AI-First vs AI-Assist:

Early-career developer training in the public sector

How AI can train a new generation of public sector developers build secure, resilient digital services

zaizi

Executive summary

Artificial Intelligence (AI) is reshaping software development. The public sector should integrate AI into early-career developer training to realise productivity and efficiency gains, but also to build a resilient workforce for the future.

In summer 2025, Zaizi ran a seven-week internship with two CyberFirst bursary students.

The programme was an experiment to test two AI-enabled workflows: **AI-First**, where AI generates code from natural language prompts, and **AI-Assist**, where AI provides context-sensitive suggestions within a developer's workflow.

Key findings

1. **AI-First speeds prototyping but is fragile**
The AI-First project produced a working prototype quickly. But debugging opaque AI-generated logic took time. Mentors noted pedagogical value, as it strengthened resilience and critical thinking. But the interns' confidence with AI-First remained low.

2. **AI-Assist boosts confidence and skills**
AI-Assist sped up routine coding tasks while maintaining developer ownership of design and logic. Interns reported higher confidence, better understanding of code structure, and improved long-term skills growth.

3. **Mentorship and guardrails are essential**
AI worked best when paired with structured mentoring and ethical training. Workshops on accessibility, secure design, and testing provided vital guardrails. By the end of the programme, both interns rated themselves 5/5 for their ability to critically reflect on AI outputs.

4. **Sequence matters**
Starting with AI-First exposed risks and complexity, priming interns to use AI-Assist more responsibly in the second project. This order may be valuable in training pipelines.

AI-First
fast but fragile

- rapid prototyping
- hard to debug
- brittle code

AI-assist
builds confidence

- skills growth
- clear logic
- higher confidence

Mentorship & guardrails
are essential

- ethical, secure design
- code reviews
- builds resilience

Sequence
matters

- AI first then AI assist
- teaches responsibility
- strengthens reflection

Implications for public sector organisations

The findings carry significant weight for government digital teams and their delivery partners.

Our experiment found that AI-Assist is the most effective approach for public sector training. It accelerated learning and boosted developer confidence, while ensuring the code was maintainable and transparent.

AI-First, on the other hand, is valuable only as a teaching exercise and should not be the foundation for training pipelines.

These results show that AI is not just a technical upgrade: it is also a skills and workforce challenge for the public sector. For that reason, its effectiveness as a training tool depends on strong mentoring, ethical guardrails, and evaluation.

When used responsibly, AI can accelerate learning and help build a new generation of AI-literate, critically minded developers capable of delivering secure, transparent, and resilient public services.

1. Introduction

Artificial Intelligence (AI) is reshaping the way software is designed, built, and maintained.

In recent years, development teams have increasingly adopted tools that either generate large sections of code from natural language prompts (AI-First) or support developers by suggesting code as they work (AI-Assist).

These approaches are no longer experimental. A study found that developers using GitHub Copilot were able to complete tasks 55% faster than those without it (Ziegler et al., 2023).

Public sector organisations are already trialling AI in service delivery, from case management in justice systems to document digitisation in healthcare (StateTech Magazine, 2025).

AI in Public Sector

Unlike consumer technology, public-facing applications need to meet rigorous standards of accessibility, ethical use, and resilience.

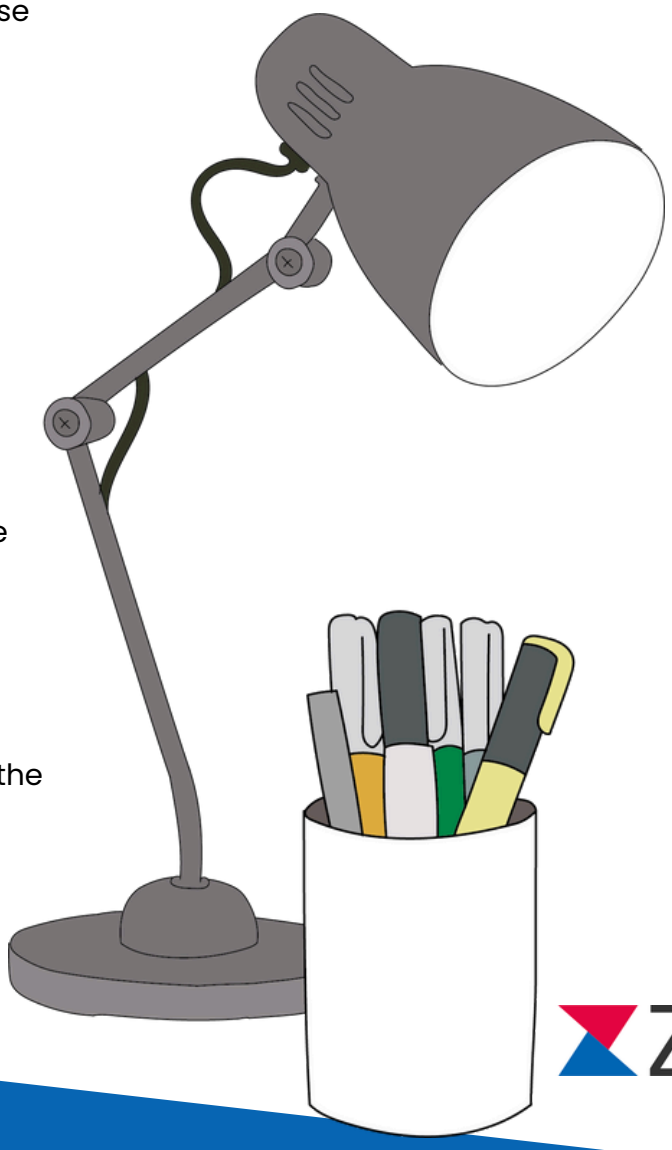
This raises a pressing question: how should educators integrate AI into the training of early-career developers?

The risks of misapplication are clear. Over-reliance on AI could create a generation of developers who struggle to debug, extend, or explain the systems they build — a critical weakness in the public sector, where accountability and continuity are non-negotiable.

Commentators have warned that teams must 'learn before they lean' on generative AI.

Junior developers are particularly at risk of skipping fundamental skills development if AI takes too much of the load (TechRadar, 2023).

At the same time, when used carefully, AI has the potential to accelerate learning, free up time for higher-order problem-solving, and embed best practices earlier in a developer's career.



Zaizi's internship programme

Against this backdrop, Zaizi designed and delivered a seven-week internship programme, working with two CyberFirst bursary students.

The aim was not simply to give the interns work experience but to run a structured comparison between AI-First and AI-Assist workflows.

We evaluated their impact on:

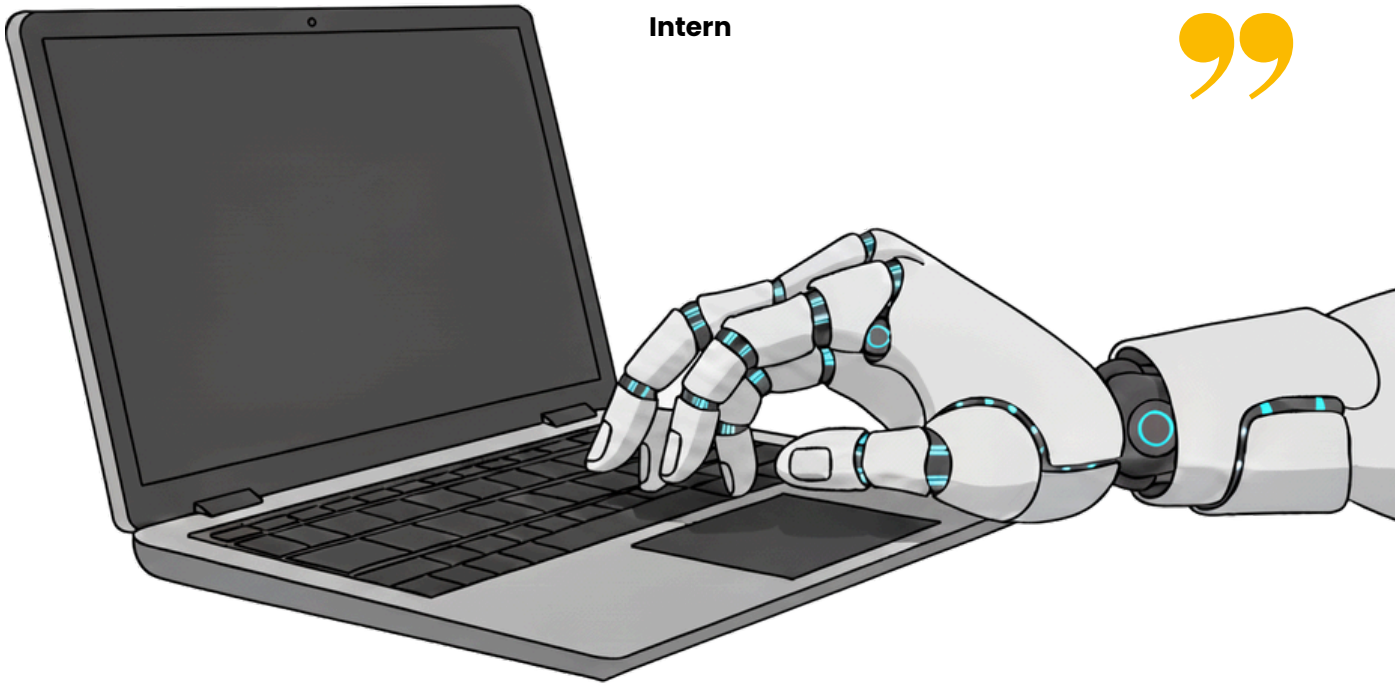
- code quality and maintainability
- developer training and skills retention
- team collaboration and mentoring
- suitability for public sector contexts

This white paper captures what we learned from that experiment. It:

- presents the design and outcomes of the programme
- compares the two AI approaches
- sets out practical recommendations for embedding AI safely and effectively in early-career developer training.

The paper is for public sector organisations and their delivery partners who want to integrate AI into their training pipelines and mentoring practices.

> The most valuable lesson was learning how to use AI tools to their greatest effect but still think for myself
>
> **Intern**



2. Background

2.1 Defining AI-First vs AI-Assist

The distinction between AI-First and AI-Assist development workflows is central to this report.

AI-First tools generate end-to-end applications from natural language prompts.

Systems such as Firebase Studio with Gemini, or frameworks like GPT-Engineer, attempt to scaffold front-end interfaces, database schemas, business logic, and sometimes even documentation.

The developer primarily serves as a curator and debugger — reviewing, correcting, or adapting what the AI has produced.

This approach promises rapid prototyping but risks producing code that is brittle, opaque, or poorly aligned with maintainability requirements.

AI-Assist tools, such as GitHub Copilot or Amazon CodeWhisperer, are embedded into integrated development environments (IDEs).

They provide context-sensitive suggestions, code snippets, tests, and refactoring hints based on the developer's activity.

The developer remains the primary author, with the AI acting as a supportive partner.

Early studies suggest that assistive tools improve productivity and developer satisfaction without removing ownership of the code (Vaithilingam et al., 2022).

The distinction is not merely technical.

It affects how developers learn, how teams collaborate, and how organisations maintain long-term control over their systems.

Comparing AI approaches

AI-FIRST	You tell the AI what you want through prompts	The AI generates the whole app	The developer's main job is to refine & review
AI-ASSIST	The developer writes the code	The AI suggests code snippets as you type	The developer remains in control of the final code

2.2 Why it matters in the public sector

Public sector digital services operate under constraints that make the responsible use of AI particularly important.

Transparency and auditability: Services must sometimes withstand parliamentary scrutiny, public inquiries, or legal challenge. AI-first generated code, if not well understood by the team, can compromise auditability.

Maintainability and continuity: Public services must remain operable for years, often beyond the tenure of the original delivery team. Maintainability requires code that is well-structured, documented, and comprehensible.

Security and ethics: Public trust depends on secure, accessible, and fair services. Over-reliance on AI-generated components can introduce vulnerabilities that are not immediately visible (NCSC, 2021).

Skills pipeline: Government services rely on a mix of in-house staff and contracted delivery partners. Ensuring early-career developers are trained to think critically about AI is essential to sustaining a resilient workforce.

The National Cyber Security Centre (NCSC) has warned that AI "can bring security benefits as well as new opportunities for attackers," stressing the need for caution in adoption (NCSC, 2021).

Meanwhile, the UK Government's Technology Code of Practice emphasises service transparency, maintainability, and security — qualities to safeguard when AI enters the development workflow (GOV.UK, 2024).

2.3 Objectives and principles of the internship

We explicitly framed the internship as an experiment to compare AI-First and AI-Assist approaches under controlled conditions.

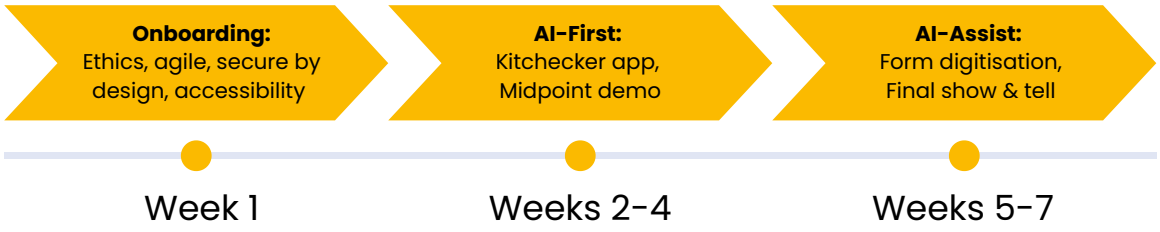
The objectives were:

- **Equity of challenge:** We designed both projects with comparable scope and complexity to enable meaningful comparison.
- **Structured autonomy:** Interns had freedom to explore within a scaffolded environment of mentorship and workshops.
- **Reflection over output:** The programme prioritised learning outcomes over polished deliverables.
- **Ethical AI usage:** We encouraged interns to critically evaluate the risks and limitations of AI, particularly in the context of public sector applications.
- **Ongoing support:** We ensured continuous guidance through weekly check-ins, code reviews, and shadowing.

We designed the principles not only to support the interns' growth but to provide a replicable framework for how public sector organisations and their delivery partners can structure training with AI.

> The first project showed me how much AI could do for you, but also how quickly you could get lost in its logic if you didn't understand it yourself.
>
> **Intern**

3. Programme design



3.1 Participants

Two CyberFirst bursary students undertook the internship, both entering their second year of undergraduate study.

CyberFirst is a government-backed outreach and education programme which provides paid cyber skills training for undergraduates. Every summer, students participate in internships with organisations like Zaizi.

The students had a firm academic grounding but limited exposure to professional engineering practices. It made them an ideal test case for exploring how AI-enabled workflows shape the learning journey of early-career developers.

The internship ran for seven weeks (July–August 2025). We structured activities around two major proof-of-concept projects and integrated workshops and mentorship throughout the process.

3.2 Duration and structure

Week 1: Onboarding and foundations
The programme began with an introduction to the company values and the aims of the internship.

We delivered workshops on ethics of AI in code, agile kanban, pair programming, Git hygiene, secure by design, accessibility in public services, code review practices, and testing fundamentals.

The interns said the week was 'hugely valuable,' giving them a baseline of professional practice before experimenting with AI.

Weeks 2–4: AI-First
For the AI-first project, the interns developed a **Kitchecker app** — a dashboard to track and verify kit and equipment against a required checklist.

The interns used Firebase Studio with Gemini integration to generate an AI-led end-to-end service (including user interface, database, logic, and documentation).

In week four, the interns presented a midpoint demo to reflect on the challenges they faced and share insights.

Weeks 5–7: AI-Assist
In the AI-assist project, the interns took an existing paper form and developed it into a **GDS-compliant digital form**, complete with question routing and validation.

They used GitHub Copilot (GPT-4.1) within a traditional coding workflow, which integrated AI assistance with hands-on development.

It concluded with a final Show & Tell where the interns presented both projects. They demonstrated the learning journey and compared the outcomes between the AI-First and AI-Assist approaches.

3.3 Mentorship and evaluation

We embedded structured support to guide learning:

- **One-to-ones:** Weekly check-ins with senior developers for technical and pastoral support.
- **Peer review and pairing**: Interns worked together to review each other's code and AI usage.
- **Demo days:** They presented their progress to mentors and the wider company.
- **Slack support:** They had daily support from the Zaizi engineering team.

Mentors highlighted that the most valuable learning occurred during code review sessions.

> The interns weren't just fixing the AI's mistakes. They were learning to articulate why the code was wrong, and how to reshape it.
>
> **Mentor**

We continually measured and assessed their learning outcomes:

- **Code reviews:** Mentors assessed code for readability, structure, testing, and maintainability.
- **Debugging exercises:** Interns extended or debugged AI-generated code live, testing resilience and understanding.
- **Self-assessment surveys:** They reflected on their confidence, skills, and perceptions of AI before and after both projects.
- **Reflection journals:** Interns logged daily experiences, challenges, and decisions about AI use.
- **Mentor reports:** Senior staff assessed the interns' skill growth, collaboration, and critical thinking.

The evaluation balanced quantitative indicators (e.g. survey confidence ratings) with qualitative insights (e.g. reflections, mentor notes).

This ensured that the internship produced not only working prototypes but meaningful evidence for public sector training policy.

4. Programme design

4.1 Project AI-First

The first project tasked the interns with using an AI-first workflow.

They generated an application primarily through natural language prompting, with the developer role focused on curating, debugging, and adapting AI outputs.

Outcomes
The project delivered a functioning proof-of-concept app and dashboard, but the path was uneven.

The AI excelled at rapid prototyping, scaffolding database schemas, and generating boilerplate documentation.

But significant challenges arose:

- Frequent Gemini errors and cache issues disrupted progress.
- Scaling problems emerged once interns tried to extend functionality.
- Opaque logic made debugging difficult

Learning impact
The interns said they spent much time untangling AI-generated code rather than writing features themselves.

> Debugging, untangling, and understanding the AI's code was the hardest part.
>
> **Intern**

Mentors noted that, while frustrating, the experience had pedagogical value. It built resilience, strengthened debugging skills, and highlighted the risks of blindly trusting AI outputs.

The interns' confidence remained only 'somewhat confident' with AI-First approaches.

It aligns with wider literature on generative AI.

Studies show that AI-generated code can contain subtle errors and non-obvious inefficiencies, requiring skilled human oversight (Pearce et al., 2022).

For early-career developers, the risk is that AI-First becomes a crutch that impedes fundamental skill development (TechRadar, 2023).

4.2 Project AI-Assist

The second project placed the interns in an AI-assisted workflow.

They used AI as a coding partner within a traditional software development process.

Outcomes
The interns successfully delivered the proof of concept, with cleaner code, higher maintainability, and fewer internal errors compared to the first project.

GitHub Copilot accelerated repetitive tasks (documentation, test scaffolds, routine syntax), while leaving design and logic to the interns.

Both interns reported feeling 'very confident' or 'more confident' in their abilities during this project.

> AI-Assist gave me a much better understanding of how the code worked. I felt like I was still driving, with AI there to back me up.
>
> **Intern**

Learning impact
In contrast with AI-First, the interns consistently linked AI-Assist with long-term skills growth.

They reported better grasp of code structure and logic, improved Git hygiene, and stronger testing skills.

Survey results backed this up:

- Greater confidence in long-term skills: both interns preferred the AI-Assist project.
- Critical reflection on AI outputs: both rated themselves 5/5.

These findings reflect wider industry research. GitHub's 2023 study found Copilot sped up tasks, while improving developer satisfaction by preserving agency over problem-solving (Ziegler et al., 2023).

Comparison chart

	Project 1: AI-First	Project 2: AI-Assist
Outcomes	Firebase + Gemini: AI-generated app using natural language prompts	GitHub Copilot: AI assistance in traditional coding workflow
Tooling & scope	Working prototype but brittle & opaque logic	Cleaner code, maintainable & fewer errors
Learning impact	Time spent debugging AI outputs, resilience built	Improved code structure, testing, Git hygiene
Intern confidence	Interns 'somewhat confident' of approach	Interns 'very confident' of approach

4.3 Key themes

Several themes emerged across both projects.

Critical reflection
Interns learned to interrogate AI outputs. They could identify over-engineered, redundant, or logically inconsistent code.

This skill is arguably more valuable than raw productivity gains, given the public sector's need for explainable systems.

Mentorship shapes learning
Mentorship turned AI failures into learning opportunities.

Rather than allowing interns to treat AI as an unquestioned authority, mentors used debugging sessions to reinforce critical thinking and professional practice.

It aligns with guidance from the NCSC, which highlights that machine learning systems can introduce vulnerabilities if not properly governed and emphasises secure-by-design principles when adopting AI (NCSC, 2021; 2023b).

In practice, mentors provided the human safeguard, helping interns identify brittle or insecure AI outputs and reshape them.

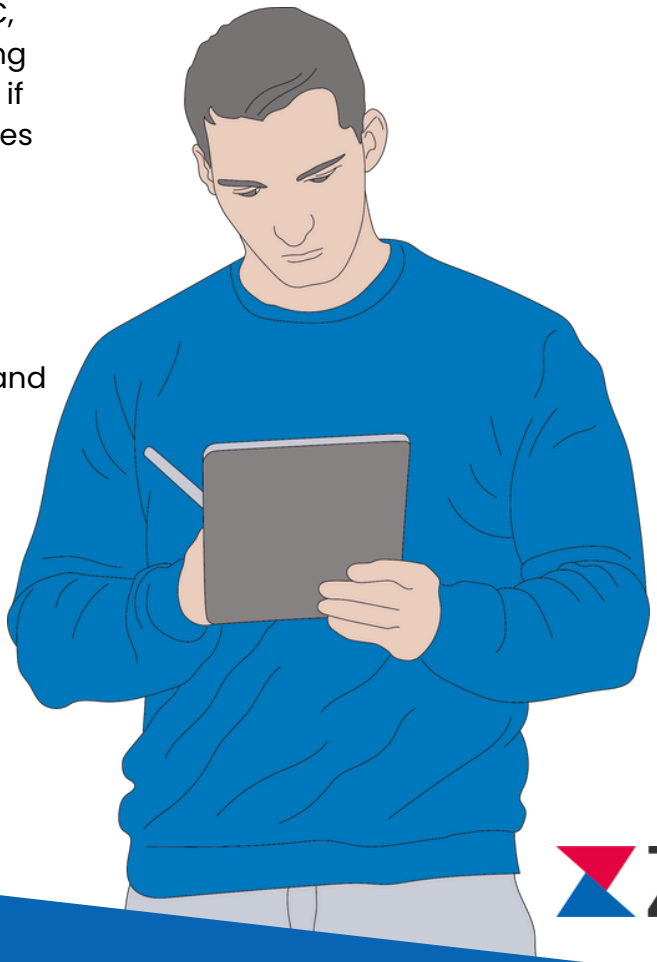
Workshops as guardrails
Workshops on ethics, security, accessibility, and testing provided a framework for navigating the complexities of AI.

Interns frequently referenced these sessions when making design decisions.

Sequence matters
Starting with AI-First allowed the interns to understand the risks and complexity of AI.

AI-Assist allowed them to apply lessons in a more controlled environment.

By the time they reached the AI-Assist project, they knew they had to manage Copilot and not mindlessly follow it.



5. Public sector training

The internship showed that AI-First and AI-Assist approaches produce very different training outcomes for early-career developers.

While both approaches have value, they are not interchangeable. Their implications are particularly significant in public sector contexts.

5.1 AI-First: High risk, selective use

AI-First tools offer striking benefits for rapid prototyping. The interns produced a working application within weeks despite their limited prior experience. But scaling beyond simple prototypes revealed brittle dependencies.

Public sector services cannot ignore these risks. Code that is hard to explain or maintain is a liability, particularly in government environments where auditability and continuity are critical (GOV.UK, 2024).

The National Cyber Security Centre (NCSC) warns that reliance on opaque AI systems may introduce security vulnerabilities that are not visible until exploited (NCSC, 2021).

AI-First should not form the foundation of early-career training.

Instead, it may serve a selective role: allow developers to quickly prototype, make them foresee potential risks, strengthen debugging skills, and illustrate the importance of code clarity.

> The first project showed me how much AI could do for you, but also how quickly you could get lost in its logic if you didn't understand it yourself.
>
> **Intern**

5.2 AI-Assist: A strong foundation for training

AI-Assist workflow offered a balance of productivity and control.

Interns described feeling more confident, better able to structure their code, and more capable of explaining their decisions.

It had clear alignment with public sector needs:

- **Transparency:** Copilot's suggestions still required explicit acceptance, ensuring developers understood and could justify changes.
- **Maintainability:** The resulting codebase was cleaner, more structured and understandable.
- **Confidence and skills retention:** Interns' survey responses confirmed that AI-Assist built greater long-term skills confidence than AI-First.

Research shows that assistive AI tools boost productivity without undermining developer agency, provided they are integrated within strong mentoring and review practices (Vaithilingam et al., 2022).

AI-Assist should be the baseline approach in onboarding and training early-career developers.

It accelerates learning while protecting critical thinking and long-term maintainability.

5.3 The non-negotiable role of mentorship and guardrails

AI is effective only when paired with structured mentoring and ethical guardrails.

- **Mentorship**: Weekly check-ins, code reviews, and pairing gave interns the confidence to challenge AI outputs rather than accept them unquestioningly.
- **Workshops**: Sessions on secure design, accessibility, and testing grounded AI experiments that follow public sector principles.
- **Critical reflection**: By the end of the programme, both interns rated themselves 5/5 for their ability to analyse and judge AI-generated outputs.

5.4 Building the Public Sector skills pipeline

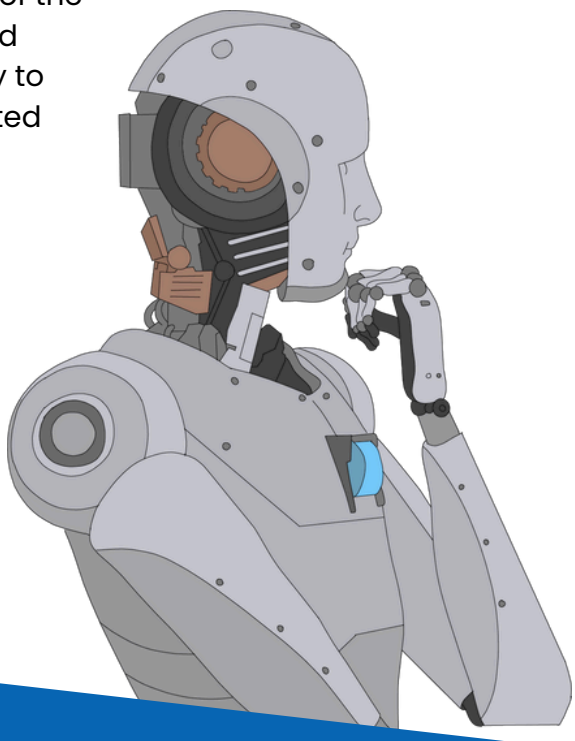
The programme highlighted the strategic role of training in sustaining a resilient public sector digital workforce.

With government services increasingly dependent on digital platforms, there is pressure on both in-house teams and delivery partners to ensure continuity of skills.

If early-career developers train in AI-First workflows without guardrails, it could create brittle systems and teams ill-equipped to extend or secure them.

By contrast, AI-Assist workflows can help cultivate a generation of developers who are both AI-enabled and critically minded.

It reflects OECD guidance (2024) for governments to treat AI not only as a tool for productivity, but as a driver of skills development, institutional capacity, and long-term governance readiness across the public sector.



6. Recommendations

The findings show AI should play a role in training early-career developers.

It should be carefully structured to align with public sector priorities of transparency, maintainability, and accountability.

Here are our recommendations.

6.1 Adopt AI-Assist as the default training approach

Public sector organisations should embed AI-Assist workflows (e.g. GitHub Copilot, Amazon CodeWhisperer) in training programmes.

These tools accelerate learning and productivity while preserving developer ownership of the code.

Why: Our interns reported greater confidence, better code comprehension, and higher skills retention.

How: Use AI-Assist in day-to-day workflows but always review, test, and explain suggestions to meet audit requirements.

6.2 Use AI-First selectively as a teaching exercise

AI-First tools should not be excluded entirely. They enable rapid prototyping and can teach valuable lessons in risk management.

Why: The AI-First project forced interns to confront brittle, opaque code and develop resilience in debugging.

How: Position AI-First tasks as controlled experiments within training or hackathon settings. The goal is reflection, not production-ready code.

6.3 Strengthen mentorship and code review practices

Early career developers need structured guidance to challenge AI outputs and learn what to accept or reject.

Why: AI increases the need for guidance. Our mentorship transformed failures into teaching moments and helped interns develop critical thinking.

How: Weekly one-to-one mentoring, structured code reviews, and opportunities for interns to present and defend their decisions.

6.4 Embed guardrails through ethical and secure-by-design training

AI must reflect public sector priorities of security, accessibility, and fairness.

Training should include mandatory sessions on the ethical use of AI and its implications for service design.

Why: Interns cited workshops on accessibility, secure design, and testing as crucial to guiding their AI use.

How: Incorporate AI-specific modules into onboarding programmes, aligned with the GOV.UK's Technology Code of Practice and NCSC's AI security guidance.

6.5 Build evaluation into the training pipelines

Training should combine self-reflection, mentor assessment, and technical evaluation.

Why: Surveys and mentor reports showed growth in confidence, skills, and critical thinking — things that code quality alone could not measure.

How: Use a mixed evaluation framework combining technical review, self-assessment, and qualitative reflections to assess impact.

6.6 Invest in AI training to build workforce resilience

AI training is not just about efficiency; it's a strategic investment in the public sector workforce.

Why: Government organisations need to create an AI-literate workforce.

How: Embed AI-Assist into graduate schemes, apprenticeships, and partner contracts. Ensure that public sector teams remain AI-literate but critically minded.

Recommendation	Why	How
Adopt AI-Assist as default	Builds confidence and comprehension	Integrate into workflows; review and test suggestions
Use AI-First selectively	Teaches risk awareness and debugging	Apply in controlled learning, not live systems
Strengthen mentorship	AI increases the need for guidance	Run check-ins, reviews, and get interns to explain decisions
Embed guardrails	Ethics and secure design workshops guide AI use	Add AI-specific onboarding aligned with standards
Evaluate training	Mixed evaluation tracks skills and confidence, not just code	Combine surveys, mentor notes, and code review
Invest in AI-enabled training	Sustains AI-literate public sector teams	Embed in graduate schemes and supplier contracts

7. Conclusion

This internship demonstrated in practice what many in the public sector recognise in theory: the way organisations introduce AI into developer training matters as much as the technology itself.

The experience confirmed that AI alone cannot teach fundamental developer skills.

It is a force multiplier for training, as long as mentorship, ethical guardrails, and structured evaluation are in place.

7.1 Key implications for public sector training

The lessons for public sector organisations and delivery partners are clear:

- embed AI-Assist as the baseline in early-career training pipelines.
- anchor AI usage with mentorship and secure-by-design principles, making sure junior developers remain critical of AI outputs and are accountable.
- use evaluation frameworks to measure not just productivity but confidence, comprehension, and ethical awareness.

This is not just about improving developer efficiency. It is about preparing the next generation of public sector technologists to build secure, transparent, and resilient systems.

As government services become increasingly digital, training developers to use AI responsibly ensures the workforce remains capable and adaptable.

7.2 Next steps: Scale and embed AI training

The internship programme provides a first step in applying AI to public sector training.

The natural next step is to expand on these insights by:

- testing AI-Assist workflows across larger cohorts.
- embedding them into graduate schemes, apprenticeships and delivery partner training.
- aligning AI-enabled training with procurement practices for delivery partners.

When implemented thoughtfully, AI can accelerate the development of a critically minded, AI-literate workforce — one capable of delivering the trustworthy digital services that citizens expect and deserve.

8. How we can help

About the Author

Seyed has over 20 years of experience delivering secure, scalable, and user-centred digital services across government.

As head of engineering at Zaizi, he specialises in platform architecture, cloud-native solutions and agile delivery.

Seyed has a strong track record of leading high-impact programmes for government organisations, including the Government Digital Service and Department for Education.

Seyed Razavi-Nematollahi
Head of Engineering
020 3582 8330
srazavi@zaizi.com

Want to discuss the findings of this whitepaper or know more about AI-assisted training ? Connect directly with Seyed to explore how these insights can help your organisation.

About Zaizi

At Zaizi, we're on a mission to make the UK the best and safest place to live and work.

We design, build and sustain secure, user-centered digital digital services for government. For nearly two decades, we've helped public sector teams digitise processes, modernise legacy systems and automate workflows.

Here's how we can help you on your AI journey:

- **Assess your AI readiness:** We can evaluate your team's current capabilities and data infrastructure to identify how best to integrate AI.
- **Integrate AI for immediate value:** We help you find the right use cases to deploy AI quickly, effectively, and in line with government best practices.
- **Create your AI training:** We can help you design an AI training program based on the proven model described in this whitepaper.

See how AI can transform your training and delivery pipelines. Contact us today.

References

CyberFirst (2025) Bursary and Student Opportunities Programme. National Cyber Security Centre. Accessed: 4 August 2025

GOV.UK (2024) Technology Code of Practice. Government Digital Service. Available at: (Accessed: 4 August 2025).

National Cyber Security Centre (2021). Machine Learning Principles. Accessed: 4 August 2025

National Cyber Security Centre (2023). Guidelines for Secure AI System Development. Joint publication with CISA and international partners. Accessed: 4 August 2025.

OECD (2024) Governing with Artificial Intelligence: Are governments ready? OECD Policy Paper. Accessed: 4 August 2025

Pearce, H., Ahmad, B., Tan, B., Dolan-Gavitt, B. and Karri, R. (2022) Asleep at the Keyboard? Assessing the Security of GitHub Copilot's Code Contributions. In: 2022 IEEE Symposium on Security and Privacy (SP), pp. 754–768.

StateTech Magazine (2025) Transforming the Public Sector: AI Opportunities and Challenges. Accessed: 8 August 2025.

TechRadar (2023) The GenAI Crutch: Why Teams Must Learn Before They Lean. Accessed: 4 August 2025

Vaithilingam, P., Zhang, T. and Glassman, E. (2022) Expectations vs. Experience: Evaluating Code Generation Tools in Practice. In: Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems (CHI '22). ACM, New York, NY, USA.

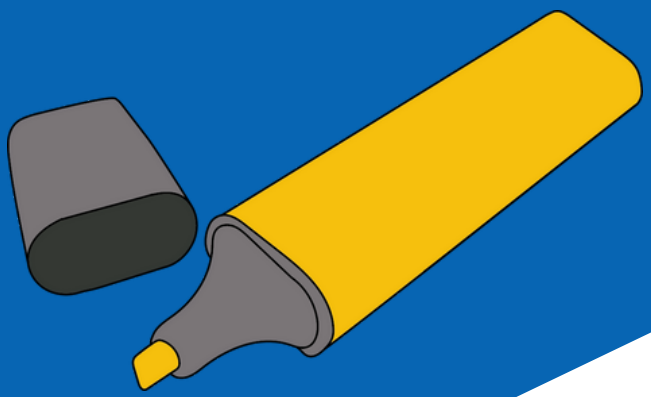
Ziegler, C., Cihon, P., Peng, S. et al. (2023) Productivity Assessment of GitHub Copilot in Software Development. arXiv preprint arXiv:2302.06590. Accessed: 4 August 2025.

AI-First vs AI-Assist:

Early-career developer training in the public sector

Kings House
174 Hammersmith Road,
London, W6 7JP

info@zaizi.com
(+44) 20 3582 8330



zaizi