Microsoft AI Tour

Microsoft

GitHub Copilot as an AI Agent in the Developer Workflow

Our scenario

Hi Ahmed,

Welcome to Zava. We are happy to have you on the team.

Ahmed, to help you get started, please begin by reviewing the open issues in our GitHub repository at https://github.com/ahmedsza/aitourdemo/issues .

There's quite a bit to tackle — especially around improving **testing**, enhancing **code quality**, and sorting through some incomplete work left by the previous developer. Feel free to review what's there, continue where it makes sense, or refactor where needed. Your fresh perspective will be valuable.

We also recommend using the **GitHub Copilot** as part of your daily workflow. It will help you:

- Understand the existing codebase faster
- Improve test coverage and overall code quality
- Automate repetitive development tasks
- Navigate and refactor the repository more efficiently

Immediate action
The Microsoft okes tell that there is something called the GitHub Coding Agent that is almost like a 'virtual developer' – and apparently you can assign issues to it. You might want to start here
Regarding the **testing and Gherkin-related issue**, **Gladys West** will be sending you a set of instructions and guidelines to get started on this.

If you need help with repository access, setting up tools, or understanding our engineering practices, the team is here to support you.
Welcome aboard, Ahmed — we have lots to build together!

Best regards,

Demo

Let the demos begin

Practices

Craft good prompts

→ Prompt Engineering

Review the generated code. You are responsible

Understand your “environment” before jumping in– code and more

→ Context engineering

Plan before implement. Break down into smaller tasks

→ Context engineering

Guide the AI Agent – guardrails, instructions, customizations

→ Context Engineering

You don’t “turn on AI”. You *design the capabilities* it has

Embrace it - don’t just use it like “another tool”

From

chat → workflow

“AI that writes code” → “AI that understands your environment and can act in safely”.

It is not a replacement for the developer..

GitHub Copilot

1. Agent=Model + Tools + Instructions
2. Craft good prompts – Prompt Engineering
3. Context is critical -> Context Engineering
4. You are responsible for reviewing the output - always review and own the code
5. Plan and Break down the work
6. Copilot is available in many places – integrated into your existing tools and workflows – VS Code, Visual Studio, JetBrains, Eclipse, Xcode, Web, CLI, etc
7. Skills, MCP, and customization capabilities create possibilities
8. It is for “everyone” – not just developers
9. It is a collaboration tool that improves over time. Keep learning and adapting

Feedback

Your feedback is valuable.

Please submit your thoughts about today's experiences at **aka.ms/MicrosoftAITour/Survey**

...or use the QR code.

Scan QR code to respond

Will put up links on http://github.com/ahmedsza/brk442

Microsoft

Extras

Primitive	Loading	Best For
Always-on Instructions	Every session	Codebase guardrails
File-based Instructions	Pattern match / description match	Area-specific rules
Prompts (Slash Commands)	User invokes	One-shot workflows
Skills	Description match → on-demand	Reusable capabilities
Custom Agents	Top-level OR as subagent	Constrained workflows
MCP	Session start	External gateways

Agent=Models+Tools+Instructions

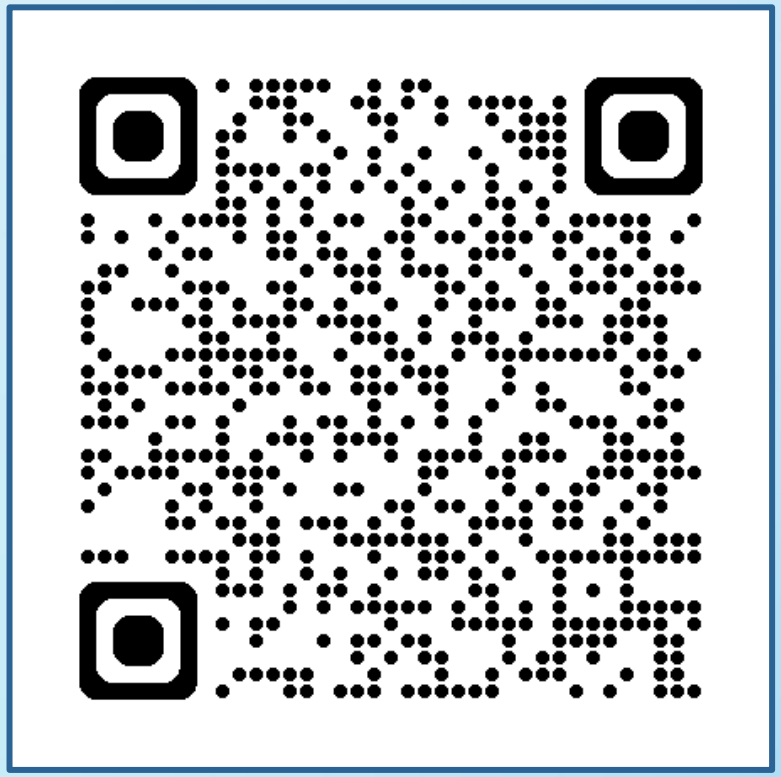
- Lots of useful capability in “non agent” mode
- Understand and Plan before implementing.
- Craft good prompts aka Prompt Engineering
- Context matters aka Context Engineering
  - Instructions
  - Agents
  - Skills
  - Prompts
  - MCP
- Leverage Copilot on the backend
  - Coding Agent
  - PR reviews and descriptions

Adopting Copilot

- Keep Learning, adapting – YouTube videos
- Microsoft Learn – search for “github copilot”
  - https://learn.microsoft.com/en-za/training/browse/?terms=github%20copilot
- Github Skills
  - https://learn.github.com/skills
- Culture, Culture, Culture
- Change Management
  - https://github.com/github/ai-adoption-playbook

We've learned that **AI adoption is fundamentally a people problem, not a technology problem.** Success isn't about buying the right tools—it's about building the human infrastructure that turns hesitant employees into confident power users.

Next steps to advance your AI expertise



aka.ms/TDMDevProductivitySkills

aka.ms/BRK442GHrepo

The Copilot effect.

github.com/features/copilot