THE COMPLETE PLAYBOOK FOR

GETTING A JOB IN TECH

The 2026 Edition

From 60+ Rejections to AWS Solutions Architect
The exact system, frameworks, and strategies
I wish someone had given me this on day one.

By Olushola Oladipupo
AWS Solutions Architect | 9x Cloud Certified
Creator of Shola's Tech Notes
sholastechnotes.com

Introduction: Why This Book Exists

I didn't study computer science. I studied Agricultural Economics in Nigeria, then Economics and Finance for my MSc in the UK.

No coding bootcamp. No family in tech. No insider connections. Just a laptop, an internet connection, and the stubborn belief that I could figure this out.

I applied to over 60 jobs. AWS roles, cloud engineering, solutions architecture - anything that sounded like the career I wanted. Every single one came back the same way: "Not enough technical experience." "Wrong background." "We've decided to move forward with other candidates."

I tried everything. Tweaked my CV. Added more skills. Got more certifications. Nothing changed.

Then a mentor told me something I'll never forget. Three words that rewired how I thought about my entire career:

> *"So what?"*

Not "what did you do?" - but "why does it matter?"

That question changed everything. I stopped listing activities and started showing impact. I stopped collecting certifications and started building things. I stopped waiting for permission and started creating proof.

Today, I'm a Solutions Architect at Amazon Web Services. I hold 9 cloud certifications - 5 AWS and 4 Azure. I've built a newsletter read by thousands of people making the same transition I made. And I've helped many career pivoters, graduates, and self-taught developers land their first (or next) tech role.

Personal Disclaimer

This book represents my personal views and experiences. I am employed by Amazon Web Services, but the content, opinions, and advice in this book are entirely my own and do not represent the views, policies, or endorsements of Amazon or AWS. All information about AWS services, certifications, and tools is based on publicly available resources. No confidential, proprietary, or non-public Amazon information is included. The frameworks and strategies in this

book are drawn from my personal career transition, my experience mentoring others, and publicly available industry knowledge - not from Amazon's internal training or processes.

Who This Book Is For

This book is for you if:

- You're thinking about switching into tech but don't know where to start
- You've been studying for months but still can't land an interview
- You have certifications but no projects to show for them
- You're applying to dozens of jobs and hearing nothing back
- You're already in tech but want to level up - better role, better pay, more impact
- You're wondering whether AI has made it harder or easier to break in

Whether you're a career pivoter coming from finance, teaching, healthcare, or retail - or a recent graduate wondering what's next - this is the playbook I wish I'd had on day one.

What You'll Get From This Book

This isn't a "10 tips to get a tech job" listicle. This is a complete, actionable system. By the time you finish reading, you'll have:

- A clear understanding of the 2026 tech job market - what's changed, what employers actually want, and where the opportunities are
- A personalised roadmap based on your situation - whether you're starting from zero or levelling up
- The exact certification path that matches your career goals (not random certs that waste your money)
- A portfolio strategy that actually impresses hiring managers - not tutorial projects everyone else has
- An AI toolkit that makes you 10x more productive - because AI isn't replacing engineers, it's replacing engineers who don't use AI
- A job search system that gets interviews - the "So what?" CV framework, networking scripts you can copy and send, and interview preparation strategies
- A 90-day action plan with weekly milestones you can follow step by step

- **BONUS:** A free 90-Day Tech Career Playbook Notion template - 7 interlinked databases (Certifications Tracker, Portfolio Projects, Networking CRM, Job Applications, Interview Prep, Sprint Planner, and Wins Journal) plus networking email templates. This is your execution system for everything in this book.

How to Use This Book

Don't read this cover to cover and do nothing. That's what everyone does with books like this. They feel productive while reading, then close the cover and nothing changes.

Instead, here's what I want you to do:

1. Read Chapter 1 first. It'll help you understand where the market is and where you fit.
2. Skip to the chapter that matches your biggest blocker right now. Certification confusion? Go to Chapter 3. Can't get interviews? Jump to Chapter 6.
3. Complete the action step at the end of every chapter. They're designed to take 30 minutes or less. Do them. That's how progress happens.
4. Use the 90-Day Sprint Plan in Chapter 8 as your weekly accountability tracker.

THE SHOLA GUARANTEE

If you follow the frameworks in this book and put in the work consistently, you'll have everything you need to be interview-ready within 90 days. Not "someday" ready. Actually ready. The kind of ready where you walk into an interview and the interviewer thinks, "This person gets it." This isn't a magic formula and your timeline will depend on your starting point - but it's a proven system that works when you commit to it.

MONEY-BACK GUARANTEE

If you follow the frameworks in this book for 30 days and genuinely feel it hasn't helped you move closer to your goal, email shola@sholastechnotes.com and I'll refund you in full. No questions, no hassle. I believe in this system because I've watched it work - for myself and for the people I've mentored.

FREE UPDATES FOR 12 MONTHS

The tech market moves fast. This book will be updated as certifications change, salary benchmarks shift, and new AI tools emerge. All buyers receive free updates for 12 months from purchase - you'll always have the latest edition.

Free Resources That Complement This Book

I've created free tools and guides that go deeper on specific topics covered here. Grab them at sholastechnotes.com:

- AWS Certification Roadmap - Visual guide to all current AWS certs, recommended paths, and study resources (sholastechnotes.com/aws-certification-roadmap)
- The DevOps Hiring Playbook - 34-page guide to landing DevOps roles, with real interview questions and frameworks (sholastechnotes.com/devops-hiring-playbook)
- The AI Coding Toolkit - Side-by-side comparison of the 5 AI coding tools from Chapter 5, with setup guides (sholastechnotes.com/ai-coding-toolkit)
- ResumeRadar - Free CV scanner that analyses your CV against real hiring criteria (resumeradar.sholastechnotes.com)

Let's get started.

Chapter 1: The New Rules of Tech in 2026

Here's the thing - the tech job market you're entering in 2026 is fundamentally different from the one that existed even two years ago. If you're following advice from 2023, you're playing a game with outdated rules. Let me show you what's actually happening.

The AI Revolution Changed Everything

AI didn't just change what we build. It changed how we get hired, how we work, and what skills actually matter.

Think about it like this: when cars replaced horses in the early 1900s, the value didn't stay with the people who could ride fastest. It shifted to the people who could drive - and eventually to the people who could design roads, build petrol stations, and reimagine transport entirely.

That's exactly what's happening in tech right now. AI is democratising technical skills. What once took years of expertise now takes a well-structured prompt. The shift isn't from smart to dumb - it's from IQ-based value (raw knowledge, memorised syntax, degrees) to agency-based value (the ability to use AI effectively to solve real problems).

The Numbers Tell the Story

Let's talk data, not vibes:

- Entry-level tech hiring has dropped significantly - Ravio's 2025 report found a 73% decline in junior positions. Companies are using AI tools to make senior developers more productive instead of hiring juniors.
- Job postings mentioning AI surged over 130% between 2024 and 2026 (Indeed Hiring Lab). Every role now expects some level of AI fluency.
- Many jobs are filled through networking and referrals - not job boards. Cold applications go through AI screening systems that reject most candidates automatically.
- Many hiring managers check portfolios before interviews. Your GitHub profile matters more than your degree.
- 56% of UK organisations plan to expand their permanent IT teams in the first half of 2026, with cloud, cybersecurity, AI, and DevOps specialists in highest demand (Robert Half, December 2025).

This shift is partly driven by AI. Research from Anthropic (2025) found that young workers aged 22 to 25 in AI-exposed occupations experienced a 14% relative decline in job-finding rates after the launch of ChatGPT. The US Bureau of Labor Statistics projects that occupations with higher AI exposure will see slower employment growth over the coming decade. But this does not mean cloud and tech careers are disappearing. Quite the opposite: with nearly half of global enterprise workloads still running on-premises or in private cloud environments (Flexera State of the Cloud Report, 2025), the cloud migration services market is projected to grow from roughly $17 billion in 2024 to over $70 billion by 2030 (Grand View Research, 2025). These workloads must move to the cloud before organisations can leverage AI at scale, which sustains strong demand for cloud professionals who can architect, secure, and optimise these migrations. What is changing is the entry point. Tools like AWS Transform now automate discovery, dependency mapping, and server replication for cloud migrations, significantly reducing manual effort. This means the repetitive, junior-level tasks that used to be someone's first role are being absorbed by automation. The system in this book exists precisely because of this shift. In a market where AI handles the routine work, your differentiation comes from certifications that prove depth, projects that demonstrate judgement, and a professional network that opens doors no algorithm can.

WHAT THIS MEANS FOR YOU

The good news: tech is still hiring. The bad news: the bar has changed. You can't just "learn to code" anymore. You need to learn to code AND build things AND use AI tools AND communicate your impact AND network strategically. This book shows you how to do all of it.

The Four Rules of AI

Before we go further, internalise these four rules. They'll shape everything in this book:

Rule 1: Always Invite AI to the Table

Every workflow you have - coding, writing, researching, debugging, preparing for interviews - should involve AI. Not because you're lazy. Because you're strategic. The best engineers in 2026 aren't the fastest coders. They're the ones who know which AI to point at which problem.

Rule 2: Be the Human in the Loop

AI generates. You validate. AI suggests. You decide. Never blindly ship what AI produces. Your judgment, your taste, your understanding of the problem - that's what makes the output valuable.

Rule 3: Treat AI Like a Person

Tell it what role to play. “Act as a senior solutions architect reviewing my system design.” “Act as a hiring manager reading this CV for the first time.” Give it context - 2 to 5 pages of context, not a one-line prompt. The quality of your output is directly proportional to the quality of your input.

Rule 4: Assume This Is the Worst AI You’ll Ever Use

Whatever AI can do today, it’ll do 10x better next year. The skills you build around AI - prompt engineering, context engineering, knowing when to use which tool - will only become more valuable. You’re not learning a tool. You’re learning a superpower.

What Employers Actually Want in 2026

I talk to hiring managers every week. Here’s what they’re actually looking for:

Then (2023–2024)

- Computer science degree or bootcamp certificate
- Generic “software engineering” skills
- Years of experience as a proxy for capability
- Perfect algorithm performance in coding interviews

Now (2026)

- Demonstrated AI fluency - you must be comfortable using Claude, Copilot, or ChatGPT
- Specific business problem-solving ability - not “I can code” but “I solved this problem”
- Portfolio projects proving real-world capability - deployed, documented, explainable
- Hybrid skills - product thinking combined with technical depth
- Communication clarity - system design reasoning matters more than syntax perfection

THE KEY INSIGHT

LinkedIn’s 2025 Talent Report highlights that hiring managers increasingly list AI fluency as a preferred or expected skill for entry-level hires. While it is not yet universal, the direction is clear: candidates who can demonstrate practical AI capability have a measurable edge.

The Salary Landscape

Let's talk money - because understanding the market helps you negotiate better and choose the right path:

UK Tech Salaries (2026)

- AI Engineers: £80,000–£130,000+
- Solutions Architects: £100,000+
- DevOps Engineering Lead: £80,000–£110,000
- Python Developers: £90,000
- Cloud Engineers (entry): £38,000–£45,000
- UX/UI Designers: £43,000–£85,000+

Global Remote Tech Salaries

- Average remote tech worker: $148,930 USD
- AI/ML Engineers: $110,000–$130,000 USD
- Cybersecurity Specialists: $110,000–$130,000 USD

The salary premium for cloud-certified professionals? According to the Global Knowledge IT Skills and Salary Report, certified cloud professionals earn an average salary of $113,000 - over $10,000 more per year than non-certified peers. That's not a rounding error - that's a significant return on a certification that costs $150–$300 and takes 6–12 weeks of study.

Remote Work in 2026

The remote work debate is settled. Here's where things stand:

- 85% of workers prefer jobs with remote flexibility; fully remote roles attract 2.5x more applicants than on-site equivalents (FlexJobs, 2024)
- Tech leads with 44% hybrid or remote options - highest of any sector
- Remote positions often trade 10–25% lower salaries for flexibility - but total compensation (no commute, location freedom, lower cost of living) frequently comes out ahead

- 85% of workers rank remote flexibility above salary in their job priorities (FlexJobs 2025 survey)

Bottom line: remote and hybrid work is the norm in tech, especially for cloud, DevOps, and AI roles. This is good news for career pivoters - you're not limited by geography.

YOUR ACTION STEP

1. Open a notes app or grab a piece of paper.
2. Write down: "The one thing I'm most confused about in the current tech job market is ___."
3. Keep that note. By Chapter 8, you'll have a clear answer and a plan to act on it.

Chapter 2: Pick Your Lane (And Go Deep)

This is where most people get stuck. They try to learn everything. Cloud. Cybersecurity. Data science. DevOps. Web development. AI. Machine learning.

Six months later, they know a little about a lot - and can't get hired for anything.

Here's the uncomfortable truth: you need to pick ONE path and go deep. Not forever. Just long enough to become genuinely competent and employable.

Why Going Wide Kills Your Chances

I see this constantly in the CVs I review. People list 50 technologies thinking it makes them look impressive. It doesn't. It makes them look scattered.

Think about it from a hiring manager's perspective. They're hiring for a cloud engineer role. They see two candidates:

Candidate A: *"Skills: Python, JavaScript, Java, React, Node.js, AWS, Azure, GCP, Docker, Kubernetes, Terraform, Ansible, MySQL, MongoDB, Redis, GraphQL, REST APIs..."*

Candidate B: *"Built and deployed a 3-tier web application on AWS using ECS Fargate, with a CI/CD pipeline via GitHub Actions and infrastructure managed through Terraform. Reduced deployment time from 45 minutes to 3 minutes."*

Candidate B gets the interview. Every time. Because they showed depth, not breadth. They showed impact, not a shopping list.

The Five Tech Domains (Pick One)

Every tech career starts in one of these five lanes. Each one leads to high-paying roles. None of them requires a computer science degree.

1. Cloud & Infrastructure

This is my lane - and I'm biased, but for good reason. Based on my experience and market research, AWS offers a broad entry point into tech. Other cloud platforms like Azure and GCP also have strong ecosystems, so choose based on your local job market and learning preferences.

- Entry roles: Cloud Support Engineer, Junior Cloud Engineer, Cloud Administrator, Associate Solutions Architect
- Growth path: Cloud Engineer → Solutions Architect → Principal Architect
- UK salary range: £38,000 (entry) to £130,000+ (senior/specialist)
- Key certifications: AWS Cloud Practitioner, AWS AI Practitioner, Solutions Architect Associate
- Why it works: No CS degree needed. Certifications open doors. Massive job market.

2. DevOps & Platform Engineering

DevOps is where cloud meets automation. If you enjoy making systems faster, more reliable, and more automated, this is your lane.

- Entry roles: Junior DevOps Engineer, Build Engineer, Release Manager
- Growth path: DevOps Engineer → Senior DevOps → Platform Engineer → Principal
- UK salary range: £38,000 (entry) to £110,000+ (lead)
- Key skills: Linux, Git, Docker, CI/CD, Terraform, Python/Bash scripting
- Why it works: Huge demand, automation mindset, clear skill progression.

3. Cybersecurity

Every organisation needs security. With AI expanding attack surfaces, the demand for security professionals is exploding.

- Entry roles: Security Analyst, SOC Analyst, IT Security Engineer
- Growth path: Analyst → Engineer → Architect → CISO
- UK salary range: £35,000 (entry) to £120,000+ (senior)
- Key certifications: CompTIA Security+, ISC2 CC, AWS Security Specialty
- Why it works: Critical demand, transferable from any background, government contracts.

4. Software Development

The classic path. If you love building things - apps, websites, APIs, tools - this is where you belong. And in 2026, AI coding tools like GitHub Copilot, Cursor, and Claude have become force multipliers. You don't need to memorise every syntax rule - you need to understand what to build and why. Think of AI as your co-intelligence: it handles the boilerplate while you focus on architecture, logic, and user experience.

- Entry roles: Junior Developer, Frontend Engineer, Backend Engineer
- Growth path: Junior → Mid → Senior → Staff Engineer → Principal
- UK salary range: £30,000 (entry) to £120,000+ (senior)
- Key languages: Python, JavaScript/TypeScript, Java, Go
- Why it works: Largest job market, remote-friendly, AI tools multiply your output.

5. Data & AI

The hottest domain in 2026. If you're drawn to data, machine learning, or AI applications, there's enormous demand.

- Entry roles: Data Analyst, Junior Data Engineer, ML Operations Engineer
- Growth path: Analyst → Engineer → ML Engineer → AI Architect
- UK salary range: £35,000 (entry) to £130,000+ (AI specialist)
- Key skills: Python, SQL, cloud platforms, basic ML concepts
- Why it works: Fastest growing domain, premium salaries, AI fluency builds on this.

How to Choose (The Decision Framework)

Still stuck? Answer these three questions:

5. What excites you most? If you could spend 2 hours this weekend learning about one of these five domains, which one would it be? Go with your gut.
6. What's your existing advantage? Worked in finance? Data & AI leverages that. Healthcare background? Cloud and security are huge in health tech. Retail experience? E-commerce development needs you.
7. Where's the local demand? Check job boards in your area. If cloud roles outnumber security roles 5 to 1, factor that in.

THE 2-WEEK TEST

Pick a domain. Spend 2 weeks learning the basics (free resources, YouTube, documentation). If you're still engaged at the end of week 2, you've found your lane. If you're bored, try the next one. Foundational skills (problem-solving, learning to learn, building things) transfer across domains. Nothing is wasted.

Going Deep: The One-Stack Strategy

Once you've picked your lane, go deep on one technology stack:

- Pick ONE cloud platform that matches your goals and local market (AWS, Azure, or GCP all work)
- Pick ONE programming language (Python is the safest bet across all five domains)
- Pick ONE tool in each critical category (Docker for containers, Terraform for IaC, GitHub Actions for CI/CD)

Master these. Build with them. Document what you build. Then expand.

The engineers who get hired fastest aren't the ones who know 50 tools. They're the ones who know 5 tools deeply and can explain why they chose each one.

YOUR ACTION STEP

1. Write down your chosen domain from the five options above.
2. Identify one transferable skill from your current or previous career that applies to this domain.
3. Find one free resource (YouTube course, documentation, Skill Builder) and bookmark it. You'll start this weekend.

Chapter 3: Certifications That Actually Matter

Let me tell you a story about how I got this wrong - and how getting it right changed everything.

When I was at NHS England, working in a data role that had nothing to do with cloud, I sat down and mapped out my own certification roadmap. Nobody told me to do it. I figured it out on my own. Not random certs. A sequence. Azure Fundamentals first. Then Azure Data Engineer. Then Azure Solutions Architect. Then Azure DevOps.

Four Azure certifications in six months. Each one built on the last - like stacking bricks instead of juggling them.

That foundation helped me land my role at AWS. Then I cleared three more AWS certifications in three months. Same principle: the right order makes the next cert easier.

Today I hold 9 cloud certifications. Not because I'm obsessed with collecting badges. Because each one served a purpose and opened a specific door.

The Certification Trap (And How to Avoid It)

Here's where most people go wrong: they collect certifications like Pokémon cards. Five or six certs, zero projects, no ability to explain what they've built. The interview won't go far.

A certification is like a driving licence. Having one doesn't make you a Formula 1 driver - but without it, you can't even get on the road.

> **THE FORMULA**
> 1 certification + 2 documented projects + ability to explain both = interview-ready. Missing any one of these three components significantly reduces your chances. Certifications get past resume filters. Projects prove you can build. Explanation proves you understand.

The AWS Certification Roadmap

As of early 2026, AWS offers 13 certifications across four tiers. The ML Specialty (MLS-C01) is being retired, so the active count is shifting to 12. Here's the current landscape:

Tier 1: Foundational ($100 USD each, no experience needed)

- AWS Cloud Practitioner (CLF-C02) - 4–6 weeks study. The front door. Covers cloud basics, economics, and security.
- AWS AI Practitioner (AIF-C01) - 3–5 weeks study. Hot in 2026. Shows AI awareness to employers.

Tier 2: Associate ($150 USD each, 1+ year recommended)

- Solutions Architect Associate (SAA-C03) - 8–12 weeks. Opens the most doors. Most popular AWS cert.
- Developer Associate (DVA-C02) - 6–10 weeks. For builders who code.
- Cloud Operations Engineer (SOA-C03) - 8–10 weeks. For ops-focused roles.
- Data Engineer Associate (DEA-C01) - 8–10 weeks. For data pipeline roles.
- Machine Learning Engineer Associate (MLA-C01) - 8–12 weeks. Replaces retiring ML Specialty.

Tier 3: Professional ($300 USD each, 2+ years recommended)

- Solutions Architect Professional (SAP-C02) - 10–16 weeks. The gold standard. Serious salary booster.
- DevOps Engineer Professional (DOP-C02) - 10–14 weeks. Hardest AWS exam.
- GenAI Developer Professional (AIP-C01) - 8–12 weeks. Brand new.

Tier 4: Specialty ($300 USD each, domain expertise recommended)

- Security Specialty (SCS-C02) - 10–14 weeks. For security-focused roles. Pairs brilliantly with CompTIA Security+.
- Advanced Networking Specialty (ANS-C01) - 10–14 weeks. Deep networking expertise for hybrid and multi-cloud architectures.
- Machine Learning Specialty (MLS-C01) - Retiring soon; being replaced by the ML Engineer Associate path. Only pursue if you need it immediately.

Recommended Paths by Career Goal

Solutions Architect Path

Cloud Practitioner → Solutions Architect Associate → Solutions Architect Professional

Best for: The broadest career path. Opens doors to SA, consulting, and architect roles.

DevOps Engineer Path

Cloud Practitioner → Developer Associate → Cloud Operations → DevOps Professional

Best for: Automation-focused engineers who love CI/CD, infrastructure, and reliability.

Cloud Security Path

Cloud Practitioner → Solutions Architect Associate → Security Specialty

Best for: Security-focused careers. CompTIA Security+ pairs well with this path.

AI/ML Engineer Path

AI Practitioner → Solutions Architect Associate → ML Engineer Associate → GenAI Developer Professional

Best for: The hottest path in 2026. Combines cloud fundamentals with AI specialisation.

How to Actually Study (Not Just Memorise)

The biggest mistake I see: studying to pass the exam instead of studying to understand the concepts. Here's the approach that works:

The 70/30 Rule

Spend 30% of your time watching tutorials and reading. Spend 70% actually doing things - building labs, breaking things, deploying services, writing code.

This is counterintuitive. Most people spend 90% watching and 10% doing. Then they pass the exam but can't explain their own projects in an interview.

Set Project-Based Goals

Don't say "I'll study AWS this month." Say "I'll deploy a real-world application on AWS in 2 weeks." Projects force understanding in a way that flashcards never will.

Set a Deadline

Give yourself 4–6 weeks per certification. Without a deadline, studying expands infinitely. Book the exam before you feel ready. The deadline creates urgency that actually helps you focus.

Learn Just Enough to Move Forward

You don't need to memorise every AWS service. There are 200+ of them. Learn the core ones deeply (S3, EC2, Lambda, IAM, VPC, CloudFormation) and know when to look up the rest.

The Salary Impact of Certifications

Let's talk ROI:

- AWS-certified professionals earn an average salary of $113,000 - over $10,000 more annually than non-certified peers (Global Knowledge)
- 80%+ of certified professionals report salary increases after certification
- Solutions Architect Associate holders earn an average of $129,000 in the US
- The ROI on a $150 certification that takes 8 weeks to earn is extraordinary

That's not a rounding error. A single certification can pay for itself within weeks of starting a new role.

YOUR ACTION STEP

1. Based on your chosen domain (Chapter 2), identify the first certification you'll pursue.
2. Set a target exam date 6 weeks from today. Write it somewhere visible.
3. Block 30 minutes a day in your calendar for study. Non-negotiable. Consistency beats intensity.

Chapter 4: Build Things That Matter

My first portfolio was embarrassing. Three tutorial projects, no READMEs, and a GitHub profile that screamed "I followed along but never built anything real." I know this because a hiring manager told me - to my face, in an interview - that my portfolio looked like everyone else's.

That feedback stung. But it was the best career advice I ever got.

I've now reviewed hundreds of portfolios, and I see the same pattern everywhere: people build tutorial projects that everyone else has already built, put them on GitHub with no documentation, and wonder why hiring managers don't respond. I was that person. Here's what I learned the hard way.

Why 500 People Can Build the Same Thing

AI made everything easier to do. Which means doing average things is no longer impressive. 500 people can build the same chatbot, the same to-do app, the same landing page.

The outliers? They do the uncomfortable, inconvenient work everyone else avoids:

- They solve real problems for real people (not imaginary problems for a portfolio)
- They document why they made every decision (not just what they built)
- They measure impact (not just list features)
- They explain trade-offs (not just describe the tech stack)

The Impact-First Project Strategy

Stop showcasing skills. Start showcasing impact.

Here's how: ask 3 people in your network - friends, family, colleagues - this question:

> *"What's the most annoying, repetitive part of your job?"*

Then build a solution for one of them. Not a perfect product. A working solution that saves them time, reduces errors, or automates something tedious.

Compare these two portfolio descriptions:

❌ *"I built a chatbot using Python and the OpenAI API."*

✅ *"I built a document scanning tool that reduced manual data entry by 6 hours per week for a real estate agency. 53 users, 12 paying customers."*

Same skill level. Completely different story. The second one gets the interview.

Projects That Impress Hiring Managers

Based on what I see coaching people through mock interviews and on LinkedIn, here are the types of projects that stand out:

> **Tip:** There are excellent project walkthroughs on YouTube you can follow step by step. AWS also offers free guided workshops at builder.aws.com/build/workshops - these are real-world scenarios built by AWS engineers and perfect for building portfolio-ready projects.

1. Cloud Resume Challenge

Build your resume as a website hosted on AWS: S3 for static hosting, CloudFront for CDN, Lambda for a visitor counter, DynamoDB for storage, CI/CD for automated deployment. This is a classic and well-respected project.

2. Containerised Web App with CI/CD

Dockerise an application, deploy to ECS or EKS, put an ALB in front. Add a GitHub Actions pipeline that auto-deploys on push. Shows end-to-end DevOps thinking.

3. Serverless API

Lambda + API Gateway + DynamoDB. Build a working REST API with no servers. Every modern team uses serverless - showing you can build with it is gold.

4. Infrastructure as Code Portfolio

Write your entire AWS setup in Terraform or CloudFormation. One command to deploy. One command to destroy. This demonstrates production mindset.

5. AI-Powered Application

Deploy a containerised AI chatbot or build a RAG (Retrieval-Augmented Generation) application. Shows you understand both cloud infrastructure AND AI integration.

The BUILD Documentation Framework

Building isn't enough. You need to document what you built and why. Here's my framework:

Background - What problem does this solve? Why does it matter?

Understanding - What was your thought process and approach?

Implementation - How did you build it? What technologies and why?

Lessons Learned - What went wrong? What would you do differently?

Deliverables - What's the final result? Show screenshots, metrics, links.

Write this up on GitHub, Medium, or LinkedIn. Your project write-ups become your portfolio.

> **THE GOLDEN RULE**
> One well-documented project beats five half-finished repos. Hiring managers spend 90 seconds on your GitHub. If they can't understand what you built and why within 90 seconds, they move on.

This is where I see the most people get stuck. They know they should build projects, but they don't know which ones to pick, how deep to go, or how to write about them. If that's you, I offer limited 1:1 coaching sessions for readers who want personalised guidance. You can book a session at mentorship.sholastechnotes.com.

The Depth Over Breadth Principle

I learned this from coaching dozens of career pivoters through DevOps interviews: the thing that separates candidates who get hired from candidates who don't is depth of understanding.

Here's the test: when a hiring manager changes the parameters of a question, can you adapt? Or do you freeze?

Candidates who freeze have only studied in a course. They memorised the answer to “What is S3?” but can’t explain what happens when you change the access policy. They know the acronym for CI/CD but can’t design a pipeline for a specific use case.

Depth means understanding WHY, not just WHAT. Why did you choose ECS over EKS? Why Terraform instead of CloudFormation? Why a NoSQL database instead of SQL for this specific workload?

Level Up Any Existing Project

Already have a basic project? Here’s how to make it interview-ready:

- Add observability - CloudWatch metrics, dashboards, alerting
- Document decisions - A README explaining why you chose each technology
- Add CI/CD - Automated testing and deployment pipeline
- Implement security - IAM roles, least privilege, encryption
- Handle failure - Health checks, auto-recovery, graceful degradation

YOUR ACTION STEP

1. Ask 2–3 people in your network: “What’s the most repetitive or annoying part of your work?”
2. Pick one problem and sketch a basic solution (even on paper). What technology would you use?
3. Create a GitHub repository for your first project. Write the Background and Understanding sections of the BUILD framework in the README before you write a single line of code.

Chapter 5: Your AI Toolkit (The 2026 Edge)

Here's what I've noticed most engineers get wrong about AI coding tools: they pick one and go all in. Cursor only. GitHub Copilot only. ChatGPT only.

That's like a carpenter who only owns a hammer.

The best engineers I work with don't have "a tool." They have a toolkit. Different tools for different problems - like choosing between a screwdriver and a drill. Both work on screws, but you wouldn't use the same one for every job.

Last Tuesday, I shipped code using three different AI tools in a single day: Cursor for refactoring a complex codebase, Claude Code for multi-file changes from the terminal, and Codex for a background task I assigned and walked away from.

Each tool did what it does best. No single tool could have done all three.

It is worth being honest about what AI means for the job market. Research shows that AI-exposed roles are seeing slower hiring growth for the youngest workers, and tools now exist that can automate tasks that once required a junior engineer. But the data also shows that AI adoption is driving increased demand for cloud infrastructure, security, and architecture skills. The professionals who thrive will be those who use AI tools to amplify their capabilities rather than compete with them on speed alone. Every tool in this chapter is chosen because it makes you more effective at the work that still requires human judgement: understanding business context, making architectural decisions, and communicating solutions to stakeholders.

The Five Tools You Should Know

I've tested every major AI coding tool on real work - not toy projects, not tutorials. Here are the five I actually reach for. I'm giving you the honest version: what each one actually does well, what it costs, and who it's for.

1. Kiro - The Full IDE Experience

This is the tool I reach for when I'm starting something from scratch. Kiro is an agentic IDE listed on AWS's website (aws.amazon.com/ai/agentic-ai) and built with AWS technology. You describe what you want to build and Kiro generates requirements, design documents, and

tasks, then executes them. The property-based testing is wild - it automatically tries to break your code before you ship it.

- Best for: Full feature development from spec to ship
- Free tier: 50 interactions/month (+500 bonus)
- Paid: From $19/month

2. Cursor - AI-Native Code Editor

My daily driver for refactoring. Cursor understands your entire codebase, not just the open file. I'll have Background Agents running tasks while I keep coding in another tab. BugBot auto-reviews PRs before my teammates even see them.

- Best for: Day-to-day coding, refactoring, large codebases
- Free tier: 2,000 completions + 50 premium requests/month
- Paid: From $20/month (free for students with .edu)

3. Claude Code - Terminal-First Agent

If you live in the terminal, this is your tool. Claude Code reads your entire repo, plans multi-file changes, runs tests, handles git workflows - all via natural language. I use it when I want full CLI control without the overhead of opening an IDE.

- Best for: Engineers who live in the terminal. Full CLI control, no IDE overhead.
- Paid: From $20/month (Pro plan required)

4. Codex - Async Delegation

This is the one I use when I want to hand off a task and walk away. Assign it a feature, bug fix, or refactor - come back to a ready pull request. Each task runs in its own isolated sandbox, so it can't break anything.

- Best for: Delegating well-defined tasks. No babysitting required.
- Paid: $20/month (ChatGPT Plus)

5. AWS Strands Agents SDK - Build Your Own

This is for when you're ready to go from using AI agents to building them. Open-source Python framework - the same SDK powering Amazon Q Developer internally. Over 14 million downloads on PyPI. This isn't experimental. It's production-grade.

- Best for: Engineers ready to go from using agents to building their own
- Cost: 100% free and open-source (Apache 2.0)

The Decision Matrix

Not sure which tool to start with? Ask yourself:

- "I want a full IDE experience" → Kiro
- "I already use VS Code daily" → Cursor
- "I live in the terminal" → Claude Code
- "I want to delegate and walk away" → Codex
- "I want to build my own AI agents" → Strands SDK
- "I'm on a tight budget" → Kiro or Cursor free tiers

Context Engineering: The Real Skill

Here's a secret most people miss: the tool doesn't matter as much as how you use it.

Context engineering is the skill of getting AI to deliver exactly what you want on the first try - not through endless back-and-forth prompting, but by providing comprehensive context upfront.

Instead of typing "build me a login page," a context engineer provides:

- The project's file structure
- The architecture and tech stack
- Testing frameworks in use
- Specific requirements and edge cases
- Examples of similar work in the codebase

That 2–5 page prompt saves hours of iteration. The quality of your output is directly proportional to the quality of your input.

Your Minimum AI Awareness Stack

Even if you're not in an AI role, every tech professional in 2026 should understand:

- What an LLM API is (OpenAI, Anthropic, AWS Bedrock) - and how to call one

- How to deploy an AI-powered application (backend + cloud + container)
- GPU vs CPU workloads - why AI apps cost more to run
- Model inference vs training - resource planning and cost implications
- Vector databases - why AI apps use them and how they're deployed

You don't need to be an AI researcher. You need to be AI-literate. That's the difference between getting hired and getting overlooked.

YOUR ACTION STEP

1. Pick ONE AI coding tool from the five above (match it to your workflow).
2. Spend 2 hours this weekend using it on a real task (not a tutorial).
3. Write a short LinkedIn post about what you learned. "This weekend I tried [tool]. Here's what surprised me." That one post starts your AI visibility.

Want the full side-by-side comparison of all five tools - pricing, features, and which one fits your workflow? Grab the free AI Coding Toolkit at sholastechnotes.com/ai-coding-toolkit.

Chapter 6: The Job Search System

Most people search for jobs. The people who get hired run a system.

I'm going to give you the exact system I've refined through my own 60+ rejection journey and through helping many people land their first tech role. It's not about applying to more jobs. It's about applying smarter.

Your CV: Impact Over Activity

Remember the "So what?" framework from my introduction? This is where it matters most.

Every single line on your CV should answer the question: "So what? What changed because of this?"

The Before and After

❌ **Before:** *"Managed cloud infrastructure and performed deployments."*

✅ **After:** *"Migrated 40+ microservices to AWS ECS, reducing deployment time by 73% and cutting infrastructure costs by £18K annually."*

❌ **Before:** *"Built a serverless application."*

✅ **After:** *"Built a serverless processing pipeline handling 50K+ daily events, reducing error rates from 12% to 0.3% with automated retry logic."*

Notice the pattern: numbers, outcomes, impact. Not activities, not responsibilities, not job descriptions.

CV Best Practices for 2026

The rules have changed. 99% of Fortune 500 companies use applicant tracking systems, and 83% of large employers plan to incorporate AI-driven screening by end of 2025 (Resume Genius, 2024). Here's what works:

- One page maximum. Recruiters spend 7 seconds on initial scan. Respect their time.
- PDF format always. Formatting survives every system.

- No graphics, images, or fancy layouts. ATS systems can't parse them. Clean, text-based formatting wins.
- Tailor every application. Match the language in the job description. If they say "CI/CD," your CV says "CI/CD" - not "continuous integration."
- Lead with your most relevant experience. If you're switching careers, lead with projects and certifications, not your previous unrelated role.
- Include a brief professional summary (2–3 lines) at the top that tells your story in seconds.

PRO TIP: USE AI TO REVIEW YOUR CV

Paste the job description and your CV into Claude or ChatGPT. Ask: "As a hiring manager for this role, what's missing from this CV? What would make you say yes to an interview?" The feedback is brutally honest and incredibly useful.

FREE TOOL: RESUMERADAR

I built ResumeRadar (resumeradar.sholastechnotes.com) - a free CV scanner that analyses your CV against real hiring criteria. Over 1,800 people have used it since launch. Upload your CV, get instant feedback on what's working and what's not. It's free because everyone deserves a fair shot at getting past the screening stage.

Real People, Real Transitions

The frameworks in this book aren't theoretical. Here are two people who followed this exact playbook:

From Help Desk to Solutions Architect (18 months). A mentee of mine started in first-line IT support, earning £28,000. No cloud experience. No certifications. They followed the one-stack strategy from Chapter 2 - picked AWS, committed to it. Earned Cloud Practitioner in 4 weeks, Solutions Architect Associate in 10. Built two portfolio projects (a serverless API and a containerised web app with CI/CD). Started posting their learning journey on LinkedIn - one of their posts caught a recruiter's eye. Within 18 months of starting their first certification, they were working as an Associate Solutions Architect earning £65,000. The system works if you work it.

From Care Support to Senior Data Analyst (12 months). Another person I coached came from a non-technical care support role. They chose the Data & AI lane, taught themselves Python and SQL in evenings, and used the "So what?" framework to rewrite their CV around measurable outcomes from their previous role (they'd been tracking patient metrics without

realising that was data analysis). They earned an Azure Data Fundamentals cert, built a data pipeline project, and landed a Senior Analyst role within a year. Their non-traditional background became their advantage - they understood the business context that pure technologists often miss.

From NHS Band 2 to Band 7 - A Career Transformation

One of the most powerful transitions I've guided was someone working in an NHS Band 2 role - entry-level, limited progression, feeling stuck. They followed the exact framework in this book: identified cloud as their domain, earned AWS Cloud Practitioner in 6 weeks, built a serverless portfolio project, and rewrote their CV using the "So what?" method. Within 8 months, they'd moved into a Band 7 technical role - a jump of five pay bands. The key wasn't just technical skills; it was positioning their existing healthcare knowledge as a strategic advantage in health-tech.

What People Are Saying

> *"I reached out to Shola after deciding to take the leap and try to break into tech and he has been nothing but a supportive mentor. He's always encouraging me, giving me genuine advice and recommendations. I really do appreciate it."*
>
> **- Ejemenare**
>
> *"If anyone is thinking about transitioning into tech or simply needs a guide to navigate the tech landscape, I wholeheartedly recommend Shola. His expertise, patience, and genuine desire to help others succeed make him a truly exceptional mentor."*
>
> **- Anuoluwapo**
>
> *"As an aspiring cloud engineer, reaching out to Shola was one of the best decisions I've made. His depth of knowledge and genuine desire to help others succeed makes him an outstanding professional."*
>
> **- Eddie**
>
> *"Shola's guidance helped me land my first role in cloud engineering. His ability to break down complex concepts and provide clear, actionable steps made all the difference in my journey."*
>
> **- Osinachi**

Networking: The 85% Rule

Industry surveys consistently show that referrals and networking fill far more roles than cold applications. LinkedIn's own data puts the figure at over 70%. Let that sink in.

You can have the best CV in the world, but if you're only doing cold applications, you're fighting over 15% of available positions while 85% are being filled through conversations you're not part of.

The Second-Order Connection Strategy

Don't reach out cold to strangers. Use second-order connections - people who know people you know.

1. Look at the company you want to work for on LinkedIn.
2. Check who in your network is connected to someone there.
3. Ask your mutual contact for an introduction. Keep it simple: "Hey, I noticed you're connected to [name] at [company]. Would you be comfortable introducing us? I'd love to learn about their experience in [role]."
4. When introduced, ask for advice, not a referral. "I'm interested in transitioning into cloud engineering. What do you wish you'd known when you started?"
5. After building rapport, ask about opportunities. Most people are happy to refer you - companies pay referral bonuses of £1,000–£5,000.

LinkedIn: Your Secret Weapon

LinkedIn isn't optional in 2026. It's where 60–70% of tech hiring conversations start.

Optimise Your Profile

- Headline: State what you ARE, not "aspiring." "Cloud Engineer | AWS Certified | Building in Public" tells recruiters exactly who you are.
- About section: Tell your story in 3–4 paragraphs. Where you started, where you are, where you're going. Include keywords from your target job descriptions.
- Featured section: Pin your best project write-ups and certifications.
- Turn on "Open to Work": LinkedIn's own numbers show a 40% increase in recruiter contact. There's no reason not to.

The Build-in-Public Strategy

Post about your learning journey. Don't need to be an expert. "Today I learned what S3 is and why every company uses it" is a perfectly valid first post.

Share your projects. Every project from Chapter 4 gets a LinkedIn post explaining what you built and why.

Engage before you create. Comment thoughtfully on 5–10 posts daily from cloud creators. Get noticed before you start posting.

I got my role at AWS partly through LinkedIn visibility. A recruiter found me through a post about cloud concepts. Your voice takes you places your CV never will.

Applying: Quality Over Quantity

Stop applying to 100 jobs with the same generic CV. Start applying to 5–10 jobs per day, each with a tailored application.

- Read the job description word by word. Match your CV language to theirs.
- Research the company. Know their products, their tech stack, their challenges.
- Write a one-paragraph cover note (not a full cover letter) explaining why this specific role at this specific company interests you.
- Follow up after 5–7 days with a polite message to the recruiter or hiring manager.

Interview Preparation

The Interview Structure in 2026

Most tech interviews follow this pattern:

6. Phone screen (30 mins) - Recruiter checks culture fit and basic qualifications
7. Technical screen (45–60 mins) - Coding or system design, often with AI tools allowed
8. System design round (60 mins) - Design a scalable system. Whiteboard or virtual.
9. Behavioural round (45–60 mins) - STAR stories about collaboration, conflict, leadership
10. Culture fit / values round (30–45 mins) - Do you align with the company's principles?

Behavioural Interviews: The STAR Framework

Prepare 8 stories using STAR format:

- Situation: Set the scene. What was happening?
- Task: What was your specific responsibility?
- Action: What did YOU do? (Not the team. You.)
- Result: What was the measurable outcome?

Cover these themes: a time you failed, a time you led, a time you disagreed, a time you delivered under pressure, a time you helped a teammate, a time you learned something quickly, a time you dealt with ambiguity, and a time you made a tough decision.

Technical Interviews: The New Rules

Here's what's changed in 2026: a growing number of companies now allow AI tools during technical interviews. Copilot, Cursor, and Claude are becoming part of the process - because that's how many engineering teams work in practice.

Translation: coding syntax matters less. System design thinking matters more. They want to see how you approach a problem, how you communicate your reasoning, and how you use tools to solve it. But here's the catch - you still need to know the fundamentals. AI tools can write code, but they can't think for you. If you don't understand networking, security basics, how databases work, or how systems talk to each other, you'll get caught out. The candidates who thrive are the ones who pair strong fundamentals with smart tool usage.

One thing I've learned from coaching people: reading about STAR stories and actually practising them out loud are completely different skills. If you can, find a friend, a peer, or a mentor to run mock interviews with you before the real thing. Practise until your stories flow naturally without notes.

YOUR ACTION STEP

1. Rewrite the top 3 bullet points on your CV using the "So what?" framework. Add numbers and outcomes.
2. Identify 5 target companies. Find at least one second-order connection at each company on LinkedIn.
3. Write one STAR story for a behavioural interview. Practise saying it out loud until it flows naturally (record yourself, it helps).

Chapter 7: Build Your Brand Before You Need It

This is the chapter most people skip. It's also the one that accelerates everything else.

Personal branding isn't about becoming an influencer. It's about making yourself findable and credible when opportunities come looking.

I built a newsletter - Shola's Tech Notes - that now reaches over 1,000 subscribers. I have 24,000+ followers on LinkedIn. I built a free CV scanner (ResumeRadar) that 1,800+ people used in its first weeks. I didn't start with any of that. I started with a single post about cloud concepts that got 12 likes.

But here's what happened: recruiters started finding me. Conference organisers started inviting me. Hiring managers started recognising my name before interviews.

Your personal brand is compound interest for your career. Every post, every project write-up, every comment you leave builds a reputation that works for you while you sleep.

The LinkedIn Playbook

Week 1: Optimise and Engage

- Update your headline to reflect who you are (not "aspiring" - you ARE)
- Write a 3-paragraph About section: where you started, where you are, where you're going
- Connect with 10 people in your target domain daily
- Leave 5 thoughtful comments daily on posts from creators in your space

Week 2: Start Creating

- Post your first learning update: "Day 1 of my [cloud/DevOps/security] journey. Here's what I learned today."
- Share a key insight from your certification study
- Post a before/after of something you built or learned

Week 3+: Build the Habit

- Post 3–5 times per week

- Rotate between: learning updates, project showcases, industry insights, and personal stories
- Every project from Chapter 4 becomes at least one LinkedIn post
- Engage with comments on your posts - community builds faster than content

> **PERSONAL BRAND PRO TIP**
> You don't need to be an expert to start posting. You need to be learning. "Today I learned what S3 is" is a perfectly valid first post. People love following someone's journey. It's authentic and refreshing in a world of polished content.

The Founder Mindset

Here's something controversial: you don't need to wait for someone to hire you to prove you can do the job.

Form an LLC. It takes less than an hour. Open a business bank account. Get a Stripe account. Build a small product - a tool, a template, a service. Get ONE person to pay you £1 for it.

Congratulations - you're now a founder with revenue. Not an "aspiring developer." A builder with customers.

Compare these two introductions:

❌ *"I'm looking for junior developer roles. I've completed several online courses."*

✅ *"I'm the founder of Pet Match Pro - 53 users, 12 paying customers, £200/month revenue. I built it with Python, AWS Lambda, and DynamoDB."*

Same person. Completely different story. The second version opens every door.

Building in Public

The most powerful career accelerator I've seen: document your journey publicly.

- Share what you're learning as you learn it
- Post your project progress - including the messy parts

- Write about mistakes you made and what you'd do differently
- Celebrate small wins publicly

Why this works: it creates visibility, builds a track record, attracts opportunities, and forces you to learn more deeply (because you have to explain things clearly to others).

I've seen people get job offers from LinkedIn posts about their learning journey. Visibility equals luck.

Communities That Accelerate Your Career

Don't go alone. Join communities where people are on the same journey:

- LinkedIn cloud/DevOps/AI groups
- Discord communities (many AWS and tech communities have active ones)
- Local meetups via meetup.com
- AWS re/Start and community events
- Tech Twitter / X communities

The people you connect with during your transition will shape your entire career trajectory. Some of my best opportunities came from people I met in communities, not job boards.

YOUR ACTION STEP

1. Update your LinkedIn headline today. Remove "aspiring" and replace with what you ARE doing.
2. Write your first learning post: "This week I learned ___ about [your domain]. Here's what surprised me."
3. Join one online community in your chosen domain and introduce yourself.

Chapter 8: The 90-Day Sprint Plan

Everything in this book comes together here. This is your week-by-week roadmap from where you are right now to interview-ready.

I built this framework based on my own journey and the patterns I've seen in hundreds of successful career transitions. It works because it's specific, time-boxed, and focuses on the actions that actually move the needle.

Your companion tool for this chapter: The 90-Day Tech Career Playbook Notion template included with this book is your execution system for everything below. It has 7 interlinked databases - Certifications Tracker, Portfolio Projects, Networking CRM, Job Applications, Interview Prep, Sprint Planner, and Wins Journal - plus networking email templates you can copy and send.

How to access your Notion template:

1. Go to Access Your 90-Day Career Playbook (Notion Template)

2. Click "Duplicate" in the top-right corner to copy it to your Notion workspace (free account works)

3. Start filling in the databases as you work through each chapter - your progress becomes your portfolio evidence

> **THE CONSISTENCY PRINCIPLE**
> 1.2 focused hours daily beats 10-hour weekend sprints. Every time. Consistency compounds. Weekend warriors burn out. Set a daily block of time - even 45 minutes before work - and protect it fiercely.

Days 1–30: Build the Foundation

This month is about choosing your path, setting up your learning environment, and earning your first credential.

Week 1: Choose and Commit

- Choose your domain (Chapter 2) and write it down
- Identify your first certification target and book the exam date (Chapter 3)
- Set up your learning environment (AWS free tier, IDE, GitHub account)

- Block 1–2 hours daily in your calendar for focused study
- Update your LinkedIn headline and write your About section

Week 2: Learn the Fundamentals

- Start your certification study (70% hands-on, 30% watching/reading)
- Complete your first hands-on lab or deployment
- Connect with 10 people in your target domain on LinkedIn daily
- Post your first learning update: “Day 7 of my [domain] journey”

Week 3: Deepen Understanding

- Continue certification study with increasing hands-on practice
- Take your first practice exam - identify weak areas
- Start engaging in one online community (Discord, LinkedIn group, meetup)
- Comment thoughtfully on 5 posts daily from people in your space

Week 4: First Milestone

- Sit your certification exam (if on 4-week track) or continue studying
- Start sketching your first project idea (Chapter 4)
- Write a LinkedIn post sharing what you’ve learned this month
- Review your progress and adjust your daily routine as needed

Days 31–60: Build and Deploy

This month is about turning knowledge into proof. Certifications opened the door. Projects prove you can walk through it.

Week 5: Start Building

- Begin your first project (pick from Chapter 4)
- Set up a GitHub repository with a proper README (BUILD framework)
- Create the project’s Background and Understanding sections first
- If you haven’t passed your cert yet, schedule the exam this week

Week 6: Build in Public

- Continue building your project - commit daily
- Share your progress on LinkedIn (screenshots, architecture diagrams, challenges)
- Add CI/CD to your project (even a basic GitHub Actions pipeline)
- Start asking people in your network about their work pain points

Week 7: Polish and Document

- Complete your first project and deploy it live
- Write the full BUILD documentation in your README
- Create a LinkedIn project showcase post
- Start your second project (aim for something that solves a real problem)

Week 8: Level Up

- Add observability, security, and documentation to Project 1
- Continue building Project 2
- Start exploring AI tools (Chapter 5) in your daily workflow
- Write a “Lessons Learned” post about what you built

Days 61–90: Search and Land

This month is about converting everything you’ve built into interviews and offers.

Week 9: Prepare Your Arsenal

- Rewrite your CV using the “So what?” framework (Chapter 6)
- Get feedback on your CV from peers, mentors, or AI
- Prepare 8 STAR stories for behavioural interviews
- Identify 20 target companies and find connections at each

Week 10: Launch Applications

- Start applying: 5–10 high-quality, tailored applications per day
- Send personalised connection requests to people at target companies
- Practice technical interviews (record yourself, use mock interview platforms)

- Continue posting on LinkedIn - visibility compounds exactly when you need it

Week 11: Interview Sprint

- Follow up on all applications from Week 10
- Practice system design questions: define Requirements, Estimate scale, design the Schema, plan the API, draw the High-level architecture, dive into Detailed design, Evaluate trade-offs, and Discuss extensions
- Do at least 2 mock interviews with friends or online partners
- Refine your “Tell me about yourself” story until it’s effortless

Week 12: Close and Convert

- Continue the application cycle - 5–10 daily
- Follow up with thank-you notes after every interview
- Evaluate offers against: salary, growth, learning opportunities, culture
- Negotiate. Always negotiate. Research market rates from this book’s salary data.

After You Land the Role

The playbook doesn’t stop when you get hired. Your first 90 days in a new role matter just as much:

Days 1–30: Learn and Integrate

- Understand the codebase, team dynamics, and company culture
- Ask questions relentlessly - nobody expects you to know everything on day one
- Set up your development environment and get comfortable with the tools
- Build relationships with your team - the people you connect with in month 1 shape your career

Days 31–60: Own and Contribute

- Ship your first feature or fix independently
- Start documenting what you learn (internal wikis, personal notes)
- Identify a small problem nobody has fixed and solve it

- Continue learning - the job is where real learning begins

Days 61–90: Impact and Grow

- Take ownership of larger tasks
- Share what you've learned with newer team members
- Start your next certification (your employer might pay for it)
- Continue building in public - your brand keeps compounding

YOUR ACTION STEP

1. Print or save this 90-day plan. Pin it where you'll see it daily.
2. Set a calendar reminder for Day 30, Day 60, and Day 90 to review your progress.
3. Start Day 1 this weekend. Not "next Monday." Not "when I feel ready." This weekend.

Chapter 9: The Mindset That Gets You Hired

We've covered the strategy, the frameworks, the tools, and the 90-day plan. But none of it matters without the right mindset.

Because here's the truth: the biggest obstacle between you and a tech career isn't your background, your qualifications, or the job market. It's the voice in your head that says "I'm not ready yet."

Lesson 1: Rejection Is Data, Not Failure

I was rejected from 60+ jobs before landing AWS. Sixty. Each rejection stung. But each one also taught me something - about my CV, my interview approach, my positioning, my communication.

The people who break into tech aren't the ones who never get rejected. They're the ones who treat every "no" as information and use it to improve for the next "yes."

Lesson 2: Stop Waiting for Permission

You don't need a degree to be a developer. You don't need 5 years of experience to be a cloud engineer. You don't need someone to hire you to prove you can do the work.

Build something. Deploy it. Get a user. Get a customer. Write about it. Share it.

Stop saying "aspiring." Start saying "I built this."

Lesson 3: The Inconvenience Is Your Competitive Advantage

Everything that feels uncomfortable - posting your first LinkedIn update, reaching out to a stranger for a coffee chat, applying to a job you don't feel "qualified" for, sharing a project that isn't perfect - that's the work most people avoid.

Which means doing it puts you ahead of everyone who doesn't.

The outliers in any field aren't the smartest. They're the ones willing to be uncomfortable more often than everyone else.

Lesson 4: Your Background Is Your Moat

My degree had nothing to do with technology. For years, I thought that was a disadvantage.

It wasn't. It was my superpower.

My non-traditional background means I explain complex technical concepts in ways that normal people understand. I communicate with business stakeholders effortlessly. I bridge the gap between technical teams and everyone else.

Whatever you did before tech - teaching, healthcare, finance, retail, military, parenting - you bring skills that pure technologists don't have. Communication. Empathy. Domain expertise. Problem-solving under pressure.

Don't hide your background. Lead with it.

Lesson 5: Consistency Beats Talent

The people who land tech jobs in 90 days aren't the most talented. They're the most consistent.

1.2 focused hours daily. Every day. No exceptions.

That's 84 hours of focused effort in 10 weeks. Enough to earn a certification, build two projects, and apply to 100+ jobs.

Most people won't do it. Which is exactly why you will.

Your Template Pack

Throughout this book, I've mentioned networking scripts, CV frameworks, and interview strategies. Here they are - 10 ready-to-use templates you can copy, personalise, and send today. No fluff. Just the exact words that work.

1. LinkedIn Connection Request (Hiring Manager)

Use this when connecting with hiring managers at companies on your target list. Personalised connection requests have a significantly higher acceptance rate than generic ones.

> Hi [First Name], I've been following [Company Name]'s work in [specific area - cloud migration, DevOps, etc.] and your approach to [specific thing they do or post about] resonated with me. I'm transitioning into [target role] and would value learning from your perspective on [specific challenge relevant to their work]. Would you be open to connecting?

2. LinkedIn Connection Request (Industry Peer)

For peers in your target space - engineers, architects, product managers in your niche. This opens doors for future collaboration and mutual learning.

> Hey [First Name], I came across your recent work on [specific project/post/article] and it aligns perfectly with what I'm building in [your area]. I'm [current role] and have been focusing on [relevant skill/domain]. Would love to connect and potentially collaborate down the line.

3. Informational Interview Request

Send this after connecting (or via email if no LinkedIn connection). Keep it tight: you're asking for 15 minutes, not an hour. Frame it as learning, not job-hunting.

> Hi [First Name], I'm currently transitioning into [target role] and I've been impressed by your background in [their specialty]. I'd value 15 minutes of your time to ask three specific questions: (1) what skills matter most day-to-day, (2) what does a typical week look like, and (3) what would you have done differently starting out. Would you have a quick slot in the next two weeks? Happy to work around your schedule.

4. Coffee Chat Follow-Up

Send within 24 hours of your chat. Reference something specific from your conversation - this tells them you were actually listening.

> Hi [First Name], thanks so much for taking the time yesterday - your point about [specific insight they shared] has shifted how I'm thinking about [relevant area]. I've already started [specific action based on their advice], and I'll keep you posted on how it goes. Really appreciated your honesty about [challenge they mentioned]. If you ever need a sounding board on [area where you can add value], my door is open.

5. CV "So What?" Bullet Formula

Every CV bullet should answer the "so what?" test: why does this matter to the hiring manager? Use this formula: [Action Verb] + [What You Did] + [Measurable Result] + [Business Impact].

Before (weak): Responsible for server maintenance.

After (strong): Automated server patch management using Terraform, reducing deployment time by 65% and closing security gaps that had been flagged in two consecutive compliance audits.

Before (weak): Worked on cloud migration project.

After (strong): Led AWS cloud migration for legacy monolithic application (15 microservices, 200+ users), delivering 40% faster query response times and reducing monthly infrastructure costs by 40% through reserved instances and auto-scaling.

Before (weak): Mentored junior developers.

After (strong): Mentored 8 junior engineers through structured Python and AWS certification programme; 7 promoted within 18 months, 5 to senior roles, with team retention improving significantly year-over-year.

CV Template: Cloud/Tech Career Changer (1-Page Format)

Here's a clean 1-page CV structure optimised for career changers entering cloud and tech roles. The key: lead with skills and projects, not with your previous job title. Let your capabilities speak before your history does.

CV TEMPLATE - ADAPT TO YOUR SITUATION

[YOUR NAME]

[City, UK] | [email] | [LinkedIn URL] | [GitHub URL]

PROFESSIONAL SUMMARY

Cloud-focused [target role] with [X] certifications and hands-on project experience in [AWS/Azure/GCP]. Transitioning from [previous field] with transferable skills in [problem-solving / data analysis / stakeholder management / process improvement]. Built [brief project highlight] using [key technologies].

CERTIFICATIONS

AWS Solutions Architect Associate (SAA-C03) - [Month Year]
AWS Cloud Practitioner (CLF-C02) - [Month Year]
[Additional relevant cert] - [Month Year]

PORTFOLIO PROJECTS

[Project Name] - [1-line description with tech stack and measurable outcome]

[Project Name] - [1-line description with tech stack and measurable outcome]

TECHNICAL SKILLS

Cloud: AWS (EC2, Lambda, S3, RDS, CloudFormation) | Languages: Python, SQL, Bash | Tools: Terraform, Docker, Git | Practices: CI/CD, IaC, Agile

PROFESSIONAL EXPERIENCE

[Most Recent Role] - [Company] - [Dates]

- [“So What?” bullet: Action + Result + Business Impact]
- [“So What?” bullet: Action + Result + Business Impact]

EDUCATION

[Degree] - [University] - [Year]

Notice: certifications and projects come BEFORE work experience. For career changers, your new skills matter more than your old title. Keep it to one page - hiring managers spend 7 seconds on a first scan. Make those seconds count.

6. STAR Interview Answer Template

Use the STAR framework for behavioural questions (“Tell me about a time when...”). This structure keeps you on track and gives the interviewer concrete proof of your abilities.

Situation (1–2 sentences): Set the scene. Who was involved? What was the challenge or deadline?

Task: What was YOUR specific responsibility? Don’t say “we fixed it” - say “I owned...”

Action (the longest part): Walk through the steps you took. Use specific tools, frameworks, and decisions. Show your thinking.

Result: What happened? Use numbers if possible. Impact on the team, customer, or business. Finish with: “And I learned [one insight].”

Interview Question Bank: 10 Real Questions with STAR Answers

Here are 10 real interview questions you’ll face in cloud and tech roles, with example STAR-format answers you can adapt to your own experience. These are based on patterns I’ve seen across dozens of mock interview sessions with my mentees.

1. “Tell me about a time you solved a complex technical problem.” In my previous role, our team’s deployment pipeline was failing 30% of the time, blocking releases for 3 days each sprint. I took ownership of diagnosing the root cause, traced it to a race condition in our containerised build process, redesigned the pipeline with sequential dependency checks, and

reduced failure rate to under 2%. This unblocked the team and saved roughly one sprint day per cycle.

2. "How do you stay current with cloud technology?" I dedicate 45 minutes each morning before work to structured learning. Last quarter, I completed the AWS Solutions Architect Associate certification while also following the AWS What's New feed and building a side project using Lambda and DynamoDB. I share my learnings on LinkedIn, which has led to conversations with other engineers and keeps me accountable to a learning rhythm.

3. "Describe a situation where you had to learn something quickly." When our team was asked to migrate a legacy application to Kubernetes with a 6-week deadline, I had no container orchestration experience. I dedicated evenings to the CKA curriculum, built a local cluster the first week, and paired with a senior engineer on our staging deployment. By week 4, I was leading the production migration. We delivered on time, and I earned my CKA certification shortly after.

4. "Tell me about a time you disagreed with a teammate." A colleague wanted to build our monitoring stack from scratch, while I advocated for using CloudWatch with custom dashboards. I prepared a comparison showing the build-vs-buy trade-offs: development time, maintenance burden, and feature parity. We agreed to prototype both approaches for one week. The managed solution won on every metric, and my colleague appreciated the evidence-based approach rather than a top-down decision.

5. "How do you handle working under pressure or tight deadlines?" During a critical production incident that affected 5,000 users, I was on call and had to coordinate the response. I set up a war room, assigned investigation threads to team members based on their expertise, communicated status updates to stakeholders every 30 minutes, and identified the root cause within 2 hours. After resolution, I led the post-mortem and implemented three preventive measures that reduced similar incidents by 80% the following quarter.

6. "What's your approach to documentation?" I treat documentation as a product. When I joined my last team, runbooks were outdated and onboarding took 3 weeks. I introduced a "docs-as-code" approach using Markdown in our repo, wrote decision records for every architectural choice, and created an onboarding guide that reduced new starter ramp-up to 5 days. The team adopted the practice, and our knowledge base became a reference for other departments.

7. "Tell me about a project you're most proud of." I built a serverless data pipeline that automated our client reporting process. Previously, a team of three spent two days each month manually compiling reports. I designed the solution using Lambda, S3, and Step Functions, tested it against 6 months of historical data, and deployed it with zero downtime. It now

generates reports automatically in under 10 minutes. The project freed up 48 person-hours per month and became a template for similar workflows across the business.

8. "How do you approach security in your work?" I follow the principle of least privilege in everything I build. On a recent project, I implemented IAM policies scoped to specific S3 buckets and Lambda functions rather than using wildcard permissions. I also set up automated security scanning in our CI/CD pipeline using Checkov, which caught 12 misconfigured resources before they reached production. Security is not a phase - it's a habit I build into every design decision.

9. "Where do you see yourself in 3–5 years?" In three years, I want to be a senior engineer who can own end-to-end system design and mentor others making the same career transition I'm making now. In five years, I'd like to be in a solutions architecture or technical leadership role where I can bridge business strategy and technical execution. I'm building toward that by earning certifications, leading projects, and developing my communication skills alongside my technical depth.

10. "Why should we hire you over other candidates?" I bring three things that are hard to find in one person: technical depth from my certifications and hands-on projects, business context from my non-traditional background, and a builder's mindset - I don't just consume knowledge, I ship things. My portfolio includes real deployed applications, not just tutorial projects. And because I've made a deliberate career transition, I bring a level of intentionality and hunger that you won't find in someone who fell into tech by default.

7. Post-Interview Thank You Email

Send within 24 hours. Keep it short (3 paragraphs max). Reference something specific from the interview - a question they asked, a problem they mentioned, a company value you discussed.

Subject: Thank you - [Role Title] conversation

Hi [First Name], Thank you for taking the time to speak with me about the [Role Title] position. I particularly enjoyed our conversation about [specific topic from interview], and it reinforced why I'm so interested in this role. You mentioned [specific challenge they're facing]. Based on my experience with [relevant experience], I believe I could [specific way you'd add value]. I'm excited about the opportunity and would love to move forward. Please let me know if you need any additional information. Best regards, [Your Name]

8. Weekly Job Search Tracker

Copy this into a spreadsheet or your Notion template (included with this book). Fill it out every Sunday evening. Track your inputs, not just outputs - consistency is the difference between landing a job in 90 days vs. 9 months.

> WEEK OF: [DATE] | Applications submitted: ___/5 target | Networking conversations: ___/3 target | Cert study hours: ___/5 target | Portfolio work: ___/3 hours target | Interviews scheduled: ___ | Wins this week: ___ | Blockers: ___ | Next week's #1 priority: ___

9. Salary Negotiation Script

When an offer comes in, they're expecting you to negotiate. These scripts cover three common scenarios. Pick the one that fits.

Scenario A: Offer is below your target range

> Thank you so much for the offer - I'm genuinely excited about the role and the team. Before I commit, I wanted to revisit compensation. Based on my [1–2 specific skills/experiences], market data for this role in [location] showing [cite Glassdoor/Ravio range], and the value I'll bring from day one, I was hoping for something closer to £[your target]. Would you be able to move to that range?

Scenario B: Offer is in range but you want more

> I really appreciate the offer at £[amount]. The role is exactly what I was hoping for. To make this a clear win for both of us, would you be able to go to £[5–10% higher]? At that level, I'm ready to sign immediately and can start on [your preferred date].

Scenario C: They say no to salary - negotiate other terms

> I understand. I'm still very keen to join the team. If salary is fixed, could we revisit this at the 6-month mark with a guaranteed review? Alternatively, is there flexibility on remote work days, professional development budget, signing bonus, extra holiday, or flexible hours?

10. 90-Day New Job Success Plan

Your first three months define how your manager and team perceive you. Use this plan on day one.

> **Days 1–30 (Learn & Quick Wins):** Schedule 1:1s with every team member + manager. Ask: "What are you working on? What's one thing that slows the team down?" Read all documentation. Start a learning log. Pick 2 small things to fix/improve (documentation, a broken pipeline step, a confusing config). Aim for 2–3 merged PRs. At day 30, frame it to your manager: "Here's what I've learned, here's what I've done, here's my plan for the next 60 days."

Days 31–60 (Own a Project): Ask your manager: “What’s a project I can own end-to-end?” This should be 2–4 weeks of focused work. Own it completely: design, implementation, testing, deployment. Weekly standups with your manager. This is where you prove you can deliver independently.

Days 61–90 (Visibility & Growth): Present your learnings at a team meeting. Volunteer for a cross-team initiative. Have 1:1s with your manager and skip-level to discuss growth. Schedule a formal 90-day review. Come prepared with evidence of your three 30-day objectives, team feedback, and your proposal for Q2 focus areas.

Your Playbook Summary

Here’s everything in one view:

1. Pick ONE domain and go deep (Chapter 2)
2. Get certified - but go beyond studying (Chapter 3)
3. Build projects that solve real problems, not tutorial projects (Chapter 4)
4. Master your AI toolkit - it’s your competitive edge (Chapter 5)
5. Run a job search system, not a job search spray (Chapter 6)
6. Build your brand before you need it (Chapter 7)
7. Follow the 90-Day Sprint Plan with daily consistency (Chapter 8)

One Last Thing

You don’t need a computer science degree. You don’t need permission. You don’t need to be “ready.”

You need a domain, a cert, two projects, a LinkedIn presence, and the stubbornness to keep going after rejection number 47.

That’s it. That’s the playbook.

Now go build something.

Want Personalised Guidance?

If you’ve read this book and you want someone to look at YOUR specific situation - your CV, your certification plan, your portfolio, your next career move - here’s how to get help:

Reply to the newsletter. Every reply goes straight to my inbox. Ask me anything about your career transition, certifications, or job search. I read every single message.

1:1 coaching (limited availability). For readers who want personalised, hands-on guidance, I offer a small number of 1:1 coaching sessions. You bring the questions. I bring 9 cloud certifications, 60+ rejections, and the experience of helping many people land their first tech role. Book a session at mentorship.sholastechnotes.com.

- Shola

Subscribe to Shola's Tech Notes for weekly insights:
sholastechnotes.com
Follow me on LinkedIn: linkedin.com/in/olushola-oladipupo
Try ResumeRadar: resumeradar.sholastechnotes.com

Bonus: Resources & Quick Reference

Free Learning Resources

- AWS Skill Builder (explore.skillbuilder.aws) - Free courses for all AWS certifications
- freeCodeCamp (freecodecamp.org) - Free coding courses in Python, JavaScript, and more
- Professor Messer (professormesser.com) - Free CompTIA certification training
- roadmap.sh - Visual career roadmaps for every tech domain
- A Cloud Guru / Pluralsight - Paid but comprehensive cloud training
- Cantrill.io - Deep-dive AWS training

Recommended Books

- System Design Interview by Alex Xu - Essential for understanding scalable systems
- The Phoenix Project by Gene Kim - Understand DevOps culture through story
- Designing Data-Intensive Applications by Martin Kleppmann - Deep technical foundations

Tools Quick Reference

- Kiro (kiro.dev) - Agentic IDE listed on AWS's website
- Cursor (cursor.com) - AI-native code editor
- Claude Code (claude.ai) - Terminal-first AI coding agent
- GitHub Actions - Free CI/CD for public repositories
- Terraform (terraform.io) - Infrastructure as Code
- Docker (docker.com) - Containerisation platform

Salary Research Resources

- Glassdoor (glassdoor.co.uk) - Salary data and company reviews
- LinkedIn Salary - Role-specific salary insights
- levels.fyi - Big tech company compensation data

- PayScale (payscale.com) - UK-specific salary benchmarks

Job Search Platforms

- LinkedIn Jobs - Primary platform for tech roles
- Indeed (indeed.co.uk) - Broad job aggregator
- Otta (otta.com) - Startup and tech-focused
- Hired (hired.com) - Companies apply to you
- WeWorkRemotely - Remote-first tech roles

Community & Networking

- Meetup.com - Local tech meetups and events
- AWS Community Builders - Apply to join the official AWS community
- Tech Twitter / X - Follow creators in your domain
- Discord communities - Real-time help and networking

Sources & References

Every statistic and claim in this book has been sourced and verified. Below is a complete list of references in order of appearance.

1. UK IT Hiring Expansion - Robert Half, UK Jobs Confidence Index, December 2025. Finding: 56% of UK organisations plan to expand permanent IT teams in H1 2026. roberthalf.com/gb/en/insights/jobs-confidence-index

2. Cloud Certification Salary Premium - Global Knowledge / Skillsoft, IT Skills and Salary Report, 2024. Finding: Certified cloud professionals earn an average salary of $113,000 USD. globalknowledge.com/us-en/content/salary-report/it-skills-and-salary-report

3. LinkedIn AI Expectations - LinkedIn, 2025 Talent Report. Finding: Hiring managers increasingly list AI fluency as a preferred or expected skill for entry-level hires. linkedin.com/business/talent/blog/talent-acquisition/future-of-recruiting-2025

4. Remote Work Preferences - FlexJobs, Remote Work Statistics and Trends Report, 2024. Finding: 85% of workers prefer jobs with remote flexibility or fully remote options. flexjobs.com/remote-work-statistics-and-hiring-trends

5. Junior Hiring Decline - Ravio, Tech Job Market Report, 2025. Finding: Entry-level positions (P1 and P2 job levels) saw a 73% decrease in hiring rates. ravio.com/blog/tech-hiring-trends

6. AI Job Posting Surge - Indeed Hiring Lab, US Labor Market Update, January 2026. Finding: Job postings mentioning AI or AI-related terms surged by more than 130% between 2024 and 2026. hiringlab.org

7. ATS Adoption Rates - Resume Genius, Applicant Tracking Systems Report, 2024. Finding: 99% of Fortune 500 companies use ATS; 83% of large employers plan AI-driven screening by end of 2025. resumegenius.com

8. AWS Solutions Architect Salary - Indeed, UK Salary Data, 2025. Finding: Average AWS Solutions Architect salary in the UK is £73,000 base (£129,000 total compensation including bonuses and benefits). uk.indeed.com/career-advice/pay-salary

9. Open to Work Badge Impact - LinkedIn, Economic Graph Research, 2024. Finding: Members with the Open to Work badge receive approximately 40% more InMail messages from recruiters. economicgraph.linkedin.com

10. AI Coding Adoption - GitHub, Octoverse Report, 2024. Finding: 92% of developers report using AI coding tools in their daily workflow. github.blog/news-insights/octoverse/octoverse-2024

All statistics were verified at the time of writing (March 2026). For the latest data, visit the original sources. Industry statistics may be updated periodically.