## ****Code Review****

## ****✅ Strengths in Your Current Setup****

* **Dynamic JD skill extraction** → Not hard‑coded, always based on the current job description and master skills CSV.
* **Skill aliases CSV** → Allows matching variants like "c plus plus" → "c++", with robust CSV loading that handles bad formats (WPS fixes included).
* **Strictness penalty** → Prevents high scores for completely unrelated resumes by reducing semantic similarity when keyword overlap is zero.
* **Fixed weights** → No confusing “Semantic Match Weight” slider; scoring is consistent for the same inputs.
* **Detailed explanations** → explain\_ats\_score() tells the user why they got the score.
* **Tone-aware recommendations** → get\_recommendations() now produces recruiter-style advice based on missing skills and similarity.
* **ATS-unfriendly content check** → Warns about tables, graphics, and HTML tags.

## ****How the Final ATS Score is Calculated****

Your ATS score is computed in ats\_score\_dynamic() in ats\_scoring.py using this

**Final Score = (sim\_weight × semantic\_similarity) + (key\_weight × keyword\_overlap)**

semantic\_similarity = SBERT cosine similarity between cleaned resume text and JD text (value between 0.0 and 1.0).

keyword\_overlap = Count of matched skills ÷ Count of JD skills (value between 0.0 and 1.0).

sim\_weight **&** key\_weight → You’ve fixed both at **0.5** now, meaning equal importance for semantic and keyword overlap.

## ****Strictness Factor****

Before the formula is applied, you have an extra step:

**if** overlap == 0: sim \*= strictness\_factor

If there are **no matched skills at all** (overlap == 0), then semantic\_similarity is multiplied by **0.5** (your default strictness\_factor), lowering the score for completely mismatched resumes.

## ****Step-by-Step Example****

**Semantic similarity** = 0.60 (SBERT says 60% similar in language/content)

**Keyword overlap** = 0.25 (25% of JD skills are found in resume)

**Weights**: sim\_weight = 0.5, key\_weight = 0.5

**Case 1 – Some skills match**

Final Score = (0.5 × 0.60) + (0.5 × 0.25) = 0.30 + 0.125 = 0.425 → 43%

**Case 2 – No skills match (strictness penalty applies)**

Strictness penalty: 0.60 × 0.5 = 0.30

Final Score = (0.5 × 0.30) + (0.5 × 0.00) = 0.15 + 0.00 = 0.15 → 15%

This **prevents unrelated jobs** (e.g., Java JD vs Python resume) from scoring high just because of generic wording.

## ****Output Bundle****

ats\_score\_dynamic() returns: **Final ATS score** (0–100)

**Details dict**: similarity → final semantic similarity (after strictness penalty if applied)

keyword\_overlap → fraction of JD skills found in resume

strictness\_factor\_applied → True/False

**Matched skills** list

**Missing skills** list

**JD skills** list (for explanation/recommendations)

These are then used in:

explain\_ats\_score() → human-readable reason for score

get\_recommendations() → condensed recruiter advice

**Bottom line:**  
Your score is **50% semantic match + 50% skill keyword overlap**, with a **penalty** if **keyword overlap is zero**, and capped missing skills output so feedback is actionable.