

WordCamp  
**torino** 2026

**Vibe Coding**  
**fatto bene**



# Maurizio Pelizzone

[maurizio@mavida.com](mailto:maurizio@mavida.com)

Con oltre 15 anni di esperienza nell'ecosistema WordPress, sono specializzato nella creazione di temi e plugin custom, con un focus verticale sull'integrazione di CRM e gestionali aziendali.

Dal 2023 ho trasformato il mio workflow integrando l'Intelligenza Artificiale per ottimizzare i processi di sviluppo.

Oggi aiuto le aziende a far dialogare WordPress con il resto del mondo digitale, rendendo il lavoro più fluido, intelligente e scalabile.



# Come viene usata l'intelligenza artificiale?

<b>Consapevolezza</b>	~90%	Sanno cos'è (ne hanno sentito parlare al telegiornale).
<b>Sperimentazione</b>	~60-70%	Hanno provato ChatGPT almeno una volta per curiosità.
<b>Utilizzo Lavorativo</b>	~40%	La usano per compiti base (email, riassunti, brainstorming).
<b>Utilizzo Professionale</b>	< 15%	La usano come parte integrante del workflow (coding, automazione).

# Il nostro lavoro è a rischio?

**#AI-FOMO**

**I ruoli più coinvolti riguardano l'analisi dei dati, la scrittura e la programmazione.**

<https://www.anthropic.com/research/labor-market-impacts>

Occupation	Observed exposure	Leading automated task
Computer programmers	74.5%	Write, update, and maintain software programs
Customer service representatives	70.1%	Confer with customers to provide info, take orders, handle complaints
Data entry keyers	67.1%	Read source documents and enter data into systems
Medical record specialists	66.7%	Compile, abstract, and code patient data
Market research analysts and marketing specialists	64.8%	Prepare reports of findings, illustrating data graphically and translating complex findings into written text
Sales representatives, wholesale and manufacturing, except technical and scientific products	62.8%	Contact customers to demonstrate products and solicit orders
Financial and investment analysts	57.2%	Inform investment decisions by analyzing financial information to forecast business, industry, or economic conditions
Software quality assurance analysts and testers	51.9%	Modify software to correct errors or improve performance
Information security analysts	48.6%	Perform risk assessments and test data processing security
Computer user support specialists	46.8%	Answer user inquiries regarding computer software or hardware operation to resolve problems

<https://www.anthropic.com/research/labor-market-impacts>

**Tutti sanno cos'è l'IA, ma...**

**meno di 1 persona su 5  
sa davvero come usarla**

## - Mi dica cosa sa fare che non possa fare con ChatGPT



**AI CAN  
ALMOST DO MY JOB**



**AI CAN ALMOST  
DO MY JOB...**



# Il paradosso di Jevons

*L'efficienza dell'IA non è una minaccia alla quantità di lavoro, ma un acceleratore di ambizione.  
Rendendo l'intelligenza una commodity a basso costo, sbloccando progetti prima impossibili e  
spostando il valore umano dall'esecuzione alla direzione strategica*

# Dicesi Vibe Coding...

**“There’s a new kind of coding I call vibe coding, where you fully give in to the vibes, embrace exponentials, and forget that the code even exists.”**

**— Andrej Karpathy, X/Twitter, 2 febbraio 2025**

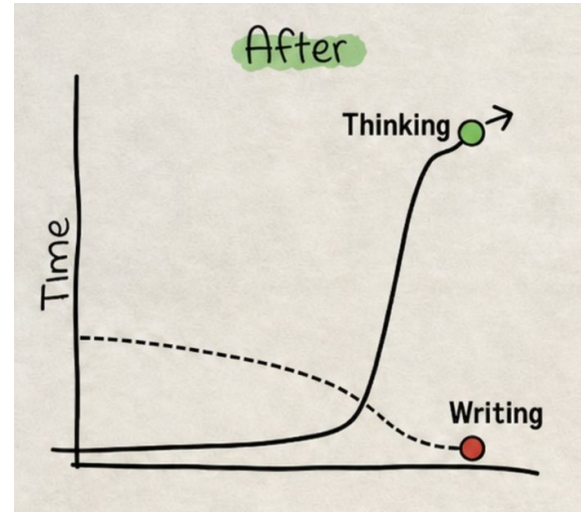
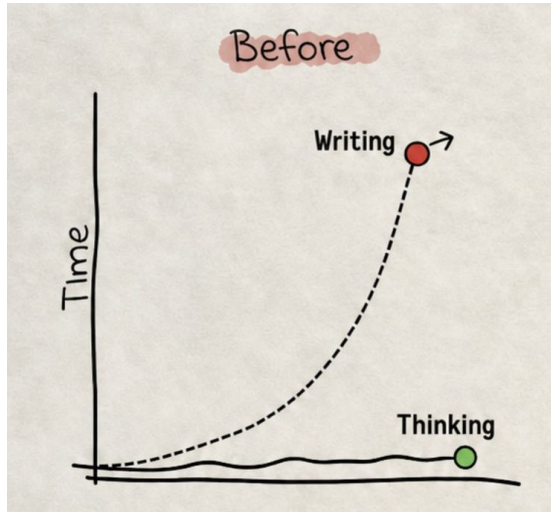
*Esiste un nuovo tipo di programmazione che chiamo “Vibe Coding” in cui si descrive l’intento e si lascia che l’AI generi il codice, concentrandosi sull’idea anziché sulla sintassi.*

**“The new default is that you are not writing the code directly 99% of the time. You are orchestrating agents who do and acting as oversight.”**

**— Andrej Karpathy, febbraio 2026**

*Adesso nel 99% dei casi non si scrive direttamente codice.  
Si orchestrano “agenti AI”  
(che pianificano, scrivono, testano e rilasciano codice).*

**La competenza tecnica non scompare: si trasforma da  
“scrivere codice” a “orchestrare chi lo scrive”.**



[https://www.facebook.com/permalink.php?story\\_fbid=pfbid02pVvaJVRtaBTKzv1WvYZdPLueARiqBf7PMBiCLrEDPg1yaBgfoS6GCaMph2CB7btel&id=61582766436550](https://www.facebook.com/permalink.php?story_fbid=pfbid02pVvaJVRtaBTKzv1WvYZdPLueARiqBf7PMBiCLrEDPg1yaBgfoS6GCaMph2CB7btel&id=61582766436550)

# Il problema

# Molti utenti usano l'AI come un oracolo...

*Si lancia un prompt generico e si ottengono 500 righe di codice.*

*Poi le incollano e, quando qualcosa si rompe, non sanno dove mettere le mani  
generando un enorme **"DEBITO TECNICO"***

~~VIBE CODING~~

**HOPE CODING**

# La soluzione

il Vibe Coding fatto bene

Vibe Coding come un  
paradigma in cui  
**l'intento umano**  
deve essere preminente ed  
usa la potenza generativa di  
un AI come **estensione** e  
non sostituzione.

Facendo Vibe Coding non si scrive più codice, ma dobbiamo imparare a **guidare, supervisionare e orchestrare** agenti AI che producono codice aderente a **standard, best practice e requisiti funzionali.**

Non si tratta di  
*“copiare e incollare”*,  
ma di una disciplina che richiede  
competenza tecnica, pensiero critico  
e **prompt engineering**.

**Sì, bravo, ma in pratica?**

DEVI DIVENTARE

# Il pilota che guida una F1 a Monte Carlo

Una macchina da F1 ha una potenza bruta incredibile (come i miliardi di parametri di un LLM). Può andare da 0 a 100 in due secondi. Ma se lasci l'acceleratore a tavoletta senza toccare il volante, finirai fuori strada al primo tornante.

Facendo Vibe Coding l'AI può generare 100 righe di codice in un istante, ma sei tu che decidi la direzione (l'intento). Se non sterzi (non dai contesto), il plugin si schianta contro un errore fatale.



immagine creata da chatgpt

DEVI DIVENTARE

# L'Architetto che progetta con cura

Un architetto non inizia mai a scavare senza un'idea chiara. Non dice "costruiamo qualcosa", ma definisce lo scopo: "Voglio un museo che sfrutti la luce naturale e trasmetta un senso di leggerezza".

Facendo Vibe Coding non devi scrivere "fammi un plugin", ma devi definire l'anima del progetto. Il codice è il tuo materiale da costruzione, la tua visione è il progetto.



immagine creata da chatgpt

# ESEMPIO

**(il vibe coding fatto bene)**

# Di nuovo al lavoro, Maurizio?

Crea Claude Opus 5.0.  
Attenzione a non fare errori



Opus 4.6 Estesa ▾



# L'IDEA

*“Voglio un plugin che aggiunga ai blocchi nativi Gutenberg delle opzioni che mi permettano di decidere se questi debbano essere visualizzati o nascosti in base al fatto che l'utente sia o meno loggato.”*

**CRITICITÀ: Cache, CDN e REST API**

## ## Contesto e obiettivo

Sviluppa un plugin WordPress chiamato **"WP Access Control Block"** che aggiunge a **tutti i blocchi Gutenberg nativi** la possibilità di controllare la visibilità in base allo stato di autenticazione dell'utente.

Il plugin permette all'autore di decidere, blocco per blocco, se il contenuto deve essere visibile a tutti, solo agli utenti loggati, oppure solo ai visitatori anonimi (non loggati). Opzionalmente, per ogni singolo blocco è possibile configurare un messaggio teaser personalizzato mostrato al posto del contenuto nascosto. Questo consente di creare pagine con contenuto misto – pubblico e riservato – senza duplicare pagine, usare shortcode, o installare plugin di membership complessi.

## ## Requisiti tecnici di base

- **PHP** >= 8.1 (usa typed properties, union types, match expression dove appropriato)
- **WordPress** >= 6.0 (i filtri Gutenberg `blocks.registerBlockType` e editor.BlockEdit` sono stabili da questa versione)`
- **Node.js** >= 18 (per il build process con `@wordpress/scripts`)`
- **Licenza** GPLv2 or later
- **Text Domain** `wp-access-control-block``
- **Prefisso funzioni/hook** `wacb_``
- **Nessuna dipendenza PHP esterna**: il plugin deve funzionare standalone lato server
- **Nessuna custom table**: usa solo gli attributi nativi dei blocchi Gutenberg e le opzioni WordPress
- **I commenti nel codice PHP e JS/JSX devono essere in italiano** per ogni funzione e per ogni passaggio logico delicato o non ovvio

## ## Build system e tooling

### ### Approccio: `@wordpress/scripts`

Il progetto usa `**@wordpress/scripts**` come build tool. Questo è il tool ufficiale di WordPress che wrappa webpack con una configurazione pre-ottimizzata per lo sviluppo di blocchi e plugin Gutenberg.

Vantaggi:

- Supporto JSX nativo
- Compilazione SCSS → CSS automatica
- Generazione automatica del file `.asset.php` con dipendenze e versione (hash) per l'enqueue
- Estrazione automatica delle dipendenze `@wordpress/*`
- Compatibilità garantita con l'ecosistema Gutenberg

...

## ## Architettura del plugin

### ### Struttura file

```
...
wp-access-control-block/
├─ wp-access-control-block.php      # File principale del plugin (bootstrap)
├─ readme.txt                       # Readme per WordPress.org (formato standard)
├─ README.md                        # Readme per GitHub (in inglese)
├─ CHANGELOG.md                     # Changelog (formato Keep a Changelog)
├─ .gitignore                       # Git ignore
├─ package.json                     # Dipendenze npm e script
├─ webpack.config.js                # Configurazione webpack custom
├─ languages/
│   └─ wp-access-control-block.pot  # Template di traduzione
├─ includes/
│   └─ BlockFilter.php              # Classe PHP - filtro render_block
│   └─ EditorAssets.php            # Classe PHP - enqueue degli asset editor
│   └─ Settings.php                 # Classe PHP - pagina impostazioni (defaults teaser)
├─ src/                              # Sorgenti (NON distribuiti, usati dal build)
│   └─ js/
│       └─ editor.jsx               # Entry point JS - registra filtri Gutenberg
│       └─ components/
│           └─ VisibilityControls.jsx # Componente pannello controllo visibilità
...

```

## ## Note importanti per l'AI

- **Usa JSX** e `@wordpress/scripts` - il codice sorgente JSX sta in `src/js/`, il build compila in `build/`
- **Non includere dipendenze** `@wordpress/` manualmente nell'`enqueue` - il file `.asset.php` generato dal build le gestisce automaticamente
- **Non nascondere contenuto via CSS o JS frontend** - solo filtraggio PHP server-side
- **Testa sempre** la funzionalità verificando che il sorgente HTML della pagina (View Source) non contenga il blocco nascosto
- **Non creare tabelle database** - usa solo attributi dei blocchi e opzioni WordPress
- **Commenti nel codice in italiano** - per ogni funzione e per i passaggi logici delicati
- **Stringhe UI traducibili** - wrappate in `__( )` da `@wordpress/i18n` in JSX e nelle funzioni PHP appropriate
- **Early return nel filtro `render_block`** - la maggior parte dei blocchi avrà visibilità `'all'`, ottimizza per quel caso
- **SCSS con BEM** - tutte le classi CSS seguono la convenzione `.wacb-block__element--modifier`
- **Il pannello dei controlli va dentro `<InspectorControls>` senza prop `group`** - questo lo posiziona naturalmente sopra "Advanced" nell'Inspector Sidebar. NON usare `<InspectorAdvancedControls>`.
- **Il teaser è configurabile per singolo blocco** - con fallback al messaggio globale delle impostazioni
- **La cartella `build/` va committata** - è necessaria per la distribuzione del plugin
- **La cartella `node_modules/` NON va committata** - è nel `.gitignore`

# PROMPT COMPLETO GENERATO DA OPUS

<https://github.com/miziomon/wp-access-control-block/blob/main/prompt.md>

# Anatomia di un prompt ben fatto

- **Contesto** — lo sfondo: qual è il progetto, chi è il pubblico (se serve), qual è lo stack tecnologico, quali vincoli di dominio si applicano.
- **Task** — l'azione richiesta. Regola d'oro: una task per volta. Mescolare richieste diverse in un singolo prompt produce output ibridi e confusi.
- **Vincoli** — cosa il modello deve fare e cosa non deve fare: lunghezza, tono, standard di codice, divieti (“non usare jQuery”, “niente librerie esterne”, “max 100 righe”).
- **Formato di output** — come vuoi ricevere la risposta: tabella Markdown, JSON con uno schema specifico, “solo codice senza spiegazioni”, lista di bullet, ecc.
- **(Opzionale) Ruolo** — “agisci come un senior backend engineer”. È il quinto ingrediente, deliberatamente messo per ultimo: nel 2026 il suo peso è significativamente ridotto.

# Alcune tecniche di Prompt Engineering

# Chain-of-Thought (CoT) — RAGIONAMENTO PASSO PASSO

Si chiede esplicitamente al modello di ragionare passo dopo passo prima di fornire la risposta finale. Migliora drasticamente le prestazioni su compiti logici, matematici e di analisi complessa.

## Esempio:

Un sito WordPress + WooCommerce è diventato lento nelle ultime settimane (TTFB > 3s sulle pagine prodotto).

Prima di proporre soluzioni, ragiona per fasi:

1. Elenca le cause più probabili di un peggioramento del TTFB su WP/WooCommerce in produzione.
2. Per ogni causa, indica come verificarla (strumento gratuito, query SQL, log da controllare).
3. Ordina le cause per probabilità + facilità di verifica.
4. SOLO DOPO aver completato i tre step sopra, proponi un piano di intervento in 5 step ordinati.

Non saltare ai consigli generici tipo "usa un caching plugin". Voglio l'analisi diagnostica prima.

**Quando usarlo:** Debugging, analisi di sicurezza, problemi logici multi-step, confronti, decisioni architetturali. Fondamentale nel coding.

# Few-shot Prompting — INSEGNARE PER ESEMPI

Invece di descrivere come vuoi l'output, mostralo con due o più esempi. Il modello riconosce il pattern, lo applica al nuovo input. È la tecnica più efficace per la voice consistency (mantenere uno stile uniforme su contenuti ricorrenti) e per la riproduzione di formati strutturati che è difficile descrivere a parole.

## Esempio:

Devo scrivere FAQ per un sito di prodotti per la prima infanzia.

Tono: confidenziale, mai paternalistico, qualche emoji ma con parsimonia. Risposta sintetica, max 3 righe.

Ti do due esempi del tono che voglio:

....

Ora scrivimi 3 FAQ nello stesso tono su questi argomenti:

1. Sterilizzazione del ciuccio
2. Allergie alimentari nei primi 12 mesi
3. Quando passare dal latte materno al latte di proseguimento

**Quando usarlo:** voice consistency, microcopy, schede prodotto, FAQ, classificazioni con etichette tue, formati strutturati difficili da descrivere a parole

# Meta Prompting — IL PROMPT PER IL PROMPT

Chiedi al modello di scriverti il prompt che dovresti dargli. È utile quando un task è nuovo o complesso, e/p non sai da dove iniziare

## **Struttura tipica:**

# Ruolo: Chi sei

# Contesto: Su cosa lavoriamo

# Vincoli: Regole da rispettare

# Formato output: Come vuoi la risposta

# Task: Cosa devi fare adesso

**Quando usarlo:** task nuovo o vago dove non sai cosa stai chiedendo: prompt complessi da formalizzare per riuso, dominio sconosciuto, brief di marketing da tradurre in spec tecnica. Le 3-5 domande di

# Panel of Experts — LA TAVOLA ROTONDA

Prezioso quando devi prendere una decisione e non vuoi accontentarti di una risposta unica, ma vuoi sentire prospettive diverse — soprattutto quelle che potrebbero non venirti in mente. Ogni esperto fornisce la propria risposta indipendente, le divergenze vengono evidenziate e discusse, e il risultato finale è una sintesi multi-prospettiva con un risultato più completo e bilanciato

## Esempio (Tavola Rotonda per lo sviluppo):

“Comportati come se stessi facendo un dibattito in una tavola rotonda con i seguenti esperti virtuali:

- Ingegnere informatico / Full Stack Developer
- Programmatore esperto in PHP
- Programmatore esperto in JavaScript
- Database Administrator e Data Engineer
- Designer esperto in UX
- Project Manager

Ogni esperto deve rispondere alla domanda con la propria opinione.

Se ci sono osservazioni discordanti, evidenziale.”

**Quando usarlo:** Decisioni architettoniche complesse, scelte tecnologiche con trade-off, analisi di requisiti multi-stakeholder, review di design, definizione di API. Particolarmente potente quando il problema tocca più discipline (backend, frontend, DB, UX, gestione progetto).

# La cassetta degli attrezzi

QUALE LLM USARE?

# Non esiste solo ChatGPT

**CLAUDE**

---

Anthropic  
<https://claude.ai>

**GEMINI**

---

Google  
<https://gemini.google.com>

**CHATGPT**

---

OpenAI  
<https://chatgpt.com>

QUALE LLM USARE?

# Non esiste solo ChatGPT #2

**GROK**

xAI  
<https://grok.com/>

**KIMI**

Moonshot AI  
<https://www.kimi.com/>

**QWEN**

Alibaba  
<https://chat.qwen.ai/>

Strumento all-in-one

# Servizi online

Lovable

Replit



**Lovable**

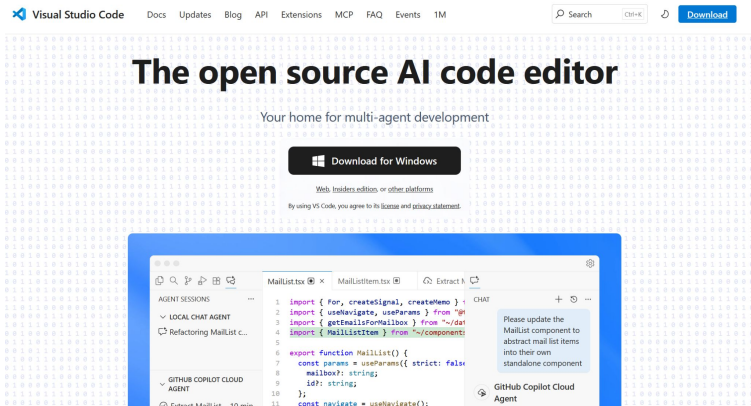


**Replit**

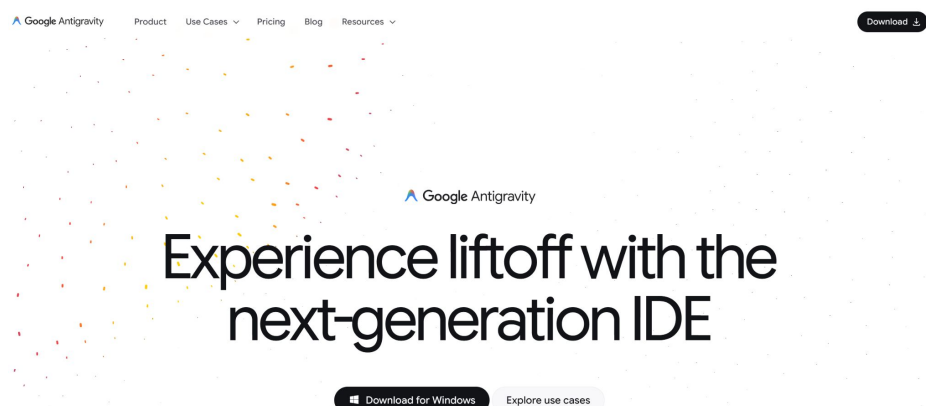
Strumenti da installare sul proprio computer

# Qualcuno ha detto IDE?

## VS Code



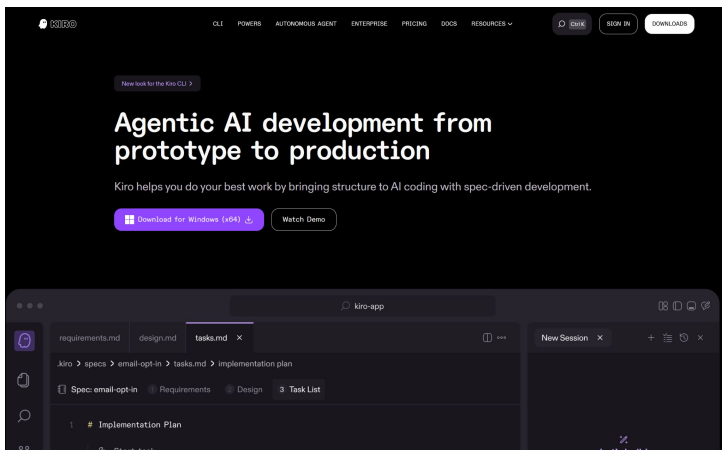
## Antigravity



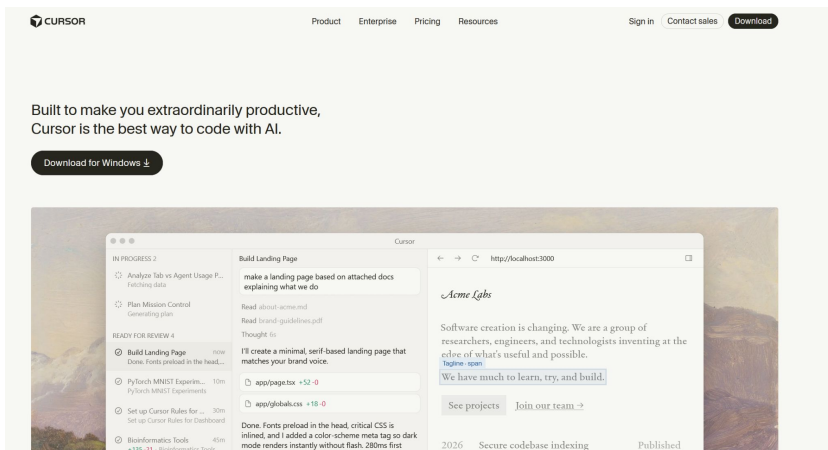
Strumenti da installare sul proprio computer

# Qualcuno ha detto IDE? #2

## Kiro



## Cursor



Per chi ama il controllo estremo

# Claude Code



Analyzing...

## Built for > coders

Work with Claude directly in your codebase. Build, debug, and ship from your IDE, Slack, or the web. Describe what you need, and Claude handles it.

Get Claude Code

irm <https://claude.ai/install.ps1> | iex

Or read the [documentation](#)



Claude Code v2.1.76  
Opus 4.6 (1M Context) · Claude Enterprise  
/Users/johnnie/taskflow

> I just joined the team. Can you give me a high-level overview of how this codebase is structured and where the main entry points are?

- I'll explore the codebase to give you a comprehensive overview.
- Explore(Explore codebase structure)
  - L Done (17 tool uses · 38.0k tokens · 28s)
- \* Clauding... (esc to interrupt)
- TaskFlow API - Codebase Overview

Welcome to the team! This is a Node.js/Express REST API for task management with team collaboration features. Here's the project structure:

```
taskflow/  
├── src/  
└── ...
```





# PRATICA

#sulemani

# DECIDIAMO INSIEME

## A

### Simple Page Access Control

---

Un plugin che permette di proteggere singole pagine o post con una restrizione per ruolo utente con login stilizzata, e log degli accessi.

## B

### Telegram Notification

---

Un plugin che si aggancia a wp\_mail e per ogni email in uscita invia una notifica via Telegram Bot API al chat ID configurato. L'utente riceve sul telefono una notifica con mittente, oggetto, e un'anteprima del contenuto.

## C

### Webhook Dispatcher

---

Un plugin che con un pannello nell'admin permetta di configurare la chiamata a URL Webhook esterni su specifici eventi (es user creation, login, post publish, plugin activated, plugin updated, etc.)

# VINCOLI NON NEGOZIABILI

**PHP 8.1 + Autoload PSR-4**

**Coding Standards con PHPCS e WPCS**

**SCSS con metodologia BEM**

**wp-env – Ambiente di sviluppo locale**

**Creazione file repo git**

**CHANGELOG.md (keepachangelog)**

**Skills (<https://github.com/Automattic/agent-skills>)**

# META PROMPT TEMPLATE

Crea un prompt dettagliato in un file md da usare con Claude Code per: .....

In aggiunta a quando ti ho detto usa questi vincoli

PHP 8.1 + Autoload PSR-4

Coding Standards con PHPCS e WPCS

Analisi del codice con PHPStan

SCSS con metodologia BEM

configurazione Ambiente di sviluppo locale - wp-env

CHANGELOG.md (keepachangelog)

Installa ed use questo Skills per lo sviluppo (<https://github.com/Automattic/agent-skills>)

Creazione file repo git e configurazione remote

Chiedi se hai domande e suggerisci miglioramenti

## REQUISITI:

Per alcuni dei vincoli sono necessari dei software di terze parti che vanno installate sul proprio computer tra cui: **PHP**, **Docker** , **Node.js**, **git**

# PROVA SUL CAMPO

- 1. Preparazione**
  - a. Brainstorming e confronto**
  - b. Creazione prompt**
  - c. Lettura, analisi, correzione e sistemazione**
- 2. Generazione codice**
  - a. Creazione piano di sviluppo (con prompt strutturato)**
  - b. ... vibe**
- 3. Test, debug e miglioramenti**
- 4. Recupero debito tecnico**

# POSTULATO DEL PELIZZONE

**25%**

**Prompt**

**25%**

**Coding**

**25%**

**Test**

**25%**

**Debito tecnico**

...vibe

# VERSO L'INFINITO E OLTRE

## (oltre la nostra confort zone)

Maurizio Pelizzone

# CLAUDE CODE

Una guida pratica per chi vuole iniziare  
ad usare Claude Code in modo professionale



Claude Code è la CLI agentica di Anthropic che vive nel terminale dello sviluppatore: legge il codice, esegue i comandi, rispetta le convenzioni del progetto.

Questa guida pratica — pensata per chi vuole iniziare a usarla in modo professionale — copre installazione e setup, comandi quotidiani, Plan Mode, prompt engineering, gestione del contesto e dei permessi, hook, server MCP, subagent e Skill personalizzate.

Tutti i contenuti sono verificati sulla documentazione ufficiale Anthropic. Esempi concreti su PHP, JavaScript e WordPress. Niente *hype*: ogni capitolo dichiara dove lo strumento conviene davvero e dove no.

Rivolto a sviluppatori web con familiarità con terminale e Git, professionisti che vogliono integrare l'AI nel proprio workflow in modo consapevole, e team tecnici che stanno valutando l'adozione di strumenti AI agentici nei processi di sviluppo.

Maurizio Pelizzone — Mavida snc  
Creative Commons BY-SA 4.0

Risorse online:

<https://github.com/miziomon/claude-code-guide>

<https://maurizio.mavida.com/guida-claude-code>

<https://leanpub.com/claude-code-guide>



WordCamp  
torino 2026

**Grazie!!!**