

TRAINING

# Hands-on AI-Assisted Development

## Mechanical and Electrical track Day 1

---

Environment setup, generative AI basics, literacy, using GitHub Copilot



## Do not redistribute these materials

---

**Redistribution of these materials is prohibited.**

These materials are copyrighted by Givery, Inc. Use them only for your own learning and work reference. Do not share them outside your company, on social media or with third parties.

During the course, do not take screenshots or recordings that show these materials.

## Today's goals

---

By the end of today, **Copilot runs on your own PC**

### On your own PC Copilot runs

Everyone signs in to VS Code, talks to Copilot in English, and checks that a reply comes back

# How to judge the goal

---

## Use these 4 points to judge for yourself

### **The environment works**

Your own GitHub username appears at the bottom left of VS Code, and the chat answers in English

### **What generative AI is**

You can answer the last quiz of the morning without hesitation

### **You know the line**

In the afternoon quiz, you avoid prompts with client-specific information

### **Hands-on experience**

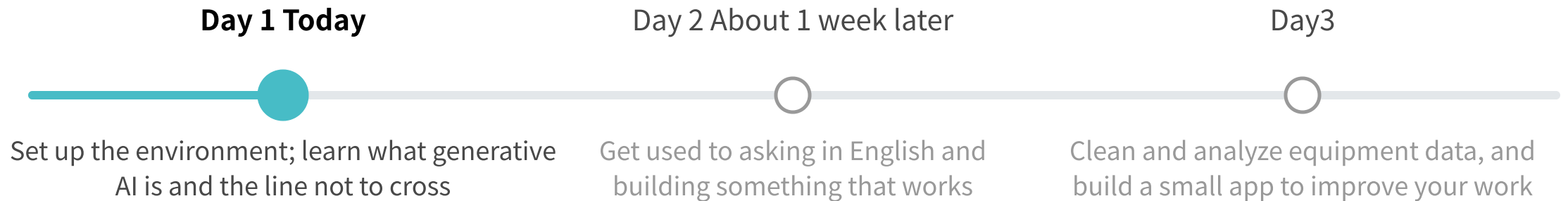
The screen you built in the afternoon exercise runs in your own browser

You do not need all 4 to be perfect. The first one is the main hurdle today.

## Where today fits in the 3 days

---

**Today is Day 1 of 3. You can catch up next week**



# Today's schedule

## Setup in the morning; concepts and practice in the afternoon

| Type             | Description                                                    | Part     |
|------------------|----------------------------------------------------------------|----------|
| <b>Morning</b>   | Environment setup + GitHub account + Copilot invite and check  | [110min] |
| <b>Morning</b>   | Break                                                          | [10min]  |
| <b>Morning</b>   | Generative AI basics                                           | [60min]  |
| <b>Afternoon</b> | Risk, literacy, governance + handling confidential information | [70min]  |
| <b>Afternoon</b> | Break                                                          | [10min]  |

Total 480min. The [10min] above is a short break; lunch is separate.

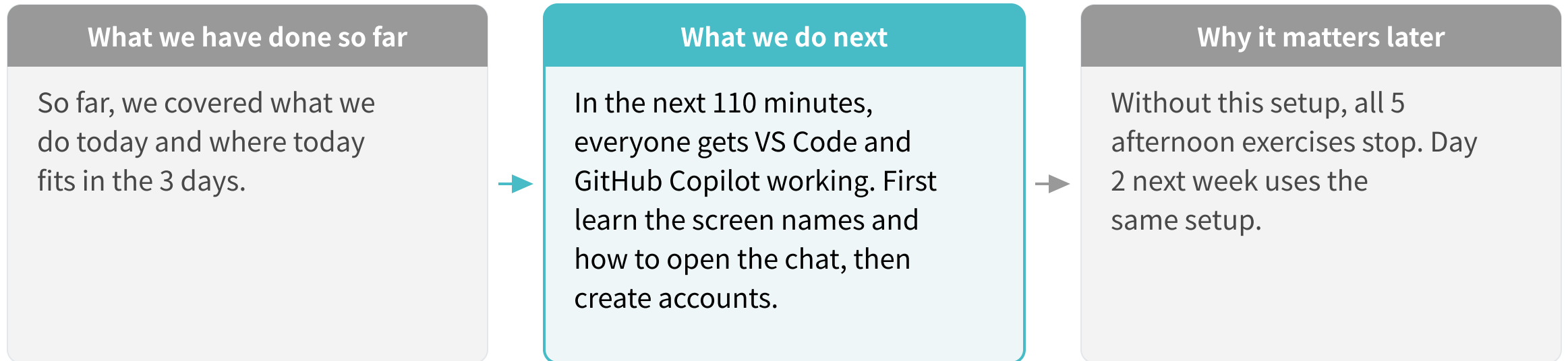
## Today's schedule (continued)

---

| Type      | Description                                       | Part     |
|-----------|---------------------------------------------------|----------|
| Afternoon | How to use VS Code                                | [40min]  |
| Afternoon | GitHub Copilot basics: 5 short hands-on exercises | [100min] |
| Afternoon | Comprehension test                                | [40min]  |
| Afternoon | Review + next week's topic                        | [40min]  |

## From introduction to setup

### This takes up most of this morning





# 01

## Environment setup

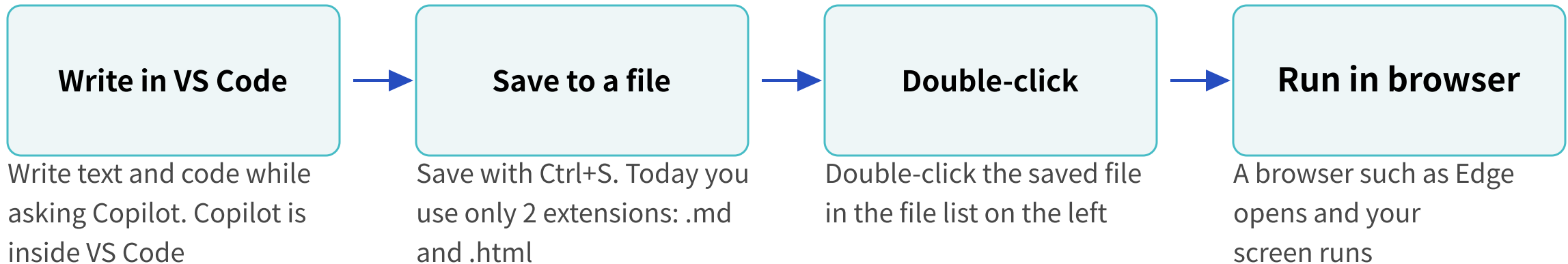
---

Get VS Code and GitHub Copilot working on your own PC

## The setup

---

**Just write and save in VS Code, then open in the browser**

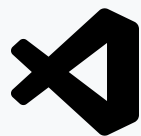


No Python or Node.js. Nothing extra to install today.

## 3 tools you use today

---

Today you use **only these 3**. Nothing extra to install



### VSCode

Where you write text and code. Copilot is inside it



### GitHub Copilot

Talk to it in English. Its Chat view opens on the right of VS Code

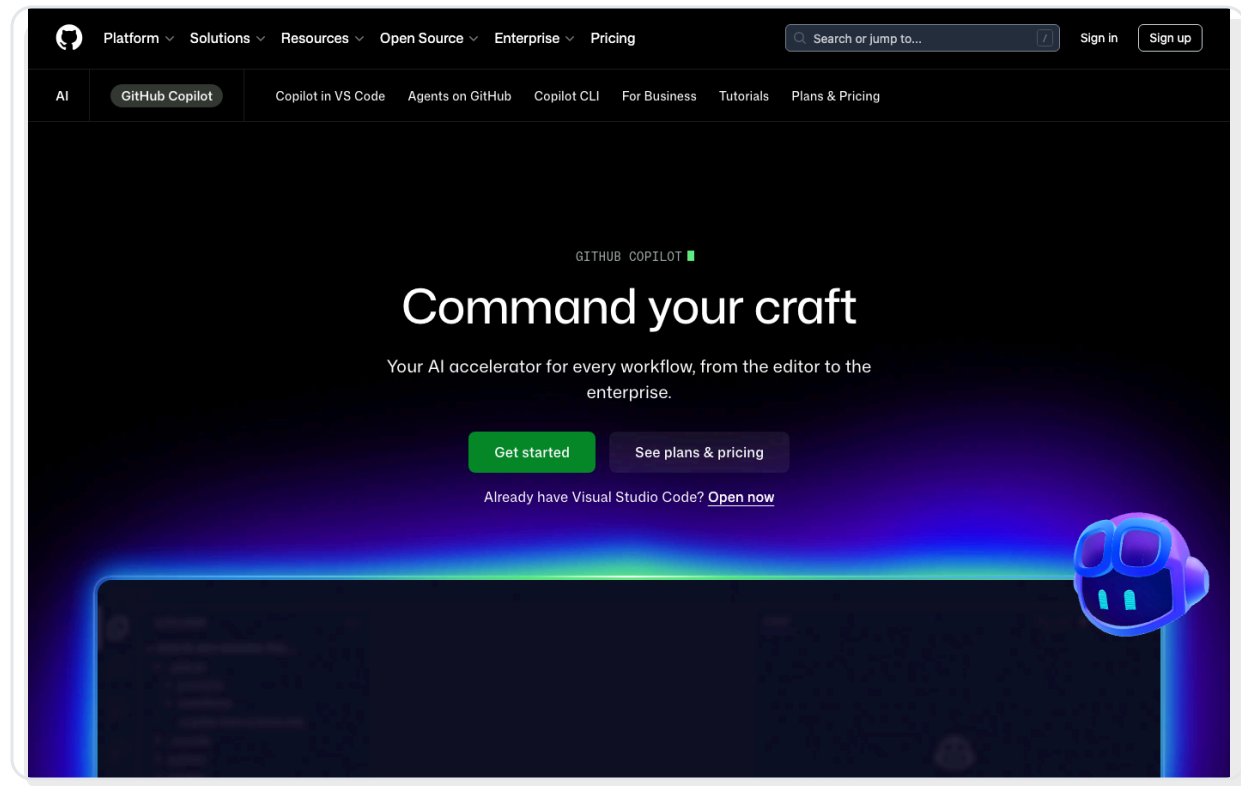


### Edge

The browser that opens and runs your files. It is already on the loaner PC

# What GitHub Copilot is

## Copilot is an assistant from GitHub, built into VS Code



GitHub Copilot official site

## How GitHub and Copilot relate

---

**You create the account on GitHub and use Copilot**



### GitHub

A service widely used in software development. You create your account here

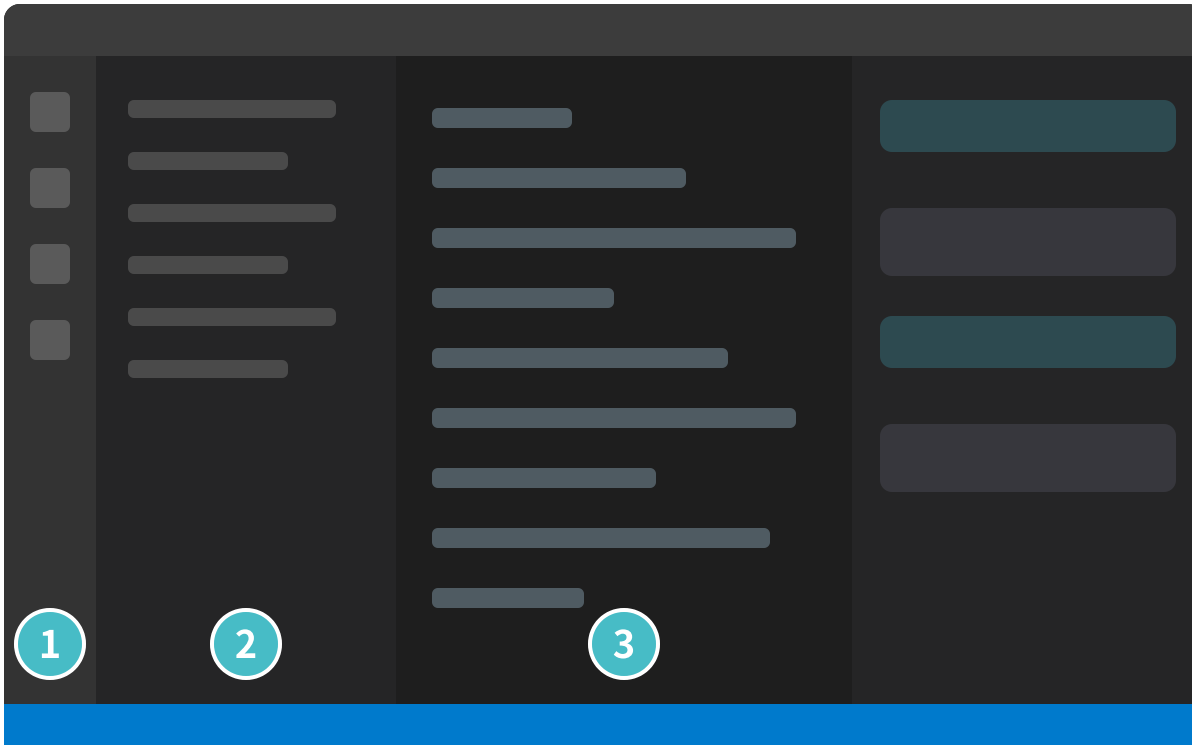


### GitHub Copilot

The assistant you use with that account. You open it inside VS Code

# The VS Code screen

## Today you use only 3 areas of this screen



### 1 Activity Bar

The column on the far left. Click the stacked-documents icon to show the file list

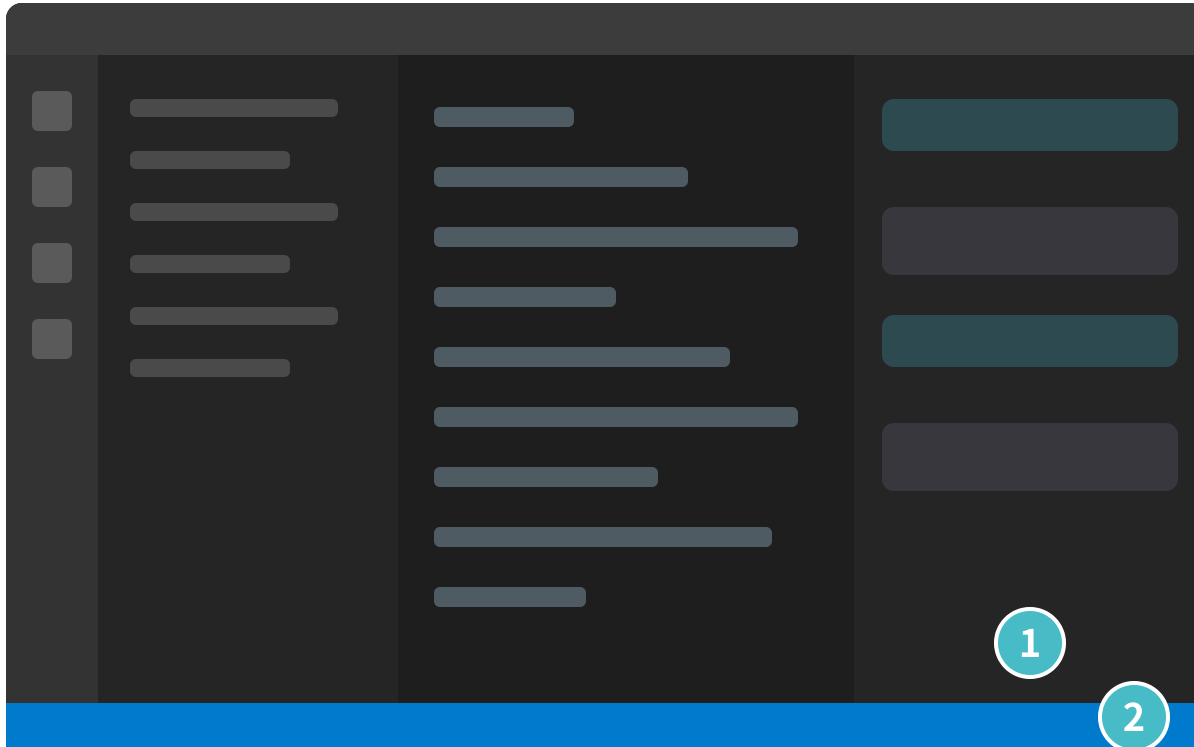
### 2 Explorer

The list of files in the open folder. Click a file to open it on the right

### 3 Editor

Where you write text and code. You can open several files in tabs, and close them with ✕

## The VS Code screen (continued)



### 1 Chat view

Where you talk to Copilot in English. It opens on the right side of the screen

### 2 Status Bar

The thin strip at the bottom edge. The account you are signed in with appears at the bottom left

## Open the handout folder and you see this screen



### 1 Explorer

Lists the contents of tpro-work. Click to open

### 2 Editor

The content of the open file. Write text here

### 3 Chat view

Where you ask Copilot. Show or hide it with the icon at the top right

### 4 Input fields

Type your request here and press Enter. You use this a lot this afternoon

## How to open the Chat view

---

### Once open, the Chat view stays on the right until you close it

- 1 Click the Copilot icon at the top right**  
It is the speech-bubble icon near the right end of the top edge
- 2 A panel opens on the right**  
A tall panel appears to the right of the editor. To close it, click the same icon again
- 3 Type in the input box at the bottom and press Enter**  
Type in plain English. A reply comes back in a few seconds

If you cannot find the icon, look for Copilot Chat in the View menu.

### Do the 4 steps in order. Do not skip any

#### Scenario

The 5 exercises this afternoon and Day 2 next week both use this setup. If you start the afternoon stuck here, not one exercise will run.

#### Steps

- 1 Start VS Code from the Start menu. If it is already open, leave it as it is
- 2 Open github.com in the browser and create a new account with your company email address. Write down the username you choose
- 3 Open the Copilot invitation email we sent in advance and accept it from the link
- 4 Go back to VS Code and sign in to GitHub from the account icon at the bottom left

#### Result

No deliverable. The result is your note of the GitHub username, and VS Code showing that username at the bottom left.

## Completion criteria for setup

### When all 4 are done, raise your hand

#### Done when

- The GitHub username you created is shown at the bottom left of VS Code
- The Copilot icon at the top right has no slash through it
- You have written down your GitHub username (you need it on Day 2 too)
- You extracted the handout ZIP to the desktop, and the folder is named tpro-work

#### If you're stuck

- No confirmation email. Check your spam folder, then contact the organizers
- The sign-in browser stays open and you do not return. Close and reopen VS Code; you may already be signed in
- The icon still has a slash: the invitation is not accepted yet. Open the email link again
- Cannot find VS Code at all. Type vs in the Start menu and it appears in the results

#### If time allows

- If you are done, extract the handout ZIP here too
- Then help your neighbor find which of Steps 1 to 4 they are stuck on

## Account creation cases

### 3 special cases. If one applies, raise your hand first

| What happens                                      | What to do                                                                                                                        |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>You already have a personal GitHub account</b> | The invitation goes to your company email. Creating a new account with your company address is the safest                         |
| <b>Asked to set up two-factor auth</b>            | Choose an authenticator app or receiving codes by SMS. If you cannot use an authenticator app on your company PC, raise your hand |
| <b>No confirmation email</b>                      | Check your spam folder; if it is not there, contact the organizers. If it does not come in 5 minutes, we move to the next step    |

Use your company email address. With a personal address, the invitation does not reach you.

## Sign-in checkpoints

---

### Check just these 2 places to see if you are signed in

#### Bottom left

Your GitHub username appears next to the account icon

#### Top right

The Copilot icon has no slash through it. A slash means you are not signed in or have not accepted the invitation

If both apply, raise your hand. It is faster than checking each screen one by one.

## Your first message

---

### Type either of these 2 lines into the chat as is

Plain English is fine. Do not worry here about whether the reply is correct.

```
Hello. What can you do? Answer in English, in 3 lines.  
List 3 things you can help me with when I write an equipment inspection report.
```

Press Enter when done. To start a new line mid-sentence, press Shift+Enter.

# Just check that a reply comes back

### Scenario

If you start the afternoon with a broken setup, all 5 exercises stop. Check the connection now, and you can focus on the content this afternoon.

### Steps

- 1 Click the Copilot icon at the top right to open the Chat view on the right
- 2 Type a line from the previous page as is and press Enter
- 3 Wait a few seconds and check that English text appears in the panel

### Result

No deliverable. You are done when the chat shows a reply in English. No need to save.

# Completion criteria for the connection check

## Done when a full reply in English appears

### Done when

- The chat returned text in English
- The reply did not stop halfway and was shown to the end

### If you're stuck

- The panel opens but no reply comes. The company network may be blocking it. Raise your hand and we will note it down
- The reply is in Japanese. Add "Answer in English." at the end and send it again
- An error appears: you were signed out. Sign in again from the account icon at bottom left
- The panel does not open when you click the icon. Close and reopen VS Code

### If time allows

- If you got a reply, next ask: "Make it shorter, in 3 lines." Seeing the conversation continue now makes the afternoon exercises faster
- Open 01\_sample\_inspection-record.md in the folder you extracted from the handout ZIP and skim the inspection items and the field for current problems

## If it still does not work

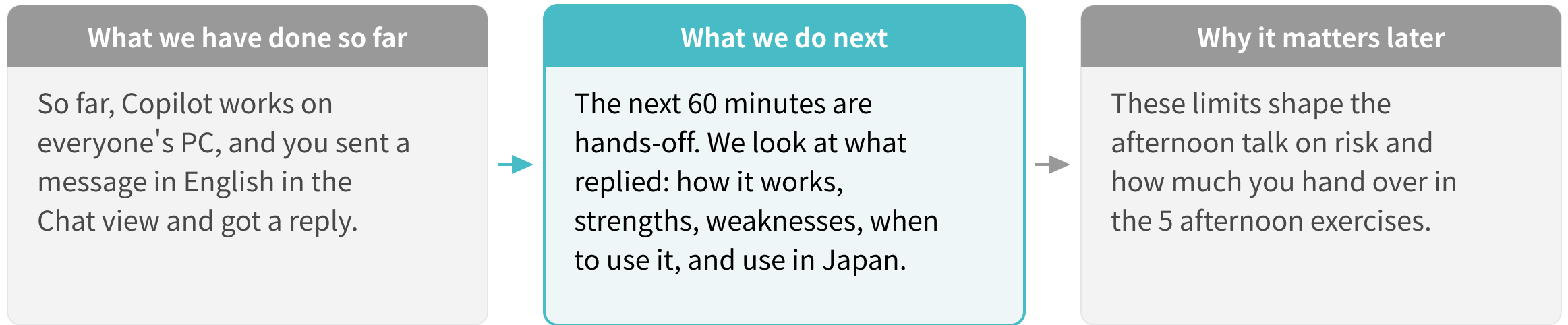
# Not fixed today? We help you one-on-one before next week

| Symptom                   | What to check                                                                     |
|---------------------------|-----------------------------------------------------------------------------------|
| <b>Chat does not open</b> | Close and reopen VS Code. If it still does not appear, raise your hand            |
| <b>Cannot sign in</b>     | Check that you accepted the invitation from the email link. Also check spam       |
| <b>No reply</b>           | The company network may be blocking it. We cannot fix it here, so we note it down |
| <b>Screen in English</b>  | It does not affect your work. Keep the VS Code display language in English        |

You can still follow the rest of the morning and the afternoon lectures. Do not rush.

## From setup to how it works

# Time to stop typing. This guides your next hands-on decisions



# 02

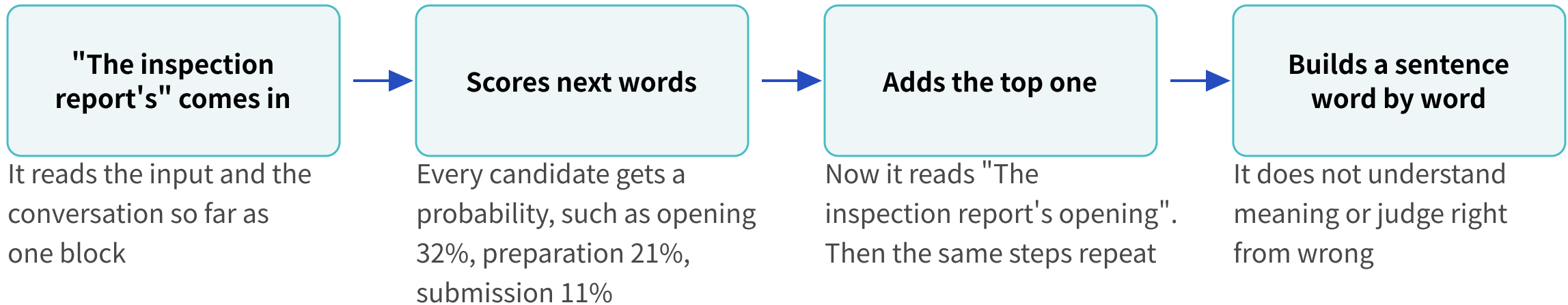
## **Generative AI basics**

---

Take home what it does well and poorly, and a clear limit on what to hand over

## How AI works

### Generative AI keeps **picking the next word by probability**



The probabilities are examples for explanation. Real values depend on the model and context.

## How it differs from search

---

### Search finds text; generative AI creates it

#### Search engine

It finds pages matching your keywords and returns them as is. The source URL is always there. The same words give much the same results.

#### Generative AI

It writes new text from learned patterns. The same question gets a different reply each time. No source is kept, so it cannot show an original page.

## Strengths and weaknesses

### Hand over shaping text, not deciding what is correct

| Type of work                               | AI use          | Reason                                                      |
|--------------------------------------------|-----------------|-------------------------------------------------------------|
| <b>Report drafts, summaries</b>            | OK to hand over | You can reread the output and fix it yourself               |
| <b>Explaining terms, advice on wording</b> | OK to hand over | If you verify it yourself, it cuts research time            |
| <b>Unifying the format</b>                 | OK to hand over | Mistakes are visible and you can fix them right away        |
| <b>Measurement math, record counts</b>     | Don't use as is | Numbers are not guaranteed; wrong digits sound just as sure |
| <b>Standards compliance</b>                | Don't use as is | It may make up the standard number itself                   |
| <b>Safety decisions</b>                    | Don't use       | Errors lead to accidents and cannot be undone later         |

## A yardstick for deciding

### When unsure, decide by whether a mistake can be undone

#### Easy to fix

Report drafts. Reread and fix them, and no one is affected

#### Fixable after checking

Researching terms and procedures. OK if you verify with primary sources

#### Drafts only from here

Client documents. You are responsible, so write the final version yourself

#### Fixing involves others

Measurement calculations. A number that spreads skews later decisions

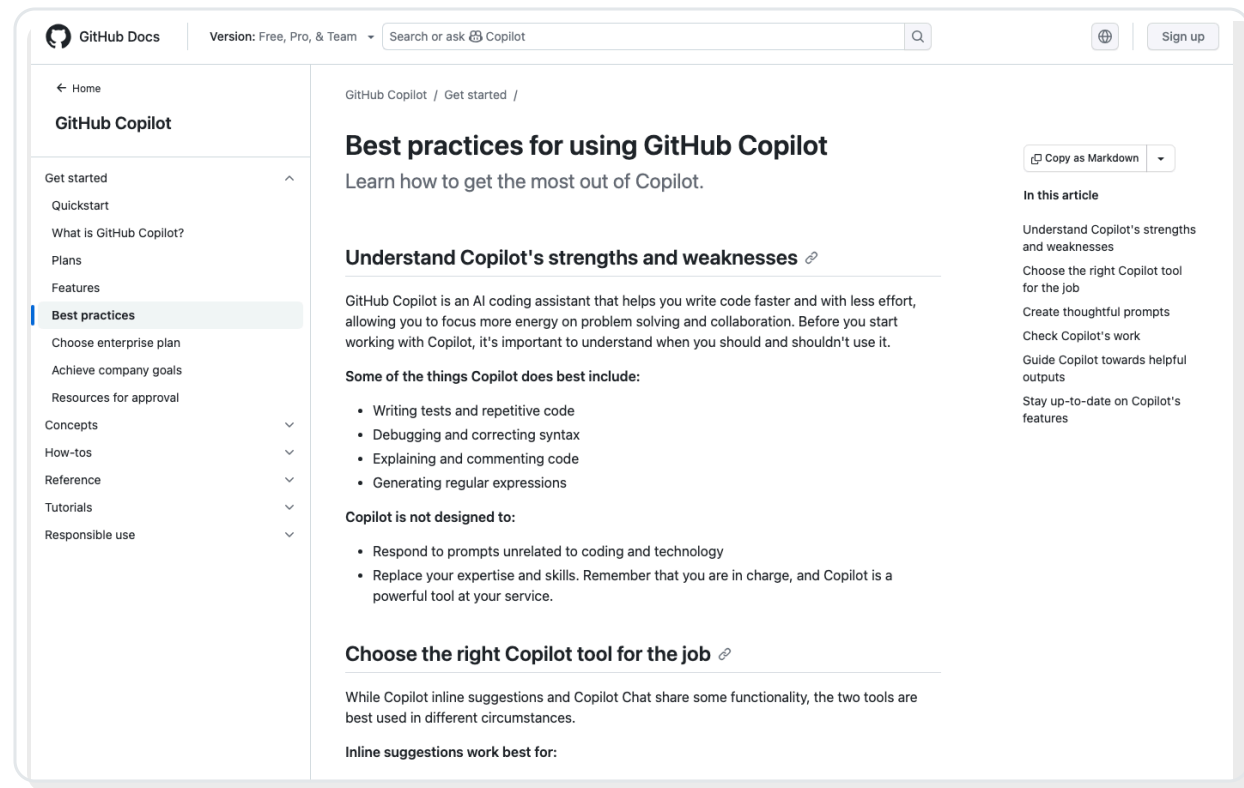
#### Cannot be undone

Safety decisions. Do not use AI here

This one yardstick is all you need to remember. It also works for cases not in the table.

# What the provider says

## The provider also states that checking facts is the user's job



Official GitHub Copilot documentation on usage precautions

## How it is used in Japan

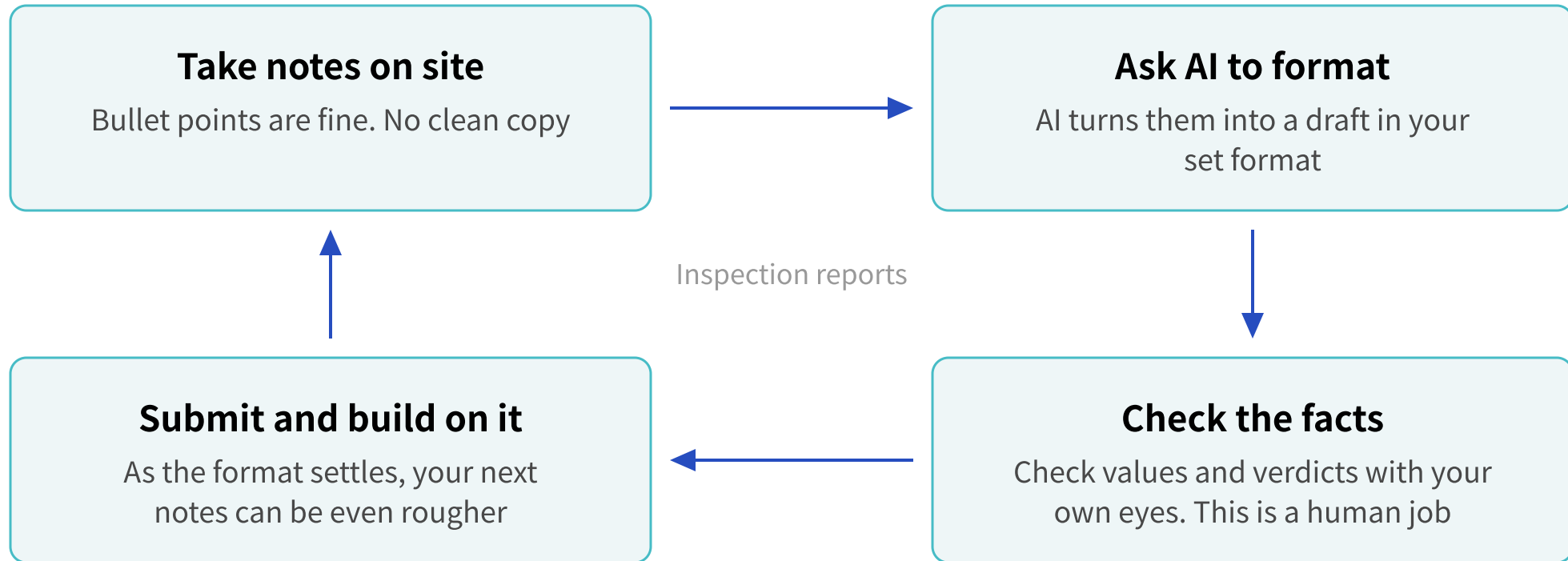
### Manufacturing and maintenance begin with drafts and searches

| Process                 | What AI does                                       | What people keep                   |
|-------------------------|----------------------------------------------------|------------------------------------|
| Daily inspection        | Drafting reports from field notes                  | Check values and verdicts; reread  |
| Using equipment manuals | Finding likely sections in thick procedure manuals | Final check in the original manual |
| Sorting fault reports   | Sorting scattered messages into fixed fields       | Isolating causes, deciding actions |

Described at the level of industry and process. No company names or reduction rates are listed.

# Inspection reporting workflow

## Turning field notes into drafts is where AI helps most



Remove client names and model numbers from notes before pasting. The afternoon covers this in detail.

## How searching changes

---

### Searching gets faster, but you still check the original

#### Before

You search thick equipment manuals by guessing from the table of contents. Text worded differently from the contents is not found. Sometimes you go around asking experts.

#### Asking AI

Ask in English: "Which chapter should I read for this symptom?" Finding the spot takes less time, and experts are not interrupted.

**Take 30 seconds to apply this to your own work**

**Of the 3 examples you just saw,  
which is closest to your own work?**

Think for just 30 seconds. We do not collect answers. In the afternoon exercises, keep that scene in mind as you work.

**Q. Which statement about generative AI is correct?**

- A** It searches the internet and returns the text it finds as is
- B** It understands the meaning of the question and calculates the correct answer
- C** It builds text by repeatedly choosing the word likely to come next, based on probability
- D** It finds the closest match in a prepared set of answers and returns it

The answer is on the next page. Think it through yourself first.

### The answer is **C**: it predicts by probability

**C**

**It builds text by repeatedly choosing the word likely to come next, based on probability**

It gives probabilities to possible next words and adds the most likely one, word by word. It does not understand meaning or judge right from wrong.

**A**

It searches the internet and returns the text it finds as is

It works differently from search. It writes text on the spot instead of finding existing text, so no source URL remains.

**B**

It understands the meaning of the question and calculates the correct answer

It does not calculate the right answer. It only picks a plausible continuation, so mistakes come in the same tone.

**D**

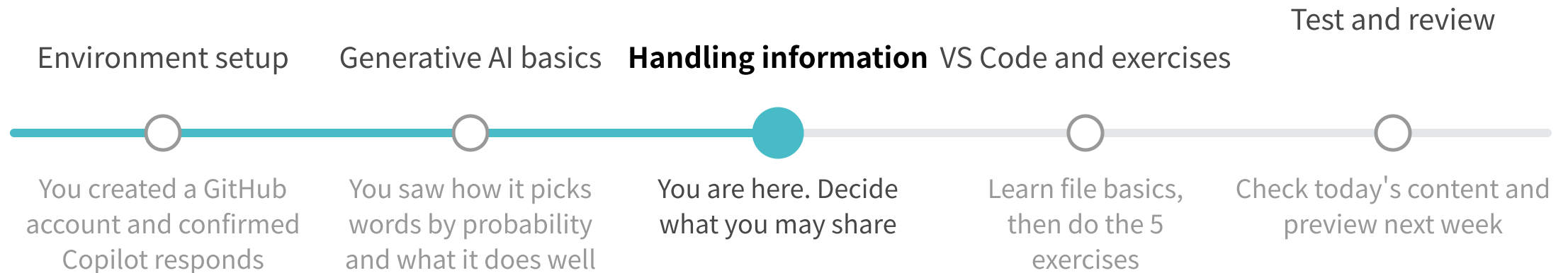
It finds the closest match in a prepared set of answers and returns it

It has no answer set. Replies differ each time because it writes them anew on the spot

Even wrong answers come back in a confident tone. Checking the content remains a human job.

## Where we are today

First this afternoon: **set your limits** before hands-on work



The afternoon is mostly exercises. For this section only, stop typing and listen.

## From basics to decisions at work

# Knowing the weaknesses is not the same as avoiding accidents

### What we have done so far

The morning ended with one line: even wrong answers come back in a confident tone. That line is our starting point. The mechanics part is over; bring only the conclusion that it picks the next word by probability.



### What we do next

Next: cases that cause accidents even when you know the weakness. How to spot plausible mistakes, where your input goes, 4 ways to hide client information, and priorities when unsure.



### Why it matters later

The 5 exercises that follow stay inside these limits. Your Day 2 work theme is written within them too. Decide today, and next week you can start right away.

# 03

## **Risk and literacy**

---

As someone who handles client information, learn to set your own limits

## Plausible mistakes

### AI answers in **the same tone** whether it is right or wrong

#### Using the answer as is

It answers with standard and table numbers, like "The tightening torque for that use is specified in JIS B 0000-0. See Table 3." The number may not exist, and you cannot tell from the text. It never sounds unsure.

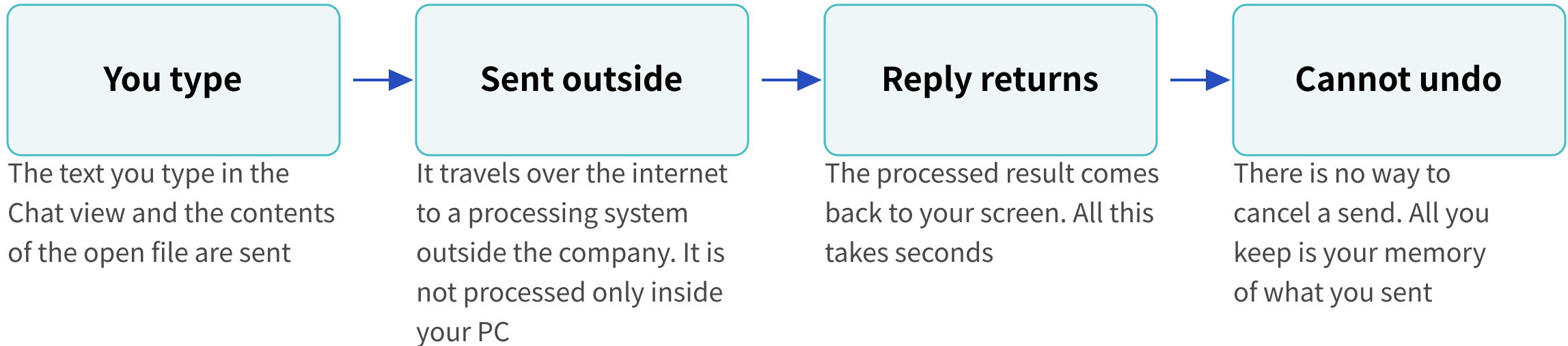
#### Re-check numbers and values before use

Use the text structure, but look up standard numbers again in primary sources such as the JISC search. If you cannot find it, do not use the answer. Check "Does this number exist?", not "Which standard covers it?"

## Where your input goes

---

**Whatever the contract says, your input is processed on servers outside the company**



## Check before sending

---

### Send only what passes these 4. If unsure, don't send

1

#### **Can the client be identified?**

Check for client, product, project and partner names. If any appear, generalize them to the industry and process level

2

#### **Are real materials mixed in?**

Do not share measurement data, drawings, internal form files or screenshots as they are. Describe the structure in words

3

#### **Are specific values left?**

Share no values such as dimensions, torque or run time. Use item names only, or unrelated dummy values

4

#### **Did you reread it yourself?**

Reread your text and judge if it is safe to share outside. If you cannot tell, do not send it

## Handling confidential information

---

When you write about work, follow **these 4 rules**

**Generalize**

Generalize names to industry and phase

**No real materials**

No real data, real files or screenshots

**Avoid specific values**

Write goals and issues, not exact figures

**Check it yourself**

Check yourself that it is safe to share

These 4 are operating rules agreed with your company. Follow them in your work from today on.

## Rewriting the same question

### You can drop specific details and keep the question intact

#### Question written as is

On the conveyor (model) for (client), the drive motor reaches 82 degrees. Check the attached measurement log and tell me the likely causes.

#### Question after generalizing

A conveyor for automotive parts has a drive motor that stays hotter than expected. List points to check from both design and maintenance.

## A ready-to-use question

### Just add the reader and length to the generalized version

This is the right side of the previous page, written so you can type it straight into the Chat view. Line 1 gives the subject, line 2 the symptom, line 3 the request, line 4 the reader and length, and line 5 states up front what you cannot share. This 5-line pattern works for any work question.

```
I would like to ask about a conveyor device for automotive parts.  
The drive motor temperature has stayed higher than expected.  
List the points to check from both the design side and the maintenance side.  
The reader is an engineer in my department. Use up to 6 bullet points, 40 characters each.  
I cannot share the client name or the model number, so answer in general terms.
```

With line 5, the AI asks back "Please tell me the model number" less often.

# Order of priority

---

## Client rules come first; the 4 rules apply beneath them

- 1 Client site rules**  
When you work at a client's workplace, its rules come first. Whether you may use generative AI, and on which devices, differs by client
- 2 Company rules**  
Next come your company's rules. Whether you must apply for use, and where, is set by your company
- 3 Today's 4 rules**  
Within the limits of the two above, use the 4 rules as practical writing rules. The 4 rules never override the rules above
- 4 When you cannot decide**  
Do not send. Ask your manager or your company's contact point; details are given on the day

Do not use it on client devices, and do not take out materials from clients. Treat these 2 as defaults until you check the rules.

### Decide the 3 tricky cases yourself, in advance

#### Scenario

Each time you write a question, you stop to ask "Can I share this?" Settling 3 tricky cases now saves you from worrying in every exercise and on Day 2. This is not about guessing the answer; it is about finding your own habits of judgment.

#### Steps

- 1 Write 1, 2 and 3 down the side of your notes or the margin of a handout.
- 2 Case 1: "Hide the client name and write 'an automotive parts maker'." Write ○ (OK to share), △ (conditional) or × (do not share).
- 3 Case 2: "Multiply all measured values by 10 and paste the shifted data." Judge it the same way.

## Cases where judgments split (continued)

---

### Steps

- 1 Case 3: "Blank out an internal inspection record form and paste it." Judge it the same way.
- 2 When all 3 are written, add one line on each reason. Writing which of the 4 rules applies is also fine.

### Result

There is nothing to submit. You are done when your notes show 3 judgments and reasons. We do not collect them or ask you to present.

## Completion criteria for the 3 cases

**1 is  $\triangle$ , 2 is  $\times$ , 3 is  $\times$ . If you can give reasons, you have it**

### Done when

- All 3 have  $\bigcirc$ ,  $\triangle$  or  $\times$  chosen by you, each with a one-line reason
- Case 1 is  $\triangle$ , with a reason like "not OK if the wording reveals the company"
- Case 2 is  $\times$ , with a reason like "relations and trends between values remain"
- Case 3 is  $\times$ , with a reason like "the form itself may be confidential"

### If you're stuck

- If you marked all 3  $\times$ : then you cannot ask anything. The answer is that 1 can be shared depending on how you generalize. Rethink how far you must generalize.
- If you marked Case 2  $\bigcirc$ : even at 10 times, ratios between values and trends stay the same. Changing digits is the same as giving the specific values.
- If you cannot state the  $\triangle$  condition in Case 1: a niche industry with few firms in Japan reveals the company. Use broad terms like automotive parts or industrial machinery.
- If you have no reason, write which of the 4 rules applies. That counts as done.

## Completion criteria for the 3 cases (continued)

---

### If time allows

- Write down one case from your own work where judgments might split. You can use it as is for your Day 2 work theme.
- Rewrite Case 1 yourself so the company name cannot be guessed. You will see how broad the industry name must be to be safe.

**Q. You want advice on a client's equipment. Which way of writing is appropriate?**

- A** Ask with the client name and model number, and paste the measurement log file
- B** Ask what to check when an automotive parts machine is hotter than expected
- C** Hide the client name and ask with a screenshot of a drawing pasted in
- D** Paste the actual measured values as is to keep the question short

The answer is on the next page. Think it through yourself first.

## Comprehension check 2

---

The answer is **B**. You can ask without specific details

**B**

**Ask what to check when an automotive parts machine is hotter than expected**

It is generalized to the industry and process level, with no client name, real data or specific values. The equipment type and symptom remain, so you still get points to check.

**A**

Write the client name and model number, and paste the measurement log  
Model numbers and measurement logs are both specific details. This breaks Rule 1 and Rule 2

**C**

Hide the client name and paste the drawing  
The drawing itself is confidential. Hiding the name does not fix that. Describe the structure in words

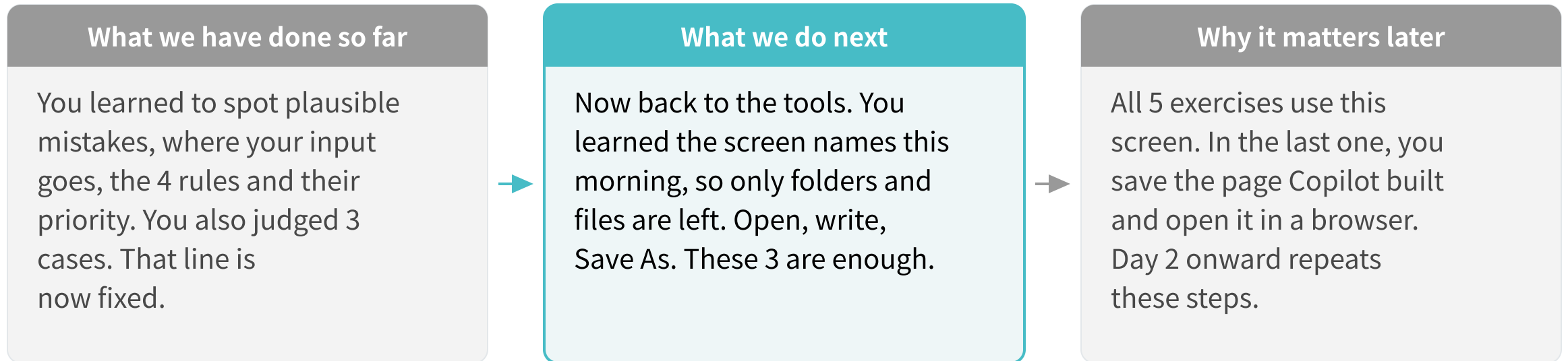
**D**

Paste the actual measured values as is  
Avoid specific values; write at the level of the goal or problem. To keep it short, give item names only

If unsure, ask yourself, "Would this text be a problem if it left the company?" If so, rewrite it.

## From decisions to hands-on steps

### From here on, it is only screen operations



# 04

## How to use VS Code

---

You will be able to open a folder, create a file and save it with Save As on your own

# Folders and files

## If you don't open a folder, no list appears on the left

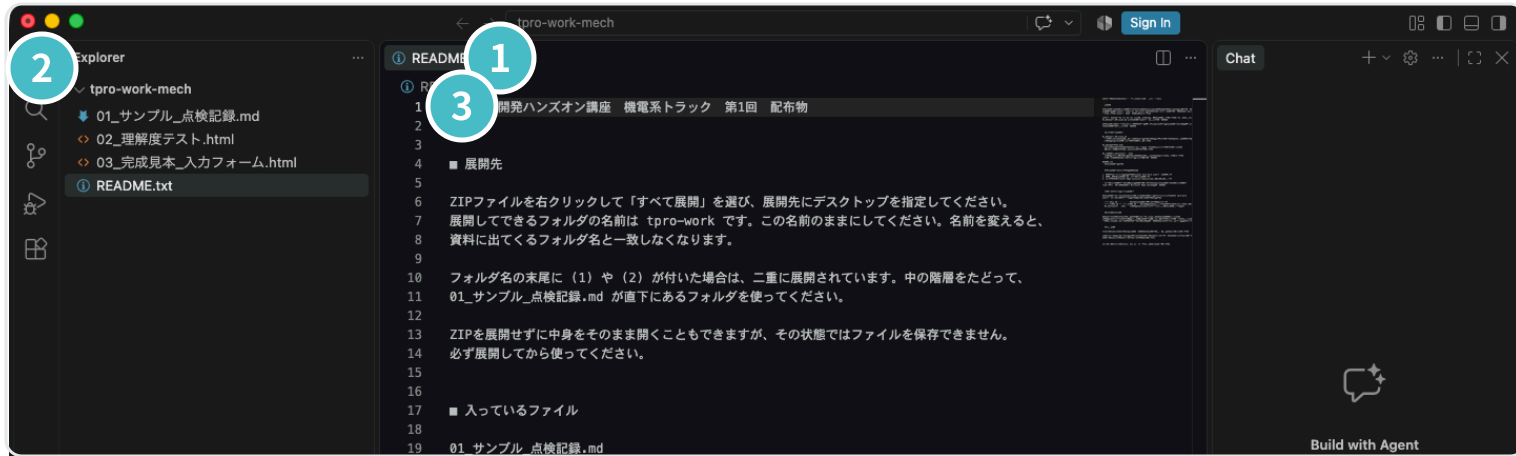
### When you open only a file

File > Open File opens just one file. The editor shows its contents, but the Explorer on the left stays empty. You cannot move to other files in the folder, and every save means looking for a location.

### When you open a folder

tpro-work opened with File > Open Folder. Everything inside is listed on the left, and one click moves between files. Saves go to this folder by default, so new files do not get lost.

## While the dot shows, the file is not saved yet



### 1 Dot on the tab

The white dot after the file name. Ctrl+S turns it into an X

### 2 Number at top left

Number of unsaved files. 1 means one is left

### 3 Text in progress

It shows on screen but is not written to the file yet

If VS Code asks whether to save when you close it, this dot was still showing

## 4 file operations

---

# Today you use only these 4 file operations

1

### **Open a folder**

From the File menu at the top, choose Open Folder and select tpro-work on the desktop. Its contents appear in the Explorer on the left

2

### **Open a file**

Click a file name in the Explorer. No need to double-click

3

### **Create a new file**

Press Ctrl+N to open an empty tab. At this point it has no name or location yet

4

### **Save As**

Press Ctrl+S, type the file name including the extension, and choose where to save it. .md is text; .html is a page you open in a browser

A dot next to the name on the tab means it is unsaved. Press Ctrl+S and the dot goes away.

## Basic operations to remember

If the text is too small to read, press **Ctrl+Shift+=** now

| What you want           | How          | When to use it                                                           |
|-------------------------|--------------|--------------------------------------------------------------------------|
| <b>Make text larger</b> | Ctrl+Shift+= | When the text on screen is too small. Ctrl+- makes it smaller again      |
| <b>Undo</b>             | Ctrl+Z       | When you paste or delete by mistake. Press again to go back further      |
| <b>Redo</b>             | Ctrl+Y       | When you went back too far                                               |
| <b>Search in a file</b> | Ctrl+F       | To find a word in the open file. The count shows at top right            |
| <b>Display language</b> | Keep English | Keep the VS Code display language in English. No language pack is needed |

Extensions can be added only if your company network can reach the download site; this depends on your company's settings. Today's content works with the English labels as is.

### The last few characters **change how a file opens**

| Extension    | Contents                            | What happens when you open it                                             |
|--------------|-------------------------------------|---------------------------------------------------------------------------|
| <b>.txt</b>  | Plain text memo                     | Text shows as is. No bold or headings                                     |
| <b>.md</b>   | Text with headings and bullet lists | VS Code shows the symbols as typed. Open the preview to read it formatted |
| <b>.html</b> | Page for a browser                  | Double-click it to open a browser that shows it as a page                 |
| <b>.csv</b>  | Table data                          | Text separated by commas. Excel can open it too                           |

Forget the extension when saving, and all text stays one color. Add it back to fix this

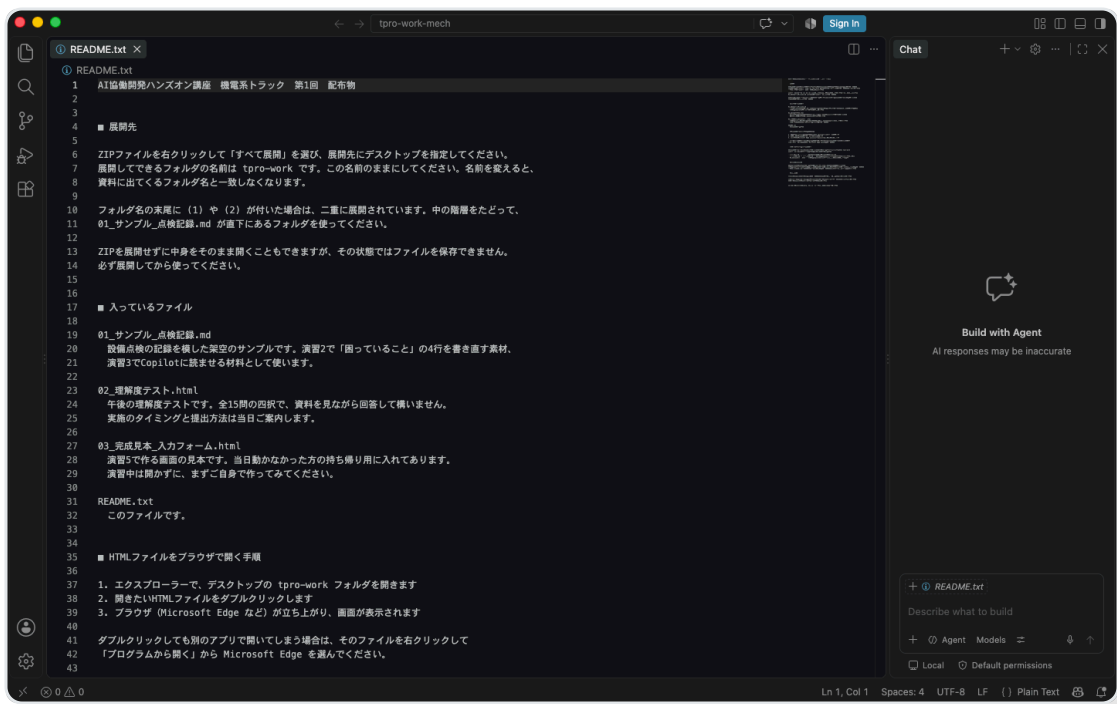
It was only closed. **All of it can be restored**

| What happened                      | How to restore                                                                |
|------------------------------------|-------------------------------------------------------------------------------|
| The file list on the left is gone  | Click the top icon on the far left. Or press Ctrl+B                           |
| Chat view is gone                  | Click the chat icon at top right. If you can't find it, look in the View menu |
| A black panel opened at the bottom | That is the Terminal. Not used today. Close it with the X at top right        |
| All text turned one color          | The extension may be missing. Use Save As again with the extension            |
| Nothing fixes it                   | Close VS Code and open it again. Saved files are not lost                     |

Nothing is broken. A stray key press can close things even if you don't remember it

# When the left list disappears

If it disappears, **press Ctrl+B once** to bring it back



## List hidden

The file list on the left is gone. It is not broken, just hidden



## List restored

Press Ctrl+B to bring the list back. Clicking the top icon on the far left does the same

# Run through the exercise steps once, in advance

### Scenario

In Exercise 3 you have Copilot read a handout file, and in Exercise 5 you save the page it builds as a file. Both start by opening a folder. If this stalls during the exercises, everyone stops, so we run through it once now.

### Steps

- 1 From the File menu at the top, choose Open Folder and select the tpro-work folder you extracted to the desktop.
- 2 In the Explorer on the left, click 01\_sample\_inspection-record.md to open it. Its contents appear on the right.
- 3 Press Ctrl+N to open a new tab and type one line with your name and today's date.

## Practice with files (continued)

---

### Steps

- 1 Press Ctrl+S, enter memo\_afternoon.md as the file name, and save it in the tpro-work folder that is open now.
- 2 Check that the dot on the tab is gone and that memo\_afternoon.md now appears in the Explorer on the left.
- 3 Finally, click the Copilot icon at the top right to open the Chat view, then click it again to close it. You use it at the start of the next section.

### Result

Save memo\_afternoon.md in the tpro-work folder on your desktop. One line of content is enough. There is nothing to submit.

## Completion criteria for file practice

### Done when the left list has memo\_afternoon.md and no dot

#### Done when

- The Explorer on the left shows both 01\_sample\_inspection-record.md and memo\_afternoon.md
- The tab for memo\_afternoon.md has no dot
- Opening 01\_sample\_inspection-record.md shows the list of inspection items on the right
- You can open and close the Chat view with the Copilot icon at the top right

#### If you're stuck

- If you cannot find the handout folder, the ZIP may not be extracted. Right-click the downloaded ZIP, choose Extract All, and try again.
- If files are saved to Documents or Downloads, check that the top of the Save screen shows tpro-work. If not, select it again from the list on the left.
- If you forgot .md in the file name, right-click the file in the Explorer, choose Rename, and add .md.
- If no list appears on the left, click the stacked-documents icon at the top left once.
- If the Save screen looks unfamiliar, it is the same screen as long as it says Save As. Type in the file name field and click Save.

## Completion criteria for file practice (continued)

---

### If time allows

- Add 3 lines to memo\_afternoon.md about what to remember from the morning, then save over it with Ctrl+S. You will see that the second save does not ask for a name.
- Close the editor tab with the x, then click memo\_afternoon.md in the Explorer to open it again. You will see that closing the tab does not delete the file.

## From practice to real use

# Build up step by step and end with something that works

### What we have done so far

So far you set up Copilot, saw that generative AI picks words by probability, and checked the 4 rules that keep client names and real data out. Just now, everyone opened a folder and saved memo\_afternoon.md.



### What we do next

In the next 100 minutes, you use those 3 places for 5 exercises: ask in English, fix a draft, ask about an open file, refine by conversation, build a working page.



### Why it matters later

The page from Exercise 5 is the starting point for next week's vibe coding intro and the Day 3 mini app. Getting this far today is enough.

# 05

## Using Copilot

---

In 5 small exercises, go from asking in English to running what you get

## Where Copilot fits

---

**Copilot is not a separate app. It sits beside where you write**

### Chat AI in a browser

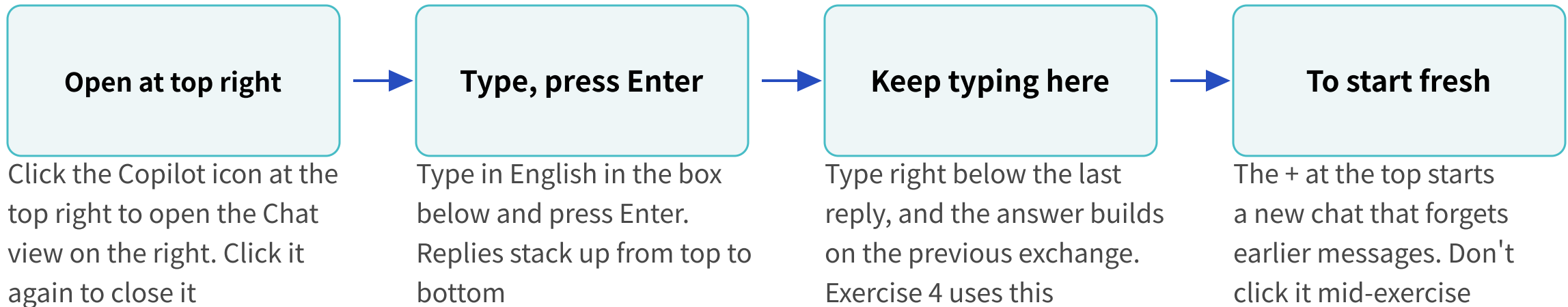
You use it in a browser tab. It sees only text you copy and paste, not your files. To get back to your document, you switch tabs.

### GitHub Copilot in VS Code

The Chat view opens on the right side of VS Code. It can use the open file's contents directly. You write and ask on the same screen, so there is no switching.

# The Chat view and continuing a conversation

## Keep typing in the same chat and earlier messages still count



If you can't find the icon, look for Copilot Chat in the View menu at the top. The icon position and shortcuts depend on the VS Code version on the day.

## Switching the model

---

You can switch, but **today we keep the default**

### Where to switch

The dropdown below the input box in the Chat view. Click it to see a list of AI models you can choose.

### Model names

Check on your screen on the day which models your company's license offers. The names change from time to time.

### Effect on today's exercises

For today's 5 exercises, the choice makes little difference. If you change it by accident, just carry on.

### When the choice matters

When you work with long code or hand over tasks with many steps. We cover this from next week, when it is needed.

## How to ask: the minimum

Just add **subject, reader and form** and the answer changes

### A request missing the 3

"Write an inspection report." This does not say which equipment, who reads it, or how long it should be. You get a general, textbook-style format you can't use for your own report as is.

### A request with the 3 added

"Openings for a monthly equipment inspection report, for my manager, 3 versions within 200 characters." Subject: the report opening. Reader: manager. Form: 3 versions, 200 characters. Now you get text you can edit and use.

## The prompt for Exercise 1

---

### Ready to type as is. Save thinking time for the next exercise

Type the line below into the Chat view as is and press Enter. To fit it to your own work, change only the subject part below. Do not include client, product or project names.

```
Write 3 openings for a monthly equipment inspection report, for my manager, 200 characters max.
```

```
(Template for rewriting)
```

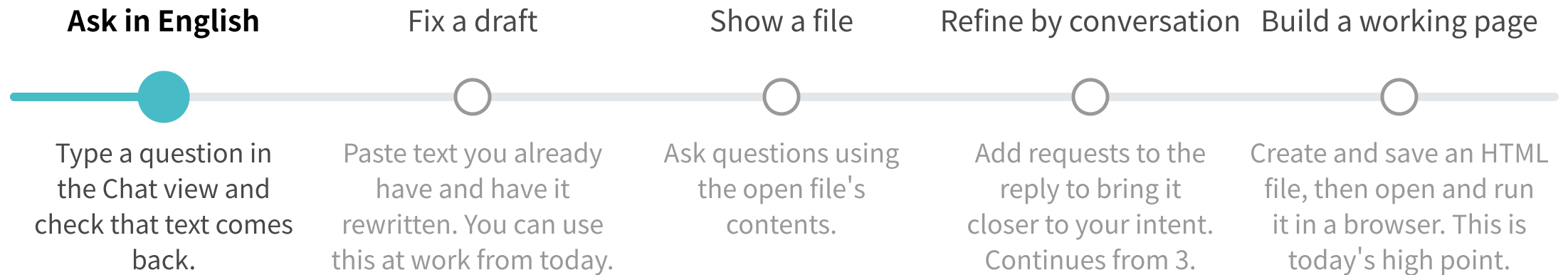
```
Create [subject] for [reader] in [format, length and number of variants].
```

```
(Example: fitted to your own work)
```

```
Write 2 requests asking another department to check an equipment noise, 150 characters max.
```

## The 5 exercises in order

Build up **one step at a time** and end with something that works



1, 2, 3 and 5 work on their own. 4 continues from 3, so if 3 did not work, start from your reply in 1.

## Exercise 1: Ask in English

At work, you first use it for **help with writing**, not code

### Scenario

You get stuck on how to start an inspection report or work notice many times a month. Instead of a blank page, get 3 drafts and pick one. You can take this use back to work soonest.

### Steps

- 1 Click the Copilot icon at the top right to open the Chat view on the right.
- 2 Click the input box and type this sentence as is: "Write 3 openings for a monthly equipment inspection report, for my manager, 200 characters max."
- 3 Press Enter to send. Text comes back within a few seconds to about 20 seconds.
- 4 Read the 3 versions and decide whether one of them would work after some editing.
- 5 If you like a reply, select that part and copy it. Press Ctrl+N to open a new file, paste it, and press Ctrl+S to save it as memo\_exercise-1.md in the tpro-work folder.

### Result

memo\_exercise-1.md (directly in the tpro-work folder). Even if you don't save it, you are done when the 3 versions appear on screen.

# Completion criteria for Exercise 1

## Done when you **decide** one of the 3 drafts works after edits

### Done when

- 3 candidate openings came back.
- You can say that one of the 3 would work in your own report after some edits.
- You asked without any client name, product name or project name.

### If you're stuck

- You get only general advice. Add the 3 conditions: subject, reader, form. Ask again.
- The reply is not in English. Send one more line, "Answer in English," to switch back.
- Enter adds a line break instead of sending. Click the send button to the right of the input box. We check on the day whether the button is there.
- It stops and nothing comes back. Check that your GitHub user name shows on the account icon at the bottom left. If not, you are signed out.

## Completion criteria for Exercise 1 (continued)

---

### If time allows

- Add 3 conditions to the same question, "the reader is my manager," "200 characters or fewer" and "no bullet points," and compare the output.
- Have it write 2 versions, "in a more formal tone" and "in a softer tone," and check how far a style instruction works.

# Fixing text you already have **pays off from today**

### Scenario

Many people spend 10 to 20 minutes each time turning rough field notes into text for a manager or another department. Hand this over, and your writing time shrinks.

### Steps

- 1 Open 01\_sample\_inspection-record.md in the file list on the left. If no folder is open yet, choose File > Open Folder and select tpro-work on the desktop.
- 2 Select the 4 lines under "What is causing trouble" at the bottom and press Ctrl+C.
- 3 Paste them into the chat and add this below: "Rewrite these 4 lines in a polite style for a report to my manager. Do not drop any of the facts."

## Exercise 2: Fix a draft (continued)

---

### Steps

- 1 Press Enter to send. Put the reply next to the original 4 lines and check by eye that no item is missing.
- 2 To try your own text, use only text you can share outside the company. Do not include client names or product names.

### Result

memo\_exercise-2.md (directly in the tpro-work folder). Saving is optional. You are done when the rewritten text is on the screen.

## Completion criteria for Exercise 2

### Done when **no original fact is lost** and only the style changes

#### Done when

- Everything in the original 4 lines is still there: copying from paper to Excel, late discovery of missing entries, searching past records, and inconsistent writing.
- The style now suits a report. The tone of rough notes is gone.
- You can judge for yourself that the rewritten text is ready to submit.

#### If you're stuck

- One item is missing. Send "Rewrite it again, keeping all 4 original items." to get it back.
- The content grows on its own and adds countermeasures that were not in the original. Add the condition "Use only what is written and do not add anything."
- Copying does not work. Drag over the 4 lines in the file, check that they are highlighted, and then press Ctrl+C.
- The file does not open. Start again from File > Open Folder and select tpro-work on the desktop.

## Completion criteria for Exercise 2 (continued)

---

### If time allows

- Have the same 4 lines written 2 ways, "30 characters for a verbal message in the company" and "politely, as a draft for the client," and compare.
- Name 2 technical terms to keep and have it rewrite the text with those terms. You see how far a term instruction works.

## Exercise 3: Show a file

# Instead of typing it all, **open the file** and Copilot reads it

### Scenario

Typing all the background into the chat is not realistic. If your material is open, the answer is built from it. This is where the time savings in later exercises and at work come from.

### Steps

- 1 In the top menu of VS Code, choose File > Open Folder.
- 2 Select the tpro-work folder you extracted to the desktop and click Open. Do not rename the folder.
- 3 Click 01\_sample\_inspection-record.md in the file list on the left to show it in the editor.
- 4 Type "Summarize the contents of this open file in 5 lines." in the chat and press Enter.
- 5 Check that the summary includes the 4 inspection items and the time spent on transcription.

### Result

memo\_exercise-3.md (directly in the tpro-work folder). Saving is optional. You are done when the summary is on the screen.

## Completion criteria for Exercise 3

### If the summary has **details unique to the file**, Copilot read it

#### Done when

- The summary names the 4 items: appearance, abnormal noise, temperature and vibration.
- The summary conveys that "transcription takes 2 hours."
- The file list on the left shows the contents of the handout folder.

#### If you're stuck

- Only general advice about inspections comes back, so the file was not read. Check that the editor tab shows 01\_sample\_inspection-record.md and send again.
- Nothing appears in the file list on the left. You opened only a file, not a folder. Start again from File > Open Folder.
- No handout folder on the desktop. The ZIP is not extracted. Right-click the ZIP, choose Extract All, and extract it to the desktop.
- The folder name ends with (1). It was extracted twice. Go down the levels and open the folder with 01\_sample\_inspection-record.md directly inside.

## Completion criteria for Exercise 3 (continued)

---

### If time allows

- Ask a follow-up: "Which of the things causing trouble in this file could AI solve?"
- Ask "If I transcribed this inspection record into Excel, what columns would you use?" and have it propose the table layout too.

## Exercise 4: Refine by conversation

### Not trying to finish in one go is the fastest way

#### Scenario

When the first reply misses the point, many people rewrite the request from scratch, but adding requests one after another gets you closer faster. You use this approach as it is in Exercise 5 and in next week's vibe coding.

#### Steps

- 1 Start with the Exercise 3 summary still on the screen. If you closed the chat, run Exercise 3 again.
- 2 Send "Make it shorter, in 3 lines." as a follow-up.
- 3 When it replies, send "Make it a bulleted list of only what is causing trouble."
- 4 On the third turn, send "Which of these should I tackle first? Give the reason in 1 line."
- 5 If Exercise 3 did not work, start from the 3 versions you got in Exercise 1 and add requests in the same way.

#### Result

memo\_exercise-4.md (directly in the tpro-work folder). Saving is optional. You are done when the 3 exchanges are on the screen.

## Completion criteria for Exercise 4

### Done when 3 or more exchanges bring it closer to your intent

#### Done when

- 3 or more exchanges on the same topic remain on the screen.
- You can say that the 3rd answer is closer to your intent than the 1st.
- You can name the one phrase, out of the 3, that changed the output most.

#### If you're stuck

- Adding requests does not change the content. Your instruction is too abstract. Use a number, like "in 3 lines" instead of "shorter."
- It seems to forget the earlier exchange and answers from zero each time. Check that you did not open a new chat. You need to continue in the same chat.
- Exercise 3 did not work, so you have no starting point. Paste the 3 versions from Exercise 1 into the chat again and add requests from there.
- Answers keep getting longer. Send "Make your last answer half as long." to fix it.

## Completion criteria for Exercise 4 (continued)

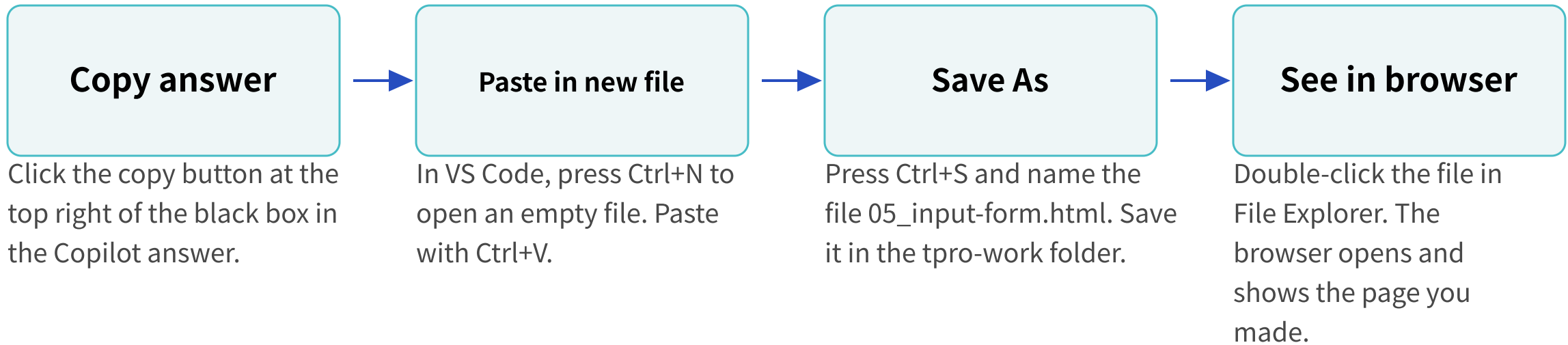
---

### If time allows

- On purpose, ask "Make it more technical and add standard numbers," and push until the content gets doubtful. You will see the plausible mistakes from this morning.
- Have the same summary written "for new hires" and "for executives," and see what the reader setting changes in the text.

## Steps in Exercise 5

Copy the answer, **save it as a file**, and open it in the browser

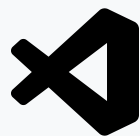


The look and position of the copy button vary by VS Code and Copilot version. If double-clicking does not open the browser, right-click and choose Open with > Microsoft Edge.

## Where you write and where you run

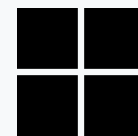
---

Save in VS Code, **open and run in the browser**



### VSCode

Where you paste the  
answer and save it as  
05\_input-form.html



### Edge

Where you double-click the  
saved file to run it

## Exercise 5: Build something that works

### What you ask for in English **runs in the browser**

#### Scenario

Writing inspection results on paper and copying them into Excel usually starts with no input screen. Ask in English for a screen that adds a name and quantity to a list. The Day 3 mini app for improving work builds on this.

#### Steps

- 1 Type this in the chat and press Enter: "Create an HTML file where I enter a name and quantity, click Add, and a row is added to the list below. Keep it all in one file."
- 2 The answer has code in a black box. Click the copy button at its top right.
- 3 In VS Code, press Ctrl+N to open a new file and paste with Ctrl+V.

## Exercise 5: Build something that works (continued)

### Steps

- 1 Press Ctrl+S to save. Name it 05\_input-form.html and save it in the tpro-work folder on the desktop. Type the name including .html.
- 2 Open the tpro-work folder in File Explorer and double-click 05\_input-form.html.
- 3 On the page in the browser, enter a name and quantity and click Add. It is complete when a row is added to the list below.

### Result

05\_input-form.html (directly in the tpro-work folder on the desktop). You are done when you open it in the browser and your row appears in the list.

## Completion criteria for Exercise 5

### Done when you enter values in the browser and **a row appears**

#### Done when

- 05\_input-form.html is in the tpro-work folder on the desktop.
- Double-clicking opens the browser with name and quantity fields and an Add button.
- Entering values and clicking Add adds a row to the list below.
- Rows you entered disappear when you reload the page. This is normal, as nothing saves them.

#### If you're stuck

- Double-clicking opens another app. The file name does not end in .html. Press Ctrl+S in VS Code again and name it 05\_input-form.html.
- The browser opens but the page is blank. The pasted code is cut off. Copy the whole box again from start to end.
- Clicking Add does not add a row. Send this as is and have it rebuilt: "Clicking Add does not add a row to the list. Fix it so it works in one file."
- Several code boxes came back. Send again: "Put it all in one file that I can save as is."
- You lost track of where you saved it. Right-click the VS Code tab to check the path, or press Ctrl+S and choose tpro-work again.

## Completion criteria for Exercise 5 (continued)

---

### If time allows

- Once it works, add one request, "Do not allow adding when the quantity is empty," save over the same file, and reload the browser.
- Add a delete button to each row of the list. If it fails, it does not affect next week.
- Show the total quantity below the list.

**The name for this approach**

---

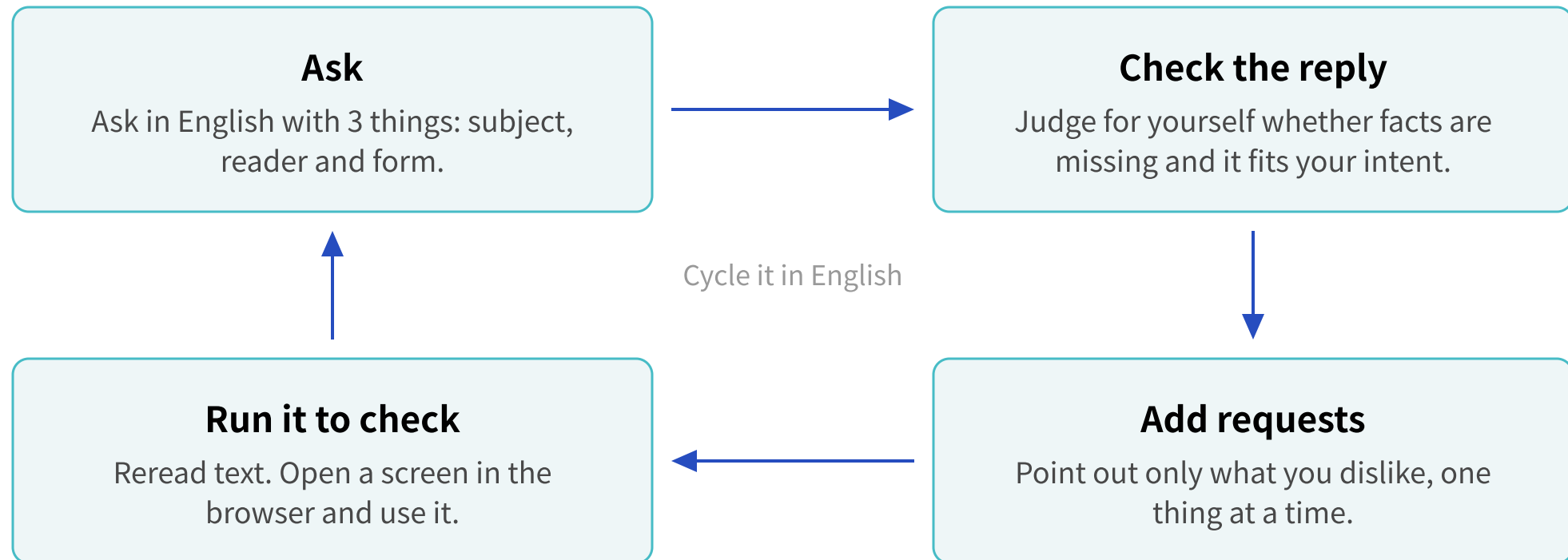
**What you just did has a name**

## **Without writing code Build it by asking in English**

This approach is called vibe coding. It is what you did in Exercise 5. On Day 2 next week, you practice it for the whole day.

## What you learned in the 5 exercises

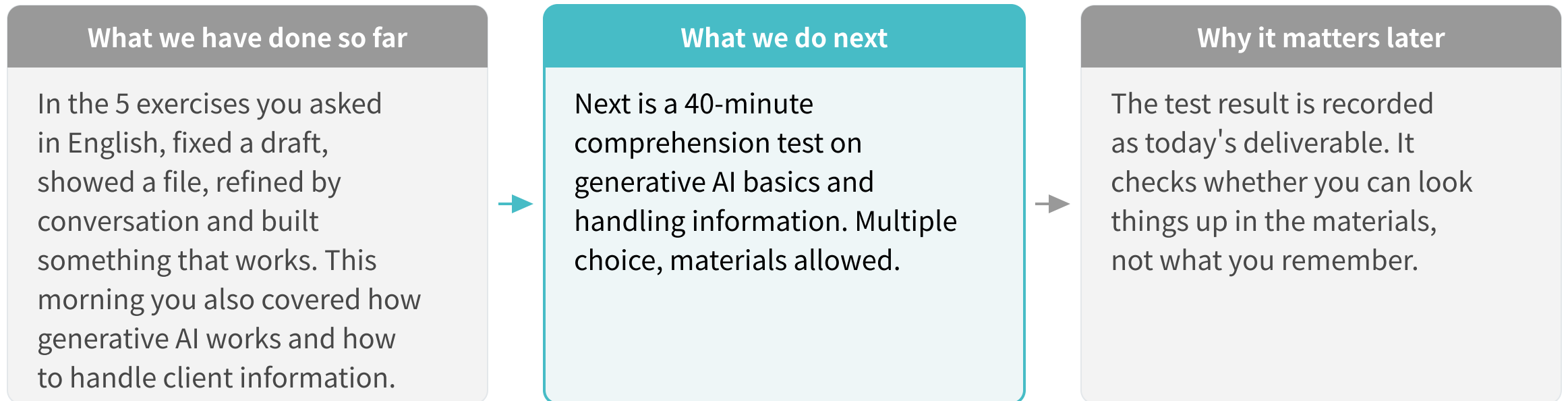
**Cycling these 4 steps is how you use it from tomorrow**



Think of one task at work tomorrow where you could cycle these 4 steps. No need to write it down.

## From exercises to the check

Now check your hands-on work as **today's deliverable**





# 06

## **Comprehension test**

---

A review test that checks today's topics with the materials open

## Comprehension test overview

A review test designed so **everyone can get full marks**

| Item    | Description                                                                            |
|---------|----------------------------------------------------------------------------------------|
| Scope   | Today's generative AI basics and information handling. No questions on exercise steps. |
| Format  | 15 multiple-choice questions. You may use today's materials.                           |
| Part    | 30 minutes. If you finish early, please wait where you are.                            |
| Talking | Please do not consult other participants. You may go back to the materials freely.     |
| Results | We grade the tests. Results are returned in the way your company prefers.              |

If you are unsure, check the materials. The test checks whether you can look things up, not memory.

# Answer 15 questions in 30 minutes, **using the materials**

**1****Receive**

Get the comprehension test. We will tell you on the day how it is handed out.

**2****Answer**

Answer 15 questions in 30 minutes. You may go back to today's materials to check while you work.

**3****If you're stuck**

If a question is unclear, raise your hand. We cannot give answers, but we explain how to read it.

**4****When you finish**

If you finish early, please stay where you are. We move on when everyone is done.

**5****Submit**

Submit when time is called. We will tell you on the day how to submit.

## From the test to wrap-up

# Decide here what you bring next week

### What we have done so far

You set up the tools, learned how generative AI works and how to handle information, did 5 Copilot exercises and took the test. You built something that works.



### What we do next

The next 40 minutes are for sorting out today in your own words. Answer 3 questions in your notes. You do not submit them.



### Why it matters later

Your notes are material for describing your work theme next week. Next week starts with 40 minutes of review, so today's sticking points can wait.



# 07

## **Review and next time**

---

Sort out today in your own words and decide what to bring next week

### Write notes on 3 questions **in your own words**

#### Scenario

What you just heard fades fastest. Only what you put in your own words is still there a week later. These notes are where you start describing your work theme next week.

#### Steps

- 1 Have a sheet of paper ready, or a new file opened with Ctrl+N in VS Code.
- 2 Answer question 1: "What surprised you most today?" One line is enough.
- 3 Answer question 2: "If you try one thing at work tomorrow, which task would you use it for?" The task name is enough. Do not write client or product names.
- 4 Answer question 3: "What are you still unsure about, or still do not understand?" We use this next week.
- 5 When you finish, share only your answer to question 3 out loud, as far as you are comfortable. Volunteers raise their hand.

#### Result

Your notes on paper or in VS Code. No submission. Take them home and bring them next week.

# Completion criteria for the review

## One line for each of the 3 is enough

### Done when

- Each of the 3 questions has at least one line.
- The task name in question 2 has no client, product or project name.
- The worry in question 3 is a question you can ask next week.

### If you're stuck

- Nothing comes to mind for question 1. Recall one moment in today's exercises when you thought "Oh, really?" An impression is fine.
- Question 2 is too broad. Narrow it from a team issue to one task you do tomorrow.
- Nothing for question 3. It is not that you understand everything; it is just not in words yet. Name one part of today you want explained again.
- Nowhere to write. Press Ctrl+N in VS Code and write there. Saving is optional.

## Completion criteria for the review (continued)

---

### If time allows

- For the task in question 2, ask Copilot: "If AI helped me with this task, where should I start asking?"
- Add one line on which input task in your work the 05\_input-form.html you made today could replace.

## Checking today's goals

---

### Review this morning's 4 goals **in a way you can judge**

#### **Environment works**

Your GitHub username shows at the bottom left, and the chat replies when you type in English. You keep this setup for next week.

#### **You know how it works**

It picks the next word by probability, so mistakes also come back in a confident tone. Done if you could answer the morning quiz.

#### **You know the line**

Generalize client, product and project names to industry and process. Do not share real data or real files. If in doubt, do not send.

#### **You have tried it**

The 05\_input-form.html from Exercise 5 ran in the browser. If yours did not, we cover it at the start of next week.

## Where today fits and the next 2 days

Today is **Day 1** of 3. Building starts next week

### Day 1 Today

Set up the tools, learned how generative AI works and how to handle information, and did 5 Copilot exercises.

### Day 2 Next week

Intro to vibe coding and prompt basics. Includes an exercise where AI cleans up a messy table.

### Day 3 After that

Analyze dummy data and build a mini app to improve your work. Pick one sample and get it running in time.

About a week passes between days. Next week starts with 40 minutes of review and a setup check.

## Next week's content

### Next week, you **build things with your own hands**

| Next week's plan            | Description                                                                                                     |
|-----------------------------|-----------------------------------------------------------------------------------------------------------------|
| <b>Day 1 review, setup</b>  | After a week away, we start by getting your environment working again. We also cover where you got stuck today. |
| <b>Intro to vibe coding</b> | Practice the approach from Exercise 5 for 130 minutes. You can build a work tool or even a game.                |
| <b>Prompt basics</b>        | Add conditions and examples to today's 3 parts: target, reader and format.                                      |
| <b>Data preparation</b>     | Have AI clean up tables broken by blanks and merged cells, like the ones at work.                               |
| <b>Your work theme</b>      | Write down where AI could help in your own work. No submission and no presentation.                             |

**Before next week**

---

**Just one piece of homework. No need to write it down**

## **Think of one task you find tedious**

One task in your work that feels like a hassle every time. We use it as the topic in next week's intro to vibe coding. No need to write it down; just keep it in mind.

The task you wrote for question 2 in today's review works as is. Bring your notes.



Givery, Inc.

Teito Shibuya Bldg. 8F, 15-13 Nampeidai-cho, Shibuya-ku, Tokyo 150-0036, Japan