Machine Learning Engineering - Phase 1

Amazon Bestsellers REST API & Data Analytics

Victor Filizola

March 1, 2026

Abstract

This document details the architecture, API endpoints, and analytical capabilities of the Amazon Bestsellers REST API. The project fulfills the requirements for the Machine Learning Engineering Phase 1 challenge, providing a complete data pipeline from web scraping to statistical analysis.

Contents

1 Project Links

- **GitHub Repository:** https://github.com/VictorFilizola/Tech-Challenge-FIAP.git
- **Live API URL:** https://tech-challenge-fiap-hhae.onrender.com/docs
- **Architecture & Demo Video:** https://youtube.com/dummy

2 Architecture Design

The system is built on a decoupled architecture, separating the heavy data ingestion process from the fast data serving and analytical layers.

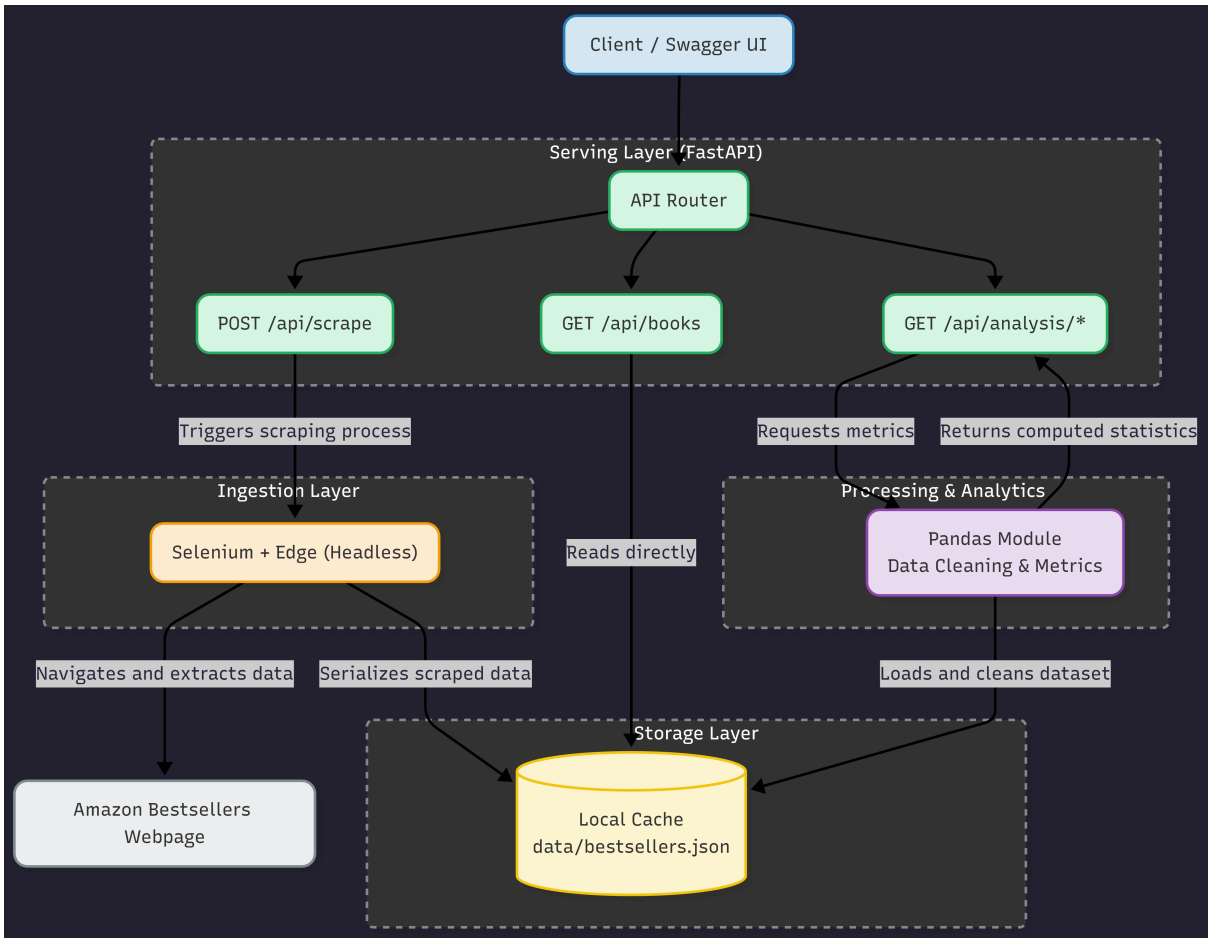

Figure 1: Project Architecture Mapping

2.1 Data Pipeline Flow

1. **Ingestion Layer (Selenium & Edge):** A headless browser navigates to the Amazon Bestsellers page, executes JavaScript to bypass lazy-loading, and extracts up to 100 book records (Title, Author, Rating, Review Count, Price).

2. **Storage Layer (Local JSON):** The scraped data is serialized and cached locally in a `data/bestsellers.json` file. This prevents IP blocking from Amazon and ensures millisecond response times for end users.
3. **Processing & Analytics Layer (Pandas):** The API reads the cached JSON, cleans the string data into numeric formats, and computes statistical metrics on the fly.
4. **Serving Layer (FastAPI):** Exposes the raw data and analytical views via a modern, self-documenting REST API.

3 API Documentation

The API is built using FastAPI, which automatically generates interactive Swagger UI documentation at the `/docs` endpoint. Below is the specification of the available routes.

3.1 System Routes

- `GET /health`: Returns the operational status of the API.
- `GET /info`: Returns metadata about the locally cached data, including the timestamp of the last successful scrape.

3.2 Data Ingestion

- `POST /api/scrape`: Triggers the Selenium web scraper. It navigates through multiple pages, collects the top 100 books, and overwrites the local JSON cache. *Note: This request takes several seconds to resolve due to browser execution.*
- `GET /api/books`: Instantly retrieves the raw list of books from the local cache.

4 Analytical Views

To provide a comprehensive descriptive analysis of the book sales, the monolithic analytics requirement was split into domain-specific endpoints.

4.1 `GET /api/analysis/general`

Provides high-level descriptive statistics of the scraped dataset.

- Total books analyzed.
- Average rating across the board.
- Total cumulative reviews across all best-selling books.

4.2 GET /api/analysis/pricing

Analyzes the financial distribution of the bestsellers.

- Average and Median prices (identifying right-skewed pricing).
- Minimum and Maximum price boundaries (identifying outliers like box sets).
- 25th and 75th percentiles to establish standard market rates.

4.3 GET /api/analysis/authors

Evaluates market share and author dominance within the top 100 list.

- Number of unique authors.
- The top 5 authors by sheer volume of books on the list.
- The combined market share percentage of those top 5 authors.

4.4 GET /api/analysis/engagement

Assesses the "Hype vs. Quality" matrix.

- Identifies the top 5 most heavily reviewed books.
- Identifies the highest-rated books (4.8 stars or higher) sorted by the highest volume of reviews, showcasing universally acclaimed titles.