Technical Challenge

Overview

Build a Pokemon Browser application that demonstrates MVVM architecture, service layer design, and WPF/XAML proficiency.

Time Allocation: 3 days from receipt of this challenge

Functional Requirements

1. **Search & Display**
   a. Search bar to find Pokemon by name
   b. Display a list of Pokemon with basic info (name, image, types)
   c. Click on a Pokemon to view detailed information in a separate panel/view
2. **Data to Display**
   a. List view: Pokemon name, sprite image, primary type
   b. Detail view: Name, image, height, weight, all types, and base stats (HP, Attack, Defense)
3. **API Integration**
   a. Use PokeAPI (https://pokeapi.co/) - free, no auth required
   b. Endpoints needed:
      i. `GET https://pokeapi.co/api/v2/pokemon?limit=151` (list)
      ii. `GET https://pokeapi.co/api/v2/pokemon/{name}` (details)

Technical Requirements

1. **Architecture**
   a. Implement MVVM pattern (no code-behind except window initialization)
   b. Service layer for API communication
   c. Models for data representation
   d. ViewModels with `INotifyPropertyChanged`
2. **Service Layer**
   a. `IPokemonService` interface
   b. Async API calls
   c. Proper error handling
   d. DTO → ViewModel mapping
3. **XAML/UI**
   a. Data binding (no manual UI updates)
   b. Commands for user interactions (not click events)
   c. At least one `DataTemplate` for list items
   d. Responsive layout

4. **Expected Patterns**
    a. Dependency injection (constructor injection minimum)
    b. Async/await for API calls
    c. `ICommand` implementation (RelayCommand/DelegateCommand)
    d. Proper use of `ObservableCollection`

Bonus Points (Optional)

- Loading indicator during API calls
- Caching to avoid redundant API calls
- Unit tests for service layer
- Custom styling/theme
- Image caching for Pokemon sprites

Deliverables

1. **Public GitHub Repository**
    a. Complete Visual Studio solution committed to a public GitHub repo
    b. Clean commit history showing your development process
    c. Share the repository URL with us upon completion
2. **README.md** in the repository explaining:
    a. How to run the application
    b. Architecture decisions made
    c. Any assumptions or trade-offs

Submission

Please share your GitHub repository URL within **3 days** of receiving this challenge.