

THE TECH ACADEMY · OCTOBER 9, 2026

Inside the Role of a Forward Deployed Engineer

How my career led to forward-deployed engineering, and what you can practice as you work toward your first role in tech.

William "MJ" Hill

Engineer, technical educator, and speaker. My work spans national labs, developer tools, healthcare, education, and RenderATL.

WEBSITE

mjhilldigital.com

LINKEDIN

linkedin.com/in/whill3

For aspiring engineers

We will cover the role, real projects, and practical ways to build skills and make your work known. Presented at The Tech Academy on October 9, 2026.

EMAIL

mjhilldigital@gmail.com

GITHUB

github.com/William-Hill

SECTION 01

My background and the role

Who I am, what forward-deployed engineers do, and what they deliver.

I'm MJ Hill

I have worked in software engineering since 2013. Today, my work combines building systems with understanding the people and organizations that need them.

What I build

- I build web applications, data pipelines that move information, and integrations that connect systems.
- My work includes AI tools, testing, making software available to users, and support.

Who I work with

- I work with scientists, developers, healthcare teams, educators, and conference organizers.
- Their knowledge helps me decide what to build.

What I do now

- At RenderATL, I work on engineering and technical revenue.
- Through /dev/color's Gates engagements, I work with institutions to define problems and deliver software.

The thread through my career is learning enough about someone else's work to make my engineering useful to them.

The experience behind the current role

PERIOD	WORK	WHAT IT PREPARED ME TO DO
2013-2020	Lawrence Livermore National Laboratory	Build for scientific experts and lead work across an international team.
2020-2022	New Relic	Ship developer tools, support production systems, and explain technical work publicly.
2022-2025	Meroxa, progressing to Staff Engineer	Gather customer requirements and deliver data integrations through implementation.
2025-2026	Zocdoc and /dev/color engagements	Handle healthcare workflows and build with education stakeholders.
2026	RenderATL	Connect engineering, technical education, and relationships to a conference’s needs.

Consulting taught me to learn a client’s domain, agree on a useful result, and build around their expertise.

Consulting across different industries

Each engagement meant learning a different business, working with its experts, and deciding where engineering could help.

Nevados · Solar energy

Built tools to process site data and generate solar-panel rotation schedules, plus visualization features for the customer web app.

Honey2Cocoa · Beauty technology

Advised an AI beauty startup matching makeup to skin tone for people of color. Guided technical strategy and reviewed the development team's computer-vision model and product-matching approach.

Squared2020 · Sports analytics

Built Dr. Justin Jacobs' NBA Running Net Points application: processing play-by-play data and creating an interactive view of player performance during a game.

Mela AI · Personal AI assistants

Advised the CEO and COO on architecture and AI strategy. Built a proof of concept with collaborating AI agents covering areas such as finances, health, and relationships.

My contribution varied: hands-on software delivery, technical strategy, or helping a founder work with a development team.

What is a forward-deployed engineer?

A forward-deployed engineer works closely with a customer or partner team to define a problem, build software in their environment, and help them use it successfully.

“Forward deployed”

- Work closely with the people doing the work, on-site or remotely.
- Their systems and operating constraints shape the assignment.

The engineering responsibility

- Build, integrate, test, deploy, debug, and document software.
- Use customer conversations to guide decisions throughout delivery.

The result

- Check that the software works in context and helps its users.
- Use deployment lessons to improve the product or the next implementation.

A stakeholder is someone affected by the work or responsible for a decision. Their expertise is part of the information you need to build.

This sounds like consulting

This sounds like consulting

A common difference is who you represent and what you are hired to deliver. Consider a fictional university that wants to help students get answers to enrollment questions.

Software consultant · Hired by the university

- Investigate the problem and help the university choose an approach: improve its help pages, buy a tool, or build a system.
- If implementation is in scope, build the solution, connect university data, and test it with staff.
- Deliver the agreed result for the university; the project may span several products.

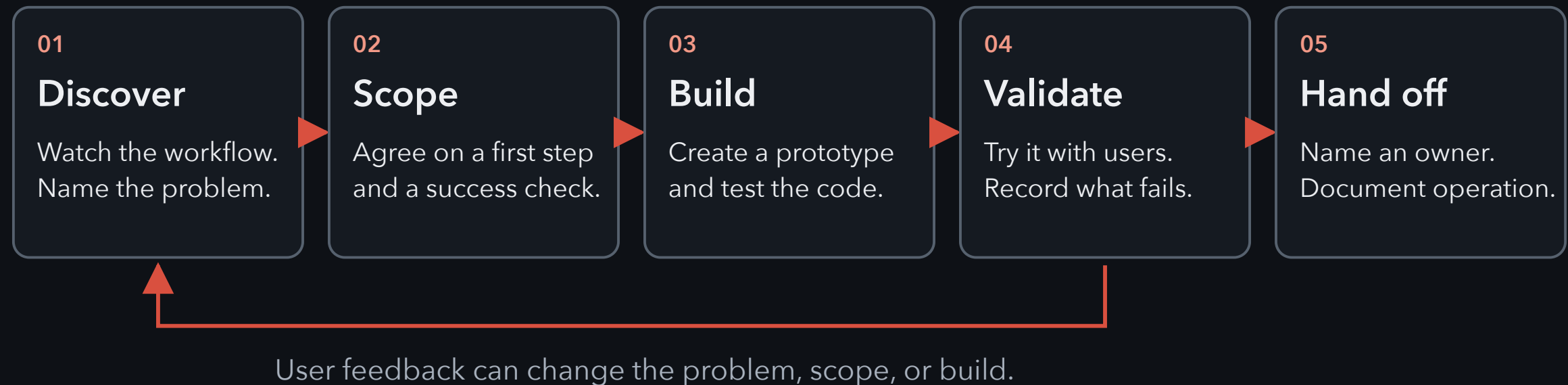
Vendor FDE · Sent by the software company

- Help the university make your company's AI platform useful for enrollment support.
- Build the integration, test answers with staff, and help launch it in their workflow.
- Bring missing features and recurring problems back to your product team so the platform improves.

In this example, the consultant delivers the university's chosen solution; the vendor FDE makes their company's platform work for the university. Both can build and deploy.

What does the engineer produce?

The work runs from understanding a problem to leaving a system people can use. The conversation continues while the software takes shape.



This is a practical way to organize the work. A small engagement may repeat these steps several times.

SECTION 02

From earlier experience to FDE work

First, the skills I built at New Relic and through consulting. Then, how I use those skills in my current work with Gates engagements and RenderATL.

New Relic taught me to ask useful questions under pressure

Before my current FDE work, I practiced asking useful questions on call at New Relic: responding when a production system needed attention. I now use that skill in time-limited stakeholder conversations.

The skill I carried forward

With limited time, I had to think about which answer would change my next action. A long list of questions is less useful than a question that helps narrow the problem.

Questions you can practice

Who is affected? What can they no longer do? What changed?
Which observation would help distinguish the likely causes? How will we know the service has recovered?

These questions illustrate the skill. They are not a reconstruction of a specific New Relic incident.

Earlier consulting taught me to learn someone else's work

You can bring engineering skills into an unfamiliar field. Start by learning what the people in that field are trying to accomplish.

Nevados · solar engineering

I built tools to parse geospatial point files and generate solar-panel rotation schedules. Understanding what the data represented helped me judge whether the output was useful.

Squared2020 · basketball analytics

Dr. Justin Jacobs supplied the Running Net Points methodology. I built data processing, an API, and an interactive application to explore it. An API lets software request information from another system.

Honey 2 Cocoa · learning an unfamiliar domain

I came in with no knowledge of makeup. I asked questions to understand how the company matched products to skin tones and what made a match useful. That understanding helped me advise on technical strategy and review the development team's approach.

On-call work taught me to narrow a problem under pressure. Consulting taught me to learn an unfamiliar domain. I now bring both skills into my FDE assignments.

How that experience became my FDE work

Those earlier roles prepared me to take responsibility from the first stakeholder conversation through building, testing, and helping people use the result. Here is where I do that work now.

What carried forward

- From New Relic: ask which answer will change the next action.
- From consulting: learn the domain, agree on the problem, and earn trust.

Gates engagements · External institutions

- Through /dev/color, I work with university and foundation representatives on workflows, requirements, and prototypes.
- I deliver software, diagrams, and documentation. The Purdue example next shows that process in practice.

RenderATL · Inside the organization

- I work with conference teams on registration, streaming, and operational needs.
- I also use technical education and speaking to support the conference's relationships and growth.

The skills connect the two chapters of my career. The next cases show how I apply them in my current FDE work.

At Purdue, the walkthrough gave us concrete requirements

At the June 2026 STEPs hackathon, three Purdue Global participants were skeptical that the learning-support platform fit their adult learners. I ran a live design sprint with them.

NEED THE WALKTHROUGH EXPOSED	TEAM CHANGE	WHY IT MATTERED
Instructors needed to see the student experience.	An instructor view-as-student preview.	Staff could inspect the experience they were supporting.
Faculty needed control over the amount of AI help.	Faculty-defined stopping points.	The tool could respect instructional choices.
Learners needed additional ways to interact.	Voice questions, read-aloud support, and accessibility fixes.	The interface supported more ways to use the tool.

I ran the sprint and co-authored the report; we built the features as a team. The report records working changes in under an hour. Institutional rollout and safe handling of real student data still needed separate work.

RenderATL needs engineering that serves the conference

At RenderATL, I work across technical delivery, education, and business relationships. The conference needs people to discover it and needs to deliver the experience it offers.

BUSINESS NEED	MY WORK	HOW THEY CONNECT
Operate the event	Session RSVP, capacity, waitlisting, and authenticated streaming.	Software supports access and participation.
Coordinate the organization	Sponsor workflows and data migration/reconciliation.	Teams need consistent information to fulfill commitments.
Reach technical communities	RenderATL University, speaker interviews, and external speaking.	Education and relationships give people reasons to learn about the conference.

This is embedded work inside one organization. It complements the external institutional work I do through /dev/color.

Event requirements become engineering decisions

Attendees, organizers, and streaming viewers have different access needs. Understanding each perspective helps define the rules the system must enforce.

PERSON	WHAT THEY NEED	ENGINEERING CONSEQUENCE
Attendee	Know whether they have a seat or are waiting.	Show an accurate registration status.
Organizer	Keep registrations within session capacity.	Enforce a capacity rule and provide a waiting-list path.
Streaming viewer	Access the content their ticket permits.	Check ticket entitlement before playback.

I built RSVP/capacity and gated streaming for RenderATL. These examples show how an operational requirement becomes behavior the software must enforce.

External speaking supports a different part of the same need

When I speak at other conferences, I can introduce RenderATL to relevant communities while sharing useful technical work.

The concrete activities

I developed RenderATL University’s three-part workshop series, hosted four speaker interviews, and included a RenderATL invitation in my WeAreDevelopers presentation.

The purpose

These activities demonstrate technical education, build relationships, and create opportunities for people to consider participating in RenderATL.

How I would measure the contribution

I have not measured how many ticket sales came from external talks. Referral links, registration sources, or qualified conversations could help evaluate that contribution.

Communication can serve a business need alongside code. Speaking alone does not establish FDE delivery; it is one part of my broader assignment.

SECTION 03

Technical Skills

Coding, data, system design, and AI: skills I use and ways you can practice them.

Coding skills • Turn a requirement into working software

An FDE needs to read existing code, build useful changes, and debug them. Start with one language and learn to explain how your code works.

Skills to build

Practice functions, data structures, and error handling. Read and write API requests. Use Git, write tests, and trace a bug from its symptom to its cause.

Where I used them

At New Relic, I used Python to process and send log data. At Meroxa, I built connectors in Go. The languages changed; reading code, debugging, and checking behavior carried across.

A project to practice

Build a small registration API that accepts a name and email. Validate input, reject duplicate registrations, and write tests for successful requests and invalid data.

Choose one small feature you can build, test, and explain. Use AI for help, then read the changes and verify what the code actually does.

Data skills · Understand what flows through the system

My work at Meroxa and on Squared2020 depended on collecting, transforming, and delivering data that another system or person could use.

Skills to build

Use SQL to query and join tables. Model records and relationships. Read JSON or CSV, call an API, and check for missing or duplicate data.

Where I used them

At Meroxa, I built connectors that move data between systems. For Squared2020, I processed basketball play-by-play data and stored it for an interactive application.

A project to practice

Load a small public dataset into a database. Answer a question with SQL, expose the result through an API, and test missing fields and duplicate records.

Follow one record from its source to the user. Explain each transformation and how you would notice an incorrect result.

System design • Decide how the pieces work together

System design means choosing components, their responsibilities, and how they communicate under real constraints.

Skills to build

Draw the path from user request to stored data. Decide where rules live, who can access each operation, and what happens when a dependency fails.

Where I used them

At New Relic, we processed S3 log files in pieces to fit the service's resource limits. At RenderATL, registration needed capacity rules and streaming needed ticket-access checks.

A project to practice

Draw a registration form, its API, and its database. Show where the seat limit is enforced. Explain how duplicate requests, simultaneous bookings, and failures are handled.

Explain the tradeoff: why does this design fit the users, deadline, expected load, and people who will maintain it?

AI skills • Build, evaluate, and verify

At Mela AI, I built a prototype with collaborating AI agents. In Gates engagements, I helped turn institutional feedback into AI-tool behavior. Those projects require engineering judgment as well as model use.

Skills to build

Call a model through an API, give it relevant context, and connect only the tools it needs. Define what it may read or change and when a person must decide.

Evaluate the behavior

Create sample inputs with expected results. Check accuracy, unsupported answers, and failure behavior. Compare changes against the same examples instead of trusting a convincing demo.

A project to practice

Build an assistant over a small set of public documents. Write ten questions and expected answers, including questions the documents cannot answer. Check its answers and references.

AI is a force multiplier. Practice judging and testing its output while building your programming, data, and debugging skills.

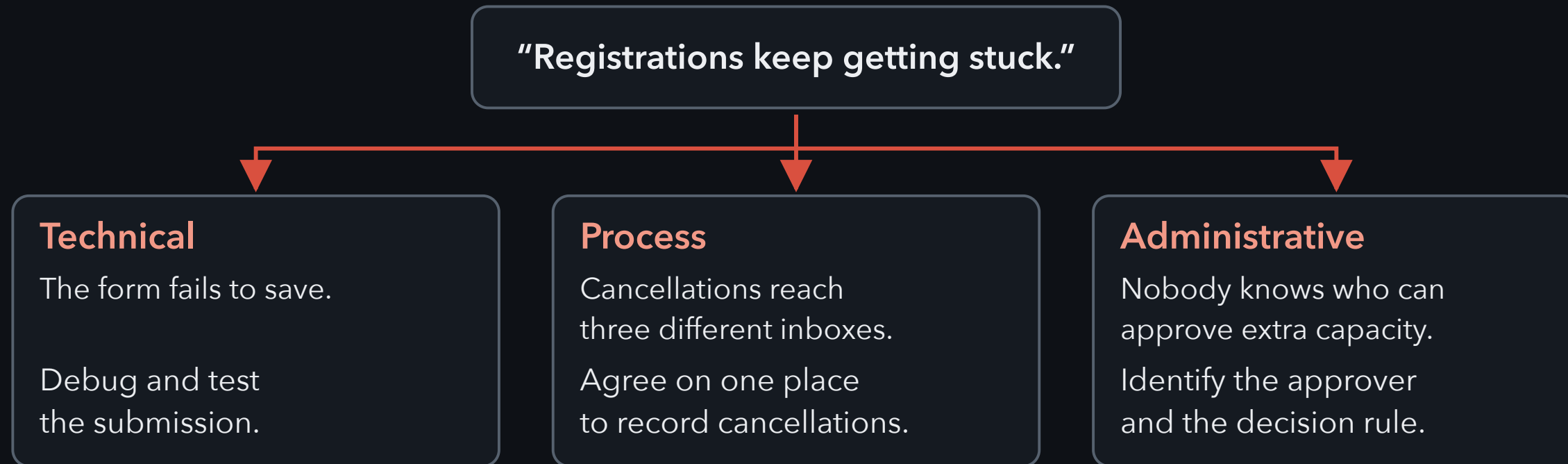
SECTION 04

The human side

Listen, understand the workflow, and earn buy-in for a useful change.

The human side is part of the engineering

Empathy and listening help you trace a complaint to its cause. This fictional registration example shows three places to look; more than one may apply.



An elegant technical solution can still fail if it solves the wrong problem or asks people to work in a way they cannot sustain.

How to ask better questions on your next project

“People like to talk about their problems.” Give them room to explain what happens before you offer a solution.

QUESTION

WHY IT MATTERS

“Can you walk me through the last time this happened?”

Listen for the actual steps, workarounds, and handoffs.

“Which part is most frustrating, and why?”

Understand the effect on their time, confidence, or workload.

“What have you already tried?”

Respect their experience and learn why earlier attempts fell short.

“Who makes the decision, and who has to do the work?”

Find the approver, day-to-day user, and support owner.

“What would make a first version worth trying?”

Agree on a useful change and a way to check it.

Listen without interrupting, then summarize: “What I’m hearing is _____. Did I get that right?” Let them correct your understanding.

Get buy-in before you build

Buy-in means the people using, approving, and supporting the change understand it and agree to try it. A working demo is only part of that conversation.

ASK ABOUT	FICTIONAL REGISTRATION EXAMPLE	AGREE ON A NEXT STEP
The process	Cancellations arrive in three inboxes.	Choose one place to record them before automating.
Authority	A volunteer cannot approve extra seats.	Find the person who owns the capacity decision.
Daily use	Nobody has time to maintain a second system.	Try the existing sheet with one named owner.
A useful result	The organizer needs a trustworthy attendance list.	Test a full room and a cancellation together.

Ask: “Does this fit how you work? Who needs to agree? Who will maintain it?” If people object, understand the constraint and revise the plan with them.

SECTION 05

Interactive exercise

Try a registration problem, then compare your approach with one possible solution.

Exercise: a workshop needs a registration process

Fictional scenario: a volunteer organizer asks for a registration form. The event is next week, the room has 30 seats, and the organizer currently uses a spreadsheet.

The complication

People sometimes cancel. The organizer wants to avoid overbooking and know whom to contact when space opens.

Your task • 2-minute exercise

After the setup, take 70 seconds to write two questions that could change the design, a smallest useful version, and one test. Then we will hear a brief answer.

A constraint on your answer

Consider the organizer's time and authority as well as the software. Use an existing tool, a process change, or custom code, and explain why the choice fits.

The next slide contains one possible approach. Try your own answer before moving on.

One possible answer to the exercise

The solution depends on what the organizer can operate next week. Here is a small version with explicit assumptions.

Questions and assumptions	First version	Validation
• Can the organizer manage seat offers manually? How many registrations are expected? • Assume a small event and an organizer who agrees to manage cancellations and offers.	• A form adds entries to a sheet as waiting. Only the organizer changes status. • Offer a seat only when confirmed plus reserved places total less than 30. • Reserve an offer for 24 hours. Acceptance confirms it; a decline or expiry releases it.	• Test a full room, cancellation, acceptance, and expiry. Keep confirmed plus reserved places at or below 30. • Match messages to the sheet. Release cancelled or expired places before making the next offer.

If volume or simultaneous registrations makes manual control unreliable, revisit the design. The assumptions tell you when the simple version stops being sufficient.

SECTION 06

Your next steps

Build relationships, learn by teaching, and practice on a project you can explain.

The technical foundations to practice

A useful small project exercises more than the interface. Build enough of the system to understand how its pieces work together.

FOUNDATION	PRACTICE IT IN THE REGISTRATION PROJECT
Programming and debugging	Validate inputs and explain errors clearly.
Data modeling	Represent a registration with waiting, reserved, or confirmed status.
APIs and integration	Connect a form to storage or another service.
Testing	Check full capacity, cancellations, duplicates, and invalid input.
Deployment and operation	Run the project somewhere usable and document how to find failures.

For an AI-specific role, add model evaluation and boundaries around data and actions. Those build on ordinary software engineering skills.

Make staying current a habit you can sustain

Tools will keep changing. Give yourself a small, repeatable way to try new ideas while continuing to build your engineering foundation.

Read one primary source

- Read an official tutorial, release note, or documentation update for a tool you use.
- Identify the change, the problem it addresses, and its limits.

Run one small experiment

- Use sample data in a project you understand.
- For AI, compare output with an expected result and record a failure case.

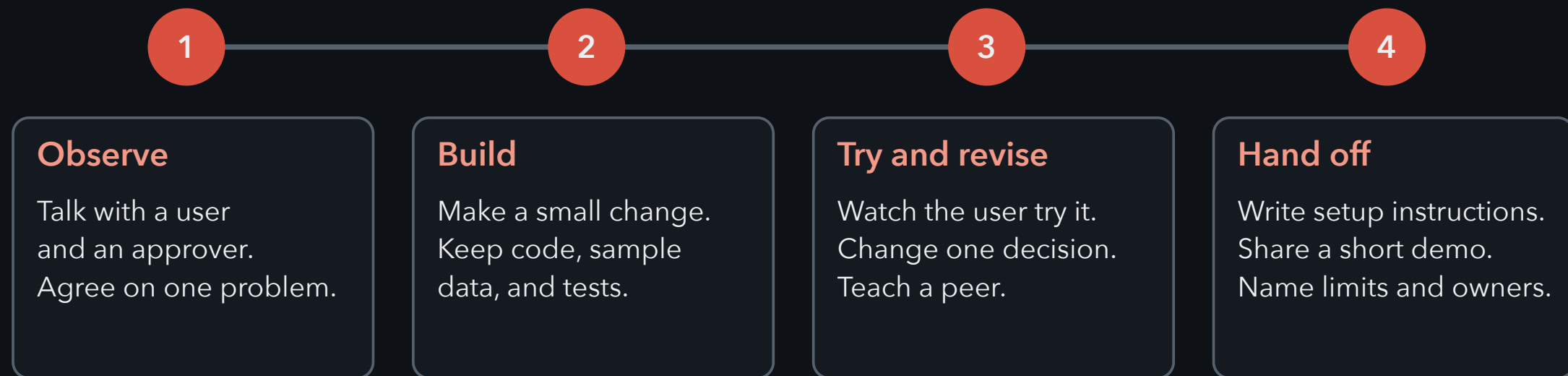
Share what you learned

- Write a short note or show a classmate.
- Explain what worked, what you checked, and when you would use it. Investigate their questions.

A suggested weekly routine: 15 minutes reading, 30 minutes trying, and 15 minutes explaining. Adjust it to the time you have.

A four-week project you can explain in an interview

Use this as a suggested learning plan. Choose a willing user and a small workflow you can safely observe and improve.



A suggested four-week plan: keep evidence of your decisions and results.

Keep sensitive data out of a public portfolio. Label a prototype as a prototype and separate observed results from intended benefits.

Describe your experience with STAR

STAR means Situation, Task, Action, Result. Use it to explain the problem, your responsibility, your decisions, and what happened. This is a fictional registration-project example.

S · Situation

Set the context. "The organizer could not tell who had a confirmed seat and who was waiting."

T · Task

Name your responsibility. "I needed to give the organizer a reliable list before next week's event."

A · Action

Explain what you did and why. "I agreed on status rules with the organizer, updated the sheet, and tested cancellations and seat offers."

R · Result

Describe what you observed. "In our tests, the list stayed within capacity and showed the correct status after a cancellation."

Use your own evidence. Be clear about your contribution, separate a test result from live impact, and say what you learned if the result fell short.

Yapping

“It’s who knows what you know.” A lot of my opportunities have come from people knowing my background and the work I’ve done.

Give people something to know

- Show a project, a small demo, or a problem you solved.
- Use a class discussion, meetup, community chat, or short post to make your skills visible.

Lead with value

- Listen to what someone is working on; share a useful example or answer a question you understand.
- Be clear about what you know and what you are still learning.

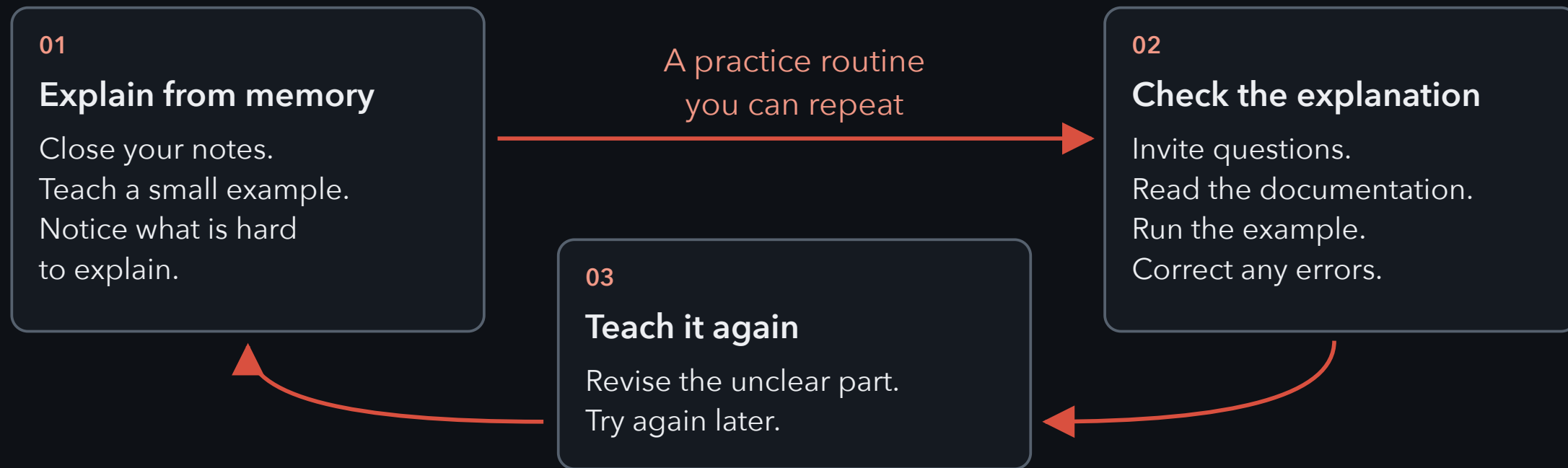
Keep the connection going

- Follow up with the resource you promised or explain how their advice helped.
- Show interest when you do not need a referral. Relationships take time and attention.

Try this: “I’m learning _____. I built _____ to help with _____. Here’s what surprised me.” Then ask what the other person is working on.

One of the best ways I learn is to teach

Teaching at Hackbright, running workshops, and speaking at conferences have all helped me learn. Explaining an idea makes me find the parts I cannot yet explain clearly.



Koh, Lee, and Lim (2018): in one experiment, teaching without notes and recall practice outperformed teaching with notes or a control task on comprehension one week later. This diagram is a suggested routine, not a percentage scale.

Your first step can be small

This week, pick one useful project, teach someone one thing you learned, and let someone see what you can do.

Keep building

Practice programming, testing, and debugging on a real workflow. Use AI where you can check its work.

Keep talking to people

Ask about their work, share something useful, and follow up. Let your projects and explanations give people a reason to remember your skills.

Stay in touch

William “MJ” Hill · mjhilldigital.com · github.com/William-Hill

Q&A: what are you building, what are you trying to learn, or what feels hardest about getting started?

Thank you

WEBSITE

mjhilldigital.com

LINKEDIN

linkedin.com/in/whill3

EMAIL

mjhilldigital@gmail.com

GITHUB

github.com/William-Hill

Q&A

WEBSITE

mjhilldigital.com

LINKEDIN

[linkedin.com/in/whill3](https://www.linkedin.com/in/whill3)

EMAIL

mjhilldigital@gmail.com

GITHUB

github.com/William-Hill