Agentic CV Verification System **Final Version**

1 Task Definition and Design Target

This homework asks for an agentic AI verification system that reads CV PDFs, queries social profiles through MCP tools, detects inconsistencies, and outputs reliability scores in $[0, 1]$. The grading has two parts: system design quality and testing performance on unseen CVs. Because the decision rule in the notebook uses a threshold of 0.5, the practical engineering target is not only generating explanations but also producing stable, well-calibrated scores around this threshold.

The current implementation is the second version of my solution. I first built a fully rule-based baseline in hw2_solution.py, then redesigned the pipeline to improve matching robustness and reduce false contradictions caused by wrong profile linkage. This report documents the final version and explicitly explains why the redesign was necessary.

2 Final System Architecture

The final system has four connected stages: CV ingestion, candidate schema extraction, MCP evidence retrieval, and score synthesis. The ingestion stage reads each `CV_i.pdf` and normalizes text so downstream prompts and rules are less sensitive to formatting artifacts. The extraction stage builds a structured candidate object with name variants, city and country, title and company, education, experience, skills, and query keywords.

Evidence retrieval is implemented as two parallel channels, LinkedIn and Facebook. For LinkedIn, the agent calls `search_linkedin_people`, `get_linkedin_profile`, and optionally `get_linkedin_interaction`. For Facebook, it calls `search_facebook_users` and `get_facebook_profile`. The two channels are intentionally separated because they have different signal quality. LinkedIn is the primary professional evidence source, while Facebook is treated as auxiliary and heavily gated.

The synthesis stage combines deterministic consistency checks with LLM-based semantic judgment. Structured rules check agreement on company, school, title, location, and skills. The LLM receives compact JSON evidence and returns a score and short findings. The final score is blended as

$$s_{\text{final}} = 0.6\, s_{\text{rule}} + 0.4\, s_{\text{llm}},$$

which keeps outputs traceable while still capturing semantic contradictions that are hard to encode with fixed rules only.

3 Agent Workflow and MCP Tool Strategy

For each CV, the workflow starts by extracting a candidate schema from CV text. Search queries are then built from full name, name variants, and a small number of keywords. The retrieval logic does not rely on a single location string. Instead, it expands location into multiple candidates derived from city, country, and cleaned location fragments, which makes `search_linkedin_people` less fragile to formatting mismatch.

After collecting search hits, the system deduplicates IDs and performs ranking. The important design choice is that ranking is two-stage. Stage one uses lightweight similarity on search snippets to select a short list. Stage two fetches several top full profiles and computes CV-to-profile consistency on structured fields; the profile with the highest consistency is retained. If the best consistency is below threshold, the channel is treated as no reliable match rather than forced evidence. This mechanism directly addresses same-name collision cases.

The workflow also includes reliability controls for MCP access. Tool loading has retry and timeout protection, and individual tool calls use bounded retry with fallback behavior. This prevents transient protocol or network failures from collapsing the whole batch and improves reproducibility when running the notebook multiple times.

4 Scoring Logic and Decision Output

The scoring logic is designed for conservative judgment under partial evidence. A rule score is computed from structured consistency, with LinkedIn taking precedence when available and Facebook used as backup when LinkedIn is missing. Facebook evidence is included in the LLM context only when Facebook match confidence is high enough; low-confidence Facebook matches are excluded to avoid introducing noisy contradictions.

The LLM receives explicit instructions that missing public data should be treated as uncertainty rather than automatic failure. It returns `score`, `verdict`, and `key_findings`. The blend 0.6/0.4 between rule and LLM output gives decision stability while preserving semantic flexibility. Final decisions are generated by the required threshold rule: score > 0.5 is reliable, otherwise problematic.

This design also makes debugging straightforward. For each CV, the system stores selected profile IDs, rule score, short findings, and final score. When a misclassification happens, I can inspect whether the root cause is extraction error, bad profile linking, or scoring bias, then adjust that specific stage without rewriting the whole pipeline.

5 Why a New Version: Advantages over Version 1

The first version in `hw2 part1` was a clean deterministic baseline, but it had structural limitations. It relied on regex-style extraction, selected one best social candidate too early, and then applied a fixed penalty model (`base - penalty`). In ambiguous name scenarios, an early wrong profile choice could trigger many downstream contradiction penalties. This behavior made results sensitive to retrieval noise and location-string formatting. The first report already showed several near-threshold outcomes, which indicated limited margin even when final labels

looked acceptable on the small sample.

The new version was built to fix those failure modes directly. First, candidate extraction was upgraded from pure regex parsing to LLM-based structured extraction with deterministic fallback, improving robustness to diverse CV layouts. Second, retrieval moved from single-hit selection to multi-candidate profile fetching and consistency reranking, which reduces same-name mismatch risk. Third, low-confidence matches are explicitly gated out, especially for Facebook, so weak social signals do not dominate decisions. Fourth, scoring moved from one-direction penalty subtraction to a calibrated fusion of rule evidence and semantic reasoning, which better handles mixed evidence and missing fields.

Compared with version 1, the new pipeline is more robust in three practical dimensions: identity disambiguation, tolerance to noisy or incomplete social data, and score calibration near the 0.5 boundary. This is the core reason for doing a second version. The redesign is not cosmetic; it is a targeted response to the specific error pattern observed in the first implementation, namely mis-linking and over-penalization under ambiguity.

Aspect	Version 1 (`hw2 part1`)	Final Version
CV extraction	Mostly regex and heuristics	LLM JSON extraction + fallback normalization
Profile selection	Choose one best search hit early	Fetch top profiles, rerank by consistency, then gate
Facebook usage	Participates in penalties directly	Low-confidence Facebook excluded from evidence
Score construction	Fixed `base - penalty`	0.6 rule + 0.4 LLM blended score
Robustness controls	Basic retry	Async retrieval, retries, timeout-aware fallback

Table 1: Key engineering differences between the first and final implementations.

6 Sample Verification Result

On the provided five CV files, the final notebook output is

$$\{\texttt{decisions} : [1, 1, 1, 0, 0],\ \texttt{correct} : 5,\ \texttt{total} : 5,\ \texttt{final\_score} : 1.0\}.$$

This exactly matches the given ground-truth labels under the required 0.5 threshold. In practical terms, the first three CVs are judged reliable and the last two are flagged as problematic.

Beyond this sample result, the main value of the final version is decision quality under ambiguity. The revised retrieval and gating design reduces accidental contradictions from wrong profile linkage, so score differences are more likely to reflect true CV–profile consistency rather than search noise.

7 Conclusion

This report presents the final version of an MCP-connected agentic CV verification system and explains the redesign from the first version. The new implementation improves robustness by combining better candidate extraction, multi-profile consistency reranking, confidence gating for noisy channels, and blended rule–LLM scoring. It satisfies the assignment requirements on architecture, workflow, and sample result reporting, and produces fully correct decisions on the provided five-CV evaluation set.