KoBo Data Automation Suite

Python Workflows for Field Data Synchronization, Change Tracking, Validation, and Documentation

Role: Python Automation Developer & Information Management Specialist
Platform: KoBoToolbox API, Python, Excel, Telegram, PDF
Operational Context: Humanitarian field-data operations
Historical Scale: Approximately 3,000 records processed across KoBo/Python workflows.

Project Overview

Field-data operations generated recurring manual tasks around KoBoToolbox submissions, Excel updates, validation reviews, attachments, and documentation.

I developed a set of Python automation workflows to reduce this repetitive work and create more structured, traceable outputs.

The automation suite covered four main operational areas:

1. Incremental synchronization and attachment processing.
2. Excel-based change tracking and structured update logs.
3. Bulk validation-status updates with success and failure reporting.
4. Automated daily and monthly reporting with role-based delivery through Telegram and email.

The original production environment is no longer accessible. The screenshots in this document are reconstructed offline demonstrations based on the same automation logic, using synthetic data.

The Operational Challenge

The manual workflow required staff to:

- Check KoBo repeatedly for new submissions.
- Download attachments individually.
- Organize images by case or household.
- Compare updated submissions with existing Excel records.
- Identify which fields had changed.
- Update validation statuses record by record.
- Track failed or incomplete operations manually.
- Prepare consistent documentation for review and archiving.

This created repeated administrative work and increased the risk of inconsistent file organization and incomplete update tracking.

Automation Workflow 1

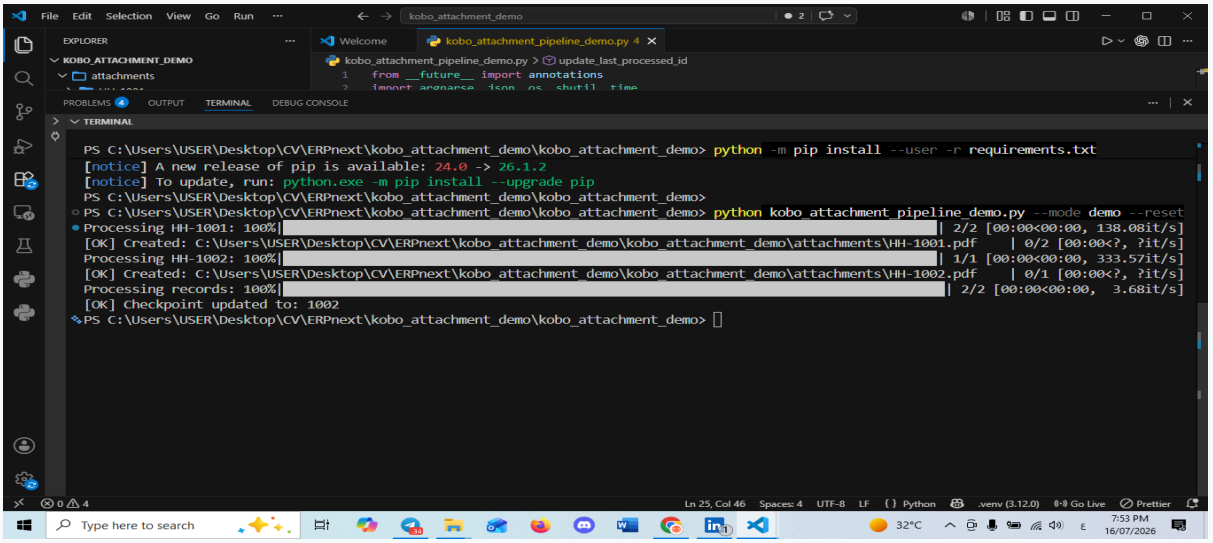
Incremental Attachment Processing

The first workflow:

- Reads the most recently processed submission ID.
- Retrieves only newer KoBo records.
- Extracts the household or case identifier.
- Downloads image attachments.
- Creates a dedicated folder for each case.
- Consolidates related images into one PDF.
- Saves the latest processed ID for the next run.

KoBo Submissions
↓
Filter New Records
↓
Extract Case Code
↓
Download Attachments
↓
Organize by Case
↓
Generate Case PDF
↓
Update Checkpoint

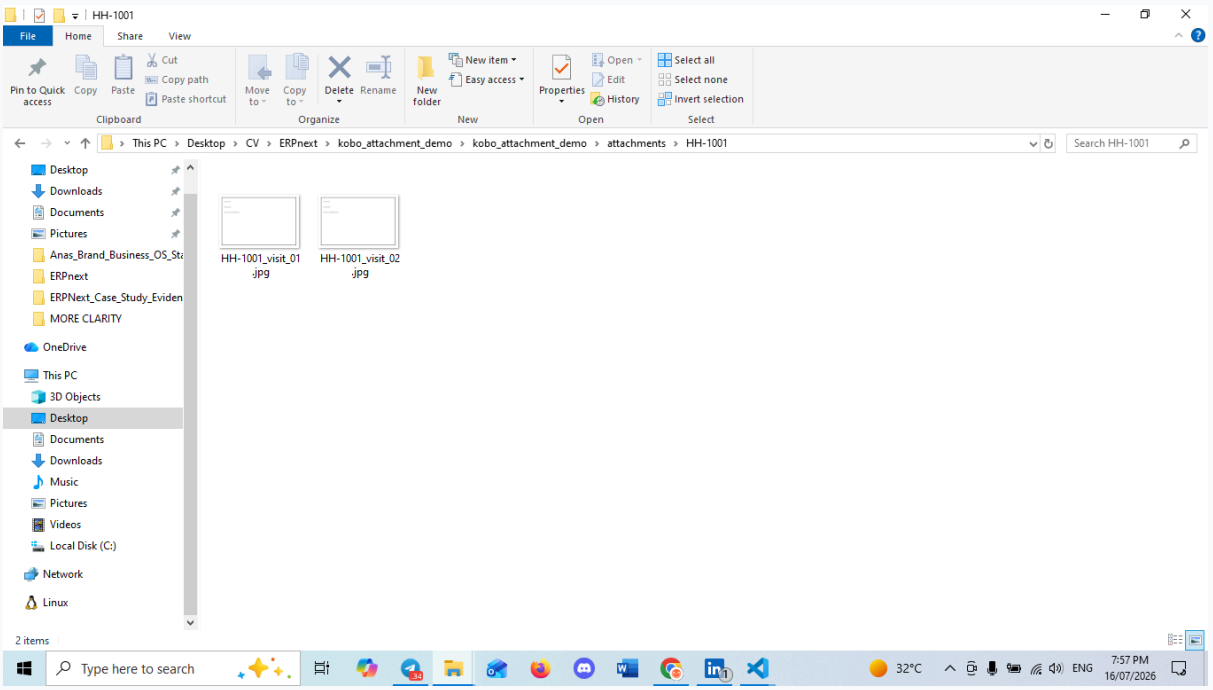
[IMAGE 1 — Incremental Pipeline Run]



Caption

Reconstructed pipeline execution showing successful processing of new records, case-level PDF generation, and checkpoint advancement.

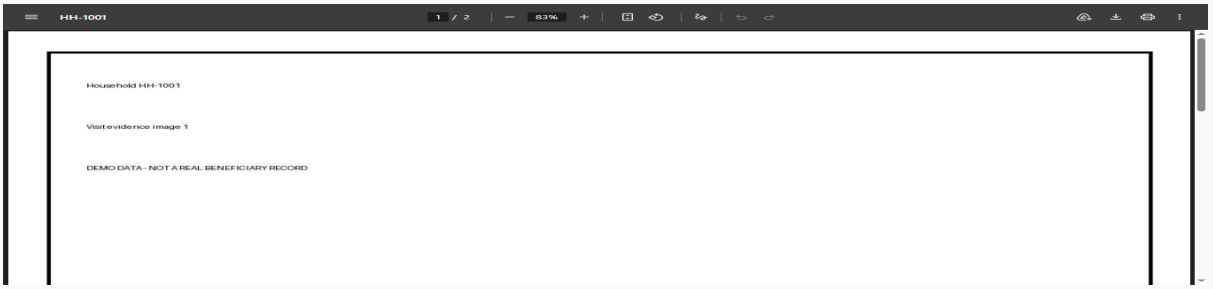
[IMAGE 2 — Attachments Organized by Case]



Caption

Synthetic attachments automatically organized inside a folder named after the household identifier.

[IMAGE 3 — Consolidated Case PDF]



Caption

Multiple synthetic field attachments automatically consolidated into one structured PDF per case.

From Manual Data Handling to Traceable Automation

Automation Workflow 2

Excel Change Tracking and Structured Updates

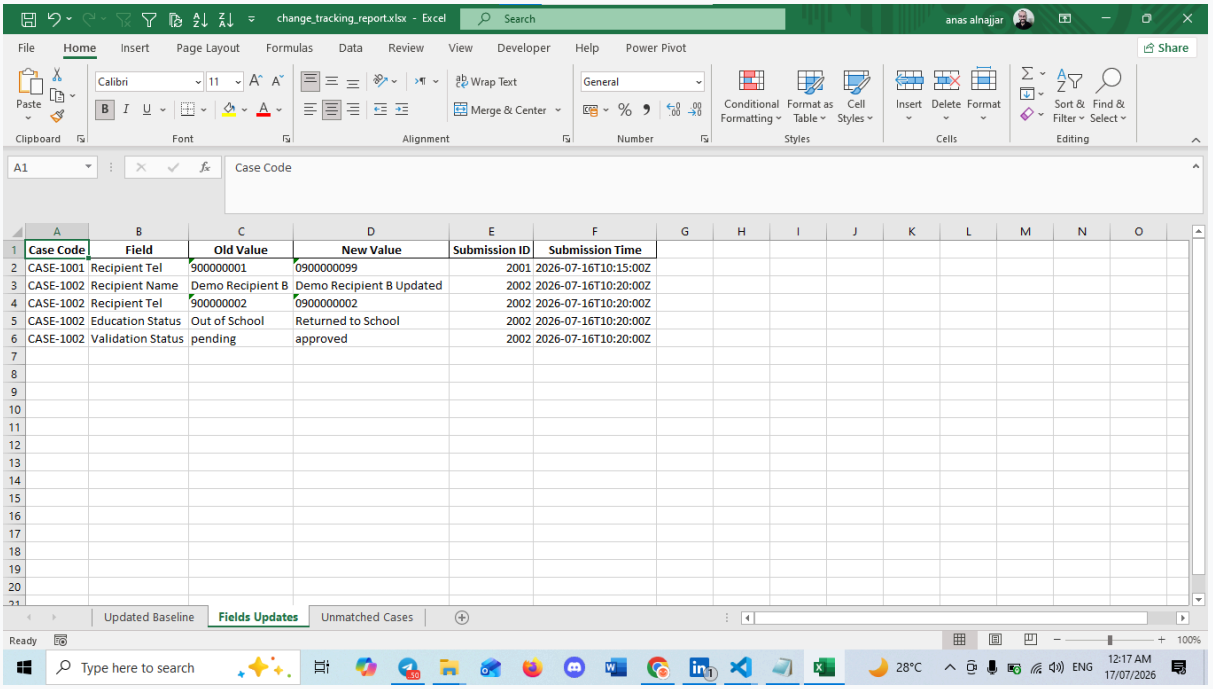
The second workflow compared new KoBo submissions against an existing Excel baseline.

For each matched case, the automation:

- Identified the corresponding case using a unique case code.
- Compared selected KoBo fields with the current Excel values.
- Recorded the old and new values.
- Logged the field that changed.
- Stored the KoBo submission ID and submission time.
- Updated the baseline record.
- Identified submissions that did not match an existing case.
- Prepared a notification summary for Telegram.

This created a structured audit trail rather than silently overwriting existing information.

[IMAGE 4 — Field-Level Change Tracking]



Case Code	Field	Old Value	New Value	Submission ID	Submission Time
CASE-1001	Recipient Tel	900000001	0900000099	2001	2026-07-16T10:15:00Z
CASE-1002	Recipient Name	Demo Recipient B	Demo Recipient B Updated	2002	2026-07-16T10:20:00Z
CASE-1002	Recipient Tel	900000002	0900000002	2002	2026-07-16T10:20:00Z
CASE-1002	Education Status	Out of School	Returned to School	2002	2026-07-16T10:20:00Z
CASE-1002	Validation Status	pending	approved	2002	2026-07-16T10:20:00Z

Caption

> Reconstructed change-tracking report showing the case code, changed field, previous value, new value, submission ID, and update timestamp.

What it demonstrates

- Field-level comparison.
- Old/new value tracking.
- Structured audit history.
- Identification of unmatched records.
- Preparation of automated update notifications.

Automation Workflow 3

Bulk Validation Status Updates

The third workflow automated the process of updating KoBo validation statuses in bulk.

The script reads a structured input containing:

- Submission ID.
- Required validation status.

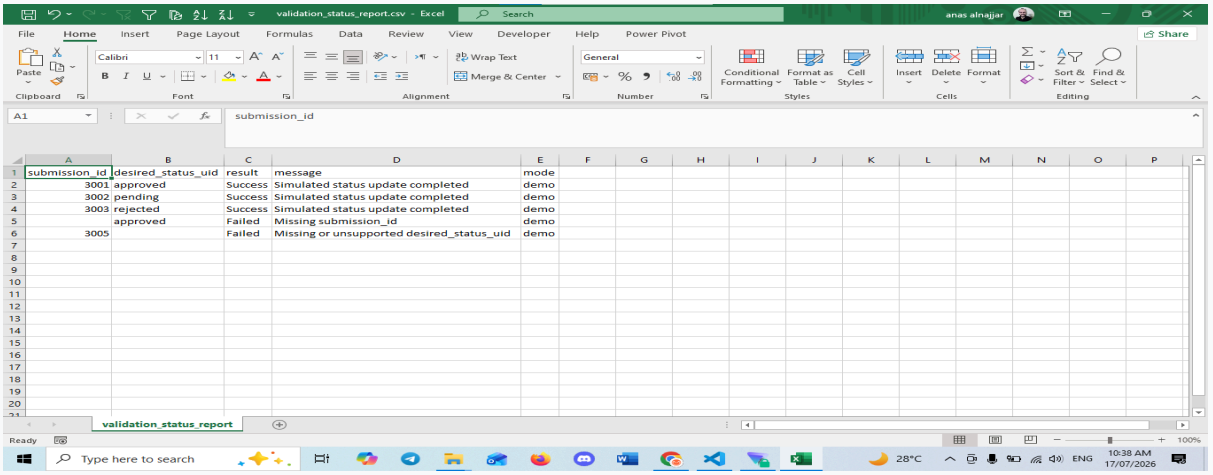
It then processes each record and produces an audit report containing:

- Requested status.

- Success or failure result.
- Error explanation.
- Execution mode.

The workflow validates required inputs before attempting an update, allowing incomplete records to be reported instead of silently skipped.

[IMAGE 5 — Bulk Validation Status Audit]



submission_id	desired_status_uid	result	message	mode
3001	approved	Success	Simulated status update completed	demo
3002	pending	Success	Simulated status update completed	demo
3003	rejected	Success	Simulated status update completed	demo
	approved	Failed	Missing submission_id	demo
3005		Failed	Missing or unsupported desired_status_uid	demo

Caption

> Reconstructed audit report showing successful bulk validation updates and intentionally failed records with clear error reasons.

Automation Workflow 4

Automated Reporting & Role-Based Notification Layer

In addition to the Python automation workflows, I developed a PHP-based operational reporting and notification layer integrated with the KoBoToolbox API.

Core Capabilities

The solution was designed to:

- Retrieve current submissions from KoBoToolbox.
- Generate daily and monthly PDF summary reports.
- Calculate the number of field visits completed by each case manager.
- Deliver daily operational reports through Telegram.
- Send monthly summary reports by email.
- Restrict each supervisor’s report to the cases assigned to them.
- Manage report recipients according to roles stored in a relational database.

- Protect API credentials and prevent their exposure through public report links.

Operational Workflow

KoBo Submissions
↓
PHP API Integration
↓
Period-Based and Role-Based Filtering
↓
Daily and Monthly PDF Generation
↓
Supervisor-Specific Case Restriction
↓
Telegram or Email Delivery

Operational Value

This layer extended the automation suite from data processing into scheduled operational communication. It enabled managers and supervisors to receive relevant summaries without manually downloading, filtering, and preparing KoBo data.

The role-based filtering ensured that supervisors received information related only to their assigned cases, while management-level recipients could receive broader operational summaries.

Evidence Clarification

> The PHP implementation presented in this case study is a sanitized reconstruction of operational logic previously developed for field-data workflows. Credentials, organization identifiers, staff information, and beneficiary data have been removed.

My Role

I designed and implemented the automation logic, including:

- KoBo API integration.
- Incremental submission retrieval.
- Checkpoint management.
- Attachment downloading and retry handling.
- Case-code-based directory organization.
- Image conversion and PDF generation.
- Excel baseline comparison.
- Field-level old/new change logging.
- Unmatched-record detection.
- Telegram notification logic.

- Bulk validation-status updates.
- Success and failure reporting.
- Environment-variable configuration.
- Safe offline reconstruction using synthetic data.

Operational Value

The automation suite was designed to:

- Reduce repetitive manual downloads.
- Avoid processing the same submissions repeatedly.
- Improve attachment organization.
- Consolidate case documentation.
- Track data changes transparently.
- Support validation and review workflows.
- Produce clear success and failure reports.
- Improve traceability across KoBo and Excel operations.
- Run locally in connectivity-constrained environments.

No unsupported percentage of time saved is claimed.

Evidence Clarification

The code reflects automation workflows previously developed for real field-data operations.

The screenshots shown here are reconstructed offline demonstrations using synthetic data because access to the original KoBo production environment is no longer available.

They demonstrate the technical logic, workflow, and output structure, not the original production interface. The previous attachment-focused case study already documented the reconstructed evidence approach.

Technical Environment

Python · PHP · KoBoToolbox API · Requests · Pandas · OpenPyXL · Pillow · TCPDF · Telegram Bot API · SMTP Email Delivery · SQL · Environment Variables · Incremental Synchronization · Role-Based Reporting · Excel Audit Logs · PDF Generation

Reusable Professional Claim

> Developed Python and PHP automation workflows integrated with the KoBoToolbox API to synchronize field data incrementally, organize and consolidate case attachments, track field-level changes against Excel baselines, update validation statuses in bulk, and deliver role-specific daily and monthly PDF reports through Telegram and email.