PROJECT 2 OF 5 | START HERE LEARNER GUIDE

Continuous Evidence & Control Monitoring

Engineer assurance

Harborline Health still collects screenshots when an auditor asks whether privileged production access has MFA. Leadership wants a repeatable evaluation from source data, with evidence someone else could re-perform. You will design and prototype that monitoring workflow, then find out what a green dashboard was hiding.

> **Companion video**
> Watch the walkthrough before you begin: https://youtu.be/SdpSWPNzRb0

Your project folder

Item	What it is for
START_HERE_Learner_Guide.pdf	This guide. Read it stage by stage.
Master_Workbook.xlsx	Your main workspace. Every stage has a tab.
1_Assess_Evidence	Current-state evidence for Stage 1.
3_Build_Source_Data	The eight source extracts, the stated control brief, and the data dictionary.
4_Break_Incident_Evidence_Open_After_Build	Incident evidence for Stage 4. Do not open until Build is complete.
Deliverable_Templates	Word templates: incident analysis (Stage 4) and Control Health Brief (Stage 6).
Optional_Python_Path	Optional. The Python version of the Build stage.
Optional_Deep_Dive_Export_Replay	Optional Deep Dive. A later auth extract to prove your monitor is data-driven.

Contents

Welcome to Project 2 . . . 3
Stage 1: Assess . . . 6
Stage 2: Design . . . 8
Stage 3: Build . . . 10
Stage 4: Break . . . 13
Stage 5: Improve . . . 15
Stage 6: Defend . . . 16
Stage 7: Career Translation . . . 17
Before you call this finished . . . 18
Optional: the Python Build path . . . 19

Welcome to Project 2

Read this once before Stage 1. It takes about ten minutes.

The scenario

Harborline's control **HL-CC-PRIV-MFA-01** says privileged production access requires MFA (multi-factor authentication). Today GRC asks Engineering for screenshots each quarter, and a weekly job reports 100 percent pass. Nobody can say, without a hunt, who was in the privileged production population last Tuesday, or whether MFA was actually required for that population.

You will build a monitor for that control: a defined population, a four-status evaluation, approved treatments, unknowns, a control over the monitor itself, and an evidence package. You are not building a production continuous-controls-monitoring platform. Project 2 is self-contained; you do not need Project 1.

Words you need first

Term	Meaning in this project
MFA	Multi-factor authentication: a second proof of identity beyond a password.
Population	The exact set of identities the control must cover. A correct rule on the wrong population produces false assurance.
Monitoring versus testing	Testing samples evidence after the fact. Monitoring evaluates the defined population on a cadence from source data and keeps re-performable evidence. Continuous does not mean real-time.
PASS	The technical MFA requirement is met.
FAIL	The requirement is not met, the identity is evaluable, and there is no valid approved exception.
WATCH	The requirement is not met, but a valid approved exception exists. A risk treatment, never a technical PASS.
UNKNOWN	The control cannot be evaluated for this identity: evidence missing, blank, conflicting, or stale, not on the roster, or a non-human account.
Monitoring assurance	A control over the monitor itself: expected sources arrived, extracts are fresh, the population count is near baseline, and the run completed.
Evidence package	What a run keeps so another reviewer can re-perform it: population, sources, extraction times, rule version, results, treatments, and unknowns.
Review freeze	2026-09-15 12:00 UTC. All math uses this timestamp, never your computer clock.

Two different questions

Technical control effectiveness asks: for the identities we evaluated, is MFA actually met? **Monitoring population completeness** asks: did we evaluate every identity that should have been in scope? A 100 percent PASS rate answers the first question only. Keep the two apart throughout this project.

Choose your depth and path

Option	Time	What it includes
Standard	About 4 to 6 hours	Full loop, a working monitor with evidence package, and every portfolio-critical artifact.
Deep Dive	About 8 to 12 hours	Standard plus extra Assess rows, richer retention language, Challenge prompts or a second Build path, a run-trend discussion, interview defense, and the export replay.

Build path	Tool	Completes Build?
Spreadsheet	The Build tabs in **Master_Workbook.xlsx**	Yes. A complete path.
Python	Scripts in **Optional_Python_Path** (Python 3.10+, no installs)	Yes. Recommended for coding practice.

You need one Build path, not both. Every other stage happens in the workbook and the Word templates.

A five-day plan (optional)

Day	Stages	Time
1	Assess	45 to 75 minutes
2	Design	60 to 90 minutes
3	Build (the long day)	90 to 150 minutes
4	Break and Improve	45 to 75 minutes
5	Defend and Career Translation	60 to 90 minutes

What you will produce

#	Artifact	Where it lives
1	Current-state assessment	**1 Assess**
2	Control definition (frequency defended, at least one monitoring-assurance control)	**2A Control Definition**
3	Source-to-control map	**2B Source-to-Control Map**
4	Evidence requirements and retention	**2C Evidence Requirements**
5	Working monitor with PASS, FAIL, WATCH, and UNKNOWN	Build tabs or **Optional_Python_Path**
6	Control results and exception analysis	**3 Health Summary**
7	Break investigation	**4 Break Timeline** and **Deliverable_Templates/4_Break_Incident_Analysis.docx**
8	Revised monitoring design	**5 Improve** and the Revised columns in Design
9	Exactly two funded monitoring improvements	**6 Priorities**
10	Control Health Brief (rate versus confidence)	**Deliverable_Templates/6_Defend_Control_Health_Brief.docx**
11	Career Translation (Standard set)	**7 Career Translation**

Stage 1: Assess

Find out how MFA evidence is really gathered today.

What you are trying to accomplish

Read the current-state evidence and classify what you see, so your design starts from facts rather than from the binder.

Why this matters

Auditors will ask how you know the population is complete and whether the control operated through the period. Screenshots and a green weekly email cannot answer either question. Assess is where you notice that.

Open these

What	Where to find it
Eight current-state evidence files	**1_Assess_Evidence**
Last quarter's manual evidence campaign	Tab **Evidence_Campaign** (also **3_Build_Source_Data/evidence_campaign.csv**)
Prior weekly job summaries (skim)	Tab **Control_Runs** (also **3_Build_Source_Data/control_runs.csv**)
Your assessment table	**Master_Workbook.xlsx**, tab **1 Assess**

File	What it is
1_Assess_Evidence/Stated_Control.pdf	Binder wording of HL-CC-PRIV-MFA-01
1_Assess_Evidence/Stakeholder_Notes.pdf	GRC, Engineering, IT, and the external SOX tester
1_Assess_Evidence/Evidence_Request_Q2.pdf	How GRC asked for evidence last quarter
1_Assess_Evidence/Screenshot_Log.pdf	Text stand-ins for the screenshots GRC stored
1_Assess_Evidence/Prior_Test_Results.pdf	The last test and its conclusion
1_Assess_Evidence/Exception_Tracker.pdf	Treatments as Engineering tracks them today
1_Assess_Evidence/System_Inventory.pdf	Cloud accounts as the wiki lists them
1_Assess_Evidence/Audit_PBC_List.pdf	What the auditor will ask for

The six labels (pick one per row)

Label	Use it when
Observation	A fact. Not yet a judgment.
Control gap	A rule is missing, unclear, or does not cover the risk.
Evidence gap	You cannot support a conclusion with what is stored.
Data-quality issue	Files disagree, are stale, or are incomplete.
Exception	A specific case where a stated control did not operate.
Unknown	You cannot tell yet. Say what would let you tell.

Do this

1. Read every file in **1_Assess_Evidence**.
2. Skim **Evidence_Campaign**: five evidence requests from last quarter. Notice artifact types, dates, and notes such as reused screenshots. Skim **Control_Runs**: four weekly runs that all report a population of 12.
3. On **1 Assess**, add one row per issue: source, what you noticed, one label, why, and what would confirm or close it. Aim for at least eight rows using more than one label.
4. Write the two-sentence summary: what worries you most, and what you still cannot know.
5. **Deep Dive (optional):** add rows and list the evidence you would request next.

What you produce	You are done when
• Tab **1 Assess** with at least eight labeled rows and the two-sentence summary.	☐ At least eight rows, more than one label. ☐ Two sentences: biggest worry and biggest unknown. ☐ You have not opened the Break folder.

Next: Move to Stage 2: Design.

Stage 2: Design

Define how privileged production MFA should be monitored.

What you are trying to accomplish

Write a control definition that a colleague could evaluate without you, map every source to the control, and state what evidence each run must keep.

Why this matters

Frequency, population, and UNKNOWN handling are design decisions, not dashboard settings. If the definition does not say how completeness is tested, nobody will test it. If it does not separate treatments from passes, a health percentage will hide risk.

Open these

What	Where to find it
Control definition	Tab **2A Control Definition**, column **Original (Stage 2)**
Source-to-control map	Tab **2B Source-to-Control Map**
Evidence requirements	Tab **2C Evidence Requirements**, column **Original (Stage 2)**
Harborline's current stated control (input, not your design)	**3_Build_Source_Data/Harborline_Stated_Control_Brief.pdf**
Source file meanings	**3_Build_Source_Data/Data_Dictionary.pdf**

Do not confuse the two documents

3_Build_Source_Data/Harborline_Stated_Control_Brief.pdf is Harborline's **current** stated control and the rule set Build tests. **Your** design lives on the three Design tabs.

Do this

1. Read **3_Build_Source_Data/Harborline_Stated_Control_Brief.pdf**. It contains the stated rules, the review freeze, and the stated monitoring-assurance thresholds. Your design may keep weekly cadence, change it, change the population, or change UNKNOWN rules, as long as you explain why.
2. On **2A Control Definition**, complete every row in the Original column: objective, risk, population (including how you would test completeness), scope, authoritative sources (which wins when two disagree), frequency, the technical requirement, the four statuses, owner, exception handling, the monitoring-assurance control, known limitations, and retention.
3. For frequency, answer the four questions in the row: why this cadence, what risk would justify daily, what would make quarterly insufficient, and what changes if a source updates only once per day.
4. For the monitoring-assurance control, state the variance or staleness that creates a monitoring exception. Harborline's Build uses a 25 percent population-variance threshold. Would you keep it in a real privileged-access environment? Consider population size, privilege level, hiring and termination activity, system changes, acquisitions, source reliability, frequency, and risk tolerance. There is no single correct percentage.
5. On **2B Source-to-Control Map**, add rows until a reviewer can follow data from each extract to the brief: source, field, normalization, rule, evaluation, evidence kept, exception or UNKNOWN handling, and reporting.
6. On **2C Evidence Requirements**, say what evidence would prove population completeness, the evaluation, treatment handling, that a run executed, follow-up, retention, and what a screenshot cannot replace.

What you produce	You are done when
• Original columns on **2A Control Definition** and **2C Evidence Requirements**. • Rows on **2B Source-to-Control Map**.	☐ A colleague could evaluate the control from your definition alone. ☐ They would not treat an approved exception as a technical PASS. ☐ They could say how they know a run actually happened.

Next: Move to Stage 3: Build. The monitor tests Harborline's **stated** rules.

Stage 3: Build

Build a working monitor and evidence package.

What you are trying to accomplish

Evaluate the stated population against the stated MFA rules, apply approved treatments correctly, record unknowns, and run the monitoring-assurance check on the monitor itself.

Why this matters

A monitor is useful because it evaluates the population you defined, the same way, every run, and keeps evidence another person can re-perform. The skill is encoding the rules so 2,500 identities would not change the logic: load, define population, evaluate, apply treatment, check assurance, emit evidence.

Open these

What	Where to find it
Stated rules and thresholds	**3_Build_Source_Data/Harborline_Stated_Control_Brief.pdf** and tab **3 Build Guide**
Source data (spreadsheet path)	**Cloud_Entitlements**, **Privileged_Roster**, **Auth_Config**, **Approved_Exceptions**, **CMDB_Accounts**, **Source_Manifest**, **Control_Runs**
Your working sheets	**Control_Results**, **Exceptions**, **Run_Metadata**, **3 Health Summary**
Python path	**Optional_Python_Path/README.md** and the Python section at the end of this guide

Stated rules

Review freeze	2026-09-15 12:00 UTC (`2026-09-15T12:00:00Z`)
Control ID / rule version	`HL-CC-PRIV-MFA-01` / `stated-v1-weekly`
Population	cloud_entitlements rows with privileged true, access_status active, environment_tag production. One evaluation per user_id.
Accepted MFA methods	`totp`, `hardware_key`. Not accepted: `sms`, `none`, blank.
MFA freshness	mfa_last_verified no more than 90 days before the freeze
Join key	user_id (lowercase email if you use it)

Status	When
PASS	Technical requirement met.
FAIL	Requirement not met, identity evaluable, and no valid approved exception.
WATCH	Requirement not met, but a valid approved exception exists. technical_condition stays not_met; risk_treatment is approved_exception.

Status	When
UNKNOWN	MFA evidence missing, blank, conflicting, or stale beyond 90 days; not on the privileged roster; or a service or break-glass / shared emergency account where human MFA logic does not apply.

> **An approved exception is never a technical PASS**
>
> PASS is a technical claim. An exception is a time-bound risk treatment with an approver and a compensating control. Recording it as PASS erases the residual risk. If a control owner asks you to mark an open exception as PASS so the percentage looks better, the answer is no.

Monitoring-assurance inputs (Harborline scenario values)

Expected sources	The five files named in source_manifest.json: privileged_roster.csv, auth_config.json, cloud_entitlements.csv, approved_exceptions.csv, cmdb_accounts.csv
Stale source	extracted_at more than **14 days** before the freeze
Baseline population	**12** (last successful weekly job in control_runs.csv)
Population variance threshold	Variance above **25%** versus baseline is a monitoring-assurance failure. Variance = (actual minus baseline) divided by baseline.

These are Harborline values for this exercise, not industry standards. Monitoring exceptions (missing sources, stale extracts, unexpected variance, failed runs) go in run metadata. They are not the same as a person's MFA FAIL.

> **Population completeness is not control effectiveness**
>
> Your status counts describe **technical effectiveness** for the identities you evaluated. The assurance result describes whether you can **trust the population and sources**. Report both. A high PASS rate with a failed assurance check is low confidence, not good news.

The other two source files

Control_Runs holds prior weekly job counts. Use it for the baseline and in Break and Defend. It is job history, not proof of completeness. **Evidence_Campaign** is last quarter's manual evidence campaign. It is Assess context and is not joined in Build.

> **Sanity check (not the answer key)**
>
> A complete monitor on this sample usually shows **all four statuses**, **at least one WATCH**, **at least one UNKNOWN**, and an assurance problem in run metadata (freshness or variance). Zero FAILs usually means a rule is not firing. Every exception coded PASS means the treatment distinction is missing.

Do this

1. Open **3 Build Guide**. It holds every input as a named cell: `REVIEW_AS_OF`, `MFA_MAX_AGE_DAYS`, `BASELINE_POPULATION`, `VARIANCE_THRESHOLD`, `STALE_AFTER_DAYS`, `CONTROL_ID`, and `RULE_VERSION`.
2. Choose one path: spreadsheet (continue) or Python (Python section at the end of this guide, then return to step 7).
3. **Define the population** on **Control_Results**: identities on **Cloud_Entitlements** with privileged TRUE, access_status active, and environment_tag production. One row per user_id. Sandbox is out.
4. **Evaluate the technical condition** by looking up **Auth_Config** on user_id. Met only when mfa_enabled is TRUE, the method is totp or hardware_key, and mfa_last_verified is present and no more than 90 days before the freeze. Count auth rows per user_id: two rows that disagree are a conflict.
5. **Apply treatment** from **Approved_Exceptions**: an exception is valid only if control_id matches, status is open, and expires_on is blank or on or after the freeze date.
6. **Assign status** (PASS, FAIL, WATCH, UNKNOWN) and a reason code, using the rules below. Copy FAIL and WATCH rows to **Exceptions**.
7. **Complete Run_Metadata**: every field is explained on the tab. Link values to **Source_Manifest** and your results with formulas.
8. Prove it is a monitor: change one mfa_enabled value or one exception expiry, confirm the status changes, then undo it.
9. Complete **3 Health Summary**: counts by status with the denominator, the treatment statement, the assurance result, and short lists of FAIL, WATCH, and UNKNOWN identities. Python users base this on **Optional_Python_Path/outputs/control_health_summary.md**.

What you produce	You are done when
• A working monitor (Build tabs or Python output). • The evidence package equivalent: **Control_Results**, **Exceptions**, **Run_Metadata**, **3 Health Summary**.	☐ All four statuses appear where the data supports them. ☐ Approved exceptions are WATCH, never PASS. Expired or closed exceptions do not treat a row. ☐ Service and break-glass identities required a judgment and are UNKNOWN, not FAIL. ☐ Run metadata records sources, extraction times, baseline versus actual population, run time, and assurance status. ☐ Changing a source value changes the output.

Next: Only now open the Break evidence folder and move to Stage 4.

Stage 4: Break

Investigate why a green monitor missed privileged access.

What you are trying to accomplish

Use the incident evidence to explain why the monitor looked green, what automation actually proved, what it failed to prove, and how monitoring and evidence should change.

Why this matters

Something happened that a high PASS rate may not explain. The professional skill is separating a wrong MFA rule from an incomplete population, without inventing logs or claiming a SOX outcome.

Open these

What	Where to find it
Incident evidence (four files)	**4_Break_Incident_Evidence_Open_After_Build**
Your evidence timeline	Tab **4 Break Timeline**
Your written analysis	**Deliverable_Templates/4_Break_Incident_Analysis.docx**
Your Stage 3 results and run metadata	**Control_Results**, **Run_Metadata**, **Control_Runs**

Open this evidence only after Build

Do not open **4_Break_Incident_Evidence_Open_After_Build** until you have a working monitor and have reviewed its results.

File	What it is
4_Break_Incident_Evidence_Open_After_Build/Incident_Summary.pdf	What was reported
4_Break_Incident_Evidence_Open_After_Build/Weekly_Job_Email.pdf	What GRC was told the job proved
4_Break_Incident_Evidence_Open_After_Build/Cloud_Legacy_B_Access.pdf	Access listing collected after the alert
4_Break_Incident_Evidence_Open_After_Build/CMDB_Follow_Up.pdf	Inventory follow-up

Where Break findings go

Excel	Word
4 Break Timeline: structured facts, sources, and limits.	**Deliverable_Templates/4_Break_Incident_Analysis.docx**: your written analysis. This is the deliverable.

Identities that appear only in the Break evidence were never in the freeze extract. Do not add them to **Control_Results**. They are evidence of a population gap, which belongs in your analysis.

Term	Meaning
Control design	The MFA rule as written cannot cover the risk.
Monitoring design	The job evaluated the wrong set of records.
Operating effectiveness	A designed step did not run.
Population completeness	In-scope access never entered the evaluated set.
Data quality	Sources disagree, are stale, or are wrongly tagged.

Do this

1. Confirm Build is complete.
2. Read the four files, starting with **4_Break_Incident_Evidence_Open_After_Build/Incident_Summary.pdf**.
3. On **4 Break Timeline**, log each fact in time order with its source, what it proves, and what it does not prove.
4. Open **Deliverable_Templates/4_Break_Incident_Analysis.docx**, save a copy, and answer every section: facts you accept and doubt, why the monitor looked green, what failed, classification, what automation proved, what it failed to prove, immediate containment, systemic remediation, and how monitoring and evidence should change.
5. Classify the failure as control design, monitoring design, data quality, operating effectiveness, population completeness, or a combination, and say which part is which.

What you produce	You are done when
• The timeline on **4 Break Timeline**. • Your completed copy of **Deliverable_Templates/4_Break_Incident_Analysis.docx**.	☐ You can explain what the weekly 100 percent actually covered. ☐ Your classification separates the MFA rule from the population definition. ☐ You did not invent logs or claim Harborline passed or failed SOX.

Next: Move to Stage 5: Improve.

Stage 5: Improve

Make the monitoring design learn from the incident.

What you are trying to accomplish

Revise how population completeness is tested, how sources are validated, how UNKNOWN and treatments are handled, how the monitor proves it ran, and what evidence is retained.

Why this matters

"We will remember to check legacy accounts" is not a control. A revised design must make the next gap visible on its own.

Open these

What	Where to find it
Your improve record	Tab **5 Improve**
Control definition and evidence requirements	**2A Control Definition** and **2C Evidence Requirements**, column **Revised (Stage 5)**
Source-to-control map	**2B Source-to-Control Map** (add or amend rows)

Do this

1. Fill the **Revised (Stage 5)** column on **2A Control Definition** and **2C Evidence Requirements** wherever your design changed. Write "No change" where it did not. Keep the Original column untouched.
2. Add or amend rows on **2B Source-to-Control Map** for any new source or reconciliation step.
3. On **5 Improve**, answer each prompt: population completeness, source validation, UNKNOWN and treatments, monitoring assurance, and evidence retention.
4. Write the key sentence: **What changed in my monitoring design after the incident, and why?**
5. Record what you are explicitly not changing and the residual risk.

What you produce	You are done when
• Completed **5 Improve** and the Revised columns.	☐ A reader of the revised design knows how completeness is tested, what happens when CMDB tags disagree with how an account is used, how UNKNOWN and treatments are handled, and how you would know the next run executed.

Next: Move to Stage 6: Defend.

Stage 6: Defend

Separate the health rate from your confidence in it.

What you are trying to accomplish

Choose exactly two monitoring improvements to fund and write a one-page Control Health Brief for leadership.

Why this matters

A pass rate among evaluated records is not the same as confidence in the population and the evidence. Leadership will read one page and ask what to fund. Harborline can fund only two monitoring improvements this quarter.

Open these

What	Where to find it
Your two funded items	Tab **6 Priorities**
Your one-page Control Health Brief	**Deliverable_Templates/6_Defend_Control_Health_Brief.docx**
Inputs	**3 Health Summary**, **Run_Metadata**, **Control_Runs**, **5 Improve**

Do this

1. On **6 Priorities**, choose exactly two items. For each, explain what and why: risk, coverage, reliability, audit value, effort, and dependencies. The tab lists an illustrative menu; you may rename or combine, but you still pick two.
2. Describe what you are deferring and the residual risk of waiting.
3. Open **Deliverable_Templates/6_Defend_Control_Health_Brief.docx**, save a copy, and complete every section on one page. Do not lead with a single percentage. If you quote a rate, state the denominator. State that WATCH items are not technical passes. Report the monitoring-assurance result and your confidence separately from the compliance rate.
4. Optional trend: **Control_Runs** shows prior job counts. Treat it as job history, not proof of completeness.
5. End with the decision you need from leadership. Write for a CISO, Head of Engineering, or SOX lead.

What you produce	You are done when
• Two funded items on **6 Priorities**. • Your one-page copy of **Deliverable_Templates/6_Defend_Control_Health_Brief.docx**.	☐ Exactly two items are funded and deferred risk is named. ☐ The brief separates rate from confidence. ☐ It fits on one page.

Next: Move to Stage 7: Career Translation.

Stage 7: Career Translation

Turn the work into an honest professional story.

What you are trying to accomplish

Write the Standard career set: honesty sentence, portfolio description, resume bullets, 30-second explanation, and a STAR story covering the full loop.

Why this matters

Employers want to hear what PASS actually proves, why UNKNOWN is useful, and how you know a monitor is working. This stage turns the project into answers you can give without notes.

Open these

What	Where to find it
Prompts and writing areas	Tab **7 Career Translation**

Results you may and may not claim

Allowed: identified X technical FAILs and kept approved exceptions as WATCH; recorded X UNKNOWN identities a PASS rate would have hidden; produced a re-performable evidence package; showed monitoring assurance failed even when some identities met MFA.

Not allowed: "implemented CCM for Harborline as a client," "improved control health to 100%," or the scenario's 40 to 60 person-hour estimate as your own KPI.

Do this

1. Say the honesty sentence: *"This was an independent portfolio project where I simulated continuous control monitoring for a fictional company. I assessed screenshot-based evidence, designed a privileged MFA monitor, built an evidence package, investigated a population-completeness incident, improved the design, and wrote a management recommendation."*
2. Write the 150 to 200 word description, two or three bullets, the 30-second explanation, and STAR on **7 Career Translation**. Counts must match your output.
3. **Deep Dive (optional):** README paragraph, skills, the ten interview questions, and the defense drill.

What you produce	You are done when
• The Standard career set on **7 Career Translation**.	☐ You can say the honesty sentence and STAR without reading. ☐ No invented employment or KPIs.

Next: Complete the final checklist.

Before you call this finished

A final check across all seven stages.

- ☐ Current-state assessment
- ☐ Control definition with frequency defended and a monitoring-assurance control
- ☐ Source-to-control map
- ☐ Evidence requirements and retention
- ☐ Working monitor with four statuses
- ☐ Health summary (PASS is technical; WATCH is treatment)
- ☐ Incident analysis
- ☐ Revised monitoring design
- ☐ Exactly two funded improvements
- ☐ Control Health Brief
- ☐ Career Translation Standard set

Reasoning questions

- Can I explain why this is a population evaluation, not a screenshot sample?
- What evidence did I trust and why was it re-performable? What did I not trust?
- During Break, was the MFA logic wrong, or was the population incomplete?
- Did I reconsider PASS versus WATCH versus UNKNOWN after I saw the results?
- What residual monitoring risk remains after my two funded items?
- What alternative frequency or population definition did I reject, and why?
- Can I defend the brief to someone who only wants a green percentage?

Optional: the Python Build path

Same data, same rules, same evidence package.

Step	File	Control skill
1	**guided/01_load_sources.py**	Load every extract the run will use
2	**guided/02_define_population.py**	Decide who is in the evaluation
3	**guided/03_normalize_mfa.py**	Make MFA fields comparable
4	**guided/04_reconcile_sources.py**	Join entitlements, roster, and auth
5	**guided/05_evaluate_control.py**	PASS, FAIL, WATCH, UNKNOWN
6	**guided/06_assurance_and_exceptions.py**	Treatments plus a control over the monitor
7	**guided/07_emit_evidence_package.py**	Reproducible run artifacts

Paths are inside **Optional_Python_Path**. Standard minimum outputs: **outputs/run_metadata.json**, **outputs/control_results.json**, **outputs/control_health_summary.md**.

Python setup (one time)

1. Install Python 3.10 or newer from https://www.python.org/downloads/. On Windows, tick **Add Python to PATH** during setup.
2. Open a terminal. On Windows use PowerShell or Command Prompt. On macOS use Terminal.
3. Check the version: `python3 --version` (on Windows the command is often `python --version`). You want 3.10 or higher.
4. Change into the optional Python folder inside this project. The easiest way: type `cd` followed by a space, then drag the **Optional_Python_Path** folder onto the terminal window and press Enter.
5. Nothing needs to be installed. The exercises use the Python standard library only.

Run commands

```
# Guided
python3 guided/01_load_sources.py
python3 guided/validate.py --step 1
# ...repeat for steps 2 to 7

# Challenge
python3 challenge/challenge.py
python3 guided/validate.py --step 7
```

On Windows, type `python` wherever these commands say `python3`. Scripts find the data folder on their own, so you can run them from any location as long as the project folder structure is unchanged. Output is written to the **Optional_Python_Path/outputs** folder, which the scripts create if it is missing.

What you see	What it means
`NotImplementedError`	Expected. You still have a TODO in that function. Read the HINT lines next to it.

What you see	What it means
`python3: command not found`	Try `python`. If neither works, install Python and reopen the terminal.
`FileNotFoundError ... outputs/0N_...json`	Run the earlier numbered step first. Each step reads the previous step's output.
Wrong SLA or freshness results	Use the review freeze timestamp from this guide, never your computer clock.

Deep Dive: export replay (either path)

Optional_Deep_Dive_Export_Replay holds a later auth extract (exported 2026-09-16 08:00 UTC). At least one identity's MFA fields differ from the freeze file. Use it to prove your monitor is data-driven.

1. Save a copy of your freeze results (workbook copy, or copy the Python outputs folder).
2. **Spreadsheet:** replace the rows on **Auth_Config** with the rows from **Optional_Deep_Dive_Export_Replay/auth_config_replay.csv**. **Python:** rename **3_Build_Source_Data/auth_config.json** to auth_config_freeze.json, copy **Optional_Deep_Dive_Export_Replay/auth_config.json** into **3_Build_Source_Data**, and rerun.
3. Compare the two result sets. At least one evaluation should change. If nothing changes, the monitor is a typed list.
4. Restore the freeze file (and workbook rows) before you continue.