



HANDS-ON GUIDE

Mechanical and Electrical track Day 2 Hands-on Guide

Hands-on AI-Assisted Development, Mechanical and Electrical track, Day 2

This is a guide to keep at hand for the 8 exercises of Day 2. It gives the steps, how to check your work, and how to get back on track when you are stuck. The slides keep moving ahead, so come back to this page when you stall. The prompts shown on the slides are included here as they are, so you can copy them instead of retyping them.

What you do today

Today you spend more time working with your hands than listening to explanations. In the morning, you build one thing that runs in the browser, on a topic you choose. In the afternoon, you clean up a messy Excel file and take it as far as a bar chart.

Time	Description	Minutes
Morning	Day 1 review and environment check (Exercise 1)	[40min]
Morning	Introduction to vibe coding (Exercise 2)	[130min]
Morning	Break	[10min]
Afternoon	Prompt basics	[60min]
Afternoon	Rewriting prompts (Exercise 3)	[40min]
Afternoon	Break	[10min]
Afternoon	Data preparation (Exercise 4)	[80min]
Afternoon	Working with tabular data (Exercises 5, 6 and 7)	[70min]
Afternoon	Putting a work theme into words (Exercise 8)	[30min]
Afternoon	Closing	[10min]

The morning and afternoon add up to 480 minutes. The instructor writes the time to resume after each break on the board. Before you leave your seat, save the file you are working on with Ctrl+S.

You use only 3 tools today. You write files in VS Code, ask GitHub Copilot in English, and run the result in a browser such as Edge. There is no installation work during the course, and you do not need administrator rights.

What you use

You use 2 folders: the folder you are given and the folder you create yourself. If you mix them, you lose track of where files are saved, so keep their roles separate.

Handout folder

Right-click the handout ZIP, choose "Extract All", and extract it to the desktop. A folder named `tpro-work` is created. Do not move the files inside. `01_data-loader.html` runs using the parts in the `lib` folder, so if you move only the HTML file somewhere else, it stops working.

File	Role
01_data-loader.html	A page that opens Excel in the browser and lets you view it. Choose a sheet to see its table, and "Copy this table" copies it as tab-separated text.
equipment-data_2026Q1.xlsx	Fictional equipment data used in the afternoon exercises. It has 3 sheets: Operations, Inspections and Defects.
maintenance-records_2026H1.csv	Fictional maintenance records used in Exercise 6. The way Duration is written and the way Assignee names are spelled differ from row to row.
equipment-data_2026Q2.xlsx	Equipment data used in the extra exercise for people who have time left. Unlike Q1, it has a different number of machines and a different kind of mess.
lib folder	Parts that make the pages run in the browser. The folder holds <code>xlsx.full.min.js</code> (reads Excel), <code>papaparse.min.js</code> (reads CSV) and <code>chart.umd.js</code> (draws charts).
README.txt	An explanation of the handouts, and where to check when a file does not open.

If you open `equipment-data_2026Q1.xlsx` in Excel and save it over the original, the "dirty" parts of the data that the exercises work on disappear, and the exercise no longer works. To see the contents, always use `01_data-loader.html`.

The tpro-work folder you create

Create a folder named `tpro-work` on the desktop. Use half-width, lowercase characters. You put everything you make today in this folder. In the morning, `index.html` is added. In the afternoon, `prompt.txt`, `clean.csv`, `chart.html` and `theme.txt` are added.

You go back and forth between the handout folder and the `tpro-work` folder only once. In the afternoon chart exercise, you copy `chart.umd.js` from the `lib` folder into the `tpro-work` folder. You do not move anything else between them.

You use this `tpro-work` folder on Day 3 as well. Do not delete it even after the course ends.

Steps to view Excel with 01_data-loader.html

A browser cannot open files on your PC by itself. You give it a file by dragging. In the afternoon, Exercises 4, 5, 6 and 7 all start with these steps.

1. Open the `tpro-work` folder on the desktop in File Explorer
2. Double-click `01_data-loader.html`. Open it in the browser, not in VS Code
3. A dotted frame for dropping the file appears in the browser
4. Drag `equipment-data_2026Q1.xlsx` from the same folder and drop it into the dotted frame
5. The names of the 3 sheets, Operations, Inspections and Defects, are listed. Click the sheet name you want to see
6. Click "Copy this table" and that sheet is copied as tab-separated text. Paste it into the Copilot Chat input box

If double-clicking opens VS Code, right-click `01_data-loader.html` , choose "Open with", and select a browser.

The 10 practice rows used in Exercise 3 in the first half of the afternoon are also at the bottom of this page.

Exercise 1: Environment check

This exercise takes about [12min].

Scenario

It has been about a week since Day 1. Every time, some participants find that they are signed out or that the Chat view will not open. The next 130 minutes run on the same screen for everyone, so we get everyone set up first. If you fix it here, no time is lost on environment problems during the exercises.

Steps

1. Start VS Code. Open it from the Start menu or the desktop icon
2. Click the Copilot icon at the top right of the window and check that the Chat view opens on the right. This is where you ask in English
3. Look at the account icon at the bottom left. If your GitHub user name is not shown, click it and sign in again
4. Type "Hello. What can you do today?" in the chat and press Enter. Check that a reply comes back in English
5. Create a folder named `tpro-work` on the desktop, and open it in VS Code with File > Open Folder

If you open only a file in Step 5 without opening the folder, you will lose track of where it is saved later. Always open `tpro-work` with "Open Folder".

First, agree on the names of 4 parts of the screen. The vertical strip at the far left is the Activity Bar. To its right, the area that shows the contents of the folder is the Explorer. The center, where you write files, is the editor. The right side, where you type your requests, is the Chat view. The strip at the very bottom is the Status Bar, and you can see your sign-in state there.

Check your result

You are done when these 4 points are met.

- The Chat view is open on the right
- The account icon at the bottom left shows your GitHub user name
- You asked a question in English and got a reply in English
- The Explorer heading shows DAY2

No deliverable.

If it does not work

Symptom	Fix
Stuck on the sign-in screen	A browser has opened and is waiting for your approval. Approve in the tab that opened. If no approval screen appears, a network restriction is the cause, so raise your hand.
You cannot find the Copilot icon	Your VS Code version may be old. Chat is built in from 1.116 onward. If it still does not appear after you reopen VS Code, raise your hand.
You opened a file without opening the folder	If you continue like this, you will lose track of where files are saved. Start again from Step 5.
No reply comes back	First, check the notifications at the bottom right of the window. If the reply still does not come back, you can go on while looking at the screen next to you.
You do not know where you created the tpro-work folder	If you created it somewhere other than the desktop, it is faster to create it again.
The whole display is in English	Keep the display language in English.

If time allows

- Enter "Write HTML code that shows today's date on the screen" in the chat and look over the code that comes back. It has the same form as the one you use in the morning exercise.
- Compare how the reply opens with what you got on Day 1, and check whether the tone is the same.
- Right-click in the Explorer and see which menu appears. In the next exercise, you create a new file from here.

Exercise 2: Build one thing on your own topic

This exercise takes about [75min]. The breakdown is: choose a topic [5min], run the first version [25min], fix what is off [25min], add one feature [15min], write down the prompt that worked [5min].

Scenario

The feel of getting something that works without writing code is hard to get from a talk. If you go through it once yourself, then in the afternoon, when you face an Excel file, you will notice that you can ask for this too. You build the Day 3 mini app for improving your work in the same way.

Plan the build as 3 rounds. Round 1 passes if it runs. Round 2 is for the look. In round 3, you add one feature. Each round repeats the same loop: tell, run, point out what is off, have it fixed. If you try to finish in one round, it actually takes longer.

Choose a topic from the 4 below. If you thought of one at the end of Day 1, use that.

Type	What to build	Sign that it works
Count things	A parts counter. Each press of the button adds 1, and Reset sets it back to 0.	The number on the screen changes with each press
Convert units	Conversion between mm and inch, and between N · m and kgf · m. Enter a value on the left and the result appears on the right.	The number on the right follows when you change the input
Prevent missed checks	A pre-start inspection checklist. It shows how many items are checked and how many remain.	The remaining count goes down when you check an item
Game	Whack-a-mole, number guessing, rock-paper-scissors. The win or loss shows right away.	The score goes up, or a win or loss appears

Do not use real model numbers, tolerances or client names. Use fictional numbers.

Steps

1. Choose one topic. Decide within [5min] and move on.
2. Ask in 3 to 5 lines. Write 4 things: what to build, what to show on the screen, what happens when the user operates it, and that everything goes into one HTML file.
3. Right-click below the DAY2 heading in the Explorer, choose "New File", type `index.html` and press Enter
4. Click the copy button at the top right of Copilot's reply, paste it into the editor and save with Ctrl+S. If you select the code in the block by hand, part of it can be left out.
5. Open the tpro-work folder on the desktop and double-click `index.html` to open it in the browser. It does not run inside VS Code
6. When it works, save one more copy of the file at that point as `index_ok.html`. If you break it while refining it, you can come back to this copy
7. Tell it one thing that is off at a time and have it fixed. After the fix, save the file and press F5 in the browser to see the change
8. Finally, write down the prompt that worked, in one line, in `prompt.txt`. You use it in the prompt exercise in the afternoon

If you skip Step 6 and break the file while refining it, the whole exercise time is lost. Always save a copy under another name at the point where it works.

For your first prompt, you can use the following form as it is. Add the last line to every prompt all day today.

```
Build a whack-a-mole game in HTML.
- For 30 seconds, moles appear at random in a 3×3 grid
- Clicking a mole adds 1 to the score
- Show the time left and the score at the top of the screen
- Put everything in one HTML file and do not load any external files
```

If you drop the last line, the reply may contain code that loads parts from an external URL. If the company's network restrictions block it, the screen goes completely blank.

When it does not get fixed, write the parts that work and the parts that do not separately. This raises the chance of a fix. If you send only "It does not work", the AI cannot tell what to fix, so it rewrites everything and breaks even the parts that worked. You can use the following form.

```
In the whack-a-mole game I am building, the score stays at 0 when I click a mole.
The countdown of the time left is working.
Fix only the cause of the score not going up. Do not change any other behavior.
Show the whole fixed HTML again.
```

If you leave out "Do not change any other behavior", even the parts that were working may be rewritten.

Check your result

You are done when these 3 conditions are met. How polished the screen looks is not part of the criteria.

- The screen appears when you open it in the browser
- Something on the screen changes when you click or do another operation
- At least 1 of the fixes you asked for is reflected

The tpro-work folder keeps `index.html` . `index_ok.html` and `prompt.txt` are kept too. Nothing to submit.

If it does not work

Symptom	Fix
The screen is blank	First press Ctrl+S. If the screen is still white, press F12 and paste the red text shown in the Console into Copilot as it is
The code is shown as plain text when you open the file	The file name has become <code>index.html.txt</code> . In File Explorer, check View > File name extensions and fix the name
Nothing changes after the fix	You have not saved, or you have not pressed F5 in the browser
It broke after more changes	Paste the contents of <code>index_ok.html</code> back into <code>index.html</code> to go back
The topic is too big to finish	Cut it down to a size that fits on 1 screen. Reduce the number of fields or narrow it to 1 feature
You do not know where to paste the code	Check that DAY2 appears in the Explorer. If it does not, open the folder again

If no screen appears after [30min], raise your hand.

Comprehension check 1

You opened what you built in the browser, and the screen is completely white. What do you do first?

- A Ask Copilot to rebuild it from scratch
- B Paste the error message shown by F12 into Copilot as it is
- C Open it again in a different browser
- D Restart the computer

Read this after you have thought it through yourself. The correct answer is B. The screen turns completely white because processing stopped partway. In the Console that opens with F12, the place where it stopped is shown in red text. If you paste that 1 line as it is, Copilot can fix only the cause. You do not need to understand what it means. With A, even the parts that were working are rewritten, and other bugs are added. With C, the result is the same in any browser if the cause is in the code. D does not help with a forgotten save or a mistake in the code. If the screen stays white and no red text appears, the cause is a forgotten save or a wrong file name.

If time allows

- Save the score and the input in the browser so that they remain after you close the screen
- Keep the layout from breaking at a smartphone screen width
- Add a switch for difficulty or units
- Match the color scheme to the notices at your workplace, and use colors that are still readable when printed

Exercise 3: Rewriting the prompt

The working time is [30min]. The breakdown is: send a rough prompt [5min], add the 4 elements and resend [15min], and write down the differences [10min].

Scenario

In the morning, you asked for things in the order they came to mind and fixed the results as you looked at the replies. In the data cleaning this afternoon, the same way of writing fills missing values with made-up numbers and leaves text with units as it is. If you build one rewriting pattern now, you can start the afternoon just by pasting it.

The AI decides any condition you did not write and moves on. If you do not write the period, the whole period is collapsed into 1 row. If you do not write the unit of aggregation, the 8 machines become a single mean value. If you do not write how to treat blanks, plausible numbers are filled in. It is the same as a machinist deciding by their own judgment when a drawing has no dimension: whatever your instruction leaves out, the other side decides.

Build a prompt from 4 elements: role, context, constraints and output format.

Element	What you write	Example
Role	Decide the viewpoint the AI answers from. The level of detail in terms and the degree of abbreviation become consistent	As an equipment maintenance engineer at a manufacturing site

Element	What you write	Example
Context	Write why you are doing this and what you have on hand	I have operating data for 8 machines pasted from Excel. I will use it for the monthly inspection report
Constraints	Write what you do not want it to do	Do not invent numbers by guessing, and do not leave text with units
Output format	Specify the form you want back. Once the form is fixed, you can compare it with the previous result	As a table with 3 columns: Equipment ID / Operating hours / Defects count

The prompt with all 4 elements is the following 6 lines. The square brackets at the start of each line show which element the line belongs to. You can remove them when you actually send the prompt.

```
[Role] Answer as an equipment maintenance engineer at a manufacturing site.
[Context] I have operating data for 8 machines pasted from Excel. I will use it for the monthly inspection report.
[Constraint] Convert the operating hours to numbers. Move the unit h to the column name.
[Constraint] Do not fill blank cells with 0. Leave them blank, and write the number of blank cells at the end.
[Constraint] Do not guess values that are not in the data.
[Output format] Return the result as a table. Use 3 columns: Equipment ID / Operating hours / Defects count.
```

When writing all 4 elements is too much trouble, just adding these 3 lines after your prompt also changes the result.

```
- Return the result as a table. Use the columns Equipment ID / Operating hours / Defects count.
- Use only half-width digits for numbers, and write the unit in the column name.
- Leave values that are not in the original data blank. Do not fill them in by guessing.
```

If you leave out line 3, plausible numbers get filled into the blank cells. The Excel file handed out in the afternoon has 2 missing values.

Steps

1. Copy the 10-row practice table and paste it into the chat, then send only "Analyze the operating data". Skim the reply for just 30 seconds [Required, 5min]
2. Open a new chat and paste the same table again. Add the role, context, constraints and output format one by one, and send it again. Always include "Do not fill in values that are not in the original data by guessing" in the constraints [Required, 15min]
3. Write down in 3 lines in `prompt.txt` what changed between the 1st and 2nd reply. Look at 3 things: the format of the reply, how numbers are handled and the amount of explanation [Required, 10min]
4. If you finish early, remove only the line "Do not fill them in by guessing", send the table including EQ-004, whose Operating hours is blank, and check that a number gets filled into the blank
5. If you finish early, put the 3 lines you use most often at the top of `prompt.txt`, so that from next time you can use them just by pasting

Step 2 opens a new chat because, if you keep going in the same conversation, the reply comes back with the 1st instruction still in effect and no difference shows. Create a new chat with the New Chat icon at the top of the Chat view.

The 10 practice rows are near the bottom of the `01_data-loader.html` page. The equipment is EQ-001 to EQ-010, 10 machines in all, and 3 problems are planted. The Operating hours of EQ-004 is blank, the Operating hours of EQ-006 is the text "182.5 h" with a unit, and the Stops of EQ-009 is blank. This is a practice table, separate from `equipment-data_2026Q1.xlsx` , which is handed out in the afternoon.

Create `prompt.txt` by right-clicking under tpro-work in the Explorer and choosing "New File". Open it in the editor, paste the prompt, and save with Ctrl+S.

Press Enter to send a message to the chat, and Shift+Enter to start a new line in the middle. If you press Enter when pasting a table, the message is sent partway.

Check your result

You are done when these 4 points are met.

- The 2nd reply is a table with the Equipment ID, Operating hours and Defects count columns
- "182.5 h" no longer remains in the Operating hours column, and it is the number 182.5
- The Operating hours of EQ-004 is still blank, and no made-up number has been filled in
- The full text of the rewritten prompt is pasted and saved in `prompt.txt` in the tpro-work folder

Read this after you have sent the prompt twice yourself. The 1st time, you get about 5 lines of text that sum up the 10 machines. There are no numbers for each machine, and EQ-004, whose Operating hours was blank, may get a number close to the mean. The Stops total also counts blanks as 0. The 2nd time, you get a table with 3 columns, 182.5 h becomes 182.5, and EQ-004 stays blank. A count such as "1 blank" may be added at the end. The difference came from 2 parts: the constraints and the output format.

If it does not work

Symptom	Fix
The reply is text, not a table	The output format line is buried in the middle of the prompt. Move it to the end and send it again
The 2nd reply is similar to the 1st	You are still in the same conversation. Start a new conversation with the New Chat icon at the top of the Chat view, then paste
A number gets filled into EQ-004	The constraint line runs into other text. Start the line with "- " to make it a separate line
The message is sent partway when you paste a table	You are using Enter for the line break. Use Shift+Enter for a line break

Symptom	Fix
You cannot find prompt.txt	It was created somewhere other than the tpro-work folder. Create it again directly under tpro-work in the Explorer

If time allows

- Send the same request 3 ways: "Return it as a table", "Return it in CSV format" and "Return it as a bulleted list". Check which is easiest to paste into Excel
- Change only the role to "Explain it as if you were explaining to a new operator", send it, and see how the level of detail in the wording changes
- Add one line to the constraints, "Sort by Operating hours, longest first", and test whether you can ask for sorting as well

Exercise 4: Data preparation of a messy Excel file

The working time is [32min]. The breakdown is: get the table ready to paste [5min], send the prompt and the table [10min], compare with the check values [8min], fix the items that do not match [4min], and save [5min].

Scenario

Many of you probably fix the monthly equipment Excel file by hand before totaling it. You move the headers, delete the units and fix full-width digits. You test whether 1 prompt today can replace this manual rework.

`equipment-data_2026Q1.xlsx` keeps the typical mess found in shop-floor files as it is.

Sheet	Contents	Main problems
Operations	Operating hours, Stops and Defects count of 8 machines	Title in row 1, headers in row 3, a blank row in the middle. Operating hours are text with units, and there are 2 missing values
Inspections	Measured size and Result of 10 lots	Full-width digits in the measured values, dates written in 3 ways, and full-width hyphens in the Result column
Defects	Defect type and Count for each process	Process names are in vertically merged cells, and the cells from the 2nd row on are empty

This exercise uses only the Operations sheet. If you paste all 3 sheets together, the reply is cut off partway.

3 kinds of mess are easier to fix if you know them in advance.

The headers are in row 3. Row 1 holds only a title in a single cell, and row 2 is empty. From the AI's point of view, nothing tells it whether row 3 is the headers or data. A blank row in the middle can also make it conclude that the table ends there.

In merged cells, a value remains only in the first row of the range. Even if the cell looks like one block spanning 3 rows on the Excel screen, rows 2 and below are empty once the file is loaded. This mess is in the Defects sheet, and

it matters in Exercise 7.

Filling a blank with 0 changes the mean. If you treat the 1 missing Operating hours value as 0 and divide by 8 machines, the mean is about 160.7 hours, which invites the reading that this machine was stopped. If you exclude the 1 missing value and divide by 7, it is 183.64 hours. Always write the number of excluded values alongside, to show that it is a mean of 7 machines.

Steps

1. In the handout folder, double-click `01_data-loader.html` to open it in the browser, and drag and drop `equipment-data_2026Q1.xlsx` onto the dotted box. Click the Operations sheet and copy it with "Copy this table" [5min]
2. Open Copilot Chat, copy the 7-item prompt below and paste it first. Right below it, paste the contents of the Operations sheet you copied, and send both as 1 message [10min]
3. Check the table that came back against the 6 check items below, one by one [8min]
4. Tell the chat again only the items that did not match, and have them fixed. If you have it redo everything, other parts get thrown off [4min]
5. Copy the cleaned table. In the VS Code Explorer, right-click, choose New File, type `clean.csv`, paste, and save with Ctrl+S [5min]

If you send the prompt and the table as separate messages, the AI replies "Please paste the table". Put them in 1 message.

For the cleaning prompt, do not write your own. Copy this one, then paste the table.

```
As an equipment maintenance engineer, clean up the table pasted below.
1) The headers are in row 3. Ignore rows 1 and 2 and the blank rows in the middle
2) If the Process column is empty, fill it with the process name from the row above
3) Remove the unit from values like "182.5 h" and make them half-width numbers
4) Convert full-width digits like "1 0.0 5" to half-width
5) Use the form 2026-04-01 for dates
6) Do not fill blanks with 0. Leave them blank and count them
7) At the end, write the number of cells that could not be converted to numbers
```

Line 7 prevents errors. If you do not write it, cells that could not be converted are silently replaced with 0, and the mean goes down.

Check your result

Check the table that came back against these 6 items.

Check item	Expected value
Number of machines	8
Missing Operating hours	1
Missing Stops	1

Check item	Expected value
Mean Operating hours	183.64 hours
Median Operating hours	182.5 hours
Cells that could not be converted to numbers	0

If 5 or more of the 6 items match, you can move on. Be sure to match these 2: 8 machines and a mean of 183.64 hours. Also check that no "h" and no full-width digits remain in the Operating hours column.

`clean.csv` is created in the `tpro-work` folder. When you open it in VS Code, you are done if it has rows for the 8 machines and 1 header row. Even if the check values do not match, save the file for now after [25min]. The next exercise assumes this file.

If it does not work

Symptom	Fix
The number of machines is 9 or 10	It is counting the title row or the blank row. Send only item 1 of the prompt again
The mean comes out in the 160-hour range	It is filling the missing values with 0. Send only item 6 of the prompt again
The number of cells that could not be converted does not reach 0	Values with units or full-width characters remain. Follow up with "List which cells still remain"
The reply is cut off partway	You pasted all 3 sheets together. Paste only the Operations sheet again
You cannot save <code>clean.csv</code>	Check in File Explorer, under View > File name extensions, whether the file name has become <code>clean.csv.txt</code>

Comprehension check 2

What happens if you load process names that are grouped by vertically merged cells as they are, and then summarize them?

- A Every row gets the process name, just as it looks in Excel
- B Merged cells cannot be loaded, so the whole row disappears
- C Only the first row is counted with the process name, and the rest have no process name
- D The empty process names are treated as the number 0

Think it through yourself before you read this. The correct answer is C. A merged cell only looks combined on the screen, and the value is stored only in the first cell of the range. After loading, row 2 and later are empty, and when you count by process, only the first row of Molding is left. A describes how it looks on the screen, not what is in the cell. If you click the cell and look at the formula bar, it is empty. In B, the rows themselves remain. Only the Process column becomes empty, and Defect type and Count come through as

they are. The empty cells in D are not 0. They come through as empty. Turning them into 0 is a decision made by whoever summarizes the data, and the original data does not say it.

If time allows

- Write the relationship between Operating hours and Stops in 1 line. You will use it as it is in the next exercise
- Apply the same prompt to the Inspections sheet and find out which of the 7 items you must rewrite for it to work

Exercise 5: Extract out-of-spec lots

The working time is [8min]. The steps are divided into 4, at 2 minutes each.

Scenario

If you scan the measured sizes by eye to find out-of-spec lots, you misread digits. On top of that, the measured values contain full-width digits. You write the conditions and have a machine look for what people overlook.

What to know before reading the numbers

If you look only at the mean of the cleaned Operations numbers, you miss the spread.

Item	Value	How to read it
Mean	183.64 hours	The mean of 7 machines, after excluding 1 missing value
Median	182.5 hours	It is almost the same as the mean. You can read this as no machine being extremely far off
Standard deviation	18.51 or 17.13 hours	It depends on the type of calculation: 18.51 for a sample, 17.13 for a population. Either way, read it as a spread of roughly ± 18 hours
Minimum and maximum	158.0 hours and 210.0 hours	The difference is 52 hours. One machine runs about 25 hours less than the others

The correlation between Operating hours and Defects count is -0.965. The numbers show a relationship in which machines with longer operating hours have fewer defects. However, you cannot say that running a machine longer reduces defects. It may be only that the machines that run well happen to be newer. This value comes from the 7 machines left after excluding the 1 machine whose Operating hours is empty, and removing just 1 machine can change it. Always state how many machines the correlation comes from. Use the correlation as material for deciding what to investigate next.

Steps

1. Drag `equipment-data_2026Q1.xlsx` onto `01_data-loader.html`, click the Inspections sheet, copy it with "Copy this table", and paste it into Copilot Chat [2min]

2. Send the request "Convert the full-width digits in the measured values to half-width digits, and put the dates in the form 2026-04-01" [2min]
3. Then send the request "The spec for Measured size is a lower limit of 9.9 and an upper limit of 10.1. List the Lot No values that are out of spec" [2min]
4. Note down the Lot No values that come out. Also count how many OK and how many NG the Result column of the original data has [2min]

If you skip Step 2 and send only the request for Step 3, the values are compared while still full-width, and the result goes wrong. Keep the order. Write the spec as "above the upper limit of 10.1", not "10.1 or more". If you write "10.1 or more", a value of exactly 10.1 is also counted as out of spec.

No deliverable.

Check your result

You are done when 3 out-of-spec Lot No values are listed on the screen. Also check that the Result column of the original data has 6 OK and 3 NG, and that the 3 lots the AI found and the 3 NG rows in the Result column are the same Lot No values. 6 OK, 3 NG and the 1 lot that counts as neither make 10 lots in total.

Extract them yourself before you read this. The out-of-spec lots are the 3 lots L-1003, L-1006 and L-1009. All of them are above the upper limit of 10.1. One row has an empty measured value and a full-width hyphen in the Result column, and this 1 row counts as neither. If the total does not come to 10, you have missed counting something somewhere.

If it does not work

Symptom	Fix
4 lots come out, and 1 of them has an empty measured value	The lot with an empty value is counted as out of spec. Tell it again that a lot with no measured value cannot be judged
No lots come out, or the count changes every time	The values are being compared with full-width digits still in place. Start over from the work in Step 2 that converts them to half-width
More than 3 lots come out	The upper limit was written as "10.1 or more". Rewrite it as "above the upper limit of 10.1"
The result does not match the Result column	A row has a full-width hyphen in the Result column. Treat that row the same as an empty value

If time allows

- Also check that no lot is below the lower limit of 9.9. It is easy to fall into the habit of looking only at the upper limit here
- Narrow the spec to 9.95 to 10.05, see how many lots come out, and confirm that the conclusion changes with the threshold

- Run the same steps as in this exercise once more with `equipment-data_2026Q2.xlsx` , which has more machines. You can use the prompts from Q1 as they are

Exercise 6: Build a maintenance record summary tool

The working time is [35min]. The breakdown is: ask roughly and watch it fail [3min], write out the rules for unifying names and units [5min], build the tool [22min], and verify with the real data [5min].

Scenario

Maintenance records are not just for this month. Next month and the month after, you will get files with the same inconsistent spellings. Pasting each one into the chat and asking for a summary again every time is inefficient in real work. Today, instead of a one-time summary, you build **a tool that unifies names, unifies units and summarizes automatically when you drag a file onto it**. You use the "3 rounds" approach you went through in Exercise 2 as it is.

In the table, the same 90 minutes is written in 3 ways: "90分", "1.5h" and "1h 30min". If you add them without unifying the units, "1h 30min", which is not even a number, silently drops out of the total, and you get "91.5", a meaningless number made by adding "90" and "1.5". The Assignee names also come in 3 forms for the same person: "Yamada", "Taro Yamada" and "ヤマダ". You build rules for these 2 kinds of dirty data into the tool, instead of giving the chat instructions every time.

Steps

1. Drag `maintenance-records_2026H1.csv` onto `01_data-loader.html` and copy it with "Copy this table". Paste it into Copilot Chat, ask only "Summarize this" in a rough way, and confirm that the assignees split into 8 or 10 people [3min]
2. Turn the 2 points that broke with the rough request, how Duration is written (3 forms: 90分, 1.5h and 1h 30min) and the inconsistent Assignee names (3 forms: surname only, full name and katakana), into conversion rules in 1 to 2 lines, and write them in `prompt.txt` [5min]
3. Round 1: Send the request "Build an HTML tool that lists the contents of the table on the screen when I drag a maintenance records CSV file onto it", and check that it works [7min]
4. Round 2: Give it the rules you wrote in Step 2 and send the request "Change the tool so that it converts every Duration value to minutes, unifies the Assignee names, and then shows the total time per assignee and the number of records per Category" [10min]
5. Round 3: Send the request "Also show the number of valid records, excluding duplicate rows and blank rows, near the top of the screen", and tidy it into an easy-to-read form [5min]

In Step 1, do not keep working with the rough request. If you see the failure first, it becomes clear in Step 2 what you should turn into rules.

The deliverable is `maint.html` in the `tpro-work` folder. Tell Copilot the file name and have it save the file.

Check your result

Open `maint.html` in the browser and drag `maintenance-records_2026H1.csv` onto it. You are done when all of the following 4 items are true.

- The valid record count is shown as 96
- 4 assignees are shown (not 8 or 10)
- The total time is 10938 minutes (182.3 hours)
- The Category breakdown is Cleaning 24, Adjustment 21, Repair 20, Parts replacement 18 and Periodic inspection 13

Run the tool yourself before you read this. There are 4 assignees. If the 3 spellings "Yamada", "Taro Yamada" and "ヤマダ" are counted as the same person, the count does not become 8 or 10. The valid record count is 96 and the total is 10938 minutes. If you build it as a tool, then when next month's file arrives, you only drag it onto the same tool and get the same summary.

If it does not work

Symptom	Fix
The assignee count is 8 or 10	The name unification rules are not reflected in the tool. Give Copilot the rules you wrote in Step 2 again as they are, and tell it "It is still not fixed"
The total is not 10938 minutes	Duplicate rows or blank rows remain. Check that the valid record count is 96, and if it does not match, compare it with the number of rows in the original data
The total comes out as an odd decimal	The "1h 30min" form is not being converted and drops out. Rewrite the Duration conversion rule again, more specifically
The tool stays blank and shows nothing	Check the error in the browser's Developer Tools (F12), paste the error message into Copilot as it is, and have it fix the problem

If time allows

- Change `maint.html` so that it shows the total time per assignee as a bar chart. The idea is the same as the chart library you use in Exercise 7, the next exercise
- Check whether the tool keeps working without breaking when you drag a different table onto it, such as the Inspections sheet of `equipment-data_2026Q2.xlsx` (whether it is built so that you can reuse it as it is)

Exercise 7: Summarize and make a bar chart

The working time is [24min]. The breakdown is: copy the library [3min], request to fill the blanks [5min], request to summarize [5min], build and save the HTML [6min], and check in the browser [5min].

Scenario

If you report the defect count by process out loud, it does not tell people which process has the most. Today you take it to a form where showing 1 page in a meeting is enough. What you build is 1 HTML page that opens in the browser.

Steps

1. Open the `lib` folder in the handout folder and copy `chart.umd.js` into the tpro-work folder. Right-click > Copy, then right-click in tpro-work > Paste [3min]
2. In `01_data-loader.html`, show the Defects sheet and copy it with "Copy this table", then send the request "For rows where the Process column is empty, fill in the process name from the row above" [5min]
3. Then send the request "Count the number of defects for each process" [5min]
4. Paste the 5-line prompt below and have it build the HTML. Copy the code that comes back and save it in the tpro-work folder with the name `chart.html` [6min]
5. Double-click `chart.html` to open it in the browser, and check that the bar heights and the numbers above the bars match the summary result [5min]

If you skip Step 2 and send only the request for Step 3, the counts do not match. Filling the merged cells comes first.

For the chart prompt, send these 5 lines right after the summary result you pasted.

```
Build 1 HTML file that shows this summary result as a bar chart.  
Load chart.umd.js from the same folder with <script src="chart.umd.js"></script>.  
Do not load anything from a URL on the internet.  
The horizontal axis is the process name and the vertical axis is the defect count. Show the count above each bar.  
Return only the contents of the HTML. No explanation is needed.
```

If you do not write line 3, it tries to load the library from a URL on the internet, and the chart does not display in your company's environment. Add this prompt to `prompt.txt` as well. When you make another chart on Day 3, you can reuse it by rewriting only the horizontal axis and vertical axis lines.

Watch where you save. The VS Code save dialog remembers the previous folder, so the file may be saved somewhere other than tpro-work. In the Explorer, check that you can see `chart.html` inside tpro-work.

3 ways to check the numbers

Check every number the AI gives you against something at least once. Today you used all 3 of these ways.

- Check against the count. Are there 8 machines? Are there 10 lots? See whether it matches the number of rows in the original data
- Check against the total. Add up the breakdown by process and check that it matches the total
- Check against the original column. See whether the 3 NG lots the AI gave and the 3 NG rows in the Result column of the original data are the same Lot No values

Do not put a number into a report if none of these ways could check it. Stop there. You use this same order in the Day 3 report writing as it is.

Check your result

- `chart.html` is in the `tpro-work` folder, and when you double-click it, 3 bars appear in the browser
- The bar heights are in the order Molding > Cutting > Inspection
- The numbers above the bars match the summary result

Tally them yourself before you read this. The defect counts by process are Molding 33, Cutting 21 and Inspection 7. They add up to 61. If the total is not 61, you missed filling the merged cells.

If it does not work

Symptom	Fix
The browser shows a blank page	<code>chart.umd.js</code> is not in the <code>tpro-work</code> folder, or the HTML is trying to load a URL that starts with <code>http</code>
Only 1 bar appears	The empty Process cells were not filled, so only Molding was counted. Start over from the request to fill the blanks
The total count does not match	Some rows were counted while still empty. Show it the table again after the blanks were filled, and have it recount
VS Code opens when you double-click	Right-click, choose a program from the menu, and open it in the browser
The text is garbled	Set the encoding in the Status Bar to UTF-8 and save again

If the chart still does not appear after [18min], give up on Step 5 and only note down the summary result from Step 3. You can still cross-check with that.

If time allows

- Make a scatter plot of Operating hours and Defects count, and check how the correlation of -0.965 looks. The 1 machine with empty Operating hours cannot be drawn, so there are only 7 points
- Add a one-line comment at the bottom of `chart.html` saying which you could use in your own meeting: the comparison by equipment or the breakdown by process.

Exercise 8: Write your own work theme

The working time is [12min]. The breakdown is: create the file [1min], fill in 5 lines [7min], narrow down the part to hand to AI [2min], reread [2min].

Scenario

On the afternoon of Day 3, you have 130 minutes to build a mini app that improves your work. If you start looking for a topic then, no time is left to build. Decide today on one task that is a hassle in your own workplace.

Nothing is collected and nobody presents. What you write stays with you.

Steps

- 1. In the VS Code Explorer, right-click > New File > type `theme.txt` to create it [1min]
- 2. Copy the 5 headings below as they are, and fill in the right side in your own words. If you get stuck, start with the most troublesome task you did this week [7min]
- 3. Narrow what you want AI to handle to one part. Pick a part of the task, such as only the copying or only the totals, not the whole task [2min]
- 4. Reread what you wrote, and find and delete any client names, product names or exact numbers [2min]

Do not skip the reread in Step 4. Deleting things yourself after you write is itself rule 4 of the confidentiality rules.

Here is a sample of how to write it.

```
[Industry and process] Inspection process in the electrical industry
[Task name] Create reports from a list of test results
[Current way] Copy from the list into the report template by hand
[How much effort] Tens of minutes per report, dozens of reports a month
[Part for AI] Copying and formatting
```

If you write the same problem as "I re-paste CSV from our internal system by hand into test reports for Company B's controllers. 40 min each.", it names the client, the product and the internal system path. In the 5 lines above, the problem and the hope remain, and client names, product names and exact numbers are gone. Describe effort only as "tens of minutes" or "dozens".

Check your result

- Industry and process, Task name, Current way, How much effort and Part for AI each have one sentence
- Effort is described roughly, like "tens of minutes" or "dozens a month", with no exact numbers
- The part for AI is narrowed to one. It points to a piece of the task, not the whole task
- It is saved as `theme.txt` in the tpro-work folder

If it does not work

Symptom	Fix
The theme is too big to write	Narrow it to one task you did this week, not the whole work of your department.
You cannot think of a task to write about	Start by thinking about what in your own work matches the totals and charts you made today.
Without the proper names, you cannot tell what the topic is	Fill in the industry and process lines first, and the rest gets easier to write.
You are unsure whether you may write it	If you are unsure, do not write it. Leaving it blank is fine.

If time allows

- Try writing that task as 3 steps, keeping your current procedure as it is. You will use this on Day 3 when you decide which part to hand to AI.
- If someone in a different process does the same task, add one line on what differs.

Troubleshooting

This table collects problems that can happen across exercises. If none of the symptoms applies to you, you can skip this table.

Symptom	What to check	Fix
No Copilot icon	The row of icons at the top right of the screen	Reopen VS Code. If it is older than 1.116, raise your hand.
You are asked to sign in	The account at the bottom left of the screen	Sign in again with your GitHub account and approve in the browser that opens.
No reply comes back	Notifications at the bottom right of the screen	A network restriction can be the cause. If it does not recover, ask the instructor.
You cannot find the tpro-work folder	File > Open Recent	Create the tpro-work folder again and open it.
The whole display is in English	Display language	Keep the display language in English.
The screen is blank white when you open it in the browser	Console tab (F12)	Paste the lines in red text into Copilot as they are. If there is no red text, you forgot to save.
Opening the file shows the code as text	End of the file name	Turn on file name extensions and check that .txt has not been added.
The screen does not change after you fix the file	Saving and the browser	Save with Ctrl+S, then press F5 in the browser.
Japanese text is garbled	Encoding shown in the Status Bar	Change the encoding to UTF-8 and save again.
The Excel screen does not open in the browser	The app that opened the file	Right-click > Open with > choose a browser. It does not work in VS Code.
The handout HTML does not work	Where the file is placed	Check that you have not moved 01_data-loader.html to a place away from the lib folder.

The source for the VS Code update details is https://code.visualstudio.com/updates/v1_116. The contact point for outside course hours will be announced on the day.

What you take home today

Check that the tpro-work folder contains these 5 files.

File	Contents	Where it is used next
index.html	The page that runs in the browser, built in the morning vibe coding exercise	The way of working itself can be used when you build the mini app on Day 3
prompt.txt	Template prompts rewritten with the 4 elements. Prompts for formatting and charts are included too.	In the Day 3 data analysis, you can start by pasting it
clean.csv	The Operations table you cleaned up in the data preparation exercise	You open this file in the Day 3 analysis exercise
chart.html	The bar chart page. It works together with chart.umd.js in the same folder.	Change the lines for the horizontal axis and vertical axis to get a different chart
theme.txt	Your own work theme	It becomes the topic of the Day 3 mini app for improving your work

What you picked up is a way of working: tell it what you want, run it, and restate each gap one at a time. You do not fix the design first and then build. You put a screen up in 5 minutes and decide as you go. This cycle was the same in the morning game and in the afternoon Excel cleanup.

The other is the procedure for a person to check the numbers AI produces. Check by count, check by total, check against the source column. Do not put a number you could not check in the report. If you skip this, a number such as a mean of 160.7 hours, calculated with the missing value filled in as 0, goes into the report as it is.

On Day 3 you continue in the same environment and the same tpro-work folder. You analyze the shared data and turn it into a report, and you build a mini app for improving your work, chosen from the samples. `theme.txt` becomes its topic, so keep it even if it is unfinished.

Rules to follow

Do not use real data from your own work in the exercises. All the data we used today is fictional and has nothing to do with real equipment, products or companies.

Before you paste, generalize the text with 4 rules.

Rule	How to apply it
Generalize client names, product names, project names and partner names to the industry or process level	Write something like "the design process in the automotive industry" instead of a specific company name or equipment model number.

Rule	How to apply it
Do not use real data, real files or screenshots	Do not paste screens of your own Excel files or drawings. Replace the values with fictional ones, like today's exercise data.
Avoid specific values for figures and specifications, and write at the level of purpose, issues and expected effects	Write "judging dimensions takes time" instead of the actual tolerance values.
Before you submit, check yourself that the content is safe to send outside the company	Read it again when you finish. Leave out any part you are unsure about.

Write "a mass-production line for auto parts" instead of "in-vehicle connectors for Company A", and write "Equipment A" and "a value slightly over the upper limit" instead of "10.05mm on EQ-1420". Even if you generalize down to the process and industry, the formatting steps you get back do not change. If the digits and units match, that is enough for practice.

Before you press the Send button, read the text you are about to paste once more, asking whether it is OK to share outside the company. It takes 30 seconds. If you use this in the environment of the client site where you are stationed, that site's rules come first. If you are unsure about something, do not paste it. Take it back with you.

This material is a copyrighted work created by Givery, Inc. Participants may use it only for their own learning and for reference in their own work. Do not distribute it outside the company, post it on social media, or share it with third parties.



Givery, Inc. Do not redistribute this material.