HANDS-ON AI / WORKSHOP

What if your work could do itself?

Practical Google Workspace Automation

Apps Script + Gemini

Laurence Svekis

Google Developer Expert · Workspace Author & AI Educator

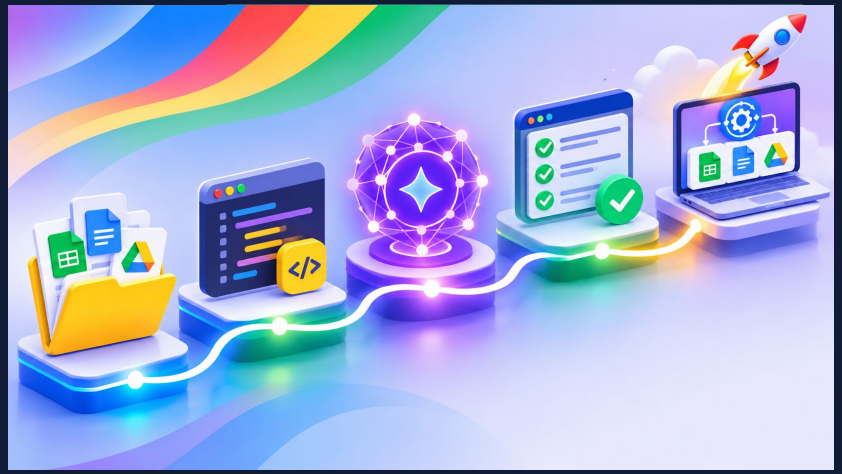


HANDS-ON WORKSHOP / RESOURCE EDITION

Hands-On AI

Google Workspace Automation | 30 Projects – Resource Edition

- Build real-world Google Workspace applications
- Automate everyday tasks with Google Apps Script
- Integrate Gemini AI into intelligent workflows
- Learn by building, testing and experimenting
- Explore 30 projects with reusable source code



What would you automate if you could?



01 / REPORTING

Reports that prepare themselves



02 / DOCUMENTS

Documents generated from data



03 / TOOL BUILDING

AI that helps you build the next tool

WHY AUTOMATE?

Get back time for work that matters

DAILY TIME SAVINGS

15 minutes × 5 times per week

ANNUAL IMPACT

65 hours

per year

KEY AREAS TO AUTOMATE

- Reports
- emails
- documents
- planning

Meet your workshop guide

01 / INSTRUCTOR

Laurence Svekis

Workshop guide and lead instructor for hands-on automation.

02 / EXPERTISE

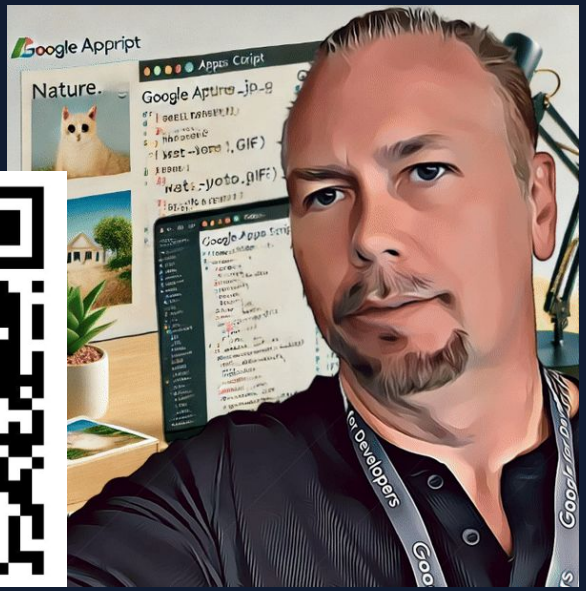
Google Developer Expert

Specializing in Google Workspace solution architecture and development.

03 / IMPACT

3 Million+ Students

Author & AI educator teaching global developers to build next-gen tools.



https://discoveryvip.com/ - Tools and Resources
https://vibelearning.ca/ - Learning with AI
https://basescripts.com/ - Blog and Work

30 applications. Your workshop toolkit.

01 / SOURCE CODE

GitHub

Complete project source code for all 30 workshop applications.



SCAN TO ACCESS

https://github.com/lsvekis/Google-Apps-Script-Workshop/tree/main

02 / STARTER FILES

Google Drive

Starter files, guides, and templates ready to copy into your Drive.



SCAN TO ACCESS

https://drive.google.com/drive/folders/1nfZzrubhrsjLsgqD99OkqSd4SsT52DIO?usp=sharing

Get your own starter files

OPTION 01

GitHub

Choose a project and inspect its complete solution code directly.

- Full source code
- Reference implementations

OPTION 02

Google Drive

Copy the starter Sheet and all required workshop templates.

- Pre-configured Sheets
- Ready-to-use assets

OPTION 03

Generator

Automate setup with execution functions.

- Run `generateWorkshop()`
- Run `generateNextBatch()` 5x

Get your workshop starter files

Create your own working copy before starting the lessons.

01

Open Google Drive

Open the shared workshop folder.

02

Download the folder

Right-click the folder → Download.

03

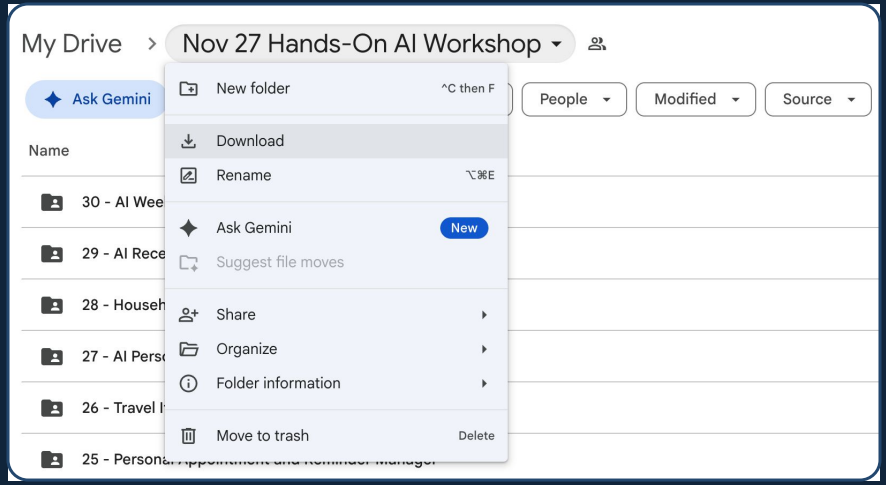
Extract the ZIP

Unzip locally; keep all lesson subfolders.

04

Keep your own copy

Do not edit or move the shared originals.



Open shared starter files ↗

https://drive.google.com/drive/u/0/folders/1nfZzrubhrsjLsgqD99OkqSd4SsT52DIO

Upload your copy to Google Drive

Keep each lesson's spreadsheet and documents together.

01

Open My Drive

Sign in with the account you will use for the workshop.

02

Upload the folder

New → Folder upload → select the extracted folder.

03

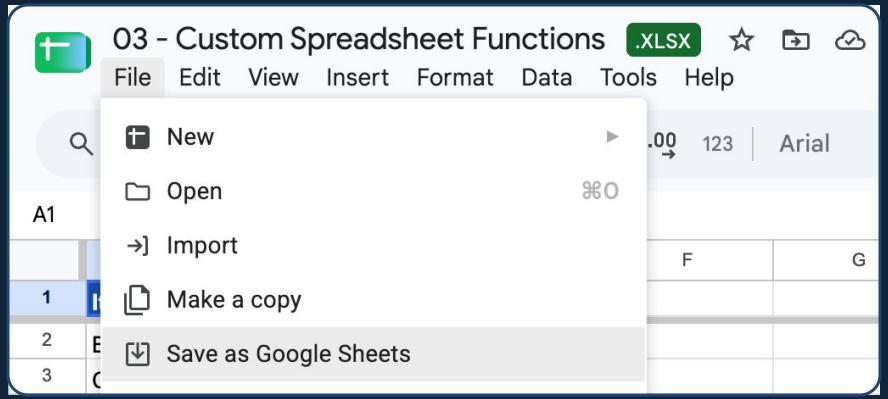
Check structure

Open a lesson folder; confirm its files are together.

04

Check formats

DOCX & XLSX require conversion before running Apps Script.



⚠ FILE CONVERSION NOTICE

Microsoft files will not run Google Apps Script directly. They must be converted to Google Workspace format (Docs / Sheets).

Script to convert all files from Microsoft to Google Workspace files.
https://github.com/lsvekis/Google-Apps-Script-Workshop/blob/main/Update%20Docx%20and%20XLSx%20to%20Doc%20and%20SHeet

Convert Office files in bulk

Use the provided conversion script on your uploaded folder.

01

Copy the script

Open the linked GitHub file; copy its raw code.

02

Create a script project

Go to script.google.com → New project; paste into Code.gs.

03

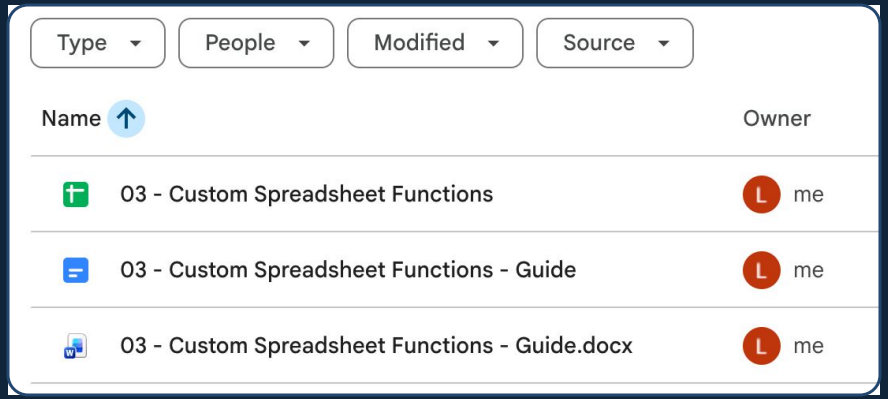
Configure ROOT_FOLDER

Paste your uploaded folder URL, not the shared folder URL.

04

Enable Drive API

Services → + → Drive API → Add.



⚠ FILE CONVERSION NOTICE

Microsoft files will not run Google Apps Script directly. They must be converted to Google Workspace format (Docs / Sheets).

Script to convert all files from Microsoft to Google Workspace files.
https://github.com/lsvekis/Google-Apps-Script-Workshop/blob/main/Update%20Docx%20and%20XLSx%20to%20Doc%20and%20SHeet

Run and verify the conversion

Native Google files are created inside the original lesson folders.

01

Run the function

Select convertWorkshopFiles → Run; authorize access.

02

Check execution log

Review converted, skipped and failed totals.

03

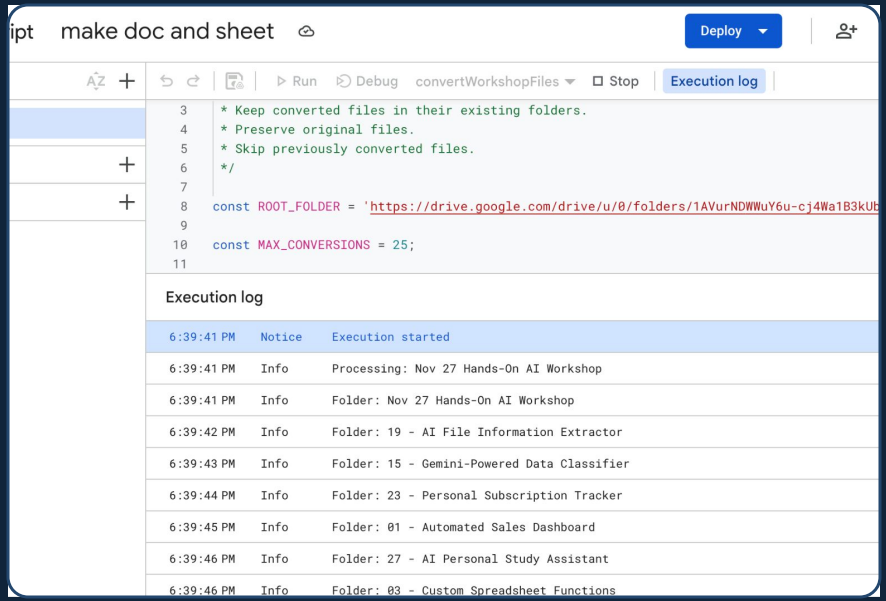
Repeat as needed

Up to 25 conversions per run; repeat until scan completes.

04

Use native files

Open converted Docs and Sheets; Office originals remain.



Get the project source code

Use GitHub to build or restore each lesson application.

01

Open the repository

Choose the folder matching your lesson.

02

Copy the source

Open the relevant script file and copy its raw contents.

03

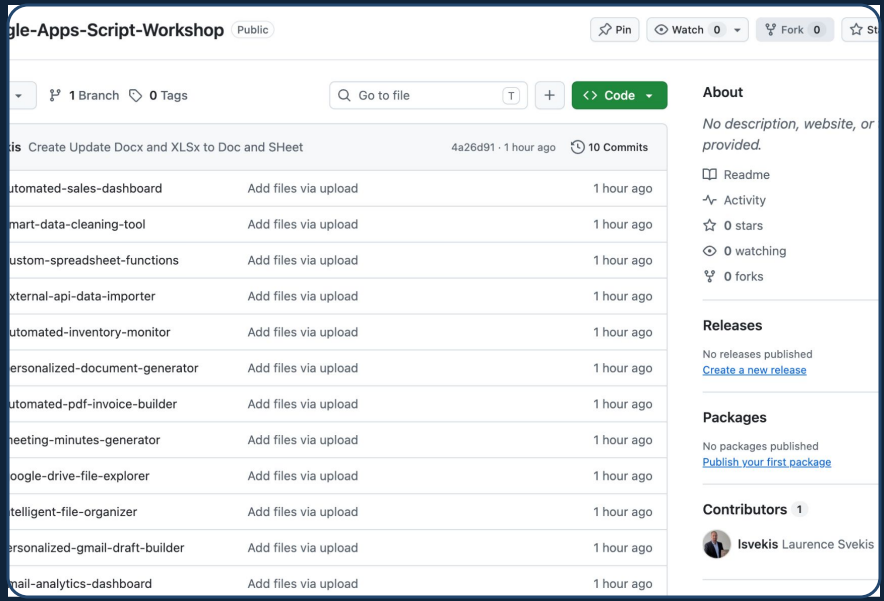
Open your lesson Sheet

Extensions → Apps Script → paste into Code.gs; save.

04

Run the lesson

Add any required HTML, services or API settings; authorize.



Open workshop GitHub repository ↗

What is Google Apps Script?

01 / CONNECTS TO

Workspace Apps

Integrates directly into your everyday Google tools:

- Google Sheets
- Google Docs
- Gmail
- Google Drive

02 / POWERED BY

Cloud Engine

Zero setup or local environment installation needed:

- Cloud-hosted JavaScript
- Built-in Google services
- Automatic security & auth
- Runs on Google servers

03 / DELIVERS

Smart Workflows

Build end-to-end automated solutions seamlessly:

- Automated reports & drafts
- Custom UI menus & dialogs
- Time-driven triggers
- Cross-app automation

Your first script in 60 seconds

01 CREATE SCRIPT
Open a blank Sheet → Extensions → Apps Script

02 PASTE CODE
Paste the code snippet into the editor window

03 EXECUTE
Save, select helloWorkspace, Run & authorize

Code.gs

```
function helloWorkspace() {

const sheet =
SpreadsheetApp.getActiveSheet();

sheet.getRange('A1').setValue('Hello
Workspace!');

}
```

TWO DIFFERENT SUPERPOWERS

Gemini: your builder and your AI engine

01 / CODE ASSISTANT

BUILD WITH AI

Use Gemini while developing to write, explain, and debug Google Apps Script.

WHEN TO USE IT

01 Building new scripts, menus, or custom functions

02 Automating rules-based math, formatting, or files

03 Zero API key or runtime AI cost required

DESIGN TIME • NO API KEY REQUIRED

02 / INTELLIGENCE ENGINE

BUILD AI INTO APPS

Embed the Gemini API in your running script to process data in real time.

WHEN TO USE IT

01 Analyzing unstructured feedback, emails, or docs

02 Summarizing, classifying, or extracting key fields

03 Drafting smart replies or dynamic insights

RUNTIME AI • GEMINI API KEY REQUIRED

Use Gemini as your coding partner at build time

01

WHAT IT IS

Prompt Gemini to write or debug Apps Script, then paste into your editor.

02

SETUP & COST

No API key, Script Properties, or runtime AI calls needed in your project.

03

HOW IT RUNS

Runs as pure JavaScript on Google servers—fast and deterministic.

</>

WHEN TO USE CODE ASSISTANT

01 **Build new tools:** Create custom functions, menus, sidebars, and triggers from scratch.

02 **Rules-based logic:** Calculate totals, format cells, copy templates, or organize Drive files.

03 **Debug & refactor:** Fix script errors, add safety checks, and speed up slow loops.

04 **Zero-cost runs:** Execute schedules or triggers repeatedly without API quota limits.

Embed Gemini API for real-time AI in your apps

01

WHAT IT IS

Your script calls the Gemini API at runtime to process live text and data.

02

SETUP & COST

Requires a Gemini API key in Script Properties; usage counts against quota.

03

HOW IT RUNS

Adds semantic reasoning where rigid if/else rules and regex break down.

AI

WHEN TO USE INTELLIGENCE ENGINE

01 **Unstructured text:** Classify feedback, triage support tickets, or score sentiment.

02 **Summarize & extract:** Condense long Docs or pull structured JSON from messy intake forms.

03 **Dynamic content:** Draft tailored Gmail replies or write data-driven executive insights.

04 **Translate & reason:** Interpret intent or normalize multi-language inputs automatically.

Prompt for a useful app, not just code

01 / INPUT SPECIFICATION

GIVE CONTEXT

Sheet name, columns, task and expected result.

- Specify sheet names & precise column layouts
- Describe the core automation task clearly
- Define expected outputs and target ranges

02 / DELIVERABLE STANDARDS

ASK FOR QUALITY

Complete commented code, setup steps, tests and safe improvements.

- Fully commented & clean code blocks
- Step-by-step setup & execution guide
- Built-in test cases & safety checks
- Optimization & error-handling recommendations

PROJECT MENU

Pick a project — all 30 are available

01 – 10 | CORE AUTOMATION

01 Automated Sales Dashboard
02 Smart Data Cleaning Tool
03 Custom Spreadsheet Functions
04 External API Data Importer
05 Automated Inventory Monitor
06 Personalized Document Generator
07 Automated PDF Invoice Builder
08 Meeting Minutes Generator
09 Google Drive File Explorer
10 Intelligent File Organizer

11 – 20 | AI & INTEGRATIONS

11 Personalized Gmail Draft Builder
12 Email Analytics Dashboard
13 Automated Task Management System
14 Interactive Workspace Sidebar
15 Gemini-Powered Data Classifier
16 AI Spreadsheet Insights Generator
17 AI Document Summarizer
18 AI Gmail Smart Reply Assistant
19 AI File Information Extractor
20 AI Business Workflow Assistant

21 – 30 | SMART ASSISTANTS

21 Personal Expense & Budget Tracker
22 Smart Meal Planner & Grocery List
23 Personal Subscription Tracker
24 AI Email Writing Assistant
25 Personal Appointment & Reminder
26 Travel Itinerary & Budget Planner
27 AI Personal Study Assistant
28 Household Maintenance Planner
29 AI Receipt & Expense Extractor
30 AI Weekly Personal Productivity

Automated Sales Dashboard

01 / PROBLEM

Manual regional totals are slow and error-prone.

02 / PRACTICAL BENEFIT

Small shops, course sales and department reporting

EXPECTED RESULT

East 160 | West 100 | Total 260

CODE SPOTLIGHT • buildDashboard.js

```
const totals = {};
input('Sales').forEach(r => {
  totals[r[1]] = (totals[r[1]] || 0)
    + num(r[3]) * num(r[4]);
});
```

TRY IT: Change East Course units from 3 to 4; total becomes 300.

Open Sales • buildDashboard()

Smart Data Cleaning Tool

01 / PROBLEM

Duplicate and malformed contacts pollute mailing lists.

02 / PRACTICAL BENEFIT

Volunteer lists, event registrations and customer records

EXPECTED RESULT

One valid unique contact; three issue rows (invalid email, duplicate, missing name).

CODE SPOTLIGHT cleanContacts.js

```
const [n,e,c] = r.map(v => String(v).trim());
const email = e.toLowerCase();
// Check missing name, email shape and
duplicates.
```

TRY IT: Correct Bob's email to bob@example.com; expect two clean rows and two issues.

Open Contacts **cleanContacts()**

Custom Spreadsheet Functions

01 / PROBLEM

Repeated pricing calculations invite formula mistakes.

02 / PRACTICAL BENEFIT

Quotes, order forms and discount calculations

EXPECTED RESULT

Net prices 45, 96, 60; bands Low, High, High.

CODE SPOTLIGHT • demoFunctions.js

```
s.getRange(n, 1, 1, 2).setFormulas([[
  `=NETPRICE(Orders!B${n},Orders!C${n},Orders!D${n})`,
  `=PRICEBAND(A${n})`
]]);
```

TRY IT: Set first discount to 0.2; net becomes 40 and band remains Low.

Open Orders • demoFunctions()

External API Data Importer

01 / PROBLEM

Copying external API records manually wastes time.

02 / PRACTICAL BENEFIT

Public data feeds and directory imports

EXPECTED RESULT

Three user rows if endpoint is available; first user Leanne Graham.

CODE SPOTLIGHT • importUsers.js

```
const res = UrlFetchApp.fetch(url,
  {muteHttpExceptions:true});
if (res.getResponseCode() !== 200)
  throw Error('HTTP ' + res.getResponseCode());
const data = JSON.parse(res.getContentText());
```

TRY IT: Change limit from 3 to 2; expect two users if endpoint succeeds.

Open Settings • importUsers()

Automated Inventory Monitor

01 / PROBLEM

Low stock can go unnoticed until supplies run out.

02 / PRACTICAL BENEFIT

Office supplies, classroom materials and small inventory

EXPECTED RESULT

A01 and A03 appear in Alerts.

CODE SPOTLIGHT • checkInventory.js

```
output('Alerts',
['SKU','Item','Stock','Reorder level'],
input('Inventory').filter(r =>
num(r[2]) <= num(r[3])));
```

TRY IT: Change A02 stock to 10; expect three alerts.

Open Inventory • checkInventory()

When should a script run?

01

Manual run

Learn and test first

STAGE 1
Development & Testing

02

Custom menu

Repeat on demand

STAGE 2
Interactive Execution

03

Time trigger

Schedule safe, repeatable work

STAGE 3
Background Automation

Personalized Document Generator

01 / PROBLEM

Writing personalized letters repeatedly is tedious.

02 / PRACTICAL BENEFIT

Welcome letters, enrollment notices and certificates

03 / EXPECTED RESULT

Two personalized Docs; template unchanged.

CODE SPOTLIGHT • generateLetters.js

```
const f = file('Letter Template')
.makeCopy('Welcome ' + name, folder());
const d = DocumentApp.openById(f.getId());
const b = d.getBody();
```

TRY IT: Add a third person; expect three newly generated letters.

Open People • generateLetters()

Automated PDF Invoice Builder

01 / PROBLEM

Manual invoice preparation repeats calculations and formatting.

02 / PRACTICAL BENEFIT

Freelancers, consulting and small service businesses

03 / EXPECTED RESULT

Two PDFs with totals 200 and 225 before tax.

CODE SPOTLIGHT • generateInvoices.js

```
const total = num(qty) * num(price);
// After creating and saving document f:
const pdf = folder().createFile(
f.getAs(MimeType.PDF)
.setName('Invoice ' + id + '.pdf'));
```

TRY IT: Change INV-101 quantity to 3; total becomes 300.

Open Invoices • generateInvoices()

Meeting Minutes Generator

01 / PROBLEM

Meeting notes and actions become scattered.

02 / PRACTICAL BENEFIT

Team meetings, clubs and project reviews

03 / EXPECTED RESULT

Three notes and two action items.

CODE SPOTLIGHT • generateMinutes.js

```
rows.forEach(r =>
b.appendParagraph(r[0] + ': ' + r[1]));
// Rows with owners become action items.
const actions = rows.filter(r => r[2]);
```

TRY IT: Assign Sam an owner and due date; expect three action rows.

Open Meeting • generateMinutes()

Google Drive File Explorer

01 / PROBLEM

Finding files and their metadata manually takes time.

02 / PRACTICAL BENEFIT

Project audits, shared folders and resource catalogs

03 / EXPECTED RESULT

Sample Brief and Sample Notes plus all other immediate lesson folder files, including guide and spreadsheet.

CODE SPOTLIGHT • listFiles.js

```
const it = folder().getFiles(), rows = [];
while (it.hasNext()) {
const f = it.next();
rows.push([f.getName(), f.getMimeType(),
f.getUrl(), f.getLastUpdated()]);
}
```

TRY IT: Add a sample file to the lesson folder; rerun and find its row.

Open Config • listFiles()

Intelligent File Organizer

01 / PROBLEM

Unorganized files are difficult to retrieve.

02 / PRACTICAL BENEFIT

Household records, finance and HR sample folders

03 / EXPECTED RESULT

Two proposed moves; MISC stays unchanged.

CODE SPOTLIGHT • previewMoves.js

```
const r = rules.find(r =>
f.getName().startsWith(r[0]));
if (r) rows.push([f.getId(),
f.getName(), r[1], false]);
```

TRY IT: Preview, approve FIN-Budget only, then apply moves and verify Finance folder.

Open Rules • previewMoves()

One template. Many documents.

01



Spreadsheet records

Source data driving automated generation

02



Reusable document template

Master layout with placeholder fields

03



Personalized documents in Drive

Final customized outputs generated automatically

Personalized Gmail Draft Builder

01 / PROBLEM

Repeated email composition takes time.

02 / PRACTICAL BENEFIT

Event invitations, follow-ups and onboarding

03 / EXPECTED RESULT

Two drafts; no sent messages.

CODE SPOTLIGHT • buildDrafts.js

```
const d = GmailApp.createDraft(email,
'Regarding ' + topic,
`Hi ${name},\nI am writing about ${topic}.\nThank
you.`);
s.getRange(i+2,4).setValue(d.getId());
```

TRY IT: Run twice; existing Draft IDs should prevent additional drafts.

Open Recipients • buildDrafts()

Email Analytics Dashboard

01 / PROBLEM

Inbox volume is hard to understand by inspection.

02 / PRACTICAL BENEFIT

Support triage and personal email workload

03 / EXPECTED RESULT

Alex 2 messages/1 unread; Jamie 1/1.

CODE SPOTLIGHT • analyzeMessages.js

```
function analyzeMessages() {
// Use sample messages, not the real inbox.
messageSummary(input('Messages'));
}
```

TRY IT: Change first sample unread value to FALSE; Alex unread count becomes zero.

Open Messages • analyzeMessages()

Automated Task Management System

01 / PROBLEM

Overdue tasks get lost in a long list.

02 / PRACTICAL BENEFIT

Team deadlines, assignments and personal chores

03 / EXPECTED RESULT

Slides Overdue; Budget Open; Review Done.

CODE SPOTLIGHT • updateTasks.js

```
function updateTasks() {
// Fixed date for repeatable workshop tests.
markTasks('2026-11-25');
}
```

TRY IT: Mark Slides Done; its priority becomes Done.

Open Tasks • updateTasks()

Interactive Workspace Sidebar

01 / PROBLEM

Editing rows directly can be awkward for nontechnical users.

02 / PRACTICAL BENEFIT

Intake forms, help requests and simple trackers

03 / EXPECTED RESULT

Submitting Test request / IT adds exactly one Open row.

CODE SPOTLIGHT • showSidebar.js

```
SpreadsheetApp.getUi().showSidebar(
HtmlService.createHtmlOutputFromFile('Sidebar')
.setTitle('New request'));
```

TRY IT: Submit an empty request; expect rejection; submit Test request / IT; expect one row.

Open Requests • showSidebar()

Connect Gemini safely

01 API KEY

Google AI Studio

Obtain an API key to enable Gemini integration.

02 SETTINGS

Apps Script

Navigate to Project Settings → Script Properties.

03 VARIABLES

Environment

Set GEMINI_API_KEY and GEMINI_MODEL properties.

Currently using : gemini-flash-lite-latest

04 VALIDATE

Testing

Use synthetic data; validate results before acting.

Create your Gemini API key

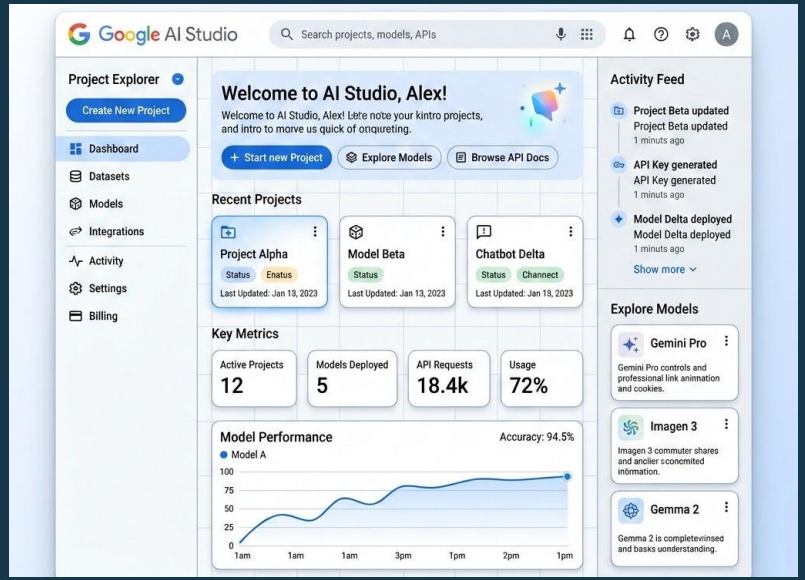
Connect your Apps Script application to Gemini.

01 **Open Google AI Studio**
https://aistudio.google.com/

02 **Create API key**
Sign in and select or create a Google Cloud project.

03 **Copy the key**
Keep it private. Never paste it into GitHub or source code.

GOOGLE AI STUDIO DASHBOARD



Configure Script Properties

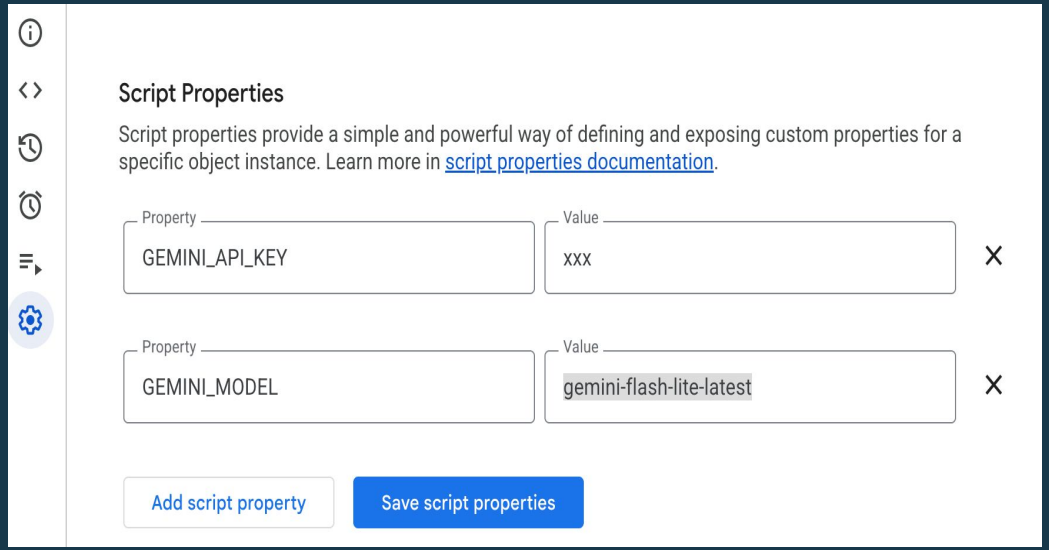
Open your lesson spreadsheet → Extensions → Apps Script.

01 Open Project Settings
Click the gear icon, then scroll to Script Properties.

02 Add GEMINI_API_KEY
Value: paste your personal API key.

03 Add GEMINI_MODEL
Use: gemini-flash-lite-latest (or gemini-2.5-flash). Save properties.

SCRIPT PROPERTIES DIALOG



Test the connection

Use the Workshop menu to check your setup.

01

Check permissions

Workshop → Check permissions verifies the lesson folder.

02

Test Gemini connection (optional)

Requires the test function and updated menu from the guide.

03

Authorize access when prompted

Review permissions; a successful test returns status: success.

Run the AI File Information Extractor

Turn unstructured Google Docs into structured spreadsheet data.

01

Prepare the lesson folder

Keep the spreadsheet and Google Docs Any related folder docs and files needed.

02

Run the application

Workshop → Run application. Open the Results tab.

03

Review the extracted data

Name • Request • Amount • Source. Ensure no errors, if errors troubleshoot the connection to Gemini

Gemini-Powered Data Classifier

01 / PROBLEM

Manual feedback classification does not scale.

02 / PRACTICAL BENEFIT

Course evaluations, customer comments and ticket routing

03 / EXPECTED RESULT

Expected categories Praise, Bug, Feature; AI output may vary.

CODE SPOTLIGHT • classifyFeedback.js

```
const result = ai(
'Return array of {id,category}; categories Praise,
Bug, Feature.',
rows);
// validateIds checks record coverage.
```

TRY IT: Add ID 4 with a clear bug report; expect four IDs and Bug for ID 4 after review.

Open Feedback • classifyFeedback()

AI Spreadsheet Insights Generator

01 / PROBLEM

Numbers alone do not explain business performance.

02 / PRACTICAL BENEFIT

Monthly budgets, project metrics and sales reviews

03 / EXPECTED RESULT

Profit 300, 400, 50; total 750; November decline identified.

CODE SPOTLIGHT • generateInsights.js

```
const rows = input('Metrics').map(
([m,r,c]) => [m,num(r)-num(c)]);
const total = rows.reduce((a,r) => a+r[1],0);
```

TRY IT: Change Nov cost to 800; profit becomes 100 and total 800.

Open Metrics • generateInsights()

AI Document Summarizer

01 / PROBLEM

Long documents take time to review.

02 / PRACTICAL BENEFIT

Project briefs, meeting notes and study material

03 / EXPECTED RESULT

Summary mentions pilot, 20 participants and unresolved budget.

CODE SPOTLIGHT • summarizeDoc.js

```
const source = DocumentApp.openById(
file('Project Brief').getId())
.getBody().getText();
```

TRY IT: Add a fictional decision to Project Brief; check summary against source.

Open Config • summarizeDoc()

AI Gmail Smart Reply Assistant

01 / PROBLEM

Responding to repetitive messages takes time.

02 / PRACTICAL BENEFIT

Support replies, scheduling and class communications

03 / EXPECTED RESULT

Two suggestions; zero drafts until approved.

CODE SPOTLIGHT • suggestReplies.js

```
const rows = input('Messages').filter(r =>
!existing.some(e =>
String(e[0]) === String(r[0])));
```

TRY IT: Generate suggestions, approve one, create drafts twice; expect one draft for that ID.

Open Messages • suggestReplies()

AI File Information Extractor

01 / PROBLEM

Extracting fields from intake forms by hand is slow.

02 / PRACTICAL BENEFIT

Training requests, applications and expense intake

03 / EXPECTED RESULT

A: Alex, Training, 250; B: Jamie, Equipment, blank amount (AI null becomes blank).

CODE SPOTLIGHT • extractFiles.js

```
const x = ai(
'Return object {name,request,amount}; amount
number or null.',
DocumentApp.openById(f.getId())
.getBody().getText());
```

TRY IT: Add Amount: 100 to Intake B; verify output against source.

Open Config • extractFiles()

AI is not the final authority

01 / SCHEMA CHECK



Check JSON shape and original IDs

Ensure structure matches expected contract before processing downstream.

02 / DATA VALIDATION



Reject unexpected values and missing fields

Filter out malformed output and enforce mandatory field constraints.

03 / HUMAN-IN-THE-LOOP



Review drafts and sensitive decisions

Mandate explicit human approval before executing critical actions.

AI Business Workflow Assistant

01 / PROBLEM

Requests get delayed when intake and replies are disconnected.

02 / PRACTICAL BENEFIT

Internal service desks, training and IT intake

03 / EXPECTED RESULT

R1 Training, R2 IT expected; one summary Doc; no email sent.

CODE SPOTLIGHT • processRequests.js

```
const rows = input('Requests').filter(r =>
r[4] === 'New' && !existing.some(e =>
String(e[0]) === String(r[0])));
```

TRY IT: Add R3 with New status; expect a new review row and report without losing existing approvals.

Open Requests • processRequests()

Capstone: connect the pieces

01 / INTAKE



Incoming requests

Gather and buffer incoming service desk and user workflow requests.

02 / PROCESSING



Gemini classification + JavaScript checks

Analyze intent using Gemini AI while running automated Apps Script validation.

03 / APPROVAL



Reviewed report and approved drafts

Generate summary reports and prepare final response drafts for human review.

Personal Expense and Budget Tracker

01 / PROBLEM

Small purchases add up without visibility.

02 / PRACTICAL BENEFIT

Household spending and simple expense reviews

03 / EXPECTED RESULT

Food 85; Transport 40; total 125

CODE SPOTLIGHT • summarizeExpenses.js

```
const m = {};
input('Expenses').forEach(r =>
m[r[1]] = (m[r[1]] || 0) + num(r[2]));
const rows = Object.entries(m);
```

TRY IT: Add Food expense 15; Food becomes 100 and overall total 140.

Open Expenses • summarizeExpenses()

Smart Meal Planner and Grocery List

01 / PROBLEM

Shopping lists duplicate ingredients and omit quantities.

02 / PRACTICAL BENEFIT

Family meal planning and shared groceries

03 / EXPECTED RESULT

Tomatoes 5 each; Lettuce 1 each

CODE SPOTLIGHT • buildGroceryList.js

```
const m = new Map();
input('Meals').forEach(r => {
const k = JSON.stringify([r[2],r[4]]);
m.set(k,(m.get(k)||0)+num(r[3]));
});
```

TRY IT: Add Tomatoes 1 each; expected Tomatoes 6 each.

Open Meals • buildGroceryList()

Personal Subscription Tracker

01 / PROBLEM

Recurring charges are hard to compare across billing cycles.

02 / PRACTICAL BENEFIT

Streaming, software and household subscriptions

03 / EXPECTED RESULT

Monthly 17; annual 204

CODE SPOTLIGHT • trackSubscriptions.js

```
const m = num(c) /
(cycle === 'Annual' ? 12 : 1);
return [s, m, m*12, day(d)];
```

TRY IT: Change Cloud annual cost to 120; total monthly 22 and annual 264.

Open Subscriptions • trackSubscriptions()

AI Email Writing Assistant

PROBLEM

Finding appropriate email wording takes time.

PRACTICAL BENEFIT

Personal invitations, appointment requests and professional messages

EXPECTED RESULT

Two suggested emails; no sends

CODE SPOTLIGHT

```
const result = validateIds(ai(
'Return array {id,subject,body}; follow requested tone.',
rows),rows);
```

TRY IT: Change first tone to Professional; review rewritten subject and body.

Open Emails • writeEmails()

Personal Appointment and Reminder Manager

PROBLEM

Upcoming appointments can be overlooked.

PRACTICAL BENEFIT

Family schedules, personal appointments and service bookings

EXPECTED RESULT

Dentist only

```
input('Appointments').filter(r =>
day(r[1]) >= '2026-11-25' &&
day(r[1]) < '2026-12-02');
```

Open Appointments • showAppointments()

Travel Itinerary and Budget Planner

PROBLEM

Travel costs are scattered across bookings.

PRACTICAL BENEFIT

Weekend trips, family vacations and business travel

EXPECTED RESULT

Day one 230; day two 20; total 250

```
const m = {};
rows.forEach(r =>
m[r[0]] = (m[r[0]] || 0) + r[2]);
```

Open Trip • planTrip()

AI Personal Study Assistant

PROBLEM

Passive reading makes it difficult to check understanding.

PRACTICAL BENEFIT

Students, professional learners and language study

EXPECTED RESULT

Four question-answer pairs

```
const q = ai(
'Return array of exactly two objects
{question,answer} grounded in notes.',
note);
if (!Array.isArray(q) || q.length !== 2)
throw Error('Expected two questions');
```

Open Notes • makeQuiz()

Household Maintenance Planner

PROBLEM

Recurring home maintenance is easy to forget.

PRACTICAL BENEFIT

Air filters, inspections and seasonal chores

EXPECTED RESULT

Filter overdue; smoke alarm test upcoming

CODE SPOTLIGHT

```
const d = new Date(day(last)+'T00:00:00Z');
d.setUTCDate(d.getUTCDate()+num(interval));
const due = d.toISOString().slice(0,10);
```

TRY IT: Set Filter last done to 2026-11-20; expect Upcoming.

Open Maintenance • planMaintenance()

AI Receipt and Expense Extractor

PROBLEM

Receipt entry is repetitive and prone to transcription errors.

PRACTICAL BENEFIT

Personal expense records and reimbursement preparation

EXPECTED RESULT

Market 23.50 CAD; Cafe 8 CAD

CODE SPOTLIGHT

```
const result = validateIds(ai(
'Return array
{id,merchant,total,currency,date}; date
YYYY-MM-DD.',
rows),rows);
```

TRY IT: Change Cafe total to 10 CAD; verify extracted total against text.

Open Receipts • extractReceipts()

AI Weekly Personal Productivity Assistant

PROBLEM

Outstanding tasks can feel disorganized across deadlines.

PRACTICAL BENEFIT

Weekly planning, household chores and personal projects

EXPECTED RESULT

Plan includes bill and book; excludes laundry

CODE SPOTLIGHT

```
const rows = input('Tasks')
  .filter(r => r[2] === 'Open')
  .map(r => [r[0],day(r[1])]);
// Pass only open tasks to Gemini.
```

TRY IT: Mark Pay bill Done; verify plan mentions only Read book.

Open Tasks • planWeek()

Automate only after it works

01

Manual run and test

Validate code logic and edge cases manually before creating automations.

02

Install a suitable trigger

Attach time-driven or event-based triggers once execution is completely reliable.

03

Monitor results and prevent duplicates

Track execution logs, set up error notifications, and enforce idempotent runs.

Your 31st application

01

Choose a repetitive task you actually do

02

Define inputs, outputs and a measurable success test

03

Build the smallest useful version

04

Improve it with Gemini and share what you learned

Your next app starts with one task



GitHub — revisit all 30 solutions

github.com/Isvekis/Google-Apps-Script-Workshop



Google Drive — copy starter files and guides

Open shared workshop folder (click here)



Starter generator — build your own workshop library

WorkshopGenerator30.txt • supplied separately with workshop materials

Thank you! Questions?

Q&A & Discussion

Have questions about the 30 projects or custom Apps Script implementations? Reach out anytime!

More Code & Repos

https://github.com/lsvekis

Explore more open-source tools, scripts, and Google Workspace automations.

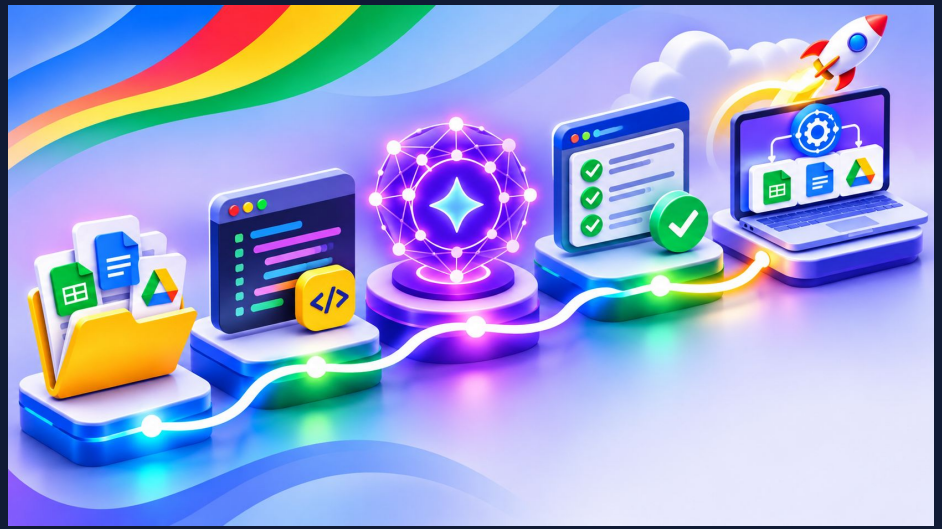
Follow My Work

basescripts.com
Linkedin
vibelearning.ca
discoveryvip.com

Stay updated with new courses, tutorials, and practical AI automation projects.

Your Learning Journey

Explore. Build. Integrate. Test. Customize.



01 Explore — Access starter files and resources

02 Build — Create Workspace applications

03 Integrate — Connect Gemini to workflows

04 Test — Validate outputs and troubleshoot

05 Customize — Adapt solutions to your needs

YOUR NEXT STEP

Your next automation starts today.

One task. One idea.
One useful application.

Explore → Build
Test → Improve

30 projects · Source code · Starter files

[GitHub: Google Apps Script Workshop]

