Vibe Coding: Automated software Generation

Relevant cells

A1	A2
B1	B2
C1	C2

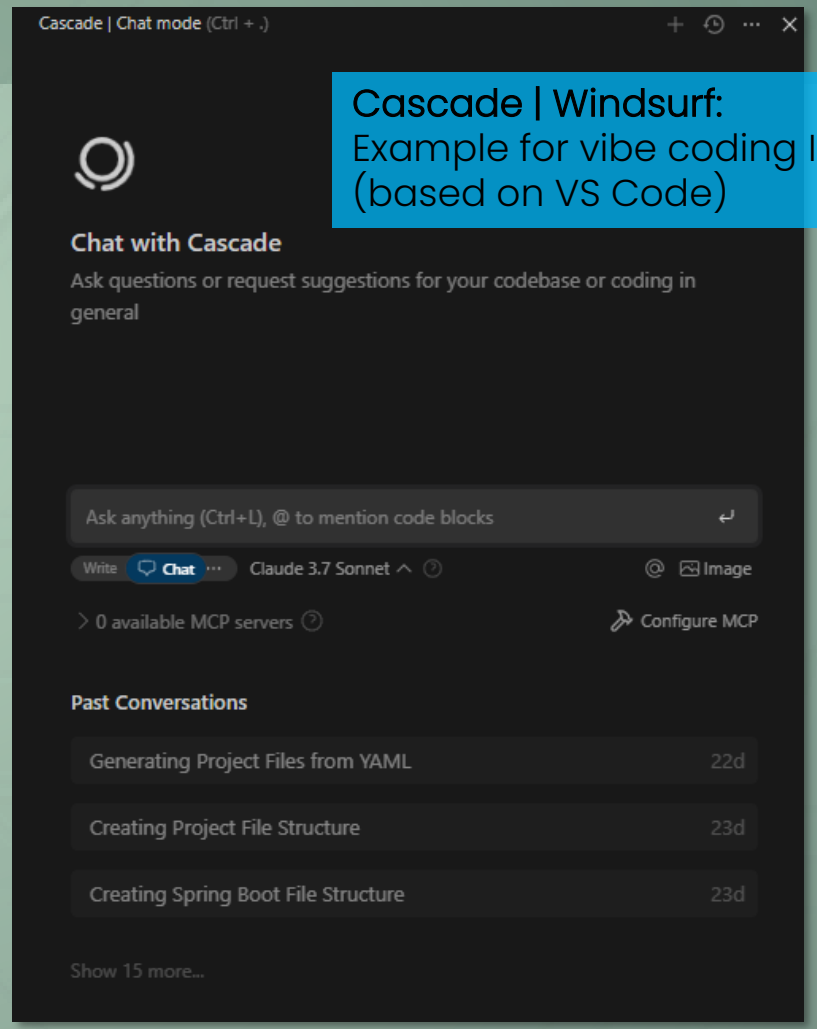
Transforming software from permanent artifacts to fluid, adaptable solutions

Development Workflow

- Describe intent in natural language
- AI analyzes and generates architecture
- Instant code synthesis and testing
- Deploy, modify, or discard rapidly

New Process Automation Perspectives

- Disposable Software:
  Generate, test, discard without loss
- Instant Adaptation:
  Real-time requirement changes
- Experimental Freedom:
  Zero-cost exploration
- Cognitive Offloading:
  Focus on problems, not syntax



Cascade | Windsurf:
Example for vibe coding IDE (based on VS Code)

Relevant cells

A1	A2
B1	B2
C1	C2

Pitfalls of Automated Code Generation

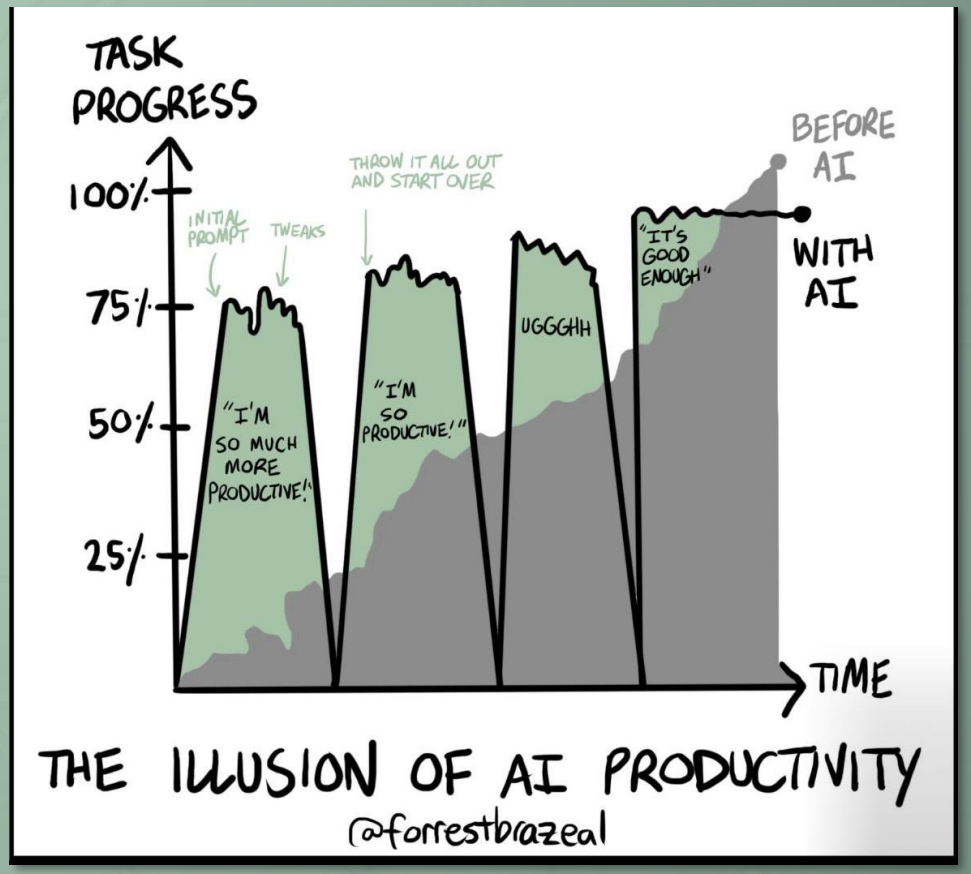
Automated code generation is fast,
but manual expertise is essential to finish and ensure quality, security, and reliability

The "First 95% vs. Last 5%" Trap

- Automated tools quickly generate most of the code (first 95%)
- The hardest and most critical 5% (edge cases, integration, compliance) often fails or is incomplete
- This may lead to iterative seeming progress and setbacks

Key Risks

- Hidden Bugs:
  Subtle errors, especially in complex logic
- Security Issues:
  Missed validations, outdated dependencies
- False Progress:
  Gives illusion of near-completion, but last steps are hardest
- Maintenance:
  Generated code can be hard to read or fix



12 Principles of Best Practice for Vibe Coding

Relevant cells

A1	A2
B1	B2
C1	C2

Vibe coding can accelerate progress, but only when used with care.
Follow these principles to ensure quality, clarity, and control.

1. Assess Your Readiness

- Ensure you have foundational programming knowledge before starting.
- Only use LLMs if you understand the language, tools, and domain.

2. Define Clear Objectives

- Start each session with a well-defined goal or problem statement.
- Guide both your thinking and the model's responses.

12 Principles of Best Practice for Vibe Coding

Relevant cells

A1	A2
B1	B2
C1	C2

Vibe coding can accelerate progress, but only when used with care.
Follow these principles to ensure quality, clarity, and control.

3. Provide Context and Use Prompt Templates

- Include relevant code, file structures, and business logic in your prompts.
- Use reusable templates to improve precision and consistency.

4. Use Different LLMs Like a Team

- Treat each LLM as a specialized team member, with strengths tailored to specific tasks.
- Switch between models depending on the task's requirements, and integrate evolving LLMs with targeted capabilities as needed.

12 Principles of Best Practice for Vibe Coding

Relevant cells

A1	A2
B1	B2
C1	C2

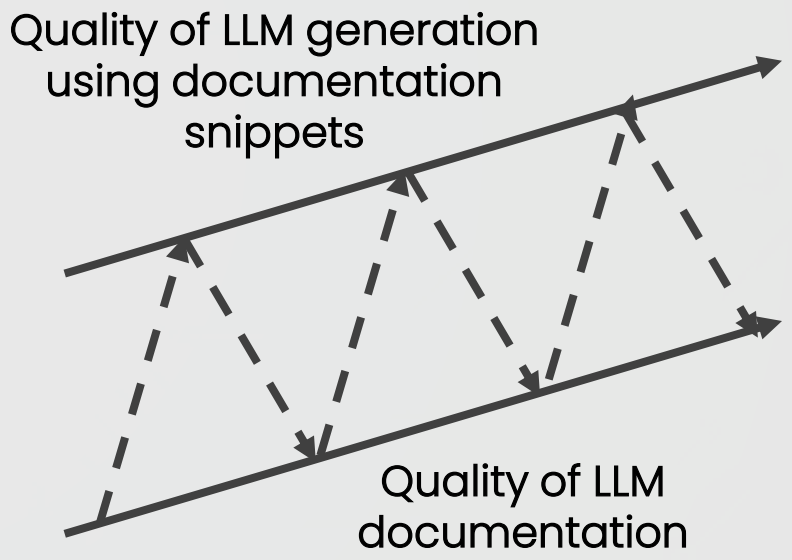
Vibe coding also introduces risks, especially when coders accept outputs they don't fully understand. To maximize quality and reliability, follow these ten principles:

5. Inject Company Documentation into Prompts

- Consult and incorporate company documentation, APIs, and contracts.
- Ensure that the generated code aligns with internal standards.

6. Create a Reinforcing Documentation Loop

- Keep records of architectural decisions, prompt strategies, and lessons learned.
- Use this documentation to refine future interactions with LLMs.



12 Principles of Best Practice for Vibe Coding

Relevant cells

A1	A2
B1	B2
C1	C2

Vibe coding also introduces risks, especially when coders accept outputs they don't fully understand. To maximize quality and reliability, follow these ten principles:

7. Recognize when vibe coding is not the right tool for the job

- Rare programming languages or frameworks - Limited training data leads to unreliable suggestions and outdated patterns
- Highly specialized tasks - Domain-specific algorithms or niche implementations where LLMs lack sufficient expertise

8. Double-Check for Hallucinations and Weaknesses in Logic

- LLMs may produce plausible but incorrect answers (hallucinations).
- Double-check responses, especially for numeric or complex logic-based queries.

https://www.agora.software/en/hallucinations-llm-and-conversational-ai/

Relevant cells

A1	A2
B1	B2
C1	C2

12 Principles of Best Practice for Vibe Coding

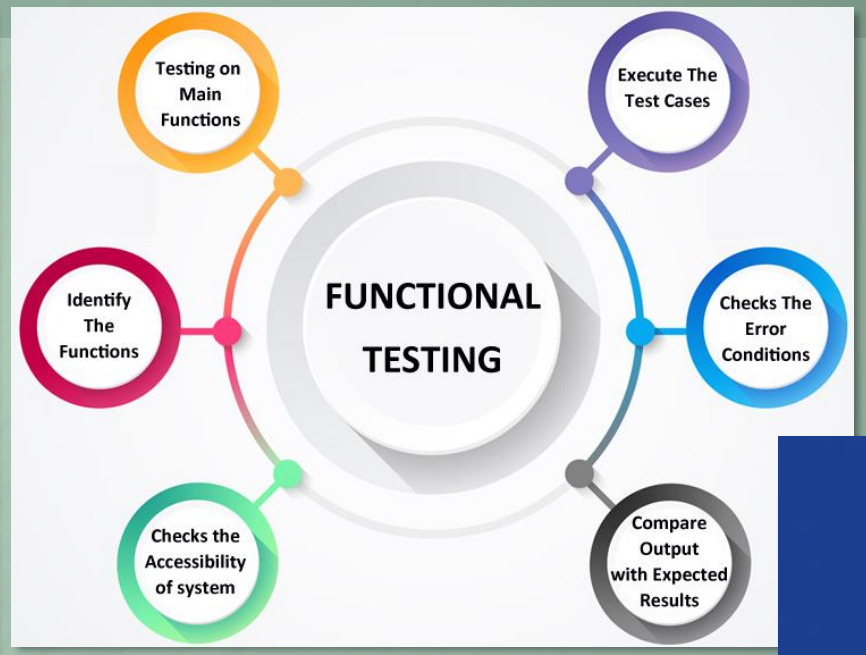
Vibe coding also introduces risks, especially when coders accept outputs they don't fully understand. To maximize quality and reliability, follow these ten principles:

9. Test Thoroughly

- Develop unit, integration, and edge-case tests for validation.
- Ensure robustness and correctness of the AI-generated code.

10. Enforce IT-Security Standards

- Audit AI-generated code for security flaws, such as input validation and data privacy issues.
- Ensure secure handling of sensitive data and internal-external system boundaries.



Relevant cells

A1	A2
B1	B2
C1	C2

12 Principles of Best Practice for Vibe Coding

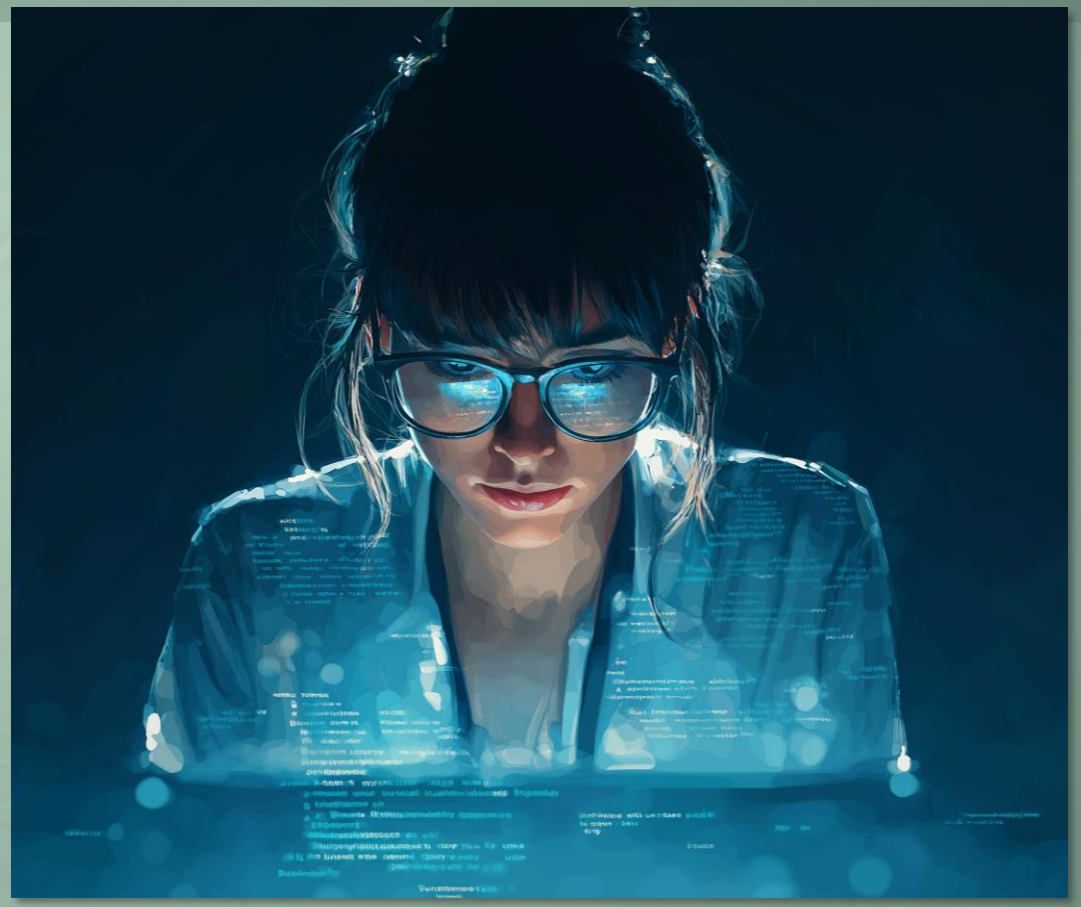
Vibe coding also introduces risks, especially when coders accept outputs they don't fully understand. To maximize quality and reliability, follow these ten principles:

11. Iterate in Small Steps

- Break down problems and generate small, manageable code segments.
- Review, test, and inspect each block before proceeding.

12. Work in Focused Sessions

- Eliminate distractions during coding.
- Dedicate uninterrupted time for better comprehension and quality control.



Use Cases of Vibe Coding

Real-World Applications of Automated Software Generation

Relevant cells

A1	A2
B1	B2
C1	C2

Rapid Prototyping

Instantly transform business ideas into functional prototypes for validation and testing.

EXAMPLE INPUT

"Create a task management app with team collaboration and deadline tracking"

Enterprise Automation

Create custom internal tools and workflow automation without IT bottlenecks.

EXAMPLE INPUT

"Employee onboarding system with document processing and approval workflows"

Educational Tools

Generate interactive learning platforms and assessment tools tailored to specific curricula.

EXAMPLE INPUT

"Interactive physics simulator for projectile motion with real-time visualization"

Scientific Computing

Generate specialized analysis tools and simulations for research without programming expertise.

EXAMPLE INPUT

"Build a Monte Carlo simulation for protein folding dynamics"

Startup MVP Development

Launch minimum viable products in days instead of months, enabling rapid market testing.

EXAMPLE INPUT

"E-commerce platform with payment processing and inventory management"

Emergency Solutions

Rapidly deploy crisis management tools and temporary solutions during emergencies.

EXAMPLE INPUT

"Contact tracing app with privacy controls and health status reporting"