AI-Powered Solutions That Fit,
Not Templates That Don't.

iwconnect.com
Global Presence: USA • EU • MK • RS
20+ Years • 350+ Professionals

WE WENT FROM CLIENT BRIEF TO DEMO IN 96 HOURS: STEAL OUR VIBE CODING PLAYBOOK

A STEP-BY-STEP PLAYBOOK FOR TRANSFORMING COMPLEX REQUIREMENTS INTO WORKING SOFTWARE AT 10X SPEED

THE MEETING THAT CHANGED HOW WE BUILD SOFTWARE

9:30 AM, MONDAY.

An international luxury retailer operating within the travel hospitality sector sends over their requirements. A few pages on a Wishlist, not more than five. No structure. Just bullets describing how they manage thousands of crew members across multiple vessels.

They'd already bought software that does the work, but the complexity and user friendliness is not something they favor for. They've also built Excel sheets that weren't built for scale. Their current system required filling out fields they didn't need just to complete forms with two relevant inputs.

The first course of action of most agencies would be to schedule discovery meetings. Create project plans. Promise delivery in 3-6 months.

9:00 AM, FRIDAY.

Marko, our lead developer at :IWConnect, is screen-sharing with their executive team. But he's not showing mockups or wireframes. He's demonstrating their complete management system. Their actual documents. Their real rotation schedules. Their live data.

The EVP interrupts:

> "Wait, can we have something like a warning in place, when someone removes a person from a certain position, so the system can blink or something to alert the manager that something is missing".

Marko adds it. During the call. In real-time.

THE VERDICT, MOMENTS AFTER THE DEMO ENDED:

> "I don't think any other company would have a better offer for software that covers all these options."

Four days. One developer. Zero traditional meetings.

This is vibe coding - our approach to AI-accelerated development that's delivering enterprise software in the time it takes most agencies to schedule their kickoff meeting.

WHY THIS PLAYBOOK EXISTS

We're not hiding our methodology. We're not protecting "trade secrets."

Here's why: The traditional software development model is broken. Clients wait months for something that might work. Developers burn out writing boilerplate code. Agencies pad timelines because that's how it's always been done.

We found a better way. And frankly, the market is big enough for everyone who's willing to evolve.

This playbook shows you exactly how Marko - one developer with the right AI workflow-delivered what typically requires a full team and three months. If you're evaluating software partners, this is what modern development looks like. If you're a developer, this is how to amplify your output by 10x.

MEET MARKO: THE DEVELOPER WHO TREATS AI LIKE A COLLEAGUE/FRIEND FROM WORK

Before we dive into the methodology, you should understand who developed this approach. Marko isn't a junior developer who learned to code from ChatGPT. He's a Lead Technical developer with over a decade of experience who realized something fundamental: AI isn't a replacement for expertise - it's an amplifier of it.

> "I talk to Claude like I'd talk to a remote colleague. 'Friend, we broke this flow when moving the steps.' And it understands exactly what I mean. But you need to know what you're building first."

Marko had previously built an employee benefits demo in three days. A recruitment platform prototype in five. When this opportunity landed on his desk, he knew exactly what to do.

THE VIBE CODING METHOD: CORE PRINCIPLES

We define vibe coding with AI as a colleague with memory and context, not as a code completion tool. You give it context links, planning documents, and screenshots. You ask for plans and targeted edits. You iterate in short, visible steps.

The difference:

TRADITIONAL AI CODING:

Initial Prompt:

> "Generate a user management system"

- Copy-paste the result
- Debug for hours
- Realize it doesn't fit your architecture

VIBE CODING:

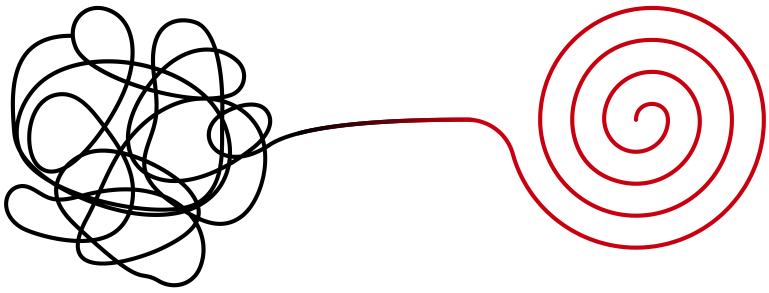
Initial Prompt:

> "We're implementing user management. Requirements are in user_management.md. Read it, summarize your understanding, ask three clarifying questions, then focus on the role-based permissions in section 2."

- Review the plan together
- Iterate with specific feedback
- Ship coherent, architected features

MARKO'S PLAYBOOK: THE EXACT PROCESS

PHASE 0: TRANSFORM CHAOS INTO STRUCTURE (2 HOURS)



The Input: Vague requirements document
The Output: Structured 50-page specification

Marko's Move:

Upload to Claude >>> "Review this request. We need to create an admin platform based on modules and roles/permissions. Give me an initial plan."

Within 20 minutes, Claude mapped:

- 10 defined modules
- 5 user roles
- Complete feature matrix
- Technical requirements
- Integration points

> "I don't start coding until Claude and I agree on what we're building. Those 2 hours of planning save 200 hours of revisions."

PHASE 1: ESTABLISH THE RULES (4 HOURS)

Planning.md

Goal
Single platform for crew management replacing Excel + bad software

Actors
- Crew Members: Upload documents, check assignments
- Managers: Review documents, plan rotations
- Medical Team: Clear health requirements
- Fleet Managers: Assign positions across vessels
- HR: Oversee entire process

Core Entities
- crew_member
- document (19 types)
- vessel
- position
- rotation_schedule

Risks
- Complex drag-drop interactions
- Real-time notifications
- Document approval workflows

Rules.md

Technical Decisions
- React JS with localStorage (no backend for demo)
- Bootstrap for consistent styling
- Reusable components for all forms
- Mobile-responsive from start
- Real data, no lorem ipsum

UI Patterns
- Status colors: Missing=Orange, Complete=Green, Pending=Yellow
- All modals follow same structure
- Drag-drop shows visual feedback
- Forms validate on blur

Variables_Guide.md

Naming Conventions
- camelCase for functions
- PascalCase for components
- snake_case for data keys
- Semantic names only (no 'temp', 'data', 'item')

Component Contract
- Every component must:
- Accept props object
- Return valid HTML
- Handle loading states
- Be mobile responsive

> "These documents become the constitution. Every prompt references them. 'Follow Rules.md' appears in 90% of my commands."

PHASE 2: STEAL THE RIGHT PATTERNS (2 HOURS)

The Insight: Don't design from scratch. Find what works.

Marko's approach:

1. Ask Claude: "Find me a free or a demo HR/crew management admin panel online that I can access and interact with"
2. Claude suggests many, Marko picks the one that makes most sense for him and he can visualize his idea into a demo application
3. Create trial account
4. Take 2 screenshots: Dashboard + Profile page
5. Feedback to Claude: "We're building with this look and feel"

> "I'm not copying their product. I'm borrowing proven UX patterns. Why reinvent what thousands of users have already validated?"

PHASE 3: BUILD IN CONVERSATION LOOPS (30+ HOURS)

This is where vibe coding diverges completely from traditional development.

The Conversation Pattern:

Marko: We're implementing [FEATURE]. Details in[DOCUMENT].md

Claude: I understand you want [SUMMARY]. Questions: [1, 2, 3]

Marko: Correct, except [CLARIFICATION]

Claude: I'll implement by [APPROACH]

Marko: Perfect, you're doing great

[**Claude** generates code]

Marko: This broke when [SCENARIO] + [SCREENSHOT]

Claude: I see the issue. The problem is [EXPLANATION]. Fixing...

Real Example from the Project:

Marko: When moving with the steps, they are not completed before we mark them as missing. Like in the image, onboarding docs and medical are skipped and should be flagged orange. Can you check?

Claude: [Analyzes image, identifies state management issue, fixes in 47 files]

The Power of Positive Feedback:
Throughout the process, Marko regularly communicated with the AI:

- "Perfect"
- "You're doing great"
- "Good work"

This isn't politeness. It's prompt engineering. Claude maintains context better with positive reinforcement.

PHASE 4: THE SCREENSHOT METHOD (CONTINUOUS)

Words fail. Images don't.

Marko's Screenshot Workflow:

1. Build something close
2. Screenshot it
3. Annotate the problem
4. Tell Claude: "Make it work like this image"

Example: Instead of: "The drag and drop should show visual feedback when hovering" Marko does: [Screenshot with arrow] "When dragging here, highlight the drop zone like this"

> "I probably shared 50+ screenshots during the build. Each one saved an hour of explanation."

PHASE 5: LOAD REAL DATA EARLY (DAY 3)

Most demos use fake data. Marko loaded their actual day-to-day realistic data, real documents, live operation information.

Why This Matters:

- Client sees THEIR system, not A system
- Edge cases surface immediately
- The demo becomes emotionally real
- Trust multiplies

> "When they saw their real profile data - receiving notifications about expiring documents, everything changed. It wasn't a demo anymore. It was their future."

PHASE 6: DEMO LIKE YOU'RE ALREADY LIVE

The Traditional Demo: "Here's what we'll build..." "This would connect to..." "Eventually, this will..."

Marko's Demo: "Let me show you your system." "Click here to see your profile data." "Your documents are processing through this workflow."

The language matters. Present tense. Ownership. Reality.

THE METRICS THAT MATTER

Let's talk numbers from this actual project:

TRADITIONAL APPROACH:

- Discovery: 2 weeks
- Design: 3 weeks
- Development Sprint 1: 4 weeks
- Feedback and Iteration: 2 weeks
- Development Sprint 2: 3 weeks
- **Total: 14 weeks minimum**

MARKO'S VIBE CODING SPRINT:

- Brief to Structure: 2 hours
- Foundation: 10 hours
- Core Features: 11 hours
- Advanced Features: 10 hours
- Polish with Real Data: 8 hours
- **Total: 41 working hours (4 days)**

OUTPUT METRICS:

- Lines of code written: ~50,000
- Features delivered: 47
- User roles implemented: 5
- Document types handled: 19
- Drag-drop interfaces: 3
- Mobile responsive views: All

Efficiency Multiplier: One developer with vibe coding = 5-person traditional team

THE TOOLS STACK (AND WHY EACH MATTERS)

CLAUDE (ANTHROPIC)

- Role: Strategic planning partner
- Why Claude: Superior context handling, understands nuance
- Cost: $20/month
- ROI: Replaces 200 hours of planning meetings

CURSOR (AI-POWERED IDE)

- Role: Code implementation environment
- Why Cursor: Understands entire codebase context
- Cost: $20/month
- ROI: 10x coding speed increase

THE INTEGRATION MAGIC:

1. Plan in Claude (creates documents)
2. Reference documents in Cursor
3. Cursor implements with full context
4. Screenshots back to Claude for review
5. Iterate until perfect

> "All of this is a $40/month spend on AI tools. They saved me 160 hours. That's 25 cents per hour saved."

WHAT THIS MEANS FOR YOUR BUSINESS

IF YOU'RE EVALUATING SOFTWARE PARTNERS:

Questions to ask agencies:

1. "Can you show us a working demo in under a week?"
2. "Do you use AI acceleration in your development process?"
3. "What's your hourly output of production code?"
4. "Can you make changes during our demo call?"

If they can't answer confidently, you're looking at 2019 methodology with 2025 pricing.

IF YOU'RE A DEVELOPER:

Stop writing boilerplate. Stop debugging syntax. Stop rebuilding solved problems.

Start conversing with AI. Start reviewing instead of writing. Start shipping 10x faster.

IF YOU'RE A DEVELOPMENT AGENCY:

Your competition isn't other agencies anymore. It's one developer with the right AI workflow delivering in days what takes you months.

Adapt or become irrelevant. There's no middle ground.

THE PLAYBOOK TEMPLATE: YOUR STARTING IMPLEMENTATION GUIDE

WEEK 1: FOUNDATION

- ☑ Set up Claude + Cursor
- ☑ Create your first Planning.md
- ☑ Establish your Rules.md
- ☑ Complete one feature with vibe coding
- ☑ Document what worked

WEEK 2: ACCELERATION

- ☑ Implement screenshot method
- ☑ Create feature templates
- ☑ Build component library
- ☑ Practice conversation loops
- ☑ Ship first complete module

WEEK 3: MASTERY

- ☑ Full project in vibe coding
- ☑ Client demo with real data
- ☑ Live changes during demo
- ☑ Document time savings
- ☑ Calculate ROI

SUCCESS METRICS:

- **5x** speed increase by Week 2
- **10x** speed increase by Week 3
- **50%** reduction in bugs
- **90%** client satisfaction on first demo

THE PLAYBOOK TEMPLATE: YOUR IMPLEMENTATION GUIDE

Here's what traditional agencies don't want you to know:
80% of software development isn't solving hard problems. It's:

- Writing boilerplate (30%)
- Debugging typos (20%)
- Rebuilding common patterns (20%)
- Waiting for clarification (10%)
- Actual problem-solving (20%)

Vibe coding inverts this:

- AI writes boilerplate instantly
- Syntax errors rarely happen
- Patterns are reused automatically
- Clarification happens in conversation
- Problem-solving becomes 80% of the work

This isn't science fiction. This is Marko's Tuesday.

YOUR NEXT MOVE

The software development industry is splitting into two camps:

Camp 1: Agencies still quoting 6-month timelines
Camp 2: Teams delivering working software in days

We're firmly in Camp 2. The question is: which camp do you want building your software?

READY TO SEE YOUR COMPLEX PLATFORM IDEA BECOME REALITY IN 96 HOURS?

Let's discuss your requirements and show you exactly what's possible with a vibe coding sprint. We'll review your brief and demonstrate how we'd transform it into working software, just like we did for the cruise ship retailer, and many more.

Book Your Vibe Coding Discovery Call

IW CONNECT®