AI Coding Pattern Detector

Description

AI Coding Pattern Detector is a smart platform designed for students, competitive programmers, and interview candidates. Users can input any coding problem and receive a **comprehensive analysis of all applicable algorithmic and data-structure patterns**, along with difficulty level and step-by-step approach suggestions. It helps learners **understand problem-solving strategies, detect patterns efficiently, and prepare for coding interviews and contests**.

Tech Stack

- **Backend:** Python, FastAPI
- **AI Model:** Ollama (Mistral 7B or Deepseek-R1)
- **Frontend:** HTML, CSS, JavaScript

Target Users

- Competitive programming students
- Engineering students preparing for technical interviews
- Learners aiming to master **DSA problem-solving techniques**

Features

Core Features

- **Pattern Detection:** Automatically identifies all relevant algorithms and data-structure techniques for a given problem.
- **Difficulty Level:** Categorizes problems as Easy, Medium, or Hard.
- **Approach Suggestion:** Provides step-by-step reasoning for solving the problem.

Advanced Features

- Problem input via **PDF or text**
- **Retrieval-Augmented Generation (RAG) support** to fetch context from previous problems (optional)
- Multi-pattern detection for complex or combined techniques

Supported Algorithmic Patterns

- Sliding Window, Prefix Sum, Two Pointers, Hashing, Kadane's Algorithm, Monotonic Stack, Monotonic Queue, Sorting, Subarray Techniques, Subsequence Techniques
- Classic 1D DP, 2D DP, DP on Trees, Bitmask DP, Memoization, Tabulation, Longest Common Subsequence, Longest Increasing Subsequence, Knapsack, Matrix DP, DP + Sliding Window
- BFS, DFS, Topological Sorting, Dijkstra, Bellman-Ford, Floyd-Warshall, Kruskal, Prim, Union-Find, Graph Coloring, Tarjan SCC, Network Flow
- Interval Scheduling, Minimum Coins, Fractional Knapsack, Huffman Coding, Greedy + Sorting
- Binary Search, Ternary Search, Sliding Window Search, Two Pointers Search, Meet-in-the-middle
- Recursion, Backtracking, Subsets, Permutations, Combinations, N-Queens, Sudoku Solver, Word Search, Maze Solving, Recursive DP
- Prime Factorization, GCD, LCM, Modular Arithmetic, Combinatorics, Sieve of Eratosthenes, Modular Exponentiation, Fibonacci, Tribonacci, Probability, Counting
- Stack, Queue, Heap, Segment Tree, Fenwick Tree, Trie, Balanced BST, Linked List, HashMap, HashSet, Union-Find
- Sliding Window + Hashing, Mo's Algorithm, Heavy-Light Decomposition, Sparse Table, Rolling Hash, FFT, Meet-in-the-middle
- Bit Manipulation, XOR Tricks, Game Theory, KMP, Z-Algorithm, Convex Hull, Line Intersections, Interval DP, Matrix Chain Multiplication, Randomized Methods

---

Workflow

1. **Problem Input:** User submits a coding problem via UI (text or PDF).
2. **AI Processing:** Ollama model (Mistral 7B) receives the input and analyzes it.
3. **JSON Output Generation:** Model generates structured JSON containing:
    - Applicable patterns
    - Difficulty level
    - Step-by-step approach suggestions
4. **UI Display:** JSON output is presented neatly on the UI for easy readability and study.

Project Impact

- Accelerates **learning of DSA patterns**.
- Prepares students for **competitive programming and coding interviews**.
- Demonstrates a **practical, AI-driven solution** for common student challenges.
- Highly **interview-impressive**, showcasing AI, prompt engineering, and DSA expertise.