Pokemon Scout API - Challenge Assignment

Author: Vilmar Junior
Date: November 2025
Project: Pokemon Scout API

Challenge Overview

This project was developed as a technical challenge to demonstrate skills in full-stack development, API integration, and software architecture.

Personal Note

I'm a huge Pokemon fan - the franchise was a big part of my childhood. When I saw that this challenge involved Pokemon, I thought it was a really creative and fun theme for a technical assignment. It made the development process even more enjoyable, working with something I genuinely care about.

Requirements Met

1. Data Intake and Processing
Successfully retrieves data for all requested Pokemon: Pikachu, Dhelmise, Charizard, Parasect, Aerodactyl, and Kingler.

2. Data Sanitization and Formatting
Extracts relevant data points, sanitizes API responses, formats data consistently, and handles edge cases gracefully.

3. Database Storage
SQLite database with SQLAlchemy ORM, proper relational structure with foreign keys.

4. Flask Framework
RESTful API endpoints with clean route structure, proper error handling, and JSON responses.

5. Reusability
Works with any Pokemon name, minimal configuration needed, scalable architecture.

6. End-to-End Process
Complete workflow: API Request > Data Retrieval > Sanitization > Formatting > Database Storage > Data Export

7. Documentation
README.md with complete project documentation and CHALLENGE_NOTES.md with implementation details.

Understanding the Technology Choices

The challenge requirements specified SQLite, SQLAlchemy, and Flask. These choices make sense for a technical assessment: SQLite is lightweight and portable, SQLAlchemy demonstrates ORM skills, and Flask focuses on core web development without framework overhead.

Features Beyond Requirements

CLI Interface (scout.py), Database Viewer (view_db.py), Interactive Menu (menu.py) as bonus feature, JSON export, comprehensive error handling, type hints and docstrings.

Testing

All 6 requested Pokemon were successfully fetched and stored. Additional testing included Mewtwo, Dragonite, and Garchomp. Duplicate handling, error cases, and database integrity were verified.

Time Investment

Approximate total time: 3.5 hours including setup, implementation, testing, and documentation.

Pokemon Scout API - Challenge Assignment

Challenges Faced

PokeAPI returns nested JSON (solved with data processor), height/weight unit conversion, ensuring reusability through parameterized design.

Possible Future Improvements

Some features that could be added if needed for production: moves and items data, evolution chains, CSV export, better filtering, caching layer, PostgreSQL for larger datasets, unit tests, Docker containerization, CI/CD pipeline, web frontend.

Conclusion

I believe I've met all the requirements thoroughly and am grateful for the opportunity to take on this challenge. It was really enjoyable and valuable for testing my skills. Working with Pokemon data made the technical work even more engaging.

GitHub Repository: https://github.com/v-junior/Pokemon_project