MODUS CREATE

Vibe coding at enterprise scale

Beyond the hype. Learnings from a 10+ people 3-months experiment.

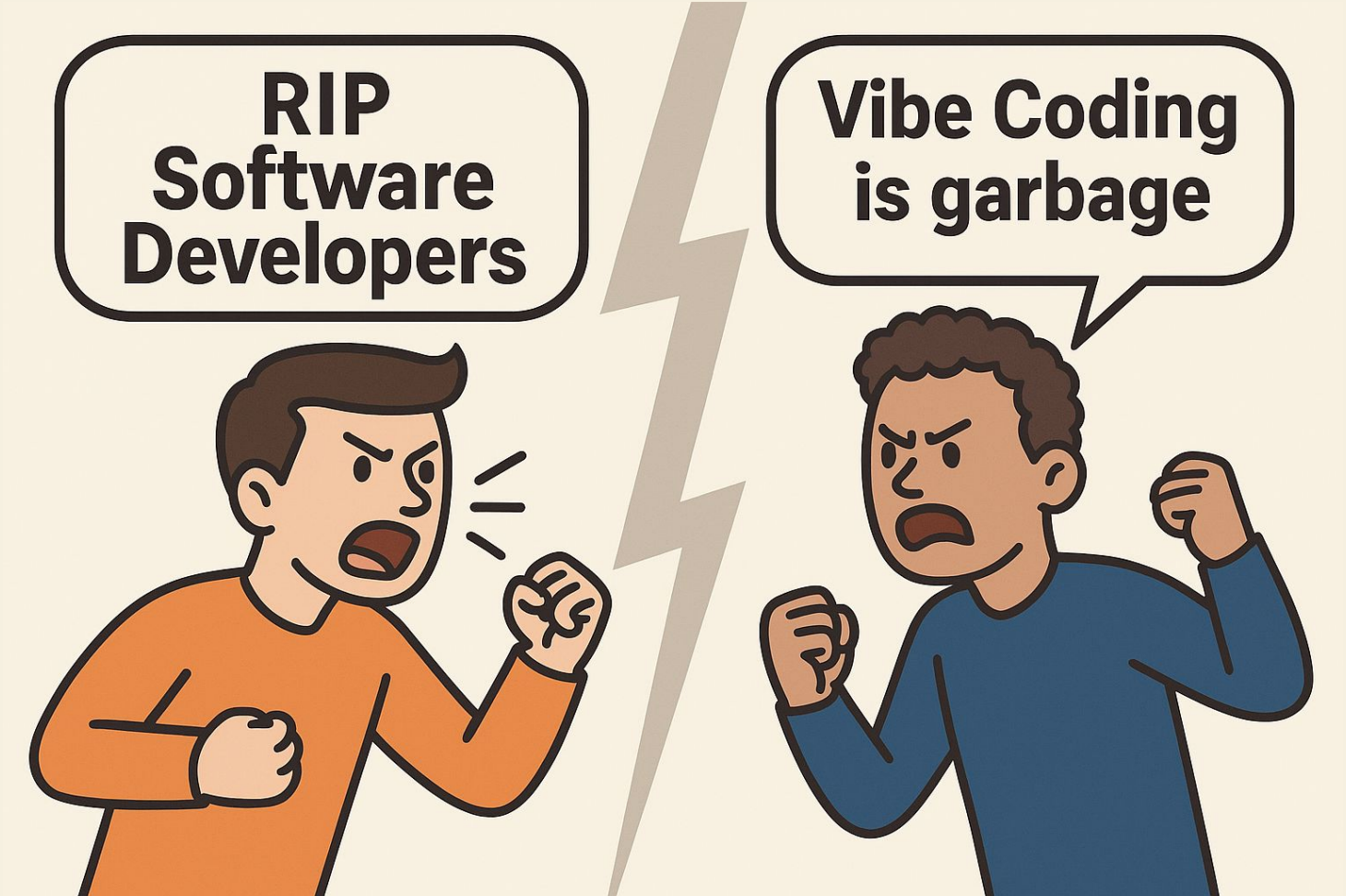
2025

Vibe coding at enterprise scale

Beyond the hype. Learnings from a 10+ people 3-months experiment.

Wesley Fuchter

- From Sao Ludgero, Brazil
- Senior Principal Engineer at Modus Create
- Working with Research & Development
- 15 years in full-stack software development
- AWS Solutions Architect
- AI-powered development practitioner
- Intersection between product, engineering, and people


RIP Software Developers
Vibe Coding is garbage

The way we build software is changing.

We vibe coded a project to see if AI could actually speed up shipping software.

No Hype.
Just learnings from a 10+ software professionals and 3-months experiment.

Hypotheses

01 Can we build the application 50% faster with a 30% reduced team compared to the traditional method?

02 Will AI coding tools mitigate dependencies on deep expertise in specific tech?

03 AI-generated code will at least match the same quality as human written code?

04 Will the team learn valuable lessons on how we integrate AI into our day-to-day workflows?

The Experiment Structure

- Healthcare product
- 10 people, two squads
- Documentation and Timesheets
- Powered by Ionic and AWS

The Results

45%

AI FASTER

30%

REDUCED HEADCOUNT

AI

LANGUAGE AGNOSTIC

QUALITY

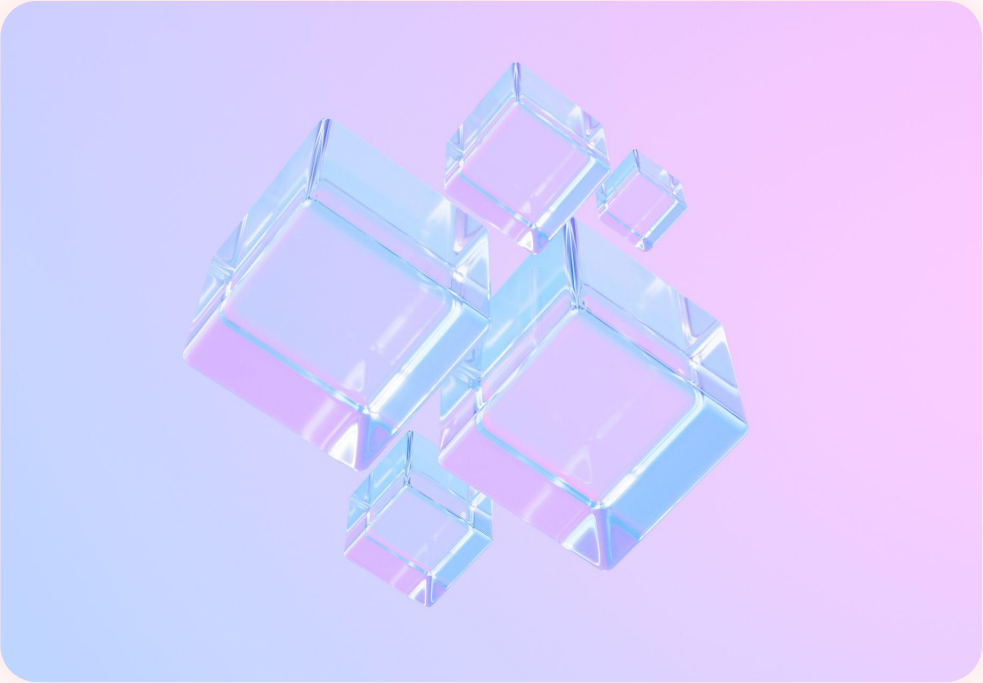
AS EXPECTED

Efficiency isn’t about cutting teams — it’s about freeing them to build more.

Feature	Time DIY	Time AI	% Faster
Project Setup Hours	34 hrs	36 hrs	**-6%**
User Authentication and Login Options	42 hrs	27 hrs	**36%**
User Registration and Account Creation	67 hrs	2 hrs	**97%**
Personalized Home Screen with Name and Avatar	28 hrs	2 hrs	**93%**
Personalized Home Screen with Latest Medical Reports and AI Assistance	102 hrs	43 hrs	**58%**
AI Health Assistant on Home Screen	74 hrs	74 hrs	**0%**
Upload Medical Report File for Easy Access	71 hrs	22 hrs	**69%**
Extract Key Medical Information from Uploaded Reports	110 hrs	20 hrs	**82%**
Reports Page	34 hrs	46 hrs	**-35%**

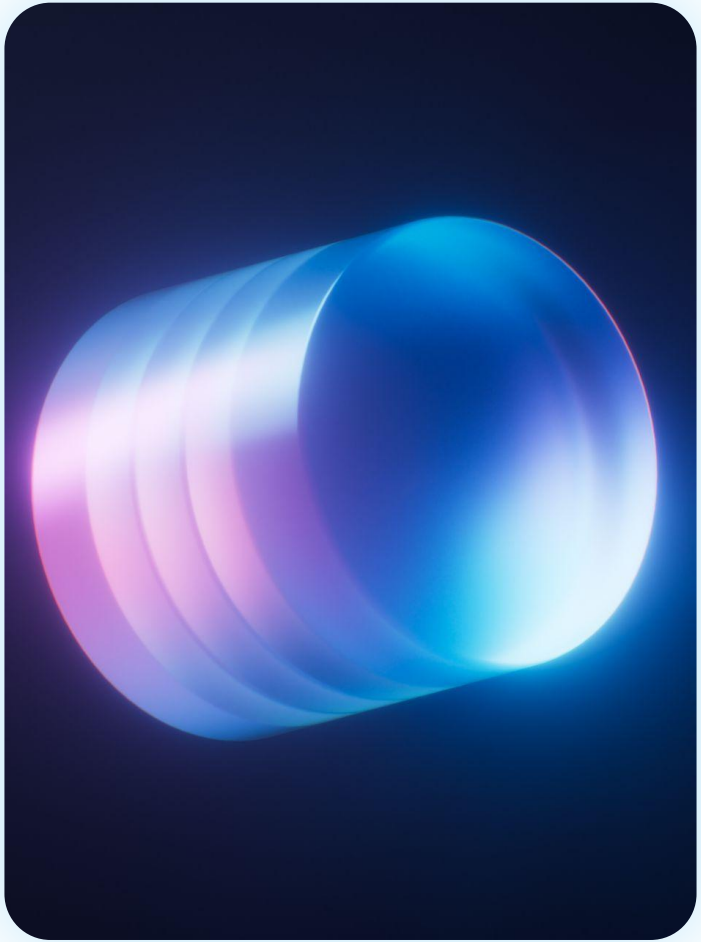
Nothing is Perfect

WE ALSO LEARNED HOW TO NOT USE AI CODING TOOLS



THE "WE DON'T KNOW WHAT WE DON'T KNOW" PROBLEM

AI can probably generate plans for a skyscraper, but you'd still want a structural engineer in the loop to make sure that the building will stand.



How We See Vibe Coding

DELIVERS THE RESULTS WITH THE QUALITY WE EXPECT.

Vibe Coding Hype

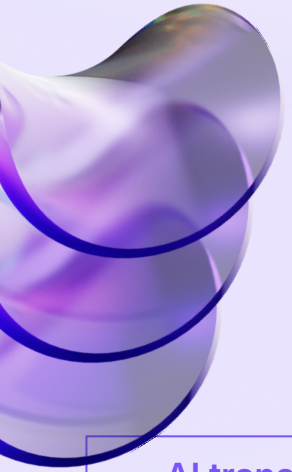
- 100% Reliance on AI
- No human review, no approve/reject code
- Not paying attention to the generated code
- No pre-planning and human reasoning

Agentic Coding

- Human being is the rockstar, AI is a tool
- Implementation planning
- Atomic Level Tasks
- Review & Validation Loops
- Human is taking care of the prompting.

AI is not a replacement for architectural clarity or domain expertise — but a multiplier of it.

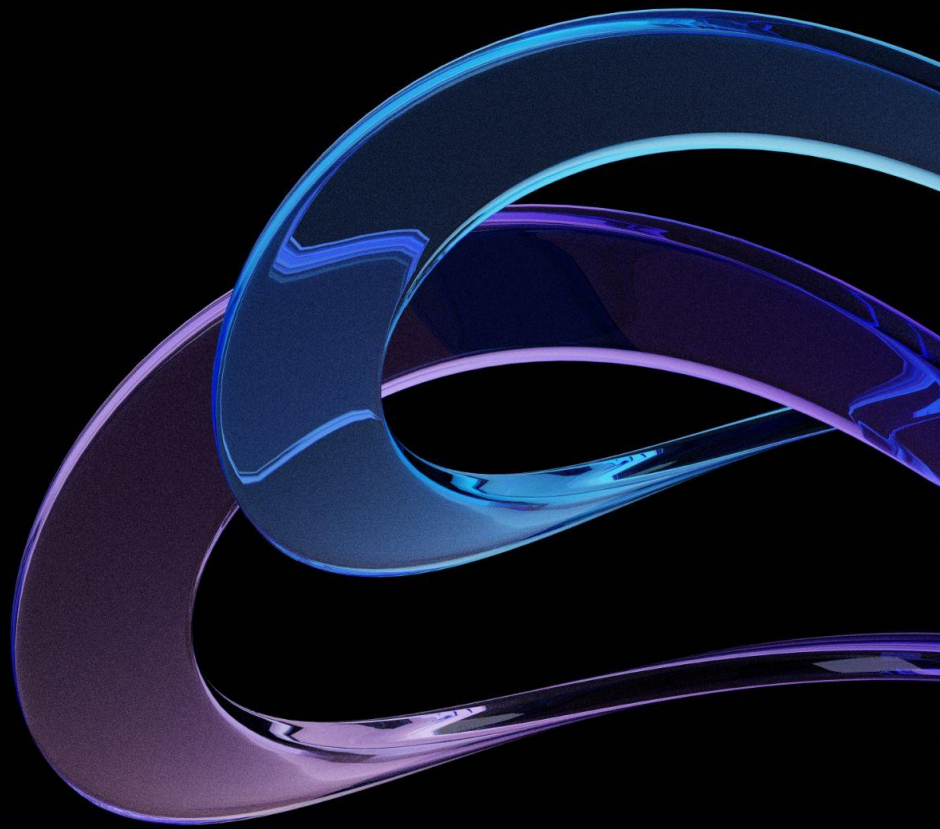
Why This Changes the Game



AI transforms how teams build by enabling faster delivery, efficient teamwork, clearer thinking, and adaptability, empowering developers to ship better products confidently in less time.

Speed	**Efficiency**
Clarity	**Adaptability**

Learnings & Workflows



Atomic Level Tasks

Each task becomes a prompt

Features broken down into tasks

Validation loop after each task

Prompt, Feedback, Iterate

Implementation plan as markdown

AI to check done tasks

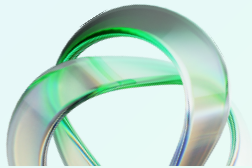
Git commit often, revert when needed

As a user, I want to see my order history so I can track my past purchases.

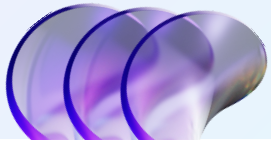
- [ ] Design and implement "/orders" GET API to fetch user-specific data
- [ ] Create database query optimized with pagination
- [ ] Implement order history UI with basic info: date, total, status
- [ ] Create detail view for individual order with line items
- [ ] Add loading, error, and empty state handling
- [ ] Write unit test for the API response shape
- [ ] Write Cypress test for frontend navigation and rendering

Rich Context

Optimize context to reduce hallucinations

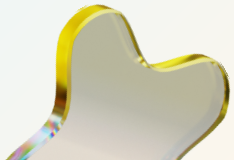


Clear the context often



Use instruction files and memory banks

Guide AI with mocks and project documentation



Leverage MCPs to feed external references

MCP Use Cases

MCP	Use Case
Figma MCP	Fetch elements from Figma while building UI components
Github MCP	Collect feedback from pull requests and turn them into a implementation plan
Database MCPs	Add the database structure to the context to get the AI to build the application code
CloudWatch MCP	Add logs from environments inside the context to use the agent to troubleshoot
Memory Bank MCP	Access a centralized memory bank that shared among agents for unified context

Good Prompts

- Clear, scoped prompts yield accurate, usable code
- Describe expected outputs: structure, naming, tests, edge cases
- Avoid vague goals. Specify what “done” looks like
- State the decisions you've made in the prompt, don't let AI make its own decisions
- Use consistent language and terms according to the documentation that exists in the context

Examples

Multi-Shot Prompt

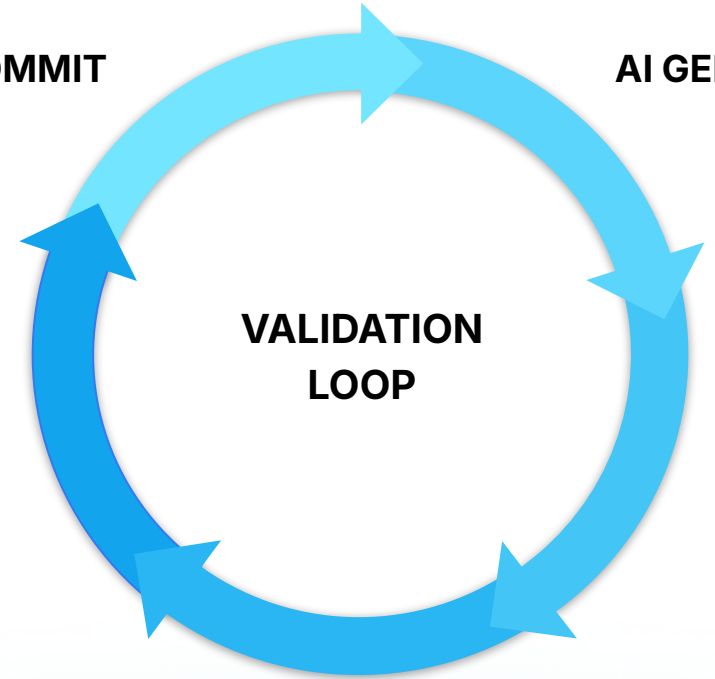
"I want to generate a TypeScript function called *filterVisibleTasks* that accepts a list of tasks and a user role, and returns only the tasks visible to that role. Tasks have properties like *visibility*, *assignedTo*, and *status*. Don't use external libraries. Include 2 unit tests in the same file using Vitest. Use the function *filterVisibleUsers* as an example."

10 Interactions Prompt

"Write a TypeScript function that deduplicates an array of user objects based on *email*. Iterate on your solution 10 times. After each iteration, review the previous output, look for improvements in clarity, performance, and correctness. Only show me the final version, along with a summary of how it evolved."

Three Experts Prompt

"You are a team of three experts: a frontend engineer, a backend engineer, and a product designer. Design a feature that lets users save articles for offline reading. Each expert should describe the feature from their point of view: UI/UX design, API/data modeling, and sync logic. Use markdown to separate answers."


GIT COMMIT
AI GENERATES CODE
VALIDATION LOOP
HUMAN MANUAL TEST
HUMAN CODE REVIEW
AUTOMATIC VALIDATIONS

Test-Driven Development

- AI plans and improves test coverage
- Tests come first — implementation follows
- Code generated to pass verified tests
- Safe, incremental logic development
- TDD rigor × AI speed

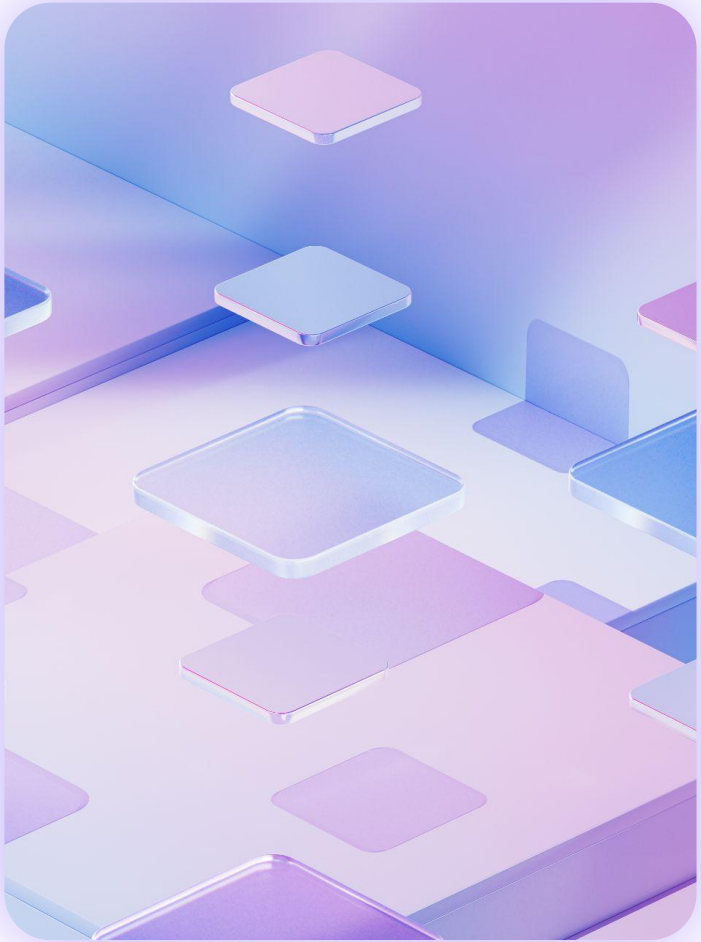
```
# TDD Plan: Calculate Invoice Total
## Setup
- [ ] Create the basic function structure `calculate_invoice_total(items,
customer_location)`
- [ ] Set up testing framework

## Basic Functionality
- [ ] Write test for empty items list returning 0
- [ ] Implement logic to return 0 for empty items list
- [ ] Write test for calculating subtotal with a single non-taxable item
- [ ] Implement logic to calculate basic subtotal (price * quantity)
- [ ] Write test for calculating subtotal with multiple non-taxable items
- [ ] Update implementation to handle multiple items

## Tax Rules
- [ ] Write test for default tax rate (5%) with a taxable item
- [ ] Implement logic to apply default tax rate
- [ ] Write test for NY tax rate (8.875%) with taxable item
- [ ] Update implementation to handle NY tax rate
- [ ] Write test for CA tax rate (7.25%) with taxable item
- [ ] Update implementation to handle CA tax rate
- [ ] Write test for mixing taxable and non-taxable items
- [ ] Update implementation to only apply tax to taxable items
```

Visual Prompting

- AI “sees” the product — mockups, UIs, diagrams
- Turns visuals into accurate, ready-to-code output
- Connects design tools and live DOM for precise iteration through MCPs



Debugging

- Diagnose and explain errors in seconds
- Pull logs and stack traces to AI context with MCPs
- Propose and validate fixes before coding
- Integrate with tests and linters for safe iteration

Exploratory & Refactoring

- Map complex systems and identify technical debt
- Plan refactors with AI before touching code
- Refactor in small, testable increments
- Validate and commit often using feedback loops
- Leave code cleaner than you found it

AI isn’t magic, but with clear prompts, narrow focused tasks, rich context, and validation loop, it can be your unfair advantage.

Handbook

Report

MODUS CREATE

THANK YOU!

Let’s make something great together.