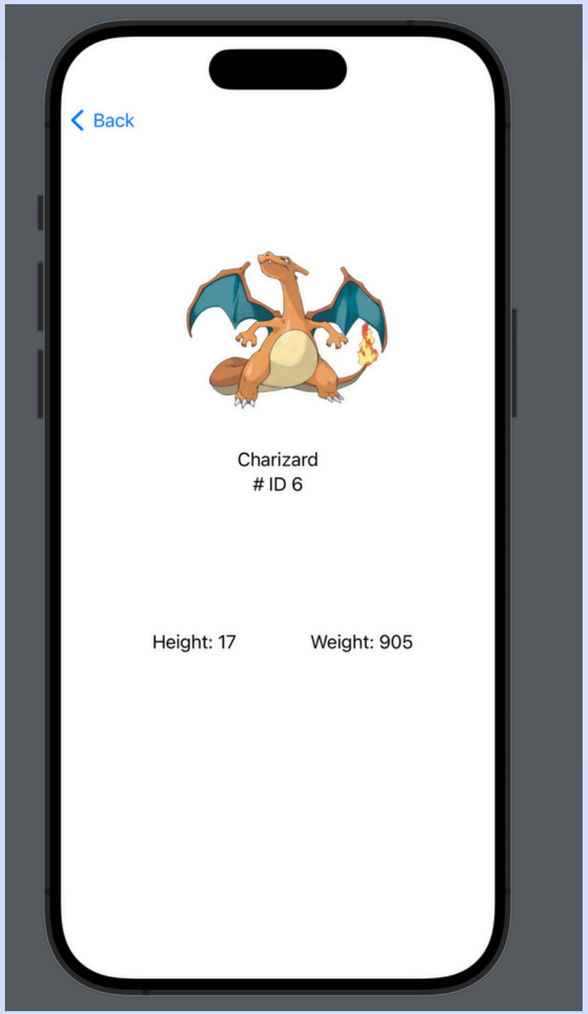
```swift
import SwiftUI
// "https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/other/official-artwork/132.png"
struct StatisticsView: View {
    @State var pokemon : Pokemon

    var body: some View {
            VStack {
                AsyncImage(url:URL(string: "https://raw.githubusercontent.com/PokeAPI/sprites/master/sprites/pokemon/other/official-artwork/\(pokemon.id).png")) 
                    image.image?.resizable()
                }
                .frame(width: 200, height: 200)
                .padding(.leading)
                Text(pokemon.name.capitalized)
                Text("# ID \(pokemon.id)")
                Spacer()
                HStack {
                    Spacer()
                    Text("  Height: \(pokemon.height)")
                    Spacer()
                    Text("  Weight: \(pokemon.weight)")
                    Spacer()
                }
                Spacer()
                Spacer()
            }
            .padding()
    }
}

#Preview {
    StatisticsView(pokemon: Pokemon(id: 1, name: "bulbasauro", weight: 12, height: 14))
}

```

Tela de Estatísticas



Controlador da API

```swift
import Foundation
func getPokemon(id: Int) async throws -> Pokemon {
    let endpoint = "https://pokeapi.co/api/v2/pokemon/\(id)"

    guard let url = URL(string: endpoint) else { throw PokeError.invalidURL }

    let (data, response) = try await URLSession.shared.data(from: url)

    guard let response = response as? HTTPURLResponse, response.statusCode == 200 else {
        throw PokeError.invalidResponse
    }

    do {
        let decoder = JSONDecoder()
        decoder.keyDecodingStrategy = .convertFromSnakeCase

        return try decoder.decode(Pokemon.self, from: data)
    } catch {
        throw PokeError.invalidData
    }

}

enum PokeError: Error {
    case invalidURL
    case invalidResponse
    case invalidData

}
```

Structs

```
1
2   import Foundation
3
4   struct Pokemon: Codable {
5       let id: Int
6       let name: String
7       let weight: Int
8       let height: Int
9
10  }
11
```