Step 6: Tune and Re-Test

Purpose and Importance

In this step, you implement improvements to the screening system based on the issues identified, and then re-run tests to validate that those changes have resolved the problems. This is where you operationalise the insights from your analysis. Regulators treat this tuning process very seriously, the EBA's guidelines explicitly make "calibration" (tuning) a regulatory expectation and even before these guidelines, it was understood that sanctions screening is not a "set and forget" system. The FCA has noted that firms should continually seek to enhance their systems to strike the right balance. The goal of tuning is to reduce false negatives to zero (or as close as humanly possible) while optimising and reducing false positives as much as you can without introducing new risks. Because tuning can sometimes have unintended effects (fixing one thing can break another), re-testing is crucial to ensure that each change indeed improves the system and doesn't cause new problems. This iterative test-adjust cycle might need multiple rounds until you're satisfied that objectives are met. By the end of this step, you should have an updated configuration that demonstrably performs better, with evidence to show regulators that you took corrective actions as part of a controlled, well-documented process.

Key Activities and Decisions

During tune and re-test:

- **Develop a targeted action plan**: From the analysis, decide on specific changes to make. Prioritise critical fixes first (e.g. anything causing misses). Your plan might include:
    - Adjusting matching algorithm settings (e.g. thresholds or sensitivity levels). For example, if names with one-letter differences were missed, you might lower the match threshold or increase fuzziness.
    - Enabling or configuring additional matching features (e.g. turning on phonetic matching or expanding the dictionary of equivalent characters if such options exist).
    - Editing or updating reference data: Perhaps adding missing aliases or alternative names to your sanctions list feed manually (as a stop gap if the official source lacked them) or removing overly broad entries if appropriate and allowed.
    - Modifying ignore lists or stop-words: If you found that ignoring "Ltd" or common words contributed to misses, you might remove or alter that rule.
    - Introducing new rules: e.g. if false positives are high for common names, implement a rule that requires a secondary identifier match (like date of birth) for those to alert, if your system can do that.
    - Fine-tuning specific fields: e.g. treat last name matching differently than first name if needed or adjust how abbreviations are handled.
    - If an issue is data-related (like a list omission), plan how to rectify that source or include supplemental data. If an issue is process-related (like list update frequency), plan to change the process as part of overall remediation (though that might be outside the system config tuning).

Write down each intended change, linking it to the issue it addresses. For example: "Issue: Missed hyphenated names, Action: Remove hyphen from delimiter list or treat ' ' and '-' equally in matching logic." and "Issue: Too many false positive on 'Ltd', Action: Remove 'Ltd' from ignore words so it contributes to match (reducing partial matches)".

- **Change one thing at a time (ideally)**: To the extent possible, implement one change and test, rather than many changes at once. This isolates the effect of each change. For instance, if you lower the threshold and also update the list data at the same time and then things improve, you might not be sure which change was the key. If constraints force multiple simultaneous changes, be extra diligent in analysing the subsequent test to ensure each issue is indeed resolved.
- **Obtain necessary approvals**: Before applying changes, follow your governance. This might mean getting sign-off from a compliance head or documenting the change request formally. Regulators want proof that tuning is controlled. A simple approach, document the changes in an email or change log and have the relevant authority say "Approved to proceed." If changes are substantial (e.g. completely different algorithm or turning off a list), weigh the compliance implications and ensure risk owners approve.
- **Implement configuration changes**: Work with IT or directly on the system (if you have access) to make the adjustments. Do this in the test environment (or a configuration clone) first. Document exactly what you did (e.g., "Threshold X changed from 80 to 70 on [date] by [user]"). This will go into your change log for audit purposes. If it's a vendor system and something isn't user configurable, liaise with the vendor for a solution or patch.
- **Run re-tests to validate changes**: After each set of changes, re-run the relevant test cases to see if the issue is fixed. You don't always need to run the entire dataset, especially if changes are specific. For example, if you fixed hyphen handling, re-run the hyphenated name tests to check they alert now. If you lowered threshold to catch one-letter-off names, re-run those particular cases that failed before. Also, rerun a sample of other cases (including ones that passed before) to make sure they still pass (no new false positives or erroneous behaviour, a mini regression test). Take notes:
    - Did the false negatives get eliminated? If not fully, you may need to adjust further or try a different approach.
    - What happened to the false positives? Perhaps threshold change might increase false positives, check if any previously clean cases now started alerting unexpectedly.
    - Any new anomalies? Occasionally, a change can cause something odd to appear (e.g. enabling phonetic matching might suddenly flag a clean name that sounds like a sanction, note if that happens).
- **Iterate if needed**: It's common that one round of tuning won't perfectly solve everything. Maybe you fixed all misses, but false positives went up slightly, you might then tweak another setting to counterbalance. Or you address top priority issues first, then move to secondary ones. Each iteration, run tests (you can run the whole set or targeted subsets depending on confidence). Keep iterating until:
    - All critical false negatives are resolved (this is a must before concluding).

  - False positives are reduced as much as feasible given the false negative constraint (this is often a balancing act. If you've dialled up the system to catch all, now see how much you can tighten without reintroducing misses).
  - The outcomes meet the success criteria in your objectives (or very close, with conscious acceptance of any minor deviation if necessary).

- **Consider external input for complex fixes**: If you encounter an issue that isn't easily solved (e.g. an inherent limitation of the system), reach out for advice. Vendor support might have suggestions or a patch. Or an external consultant might confirm "this system can't do X well." If something cannot be fixed via tuning (rare in name screening, but possible), document that and consider compensating controls (outside the system) or note it as a requirement for a future system upgrade.
- **Maintain a change log**: Throughout this step, maintain a detailed log of changes. Include:
  - What was changed (setting name, new value vs old value).
  - Why it was changed (tie to issue).
  - Who approved and who implemented.
  - Date of change. This change log is essential for documentation and proving that you managed changes properly. It will be part of your audit trail (and help if you need to undo anything).
- **Plan deployment to production**: Once satisfied in the test environment, plan to roll out these changes to the live system. Follow normal change management protocols (schedule a maintenance window if needed, communicate to users, etc.). Ensure production gets the same changes and update your documentation that production now has these new settings as of X date.
- **Update documentation**: As you tune, you might update parts of your policy or procedures. For instance, if you change threshold values, update any internal documentation that references them (like a model document). Also, draft notes for the final report about what you changed and the effect on results.

Practical Tips and Examples:

- **One change at a time – example**: Suppose analysis showed two problems: (a) threshold too high causing misses on 2 names, and (b) a stop-word causing misses on hyphenated names. Change (a) first: lower threshold from 80 to 75. Re-run those 2 names plus a few others. If they alert now (good) but you notice maybe 1 new false positive popped up, hold that thought. Then change (b): adjust stop-word handling. Re-run hyphen names. Now if all good, re-run the entire set to see the combined effect. Because you isolated changes, if something's off, you know which change likely caused it.
- **Peer review each change**: If working in a team, have one person suggest a change and another sanity check it. Example: "If we lower threshold, are we sure it won't flood us with many trivial matches?", maybe test a borderline case or two to be sure.
- **Gradual tuning**: If lowering a threshold, sometimes do it in small increments and test each time. E.g. go from 80 to 75, test, if still missing one case at 75, go to 70, etc. rather than a big jump which might go too far.

- **Think creatively**: There might be more than one way to address an issue. E.g. for false positives on common names, one approach is raising the threshold (risking misses), another is requiring an extra data point match (DOB) for common names, another is maybe whitelisting certain known innocuous names if appropriate. Choose an approach that fits your system's capabilities and regulatory comfort. Document why you chose it.
- **Don't forget list updates**: If you discovered your list source lacked aliases, a quick fix might be to manually add those aliases to a local list. Or switch to a better source if possible. For example, "Added alias 'X' to internal watchlist to ensure system catches it until vendor feed includes it." Just be sure to manage that so it stays updated.
- **Testing for unintended effects**: After tuning, test not only your original dataset but also consider if there are any new scenarios you should test given changes. For example, if you enabled phonetic matching, test a name that is phonetically similar to a sanction but not actually (to see if it creates a new false positive).
- **Example of iterative improvement**: Initially, 5 misses and 10 false positives. After first tuning round, 0 misses and 15 false positives (some new false positives came in). After second tuning (e.g. adding a minor rule to cut a false positive pattern), 0 misses and 8 false positives. Perhaps that meets your goal (if you aimed for, at most 10 false positives). If you think 8 can be pushed lower without introducing risk, you can try another tweak but be mindful of diminishing returns.
- **When to stop tuning**: In theory, you could keep tweaking indefinitely. Set a reasonable threshold. No misses (non-negotiable) and false positives reduced to a level stakeholders can manage/accept (perhaps comparisons to industry benchmarks or internal capacity help determine this). Once you hit that, plus your objectives and additional tweaks start having negligible benefit or too high risk of new misses, it's time to conclude tuning for now. Document that rationale.
- **Ensure independence in validation**: Ideally, an independent party (like a compliance officer not involved in tuning) observes or double-checks the re-test results to confirm improvements. This can be as simple as them reviewing the before and after metrics. It adds credibility that the tuning was successful and unbiased.

Common Pitfalls:

- **Over-tuning (causing new gaps)**: In efforts to eliminate false positives, you might make settings so tight that you inadvertently create new false negatives. Always re-check for that. Sanctions compliance errs on side of caution. Regulators prefer you handle some false positives rather than miss a true hit.
- **Fixing symptoms, not causes**: E.g. if the system missed "Mohamed" vs "Mohammed", simply whitelisting that one instance solves it for that name but not the general problem. It's usually better to adjust the algorithm to handle double vs single M generally, rather than patch one name. Use whitelists/blacklists sparingly, broad solutions are more robust.
- **Not documenting changes and rationale**: This cannot be overstated. If an auditor later asks "why did you lower the threshold?" and no one remembers, it's a governance fail. Making changes quickly and forgetting to log details is a pitfall. Maintain the discipline of updating the change log in real-time.
- **Ignoring system constraints**: Some tuning might be limited by the system. If the tool doesn't support a needed feature (say phonetic matching for certain scripts). You might need to note that as a limitation

and possibly plan a longer-term upgrade or vendor request. Don't spend excessive time chasing an impossible fix. Implement a temporary workaround and mark it for future improvement.

- **Communication breakdown**: If your changes significantly alter alert volume or nature, inform the team that handles alerts. For instance, after tuning you might get more alerts of type X and fewer of type Y, the team should be prepared for that shift. Not telling them could lead to confusion or mishandling of alerts.
- **Rushing without validation**: Declaring victory after changes without proper re-test is a big risk. For example, "We think lowering threshold fixed it", you must prove it by re-testing. Regulators will ask, "How did you confirm the changes worked?" and you need that evidence.
- **Changing too many things at once**: As mentioned, that can obscure cause and effect. It also raises risk of a mistake. If you must change multiple settings (say vendor applies an update that bundles changes), try to test extensively after to ensure all known scenarios still hold.
- **Neglecting policy/doc updates**: If your sanctioned screening methodology document says "threshold at 80" and you changed it to 70, update that doc to avoid inconsistencies in documentation.
- **Not engaging governance for significant changes**: For example, if you decide to remove a sanctions list from the screening (maybe because it was producing only false positives and not legally required for you), that's a major decision that requires approval beyond the project team. Ensure such strategic changes are vetted at the right level, otherwise, you could face internal or external criticism.

compliance assist

Checklist – Tune and Re-Test:

- [ ] **Action plan derived**: A concrete list of configuration changes (tuning adjustments, data updates, rule changes) has been formulated to address each identified issue.
- [ ] **Approvals obtained**: All planned changes have been reviewed and approved according to internal governance (documented via email or change request forms with management sign-off).
- [ ] **Changes implemented (test env)**: The system configuration has been updated in the test environment as per the plan. Each change is logged with details (old value -> new value, date, by whom, why).
- [ ] **Focused re-testing performed**: After changes, relevant test cases (especially those that previously failed) were run again. Results confirm that intended fixes are effective (e.g., previous false negatives now alerting, false positives reduced).
- [ ] **No new issues introduced**: Re-testing of a broader set (or full set) of cases confirms that changes did not cause new false negatives or other unintended consequences. Any minor new false positives or quirks have been noted and are deemed acceptable or will be handled by further tuning.
- [ ] **Iterative adjustments made**: If the first change didn't fully resolve an issue or caused a side-effect, further tuning was done. The process repeated until all critical issues were resolved and performance goals met. Each iteration's changes are also documented.
- [ ] **Objectives achieved or aligned**: Compare post-tuning results with the original objectives. Ensure all critical objectives (like "no misses") are achieved. Efficiency targets are met or a rationale is documented if there's a slight shortfall (with acceptance by risk owners).
- [ ] **Final configuration set**: The team is confident that the new configuration is the optimal balance for now, having eliminated serious gaps and improved efficiency. This is now ready to be promoted to production.
- [ ] **Change log completed**: The comprehensive log of all changes made during tuning (with date, description, rationale, approval) is finalised. This will be part of reporting.
- [ ] **Production deployment prepped**: A plan is in place to apply the tested changes to the live system (or it has already been done if appropriate), including scheduling, communications, and any necessary monitoring of the first days of the new settings.
- [ ] **Stakeholders informed**: Key stakeholders (compliance leadership, alert handling team, etc.) are informed of the successful tuning and any notable differences in how the system will behave going forward (like expected reduction in false positives, etc.).

- [ ] **Ready for documentation & audit**: The system is tuned, and all evidence of changes and improved results is collected, setting the stage for final documentation in Step 7.