



# AI Assisted Software Development

A Practitioner's View

A CODE Presents presentation by John M. Miller

# About John M. Miller



- Software Developer/Architect CODE Staffing
- Played roles of developer, architect, DevOps engineer, platform engineer, test architect, release manager
- Currently working on logistics and lead management projects
- Contributor to The Practitioner's View blog

<https://www.codemag.com/blog/AlPractitioner/>

<john.miller@codestaffing.com>



# CODE30

YEARS



30 years of  
*"Helping People Build Better Software"*

Survey!

<https://bit.ly/CP-07-2025Survey>



Subscribe!

<https://bit.ly/CP-07-2025Subscribe>



# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Reviewing Existing Code
  - Generating Documentation
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Managing Context



- The importance of context
- Add context
- Instructions markdown
  - File-based custom instructions, automatic context injection, instruction scopes, structure and authoring best practices, structure and authoring best practices
- Custom chat modes
  - Mode definition, tool integration, quick mode switching, workspace and user scope, persona-driven workflows, enhanced productivity and consistency
- Equivalents in other IDEs

# Adding Context



- Open editors
- Files & folders ...
- Instructions ...
- MCP resources ...
- Problems ...
- Symbols ...
- Image from clipboard
- Screenshot window
- Tools ...

# Adding Context Built-in Tools



- changes
- codebase
- extensions
- fetch
- findTestFiles
- githubRepo
- new
- newJupyterNotebook
- openSimpleBrowser
- problems
- runCell
- searchResults
- terminalLastCommand
- terminalSelection
- testFailure
- usages
- vscodeAPI

# Adding Context Tools Extensions



- azure\_get\_schema\_for\_Bicep
- azure\_get\_selected\_subscriptions
- azure\_get\_swa\_best\_practices
- azure\_get\_terraform\_best\_practices
- azure\_list\_activity\_logs
- azure\_open\_subscriptions
- azure\_query\_azure\_resource\_graph
- azure\_query\_learn
- azure\_recommend\_service
- azure\_sign\_out\_azure\_user
- create\_load\_test\_script
- get\_azure\_load\_test\_run\_insights
- installNotebookPackages
- listNotebookPackages
- review
- reviewStaged
- reviewUnstaged
- run\_load\_test\_in\_azure
- select\_azure\_load\_test\_run

# Copilot Instructions Files



Instruction files acts as a guide for Copilot

- Located in
  - .github/instructions – project scoped
  - User data folder – user scoped
- Helps Copilot to:
  - Understand the project's context and requirements.
  - Generate consistent, high-quality code.
  - Follow best practices and coding standards.
  - Adapt to specialized tools and workflows.

# Context and Requirements



Project Context and Requirements Instructions (project-instructions.md)

- Project specific guidance
- Project Overview
- Technology Stack
- File Structure

# Code Quality Instructions



- **Code templates** ([vue-code-template-instructions.md](#))
- **Error handling guidelines** ([vue-error-handling-instructions.md](#))
- **Validation guidelines** ([vue-validation-instructions.md](#))
- **Testing guidelines** ([vue-testing-instructions.md](#))
- **Cloud integration guidelines** ([vue-azure-integration-instructions.md](#))
- **API documentation guidelines** ([vue-api-documentation-instructions.md](#))

# Practices and Coding Standards



- **Code style and conventions** ([vue-style-instructions.md](#))
- **Best practices** ([practices-instructions.md](#))

# Specialized Tools and Workflow



- **Commands and scripts** (commands-instructions.md)
- **Specialized tools and rules** (tools-instructions.md)

# Updating Instructions



- Standards and practices change over time
- Keeping your instructions updated with changes
- Keeping your code compliant with instructions

# Context Rules



- @suggest Rule - Offer code suggestions
- @security Rule - Follow security best practices
- @performance Rule - Optimize for performance
- @accessibility Rule - Follow accessibility standards
- @testing Rule - Include test considerations
- @patterns Rule - Use design patterns
- @compatibility Rule - Consider browser/platform compatibility
- @azure Rule - Use Azure Best Practices

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Case Studies



- Case studies showing improved software reliability.
  - **GitHub Copilot Research Study (2022)**
  - **RedMonk Analysis**
  - **DX (Developer Experience) Survey**
  - **Accenture Case Study**

# Copilot Research Study (2022)

- Source: [GitHub Next: Quantifying GitHub Copilot's Impact on Developer Productivity and Satisfaction](#)
- Findings:
  - Developers using Copilot completed tasks 55% faster on average.
  - 88% of surveyed developers reported higher productivity.
  - Improved code quality and reliability due to fewer manual errors and more consistent code patterns.

# RedMonk Analysis



- **Source:** RedMonk: The Impact of GitHub Copilot on Software Development
- **Findings:**
  - Organizations using Copilot saw a reduction in code review cycles and post-deployment bugs.
  - Copilot's code suggestions helped standardize codebases, improving maintainability and reliability.

# DX Survey



- **Source:** GitHub Copilot Impact Report
- **Findings:**
  - 96% of developers reported faster task completion.
  - 73% said Copilot helped them stay in the flow, reducing context-switching errors.
  - Teams observed more reliable code due to Copilot's suggestions aligning with best practices.

# Accenture Case Study



- Source: [Accenture: How GitHub Copilot is helping Accenture deliver better software, faster](#)
- Findings:
  - Accenture teams reported fewer bugs and improved code consistency.
  - Faster onboarding for new developers, reducing the risk of reliability issues from misunderstandings.

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Vibes



- What is vibe coding?
  - Natural-language prompts drive code generation, not hand-crafted syntax.
  - The developer focuses on guiding, testing, and refining AI output rather than micromanaging every line.
  - Emphasizes experimentation and rapid prototyping over upfront design and strict correctness.
  - Code is often accepted with minimal review, allowing fast iteration but sacrificing initial structure.

See recording “[Vibe Coding Done Right: AI-Assisted Productivity with Engineering Discipline](#)” by my CODE Consulting colleague Philipp Bauer

# Before AI



- Create a restore point
- Consider branching
- Have a test plan

# AI Assisted Software Development



- AI code review
- Generating documentation
- Generating tests
- Generating code
- Managing change

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Generating Documentation



- Code comments
  - Function Header
  - Inline
- Update read me
- Generate architecture documentation

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# AI Code Review



- Explaining
- Examine the codebase and identify code that does not conform to the GitHub instructions

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - **Generating Tests**
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Automated Testing



- Using AI for automated testing
  - unit tests
  - bug detection
- How AI analyzes logs and suggests fixes before deployment

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Fixing Compliance Issues



- Address one of the findings from the reported deviations
- Implement the same changes using two different models
- Compare the implementations and report the pros and cons of each implementation

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Managing Change



- Path to production
- Code commits
- Code reviews
- Pull requests
- Issue identification and prioritization

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Risks of Bias



- What do we mean by bias in generated code
- Examples of bias in AI-generated code
- How to mitigate bias

# Agenda



- Managing Context
- Case Studies
- AI Assisted Development
  - Generating Documentation
  - Reviewing Existing Code
  - Generating Tests
  - Generating Code
  - Managing Change
- Bias Management
- Future Considerations

# Future Considerations

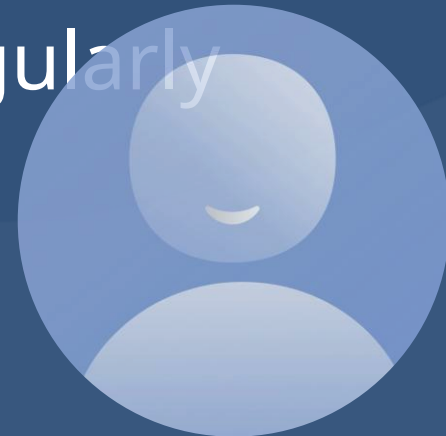


- The evolving role of AI in development.
- AI engineers vs AI engineers
  - AI engineers that build AI solutions
  - AI agents that build solutions
- The evolving role of development in the AI world
  - Is vibe coding, coding?

# Conclusion



- You control the generated code more than you might expect
- Best practices for integrating AI tools responsibly
  - Code reviews
  - Test code reviews
  - Review and comment
  - Use instructions and keep them updated
  - Evaluate your code against the instructions regularly
- Show me the prompts



# Next Steps



- NextGen AI ([`https://nextgenaiconf.com/#!/`](https://nextgenaiconf.com/#!/))
- Blog ([`https://www.codemag.com/blog/AIPractitioner/`](https://www.codemag.com/blog/AIPractitioner/))

# ONE CODE

CODE 30  
YEARS



<https://www.codemag.com/>

Survey!

<https://bit.ly/CP-07-2025Survey>



Subscribe!

<https://bit.ly/CP-07-2025Subscribe>



# Q&A



## Contact us with questions!

### CODE Contact

[codemag.com](http://codemag.com)  
[info@codemag.com](mailto:info@codemag.com)  
[facebook.com/codemag](https://facebook.com/codemag)  
[twitter.com/codemagazine](https://twitter.com/codemagazine)

### Presenter Contact

[John.miller@codestaffing.com](mailto:John.miller@codestaffing.com)