Repository Analysis Report

Repository Overview

Repository Path:
/Users/saidulmondal/Desktop/profit_pilot/repoInsight/reposage/repos/Job_recommender_system_with_MCP

Total Files Scanned: 7

Detected Languages: Python

Health Score: 72 / 100 (Grade B)

- Security: 22

- Performance: 30

- Architecture: 10

- Hygiene: 10

Architecture Summary

Architecture Type: monolith

Key Modules: main, src

Runtime Flow:
Execution starts from 'main.py' as the main entry point. The 'src' directory likely contains the core application logic and modules. Given the absence of detected separate services or inter-module communication patterns, the system appears to run as a single process without explicit service boundaries or distributed interactions.

Security Findings

- **Info:** No hardcoded secrets found in the main entry point or scanned files.
- **Medium:** Cannot determine insecure configuration or weak authentication/authorization due to lack of configuration files and no detailed source code provided.
- **Medium:** No detection of unsafe or exposed endpoints due to absence of detailed code or endpoint information.
- **Medium:** No explicit OWASP Top 10 vulnerabilities identified from the given scan and architecture summary due to limited source code access.

Performance Findings

- **Info:** No explicit N+1 query patterns detected in the scanned source code files.
- **Info:** No missing pagination detected for API endpoints or database queries in the scanned source code files.
- **Info:** No blocking or synchronous I/O operations detected in the scanned source code files.

Engineering Roadmap

Immediate Fixes

- **P0:** Ensure no hardcoded secrets exist in the entire codebase by scanning beyond main.py

Short Term

- **P1:** Perform manual code review and provide additional source code and configuration files for authentication and authorization modules
- **P1:** Analyze src directory modules for endpoint definitions and validate authentication and authorization checks on those endpoints
- **P1:** Maintain asynchronous I/O operations and monitor database query patterns during ongoing development to prevent performance degradation
- **P1:** Implement pagination for API endpoints or database queries returning large datasets as new features are developed

Medium Term

- **P2:** Conduct thorough static and dynamic security assessments on the full source code to identify OWASP Top 10 vulnerabilities
- **P2:** Consider introducing modular architecture or service boundaries in the monolith to facilitate scalability and improve maintainability