Mave: il CMS su misura che rende il cliente autonomo

Di Fabio Angelici • Pubblicato il 01 ottobre 2026 • 7 min di lettura

Struttura Sito | Manutenzione Sito | Case Studies



Questo articolo fa parte di una serie di case study sui progetti che ho seguito.

I precedenti sono Teknofibra Shop: come nasce e cresce un e-commerce custom,

ReMida Varese: costruire da zero la presenza online di un'associazione,

Chiara Netto: il portfolio che non ti aspetti e Apsvapo: da carta e penna a un gestionale su misura.

Il cliente di questo progetto, tecnicamente, non è mai stato mave.

Mave si è rivolta a un'agenzia partner per il proprio sito. L'agenzia ha disegnato tutto su Figma, e poi ha affidato a me la parte che so fare io:
mettere quel design in codice, farlo funzionare, farlo durare.
È una situazione più comune di quanto sembri, e cambia il modo in cui lavori:
chi vedi ogni giorno non è chi userà il sito domani, ma qualcuno in mezzo, e la comunicazione deve reggere su due livelli invece che uno.

Per farla reggere ho messo il sito, fin dai primi giorni, su un indirizzo di terzo livello del mio dominio.
L'agenzia poteva vederlo crescere pagina dopo pagina, senza aspettare una consegna finale per scoprire se la direzione era quella giusta.
E per tenere traccia di modifiche e richieste ho usato una kanban board che ho scritto io e che uso con i clienti:

ticket con commenti e allegati, non email sparse o messaggi su canali diversi da ricostruire a memoria ogni volta.

Niente di sofisticato.

Ma è la differenza tra un progetto che si segue e un progetto che si subisce.

La regola che ha deciso tutto il resto

Il sito di mave non è un tema Wordpress adattato.
È un CMS scritto su misura, in un framework professionale (Laravel, se il dettaglio interessa), con un pannello di amministrazione tutto suo.

Prima ancora di scrivere una riga di codice avevo già fissato una regola che avrebbe condizionato ogni scelta successiva:
ogni campo, ogni sezione, ogni elemento di ogni pagina doveva essere modificabile dal pannello fin dal momento in cui la pagina veniva creata.

Sembra scontato per un CMS. Non lo è quando le pagine arrivano una per una da un file Figma e vanno scritte a mano,
che è esattamente quello che è successo qui.
La scorciatoia sarebbe stata scrivere i testi dentro i modelli grafici: veloce da fare, e un sito che il cliente non può toccare senza chiamare qualcuno.
Un sito che si consegna, ma che non si lascia davvero.

Per questo nel codice delle pagine pubbliche non esiste un solo testo scritto a mano.
Il codice definisce la struttura, i campi, il layout. Il contenuto vero vive nel database fin dal primo momento,
inserito insieme alla pagina stessa non appena questa nasce.

Quando un pezzo di lavoro si butta via

A metà agosto avevo un sistema di template che funzionava bene: un template era un elenco di campi definiti a mano.
Faceva il suo lavoro, i test lo confermavano.

Aveva un problema che si vedeva solo usandolo: ogni template nuovo richiedeva comunque una veste grafica scritta da uno sviluppatore.
E i template esistono apposta per rendere il cliente autonomo. Un sistema in cui crearne uno richiede me non rende autonomo nessuno:
mi rende semplicemente indispensabile, che è l'esatto opposto di quello che avevo promesso.

A inizio settembre ho tolto quel sistema e l'ho sostituito: un template oggi è una sequenza di blocchi,
composta con gli stessi strumenti che si usano per costruire una pagina normale.
Nessuna pagina lo stava ancora usando, ed è stato possibile farlo senza rompere niente.
Non lo racconto perché sia una bella storia di errore ammesso. Lo racconto perché il difetto non si vedeva sulla lavagna.
Si vedeva solo nell'uso, e l'unico modo per accorgersene era usarlo davvero.

Dare la formattazione al cliente senza dargli l'HTML

Il cliente ha bisogno di mettere in grassetto una parola, di colorarla, di andare a capo dove vuole.
Il modo più ovvio sarebbe un editor che scrive HTML libero.
Ma è anche il modo che gli mette in mano lo strumento per rompere una pagina, o peggio, aprire una falla di sicurezza.

Ho scelto un piccolo insieme fisso di marcatori interpretati dal server: grassetto, corsivo, i due colori del marchio, l'andata a capo.
Combinabili tra loro, e solo quelli.
Qualunque altra cosa tra parentesi quadre resta semplicemente testo, visibile così com'è.
Se un marcatore resta aperto per errore, si vede, invece di sparire in silenzio.

La prima versione, di fine luglio, usava una sintassi diversa e non combinabile.
Bastava voler mettere una parola in grassetto e colorata insieme perché non ci fosse un modo di scriverlo.
L'ho sostituita in blocco, codice e contenuti già scritti insieme, appena è diventato chiaro che ogni nuova combinazione avrebbe richiesto una variante dedicata. Meglio rifarla una volta bene che rattopparla all'infinito.

Il giorno in cui due guasti si sono nascosti a vicenda

I primi di settembre, durante un controllo di sicurezza fatto apposta prima di considerare il sito davvero pronto,
è saltato fuori che il sito in produzione non stava inviando le intestazioni di sicurezza che avrebbe dovuto.
Non un errore nel codice: la configurazione era corretta, era la macchina a non sapere di essere in produzione.

Nessuno dei miei test automatici poteva vederlo, perché controllavano il codice, e il problema stava altrove.

L'ho trovato solo perché una mia affermazione è stata messa in dubbio,
e sono andato a controllare le intestazioni vere invece di fidarmi di quello che pensavo di sapere.

Sistemata quella variabile, il banner dei cookie ha smesso di funzionare.
Lo script che lo genera arrivava da un indirizzo che la nuova policy di sicurezza, ora finalmente attiva, non conosceva ancora.
Il primo guasto stava nascondendo il secondo da settimane.

Ho messo quell'indirizzo in un unico punto che alimenta insieme la policy di sicurezza e il controllo nel pannello,
con una prova automatica che verifica che i due non si allontanino di nuovo.

Il limite che ho scritto invece di nascondere

Il sito punta alla piena accessibilità.
Su quattro combinazioni di colore, però, il testo bianco sopra i colori del marchio non raggiunge il contrasto minimo richiesto dagli standard di accessibilità. Sono i colori scelti e approvati da mave insieme al bianco che ci sta sopra,
e cambiarli su schede già approvate avrebbe voluto dire tornare indietro su una decisione presa.

Le soluzioni esistevano tutte, le ho misurate una per una. Nessuna era complicata da applicare.
Ho scelto comunque di scrivere l'eccezione invece di sistemarla di nascosto:
i numeri esatti, dove si applica, e una frase chiara che dice che su quelle quattro combinazioni chi ha una vista ridotta non legge bene quel testo.

Non è la soluzione più elegante. È quella onesta, ed è anche l'unica che regge nel tempo:
se un domani arriva una verifica di accessibilità vera, la risposta è già scritta, e non è una scusa messa insieme all'ultimo momento.

Cosa resta, in numeri

Il progetto è durato 53 giorni, dal 16 luglio al 6 settembre. Dietro ci sono 296 commit, quasi la metà funzioni nuove,
e 729 test automatici che controllano il codice a ogni modifica: circa una riga di test ogni due di applicazione.
Non era un obiettivo scritto da nessuna parte.
È la conseguenza di una regola sola: nessuna funzione si considera finita finché una prova automatica non la copre.

Il sito oggi ha venti pagine con contenuto vero, tre lingue (italiano e inglese pubblicati, lo spagnolo ancora da scrivere ma già predisposto),
copie di sicurezza automatiche del database e dei file, e un punteggio Lighthouse di 96 su desktop e 95 su mobile.
L'unico controllo che non passa a pieni voti è, appunto, quell'eccezione di contrasto scritta e documentata.

Quello che mi porto da questo progetto non è la lista delle funzioni, per quanto lunga.
È la conferma che un cliente che vede il sito crescere giorno dopo giorno, discute ogni modifica su un ticket invece di rincorrermi via email,
e sa davvero cosa può e cosa non può ancora fare da solo, è un cliente più tranquillo.
Anche quando qualcosa va storto, come l'intestazione di sicurezza dimenticata o il contrasto che non torna: perché lo sa subito, e sa perché.

Se stai valutando un sito su misura e vuoi capire quanto di quel lavoro resterebbe davvero nelle tue mani una volta consegnato, scrivimi e ne parliamo, senza impegno.

Hai domande su come ho gestito qualche pezzo di questo progetto? Scrivilo nei commenti.

Letto originariamente su https://fabioangelici.com/blog/mave-il-cms-su-misura-che-rende-il-cliente-autonomo
PDF generato il 16 novembre 2026