Spreadsheet Formula Auditor Claude Skill

Claude skill - preview and full file inventory

File inventory (everything in product.zip, inside Spreadsheet-Formula-Auditor-Claude-Skill/)

LICENSE.txt	998 B
README.md	3.0 KB
README.txt	3.0 KB
spreadsheet-formula-auditor/SKILL.md	4.9 KB
spreadsheet-formula-auditor/examples/sample_report.md	2.2 KB
spreadsheet-formula-auditor/examples/scanner_output.txt	1.5 KB
spreadsheet-formula-auditor/scripts/audit_formulas.py	10.6 KB
spreadsheet-formula-auditor/tests/make_sample.py	2.9 KB
spreadsheet-formula-auditor/tests/sample_budget.xlsx	6.6 KB
spreadsheet-formula-auditor/tests/test_audit.py	1.4 KB

Tested here

PASS - 10 planted problems found; 12 findings total, 81 formulas scanned.

Sample: Spreadsheet-Formula-Auditor-Claude-Skill/spreadsheet-formula-auditor/SKILL.md (excerpt 1/1)

````
---
name: spreadsheet-formula-auditor
description: Audits Excel workbooks (.xlsx/.xlsm) for formula errors and risky modelling
  patterns - broken references, error values, hardcoded numbers inside formulas, formulas
  that break the pattern of their row or column, SUM ranges that stop short of the data,
  references to blank cells, volatile functions and circular references. Use when the user
  uploads or points to a spreadsheet and asks to check, audit, review, debug or
  sanity-check its formulas, totals or model logic.
---

# Spreadsheet Formula Auditor

You review the formulas in an Excel workbook and give the user a prioritised, cell-by-cell
  list of problems with a concrete fix for each one. You do not silently change the user's
  file.

## When to use

- "Check my budget spreadsheet for mistakes", "why is this total wrong", "audit the
  formulas in this model", "is anything hardcoded in here".
- Works on `.xlsx` and `.xlsm`. For `.xls` or `.csv`, ask the user to save as `.xlsx` first
  (CSV files contain no formulas).

## Step 1 - Run the scanner

If you can run code (Claude Code, or claude.ai with code execution enabled), run the
  bundled scanner first. It is deterministic and catches patterns a read-through misses:

```bash
python3 scripts/audit_formulas.py path/to/workbook.xlsx --json audit.json
```

Requirements: Python 3.8+ and `openpyxl` (`pip install openpyxl`). The script only reads
  the file.

It prints a text report and (with `--json`) writes machine-readable findings. Each finding
  has: `sheet`, `cell`, `check`, `severity` (high / medium / low), `formula`, `detail`.

If you cannot run code, do the same checks by reading the formulas the user pastes, using
  the checklist in Step 2.

## Step 2 - Interpret, do not just repeat

The scanner flags candidates. Confirm each one with judgement before reporting it:

| Check | What it means | Usually a real problem when... |
|---|---|---|
| `error_ref` | Formula contains `#REF!` | Always. A row/column/sheet was deleted. |
| `missing_sheet` | Formula points to a sheet that does not exist | Always. |
| `inconsistent_formula` | Formula differs in shape from the neighbours in its row or
  column | The neighbours are a repeating calculation (monthly columns, line items). Not a
  problem in deliberately different cells such as a header-row total. |
| `sum_range_short` | A SUM/AVERAGE over a column stops before the last number directly
  above/beside it | The total sits at the bottom of a list and rows were added later. |
| `hardcoded_constant` | A literal number inside a formula, e.g. `=B5*1.08` | The number is
````

SAMPLE

Preview excerpt only - the full file and all other files are in the download.