Abode Money
1909 Alexander Ave. #1119
Austin, TX 78722

Technical Screen: Name Verification Application

Goal

Build a small application with two independent capabilities:

1. **Target name generation** from a user-provided prompt
2. **Candidate name verification** against the **latest generated target name**

The key design constraint is that the **verifier must treat the generator as a black box**. It must not "cheat" by reusing generator chat context or calling back into the generator to decide whether a candidate matches.

Functional requirements

1) Target Name Generator

What the user can do

- Provide a free-form prompt (e.g., "Generate a random name with …")
- Trigger generation
- See the resulting **target name** (a single string)

Behavior

- Each generation produces exactly **one target name string**
- The app maintains the concept of the **latest generated target name**
- Generating again updates what "latest" means

Example prompt

> "Please generate a random Arabic sounding name with an Al and ibn both involved. The name shouldn't be longer than 5 words."

Notes

- You may use an LLM, a deterministic generator, or any other approach.

2) Name Verifier

What the user can do

- Input a candidate name
- Trigger verification
- Receive a result that includes, at minimum:
  - `match` (boolean)
  - `confidence` (numeric score, e.g., 0–1 or 0–100)
  - a short `reason` / `explanation`

Behavior

- The verifier checks the candidate name against the **latest generated target name**
- If no target name has been generated yet, return a clear error state

Critical constraint: “Black box generator”

The verifier must be implemented as if:

- The generator is **not an open, available resource** during verification
- The verifier **may only access** the **latest generated target name string**
- The verifier **must not** rely on:
  - generator chat history/context
  - calling the generator again (directly or indirectly) to evaluate a match
  - “hidden” shared LLM state between generator and verifier

How the latest target name is exposed to the verifier (in memory, file, database, API, etc.) **does not matter**—only that the verifier is **architecturally isolated** from generator internals/context and operates using only the latest name string.

Implementation requirements

Minimal interface

You can implement this as a web app, CLI app, desktop app, etc. The only requirement is that a reviewer can:

- provide a prompt → generate a target name
- provide a candidate name → verify against the latest target name and see output

Matcher evaluation

You will be graded on matching behavior using a test suite based on a list we provide later:

- `(target_name, candidate_name, should_match)` examples
- Your verifier should produce results consistent with those expectations

You're free to implement any matching strategy you think is appropriate, as long as it:

- produces deterministic outputs (or at least stable/consistent ones)
- returns `match` + `confidence` + explanation
- can be tested

Documentation

No need for any as long as it's clear how to run the application

Test Cases

Expected Matches

#	Target Name	Candidate Name	Expected Result	Rationale
1	Tyler Bliha	Tlyer Bilha	Match	Minor transposition and misspelling in both first and last name
2	Al-Hilal	alhilal	Match	Hyphen and casing differences only
3	Dargulov	Darguloff	Match	Common phonetic suffix variation (v vs ff)
4	Bob Ellensworth	Robert Ellensworth	Match	Common nickname vs formal name
5	Mohammed Al Fayed	Muhammad Alfayed	Match	Spacing and transliteration variance
6	Sarah O'Connor	Sara Oconnor	Match	Apostrophe removal and vowel simplification
7	Jonathon Smith	Jonathan Smith	Match	Common spelling variant of first name

8	Abdul Rahman ibn Saleh	Abdulrahman ibn Saleh	Match	Spacing variation within compound name
9	Al Hassan Al Saud	Al-Hasan Al Saud	Match	Minor consonant simplification and hyphenation
10	Katherine McDonald	Catherine Macdonald	Match	Phonetic first name and common Mc/Mac variation
11	Yusuf Al Qasim	Youssef Alkasim	Match	Transliteration differences in Arabic-derived names
12	Steven Johnson	Stephen Jonson	Match	Phonetic spelling differences in both names
13	Alexander Petrov	Aleksandr Petrof	Match	Slavic transliteration and phonetic variation
14	Jean-Luc Picard	Jean Luc Picard	Match	Hyphen removal
15	Mikhail Gorbachov	Mikhail Gorbachev	Match	Alternate transliteration endings
16	Elizabeth Turner	Liz Turner	Match	Common nickname shortening
17	Omar ibn Al Khattab	Omar Ibn Alkhattab	Match	Case, spacing, and compound-name variance
18	Sean O'Brien	Shawn Obrien	Match	Phonetic first name and punctuation removal

Expected Non-matches

#	Target Name	Candidate Name	Expected Result	Rationale
19	Emanuel Oscar	Belinda Oscar	No Match	Same last name but entirely different first name
20	Michael Thompson	Michelle Thompson	No Match	Similar-looking but distinct first names
21	Ali Hassan	Hassan Ali	No Match	Token order swap changes identity
22	John Smith	James Smith	No Match	Different common first names
23	Abdullah ibn Omar	Omar ibn Abdullah	No Match	Reversal of patronymic meaning
24	Maria Gonzalez	Mario Gonzalez	No Match	Gendered name difference
25	Christopher Nolan	Christian Nolan	No Match	Similar prefix but distinct names
26	Ahmed Al Rashid	Ahmed Al Rashidi	No Match	Different surname root
27	Samantha Lee	Samuel Lee	No Match	Different first name despite shared root
28	Ivan Petrov	Ilya Petrov	No Match	Distinct given names in same cultural group
29	Fatima Zahra	Zahra Fatima	No Match	Name order inversion changes identity
30	William Carter	Liam Carter	No Match	Nickname not universally equivalent without explicit mapping