compliance assist

Step 3: Build Test Dataset

Purpose and Importance

Design and assemble a comprehensive test dataset of names to challenge your screening system. The quality and representativeness of your test data will directly determine the meaningfulness of your test results. Regulators expect firms to validate systems with both real and simulated scenarios. The EBA explicitly mentions testing the system against actual sanctions list data and "manipulated sanctioned records" to verify fuzzy matching capabilities. That means you should include not just exact sanctions list entries, but also variations and misspellings that might occur in real life. Additionally, including "clean" names (that should not match) is critical to testing for false positives. In short, the test dataset should reflect the spectrum of cases the system will encounter, true hits, near-hits and clear non-hits. A well-crafted dataset will help you identify if the system is missing any sanctions (false negatives) or flagging too many innocent parties (false positives) and it provides tangible evidence of system performance.

Key Activities and Decisions

- **Include known sanctioned names (true positives)**: Gather a list of names of individuals and entities that are on sanctions lists. Prioritise those from lists relevant to your business (e.g. if you operate in UK/EU, focus on those, include U.S. OFAC names if you have U.S. exposure or just as additional challenge cases). You don't need every name on the list, a representative sample is fine, but ensure you have a mix (e.g. some well-known terrorists, some less common designated persons, a few entities). Make sure to pull the exact spelling as it appears on the official list. This forms the core of cases that should trigger alerts.
- **Create variations and aliases (fuzzy matches)**: For each sanctioned name you include, think of plausible variations a customer or criminal might use or that might appear due to data entry inconsistencies. These manipulated records will test the strength of fuzzy matching. Regulators (and criminals) are aware that list names are often matched with slight differences, so your system must handle these. Be creative but realistic, aim for variations that could genuinely occur:
    - **Spelling differences**: e.g. "Mohammad" vs "Mohamed" vs "Muhammad"; "al Qaeda" vs "al-Qaida".
    - **Name order changes**: e.g. "Smith John" instead of "John Smith" (common in some databases where last name might come first).
    - **Alternate transliterations**: especially for names originally in non-Latin scripts (Russian, Arabic, Chinese). e.g. the Russian name "Дмитрий Иванов" might appear as "Dmitrii Ivanov" or "Dmitry Ivanoff" in English. If your system is supposed to catch those, include them.
    - **Common aliases/nicknames**: e.g. if list has "Aleksandr (Alex) Petrov", test "Alex Petrov".
    - **Typographical errors**: one-letter off scenarios ("John Smyth" vs "John Smith"), missing spaces ("McCartney" vs "Mc Cartney"), swapped letters, etc.
- **Gather clean names (true negatives)**: Compile a set of names that are not sanctioned and should not cause alerts. These are equally important to test false positives. Choose names that are similar to sanctioned ones to really test the boundaries. For example, if "Robert Mugabe" is sanctioned, try "Robert Mugabee" or "Robert Mugabe Jr." as a clean test. If "Korea National Bank" is sanctioned, test "Korean National Bank" if that's an unrelated entity. Also include very common names (e.g., "Maria Garcia", "John Smith")

which might coincidentally match parts of sanctions entries, to see if your system over-flags them. Essentially, these are your negative control group.

- **Incorporate edge cases**: Think of unusual or challenging scenarios. These edge cases ensure you test beyond the "typical" Western first-name/last-name format and cover different patterns:
    - **Short names or single names**: e.g. "Umar" (some cultures have single names. Also, short names might match many others).
    - **Names with special characters or diacritics**: e.g. "José González" (with accent, vs Jose Gonzalez without).
    - **Compound or hyphenated names**: e.g. "Ali Hassan al-Majid" (and try splitting or altering it).
    - **Transposed parts**: e.g. parts of name reversed or middle name dropped/added.
    - **Entities with common words**: e.g. "General Trading Company" (see if "General" triggers anything weird).
    - **Alternative name orders/cultural naming conventions**: For instance, Chinese names might be presented as last name first in some systems and Westernised in others, include an example to see if order matters.
- **Determine quantity**: Decide on how many test cases to include. Quality is more important than sheer quantity. A typical robust test set might be on the order of 50-200 names, depending on complexity. You want enough to cover all scenario types identified, possibly with a few examples of each. If resource allows, err on the side of caution by including additional data, especially for variations (as something unexpected might pop only on a certain variant).
- **Associate each test name with expected outcome**: Create a table or spreadsheet with one row per test case: Test Name and Expected Result. Expected result can be "Alert – should match [Sanctions List Name]" or "No Alert". If possible, indicate which list or entry it should match (e.g. "Expected alert – matches UK Sanctions list entry John Smith (UID 1234)"). This will be crucial for analysis later. Essentially, this is the answer key by which you will grade the system's performance.
- **Review test data for realism**: Double-check the list with colleagues. Does each variation make sense? Are the clean names truly not sanctioned? (Ensure none of your "clean" names accidentally is on a list, that would confuse the results.) Make sure no test case is offensive or real customer data (unless anonymised/consented). Also, confirm you're testing names appropriate for your business context (e.g. if you don't deal with North Korean entities at all, you might still test one name since regulators expect it, but focus more on names from regions you deal with frequently).
- **Plan data format and loading**: Depending on how you'll run the test (next step), consider format. Often a CSV or Excel of names works or a text file. Include any secondary fields if needed to mimic real input (some systems require a full customer record with fields like country, DOB, if so, you might generate dummy data for those but with the name being the key piece). Ensure the format you prepare is compatible with how you will input it.

compliance assist

Practical Tips and Examples

- **Use official sources**: For accuracy, pull names exactly from official publications. Government's provides sanction lists, you can copy a few entries directly. The UK sanctions list, EU and OFAC SDN are all downloadable. Using these ensures you have the correct spellings and identifiers.
- **Segment your dataset**: You might tag each test case as, "List exact", "Variant", "Clean similar", "Edge case" etc. This helps later to ensure you covered each category and to analyse patterns ("all my misses were in the 'variant' category for Arabic names").
- **Automation for variants**: Consider using tooling to generate variants. Simple scripts can perform letter swaps or remove accents. However, ensure generated names still look plausible. For example, a script could take "Alexander" and remove one letter to test tolerance ("Alexnder"), but "Alexnder" might be less realistic than a common typo like "Aleksander". Use human judgment to refine automated suggestions.
- **Leverage internal data (carefully)**: If allowed, you can use some anonymised actual customer names that have historically caused issues or that you suspect might be tricky. For example, if you have a client named very similarly to a sanctioned individual, include an anonymised version of that scenario as a test (to see if your system can distinguish them). Just be sure to anonymise or get necessary permissions, since you don't want to expose real personal data in testing materials unnecessarily.
- Example test cases:
    - **Exact sanctions hits**: "Nicolás Maduro Moros" (Venezuela), "Islamic State of Iraq and the Levant (ISIL)" (entity), "Kim Jong Un".
    - **Variants**: "Nicolas Maduro" (no accent, missing second surname), "Islamic State of Iraq & Levant" (ampersand instead of 'and'), "Kim Jong-Un" (with hyphen).
    - **Transliterations**: "Dmitry Ivanov" vs listed "Dmitrii Ivanov", "Mohammed bin Laden" vs listed "Muhammad Bin Laden".
    - **Clean lookalikes**: "Maduro Motors LLC" (should be no hit though contains 'Maduro'), "Islamic Society of London" (not ISIL), "Kim Jun Un" (a similar-looking but non-sanctioned name).
    - **Edge**: "Prince (single name)", "Özil, Mesut" (name with diacritic), "Banks, Incorporated" (tests if "Bank" triggers anything).
- **Consult typologies**: If you have access to any typology reports or red flag lists (e.g. known alias patterns used by sanctioned parties to evade detection), incorporate those patterns. For instance, sanctioned Russian oligarchs often have patronymic names, test with and without the patronymic.
- **Prepare for expansion**: You might not use every test case in the first round (depending on scope defined). It's okay to have a larger pool and decide to execute a subset now and others later (just note what you did). Keep the extras, they might be useful in future or if you extend scope.

compliance assist

Common Pitfalls

- **Too few test cases**: If your dataset is too small or only very straightforward names, you may miss discovering issues. For example, testing only exact list names might show everything is fine, while in reality slight spelling differences fail. Push the boundaries with enough cases to truly challenge the system.
- **Unrealistic scenarios**: On the flip side, avoid overly contrived test cases that would never happen, as they can waste time or confuse analysis. For instance, testing "ZZZ Randomname" isn't helpful unless there's a reason. Each test should have a rationale. Regulators expect "reasonably expected" variations, not random gibberish.
- **Not testing all relevant lists**: If your system is supposed to screen against multiple lists (UK, EU, OFAC, UN), include examples from each to ensure none are being missed. A pitfall is assuming "if it catches OFAC names, it'll catch EU too", maybe not if list handling is flawed.
- **Mixing sanctioned and clean data in one test without labels**: If you feed names in and later get a jumble of alerts, it can be hard to recall which were supposed to hit. This is why the expected outcome mapping is crucial. Not having a clear expected result for each case can lead to confusion and misinterpreting results.
- **Security and privacy oversight**: Using real customer names or personal data without anonymisation could violate privacy policies or regulations. Always sanitise test data. Additionally, remember that test data might be handled by IT or others, treat any sensitive aspects appropriately (e.g. if you included a high-profile name, handle with need-to-know).
- **Forgetting special list requirements**: Some sanctions lists include not just names but other identifiers (DOB, passport). While your test primarily focuses on names, if your system uses multiple fields for matching, you might need to simulate that. If your objective is to test name-matching only, fine, but know your system's normal operational logic (so you interpret results correctly).
- **Outdated test data**: If you took names from a list a while ago, update them. Sanctions lists change frequently. Ideally, build the dataset using the latest lists available, so that your test also implicitly checks that the system's lists are up to date. (E.g. if a major name was added last week and your system misses it, that's a problem to catch.)

compliance assist

Checklist – Review Current Setup

- ☐ **Objectives clearly documented**: A written test plan or charter lists the specific goals of the testing exercise in clear terms.
- ☐ **True hit cases selected**: A set of actual sanctioned names (individuals and/or entities) from relevant sanctions lists is included.
- ☐ **Variations/aliases prepared**: For each selected sanctions name (or as many as feasible), plausible alternate spellings, transliterations and aliases have been created to test fuzzy matching.
- ☐ **Clean (non-sanction) cases included**: A set of names not on sanctions lists, especially those similar to sanctioned names or very common names, is included to test for false positives.
- ☐ **Edge cases covered**: Names with unique characteristics (special characters, single names, very short or very long names, hyphenated names, etc.) are included to challenge the system's handling of different formats.
- ☐ **Expected outcome defined**: Each test name is annotated with the expected result (e.g., "should alert – matches [list]" or "no alert"). Optionally, the target sanctions list entry or reason is noted for expected alerts.
- ☐ **Dataset reviewed for completeness**: The team cross-checks that test cases collectively address all objectives and scenario types identified in the plan (no major pattern omitted).
- ☐ **No unintended sanctions in clean set**: Double-verify that "clean" names are truly not sanctioned (to avoid misinterpreting a correct alert as a false positive).
- ☐ **Data formatted for input**: Test names (and any necessary additional fields) are compiled in a format suitable for the test execution (spreadsheet, CSV, test case tool, etc.).
- ☐ **Sensitive data handled**: Any use of real or personal data in test names is avoided or properly anonymised. The test dataset does not violate privacy or security guidelines.
- ☐ **Backup retained**: Save a copy of the test dataset and expected outcomes list in a secure location, this will be part of your documentation and needed if you re-run tests later.