CREA LAVORA VENDI

CON

WORDPRESS

WP Rocket Kit

CREA, LAVORA, VENDI SUL WEB

WP Rocket Kit

CREA, LAVORA, VENDI SUL WEB

ROCKET CODE

WordPress a livello avanzato

Benvenuto nel Rocket Code kit!

Per diventare veri professionisti del web è necessario essere all'altezza di sfide molto grandi. La prima è comprendere a fondo l'architettura di un tema WordPress

Buona lettura!

1_Come creare un tema WordPress

Prima di procedere con questo capitolo, una breve premessa. Il web è pieno di framework e temi professionali, creati da altri e super ottimizzati per essere utilizzati al meglio senza sapere niente di codice.

Personalmente utilizzo framework già realizzati e cerco di migliorarli per renderli adatti alle esigenze dei miei clienti.

Questo significa che anche se vorrai diventare un professionista del mestiere, non dovrai perderti nei codici e soprattutto non dovrai creare temi WordPress da zero ogni singola volta.

Tuttavia, comprendere come funziona WordPress nella sua componente più profonda, ti aiuterà moltissimo ogni volta che si presenterà un problema o ogni volta che un tuo cliente ti farà una richiesta che non sarà possibile accontentare senza toccare il codice.

1.1 Come funziona WordPress

Facciamo un passo indietro, e cerchiamo di capire come funziona ogni sito web.

Prima che tu possa visualizzare un sito web nel tuo computer, qualcuno ha creato una directory di cartelle all'interno di un server, che vengono lette dal tuo browser; il quale, infine, esegue tutte le indicazioni che gli sono poste dai vari file presenti nella directory.

Solitamente un sito statico (ovvero che non possiede un CMS che permette modifiche in tempo reale dei contenuti senza accedere al codice), è composto da un file index che conserva la struttura HTML del sito e che fornisce informazioni

al browser riguardo alle immagini, ai testi, ai link e a tutte le cose che deve visualizzare.

Per rendere tutto più carino, è presente un file che contiene il codice CSS, ovvero i “vestiti del sito”: colori dei testi, font, dimensioni e tutto ciò che riguarda la grafica.

Un tema WordPress è diverso da tutto questo, per una ragione essenziale: esso ha una struttura back-end che permette di modificare articoli, pagine, metadati e molto altro senza dover accedere al codice.

Per questo motivo un sito creato con WordPress ha la capacità di contenere blog, e di rimanere sempre aggiornato senza necessità di contattare il proprio WebDesiner, e impiegando la metà del tempo.

Creare un sistema dinamico, però, implica che l’architettura del tema debba avere un passo in più rispetto ad un sito statico.

Per questo motivo WordPress funziona tramite due magie:

- i file template

- il loop

Ogni file template gestisce una componente del sito, e il file index informa il browser su come e dove dovrà visualizzare i vari pezzetti del puzzle.

Il loop, invece, è un processo che estrae i vari contenuti dal database e li inserisce nel sito. Ha questo nome poiché esegue il codice ciclicamente fino a che tutte le istanze del HTML e del PHP sono soddisfatte.

Per capire meglio, recati nel mio sito web, esattamente nel blog

http://andreatasselli.net/blog/

Scorrendo potrai vedere tutti gli ultimi articoli del mio blog. Il titolo, le immagini e i contenuti sono gestiti dal loop, il quale li fa apparire solo se rispettano i suoi requisiti.

1.1.1 Da cosa è composto un tema WordPress

I file essenziali per creare un tema WordPress sono:

- index.php

- style.css

Il primo è il file principale del sito e gestisce la main area, ovvero quella parte del sito in cui vengono inserite tutte la informazioni principali. In molti casi serve anche come mappa per i file template che compongono i temi più complessi.

Il secondo file definisce gli stili del sito: margini, colori, font, disposizione dei contenuti, ecc.

Un tema normale, generalmente è composto da tre tipologie di file

- file template (nel quale rientra index.php)

- fogli di stile (nel quale rientra style.css)

- file fuction

1.1.2 File mancanti

WordPress non ti obbliga ad inserire i file template che gestiscono ogni singola parte del tema (come ad esempio comments.php, il file che gestisce i commenti), ma in questo caso utilizzerà nel sito le versioni di default.

Nel caso qualcosa non soddisfi le tue aspettative, non dovrai far altro che creare un file template per una sezione predefinita e WordPress utilizzerà quest'ultima.

1.1.3 Dove si trovano i file template

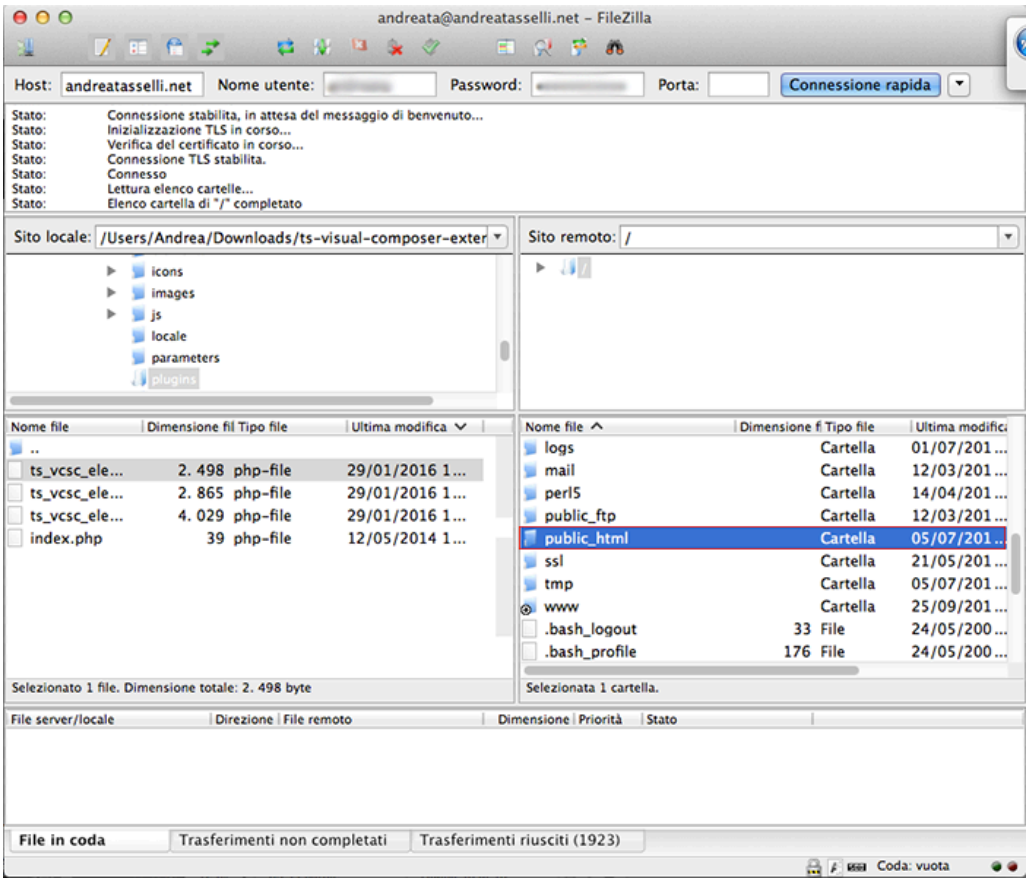
Tutti i file che andranno a comporre il tuo tema WordPress si trovano nella cartella wp-contents/themes/

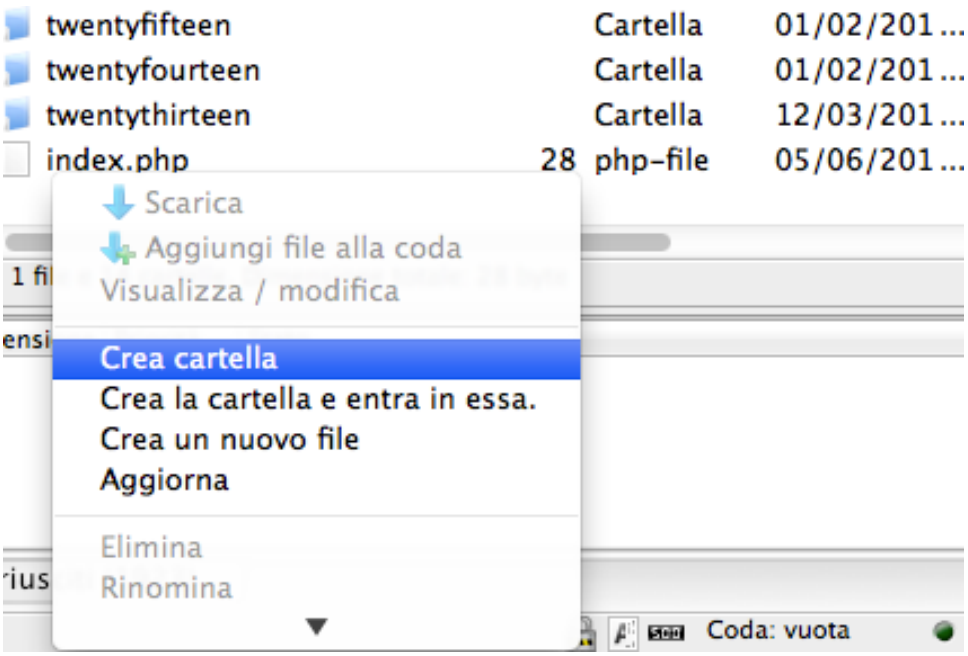
Quando andrai a creare i vari file del tuo tema, non dovrai far altro che creare una sottcartella di "themes" con il nome che avrai scelto per il tuo progetto (es. wp-contents/themes/mio-tema/

1.1.4 Passo per creare un tema WordPress

Come prima cosa, accedi tramite client FTP al database del tuo sito e crea una sottocartella chiamata "tema-esempio" nella directory dei temi del tuo sito WordPress /wp-content/themes

Per accedere tramite FTP avrai bisogno delle credenziali che verranno fornite





dal tuo servizio Hosting al momento dell'acquisto.

Il mio consiglio è quello di utilizzare FileZilla.

Host: dominio

Nome Utente: Stesso nome utilizzato per accedere al tuo hosting

Password: stessa password utilizzata per accedere al tuo account hosting

Porta: 21

Una volta dentro dovrai cercare public_html

Poi wp-content e infine themes

A questo punto, crea una sottocartella all'interno di questa directory.

Una volta creata la directory, è necessario realizzare tutti i file principali, come indicati nel prossimo paragrafo.

1.1.5 File necessari

Il secondo passo per realizzare un tema con WordPress, è creare tutti i file necessari per dare una struttura al tema.

- **index.php** Questo è il file principale del tema. Descrive la main area e specifica dove saranno inseriti gli altri file.

- **header.php** Contiene il codice della testata del tema

- **sidebar.php** Contiene le informazioni relative alla barra laterale

- **footer.php** Gestirà il piè di pagina

- **functions.php** Contiene le informazioni riguardo le funzioni principali del tema

- **style.css** Contiene le informazioni relative agli stili del tema

1.5 Inserire i crediti nel file CSS

Apri il file stule.css con il tuo editor di testo preferito e aggiungi questa sezione di commenti.

```
/*
 Theme Name:   Twenty Fifteen Child
 Theme URI:    http://example.com/twenty-fifteen-child/
 Description:  Twenty Fifteen Child Theme
 Author:       John Doe
 Author URI:   http://example.com
 Version:      1.0.0
 Tags:         light, dark, two-columns, right-sidebar, responsive-
layout, accessibility-ready
 Text Domain:  twenty-fifteen-child
*/
```

N.B. Tale sezione è l'esempio del tema di default di WordPress, e ovviamente dovrai cambiare le sezioni in base ai tuoi dati personali.

Noterai che che la sezione dei commenti inizia con /* e finisce con */. Questo serve ad indicare al browser di ignorare i commenti e di passare oltre.

Questo spazio è obbligatorio perché viene utilizzato da WordPress per mostrare tutte le informazioni del tema nella schermata temi > aspetto.

Variabili obbligatorie: Theme Name, Theme URI, Descriptio, Author, Author URI

Variabili facoltative: Tags, Template Version

Chiudi e salva il file nella sottocartella precedentemente salvata.

1.1.6 Modificare il file header.php

header.php gestisce la prima parte della pagina, quella più in alto. Esso contiene anche le meta informazioni che non verranno visualizzate dagli utenti, ma che forniscono informazioni al browser di vitale importanza per la corretta visualizzazione dei vari template.

Per prima cosa è necessario informare il browser su come dovrà interpretare il codice XHTML. Per eseguire questo passaggio è essenziale utilizzare una dichiarazione DOCTYPE.

Usando WordPress hai sostanzialmente due dichiarazioni per permettere una visualizzazione coerente su tutti i browser.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
```

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
```

Per approfondire l'argomento, ti consiglio questa guida

https://codex.wordpress.org/HTML_to_XHTML

Adesso è il momento di fare un passaggio importantissimo.

Molti neofiti di WordPress (tra cui io quando ero alle prime armi), sono convinti che esista un file unico che gestisce il codice HTML (come avviene per il file index dei siti web, basati solo su codice. Ovvero che sono assenti di un CMS che permetta di modificarli nei contenuti).

WordPress, invece, utilizza vari file template in cui il codice HTML è distribuito.

In particolare, il file header.php è il primo luogo in cui verrà aperto il tag <html>, mentre il footer.php è dove verrà chiuso.

Il passo successivo, quindi, dopo la dichiarazione DOCTYPE è inserire il tag <html> nell'header.php

Adesso è il momento delle meta informazioni del tema.

Esse saranno inserite tra <head> e </head>

Il meta tag più importante è sensa dubbio il foglio di stile. Senza di esso il file style.css non verrà processato e tutte le informazioni relative agli stili del sito non verranno visualizzate dal browser.

```
<link rel="stylesheet" type="text/css" media="all" href="<?php bloginfo('stylesheet_url'); ?>" />
```

Tag per importare caratteri dal sito, nell'area impostazioni > lettura

```
<meta charset="<?php bloginfo( 'charset' ); ?>">
```

Per visualizzare il titolo nell'area del browser dedicata al nome del sito così come risulta impostato nell'area di amministrazione di WordPress è necessario inserire questo codice:

```
<title><?php wp_title(); bloginfo('name'); ?> - <?php bloginfo('description'); ?></title>
```

Per utilizzare la meta descrizione del sito, inserita nell'area delle impostazioni di WordPress, è sufficiente aggiungere questo codice:

```
<meta name="description" content="<?php bloginfo('description'); ?>" />
```

Per permettere ai plugin di funzionare correttamente, visualizzando le loro funzioni solo in alcune sezioni del sito, è esseziale inserire

```
<?php wp_head(); ?>
```

Adesso abbiamo inserito i meta dati essenziali per permettere ai browser di visualizzare correttamente il proprio sito.

Chiudi quindi il tag head, inserendo:

```
</head>
```

A questo punto aggiungi

```
<body>
```

N.B. Ricorda di chiudere il tag body, nel footer.php inserendo

```
</body>
```

All’interno del tag “body” saranno presenti tutti i contenuti del sito, ma per definire una larghezza specifica di questi contenuti è possibile utilizzare un ulteriore tag:

```
<div id=”page”>
```

N.B. anche questo tag si chiuderà in footer.php

A questo punto è possibile inserire il codice che descriverà la testata del sito, come ad esempio il nome.

Ovviamente a questo punto sarebbe essenziale fare un po’ di esperienza con il codice per creare temi professionali.

Un primo codice molto semplice potrebbe essere questo

```
<div id=header”> <h1><a href=”<?php echo esc_url ( home_url( ‘/’ ) ); ?>” title=”<?php echo esc_attr( get_bloginfo( ‘name’, ‘display’ ) ); ?>” rel=”home”><?php bloginfo( ‘name’ ); ?></a></h1> </div>
```

Il tema di WordPress predefinito, ad esempio, utilizza questo codice

```
<div id="header">
 <div id="headerimg">
   <h1>
    <a href="<?php echo get_option('home'); ?>">
       <?php bloginfo('name'); ?></a>
   </h1>
     <div class="description">
       <?php bloginfo('description'); ?>
     </div>
  </div>
```

```
</div>
```

O, ancora più semplice, il tema classico utilizza questo:

```
<h1 id="header">
<a href="<?php bloginfo('url'); ?>"><?php bloginfo('name'); ?></a>
</h1>
```

Un semplice testo che fa da titolo, il quale rimanda alla home nel caso si faccia click su di esso.

Adesso puoi salvare il file header.php nella directory del tema e chiuderlo.

1.1.7 Modificare il file index.php

Apri il file index.php e inserisci la struttura XHTML della parte principale del sito.

In pratica dovrai andare a definire quali sono i template che dovranno essere utilizzati dal browser per visualizzare il sito (header, sidebar e footer), e come dovrà leggere i contenuti centrali tramite loop.

```
<?php get_header();? >
<?php get_sidebar(); ?>
<div id=”content”>
</div>
<?php get_footer(); ?>
```

```
 <?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
```

```
 <!-- Verifica se l'articolo corrente appartiene alla categoria 3. -->
 <!-- Se vi appartiene al div viene assegnata una classe CSS "post-cat-three". -->
```

```
 <!-- Altrimenti al div viene assegnata una classe CSS "post". -->

 <?php if ( in_category('3') ) { ?>
          <div class="post-cat-three">
 <?php } else { ?>
          <div class="post">
 <?php } ?>

 <!-- Visualizza il Titolo come un link al permalink dell'articolo. -->

 <h2><a href="<?php the_permalink() ?>" rel="bookmark" title="Link
permanente a <?php the_title_attribute(); ?>"><?php the_title(); ?></
a></h2>

 <!-- VIsualizza la data (nel formato 16 Novembre 2009) ed un link agli
altri articoli dell'autore dell'articolo. -->

 <small><?php the_time('j F Y') ?> di <?php the_author_posts_link() ?></
small>

 <!-- Visualizza il contenuto dell'articolo all'interno di un div. -->

 <div class="entry">
   <?php the_content(); ?>
 </div>

 <!-- Visualizza un elenco delle categorie dell'articolo separate da
virgole. -->

 <p class="postmetadata">Pubblicato in <?php the_category(', '); ?></p>
 </div> <!-- chiude il primo div -->
```

```
 <!-- >Termina Il Loop (ma si noti l'"else:" - si veda la riga
successiva). -->

 <?php endwhile; else: ?>

 <!-- Il primo test "if" verificava che vi fossero articoli da -->
 <!-- visualizzare.  Questo "else" dice cosa fare quando non ve ne è
alcuno. -->
 <p>Spiacente, nessun articolo corrisponde ai criteri di ricerca.</p>

 <!-- fine VERA de Il Loop. -->
 <?php endif; ?>
```

Chiudi il file index.php e caricalo nella directory dei template.

1.1.8 Modificare il file sidebar.php

Nella sidebar del proprio sito WordPress è importante inserire la sezione dei widget. In questo modo è possibile personalizzare il proprio sito WordPress in maniera illimitata.

Apri il file sidebar.php e inserisci il codice per far apparire i widget

```
<div id=”sidebar”>
<?php dynamic_sidebar( ‘primary’
 ); ?>
</div>
```

Nel capitolo dedicato ai widget e ai plugin potrai capire il loro potere e sfruttarne il potenziale.

Ovviamente non esiste una soluzione a tutto, quindi se avrai particolari esigenze che non potranno essere soddisfatte dai widget, potrai sempre creare un codice apposito e inserirlo qui.

Per far apparire le informazioni prima dei widget, inserisci il codice subito dopo il tag di apertura, mentre se vorrai farle apparire dopo i widget, inserisci il codice appena prima del tag di chiusura.

Salva il file sidebar.php e inseriscilo nella directory del tema.

1.1.9 Modificare il file function.php

Aggiungi questo codice per registrare la barra laterale con i widget.

```
<?php
add_action( ‘wigets_init’,
 ‘my_sidebar’ );
function my_sidebar() (
/* Registra la barra laterale principale’, “/
register_sidebar(
array(
 ‘idi => ‘primary’,
 ‘name’ => -( ‘primary’ ),
     register_sidebar( array(
                'id' => 'primary',
                'name' => __( 'Primary’ ),
                'description' => -( 'This is the primary sidebar' ),
                'before_widget' => '<div id="%1$s" class="widget
%2$s">',
                'after_widget' => '<h3 class=”widget_title”>',
                'before_title' => '<h2>',
                'after_title' => '</h3>'
                 )
                );
                }
```

```
                ?>
```

Salva e chiudi il file fuction.php nella directory del tema.

1.1.10 Modificare il file footer.php

Apri il file footer.php e inserisci le informazioni che vuoi far apparire nel piè di pagina.

```
<div id=”footer>
<cite>
&copy; <?php echo date
(‘Y); ?><a haref=”<?php bloginfo(‘url’);?><?php
 bloginfo(‘name’); ?></a>
 </cite>
</div>
```

Alla fine del footer, dovrai prestare attenzione a chiudere i giusti tag.

Prima di tutto:

```
<div id=”page”>
```

che avevi aperto nel header.php

Poi aggiungi l’hook

```
<?php wp_footer(); ?>
```

Per attivare il supporto dei plugin.

Infine, cosa più importante di tutte, chiudi i seguenti tag

```
</body>
</html>
```

Salva il file footer.php della directory del tema.

1.2 Vedere il proprio sito WordPress

Se avrai letto con attenzione questo capitolo dedicato allo sviluppo di un tema, avrai avuto modo di iniziare a “vedere” il tuo sito WordPress.

Adesso è venuto il momento di ricapitolare.

Un tema WordPress è costituito da una componente front-end (quella vista dagli utenti) e una componente back-end (ovvero il CMS dove andrai ad aggiungere articoli, modificare i widget, ecc.)

Ovviamente esiste un livello più nascosto, quello che solitamente nessuno osserva: le directory che contengono il codice.

WordPress è composto da quattro sezioni:

Header: la parte in alto (gestito da header.php)

Main area: dove vengono rappresentati i contenuti (gestito da index.php)

Sidebar: barra laterale dei widget (gestito da sidebar.php)

Footer: piè di pagina (gestito da footer.php)

Il codice HTML comincia nel file header e finisce nel footer (a differenza dei siti statici, dove solitamente è tutto inserito nell’index).

Importante: perché è essenziale sapere dove finisce il codice HTML?

Molto spesso, vari strumenti presenti online di estrema importanza, chiedono di inserire stringhe di codice in fondo al codice HTML prima del tag <body>

Facendo ricerche sul web, si ottengono svariati risultati, e spesso si entra in confusione. Il sapere, quindi, permette di velocizzare il processo.

La magia di WordPress avviene attraverso due caratteristiche:

- il loop dell'index

- i file template

Il loop è un processo che estrae i vari contenuti dal database e li inserisce nel sito. In questo modo il sito è sempre aggiornato con gli ultimi articoli.

I file template estraggono dati dal database del sito e generano l'HTML che deve apparire.

Un file template può generare un risultato specifico (come single.php che mostra un singolo articolo del blog), oppure può essere incluso in altri template, che funzionano come tessere di un puzzle.

Infine, WordPress ha un foglio di stile (style.css) che determina le componenti grafiche del sito, e un function.php, un file che permette di eseguire le funzioni del tema.

Questo è WordPress. Semplice, geniale e alla portata di tutti.

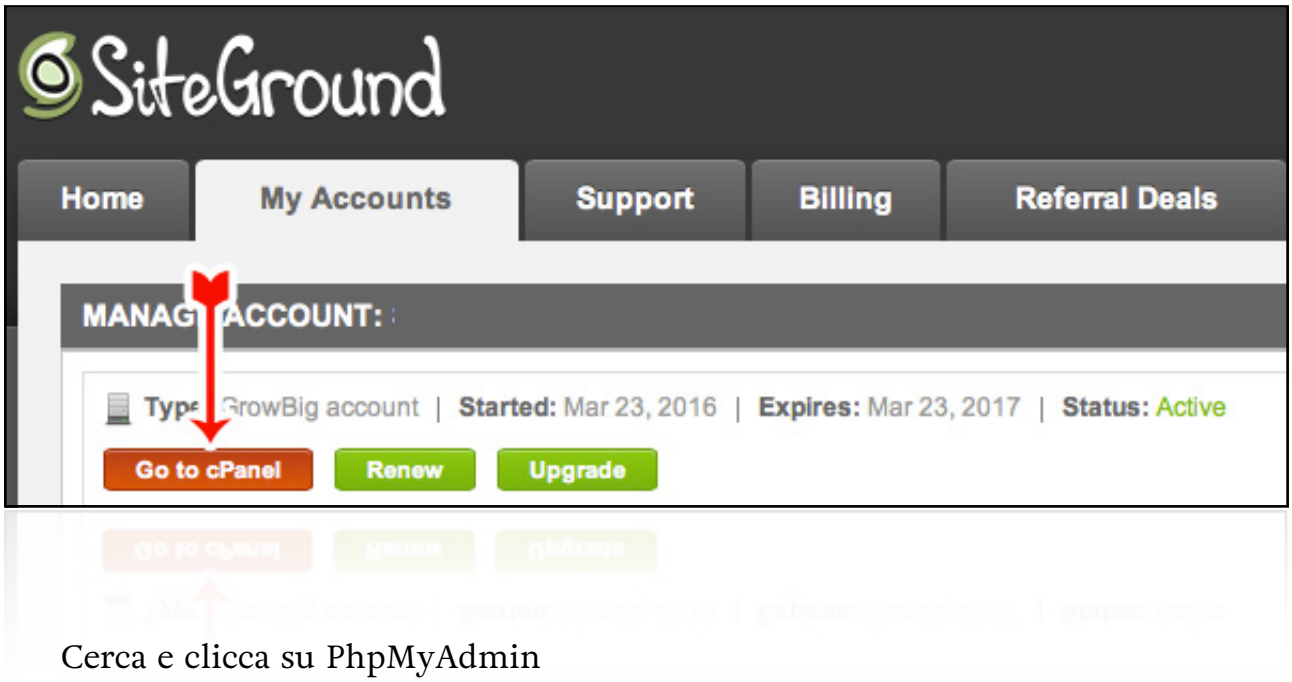
2_Hai perso la password?

Alcune volte, soprattutto quando si vendono siti realizzati con WordPress, capita che il proprio cliente cambi la password.

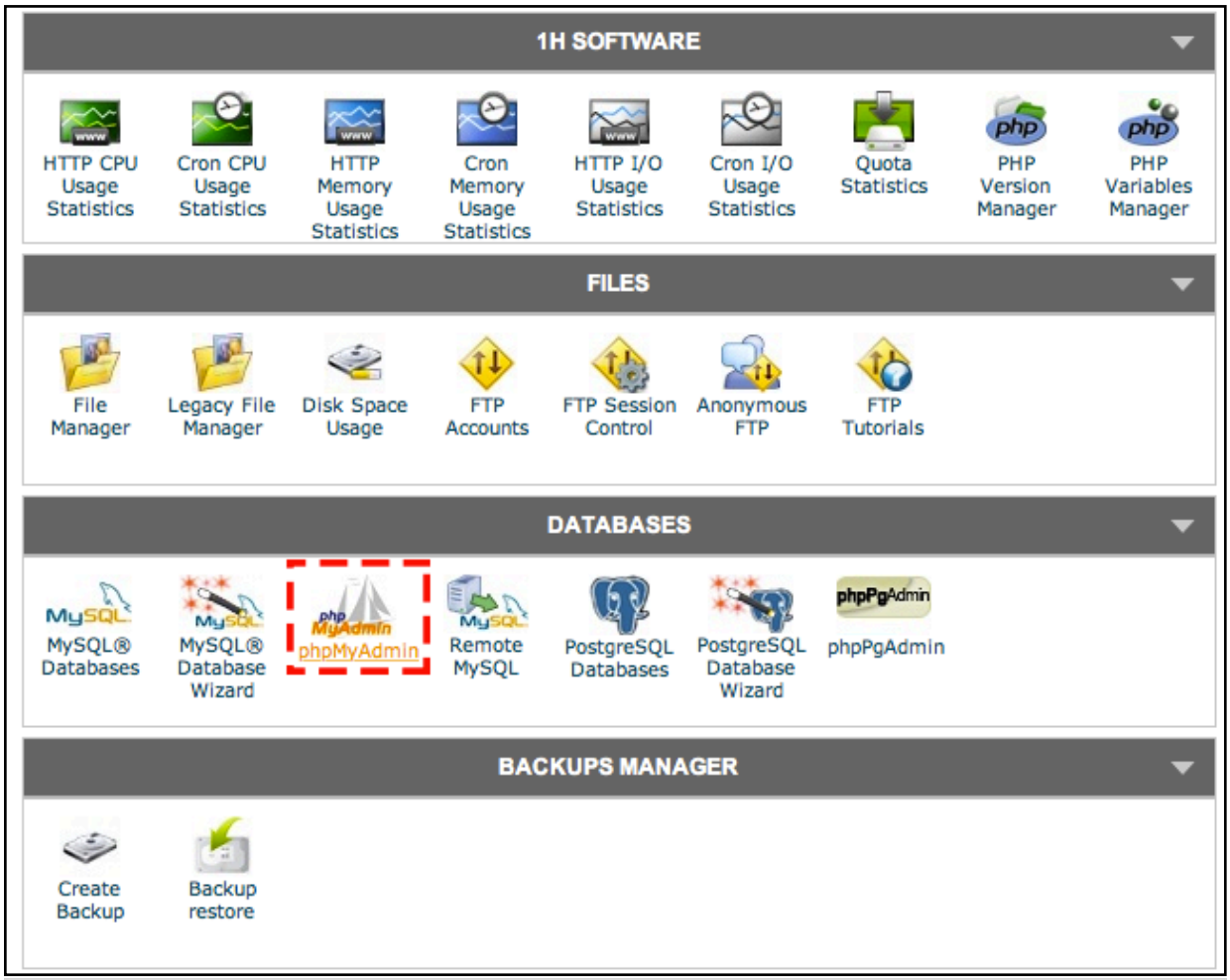
In questo modo noi non potremo più accedere al CMS. In fondo questo non sarebbe un problema, fino al momento in cui il cliente non riesca più a trovare la password e necessiti di una modifica effettuata da noi.

A quel punto si rende necessario impostare una nuova password proprio da PhpMyAdmin.

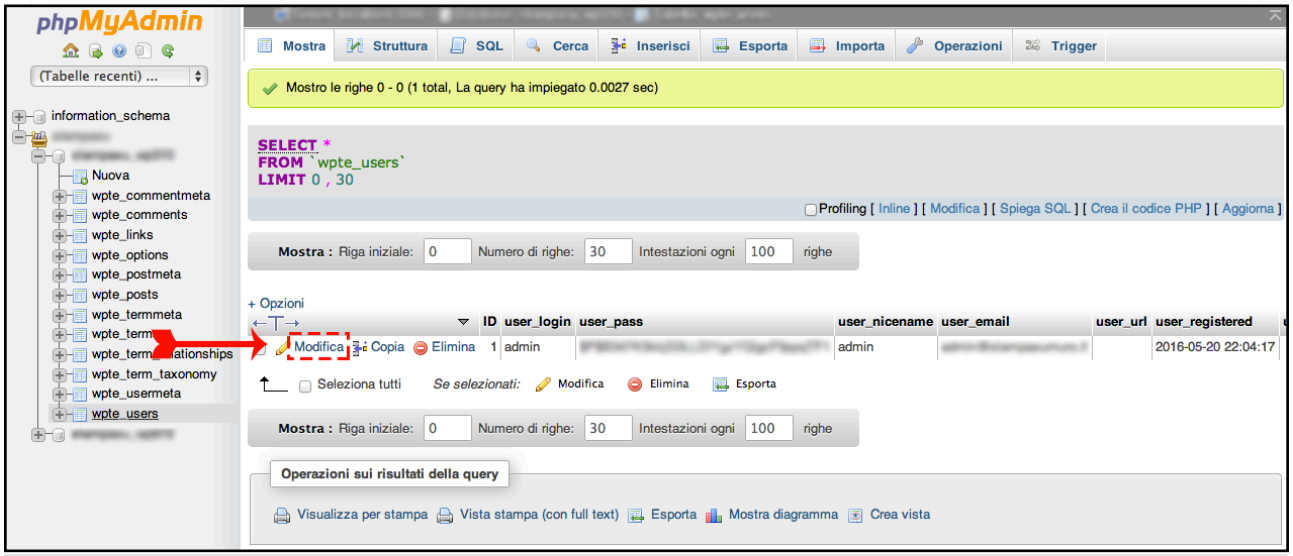
Recati sul tuo account Hosting e apri il cPanel.



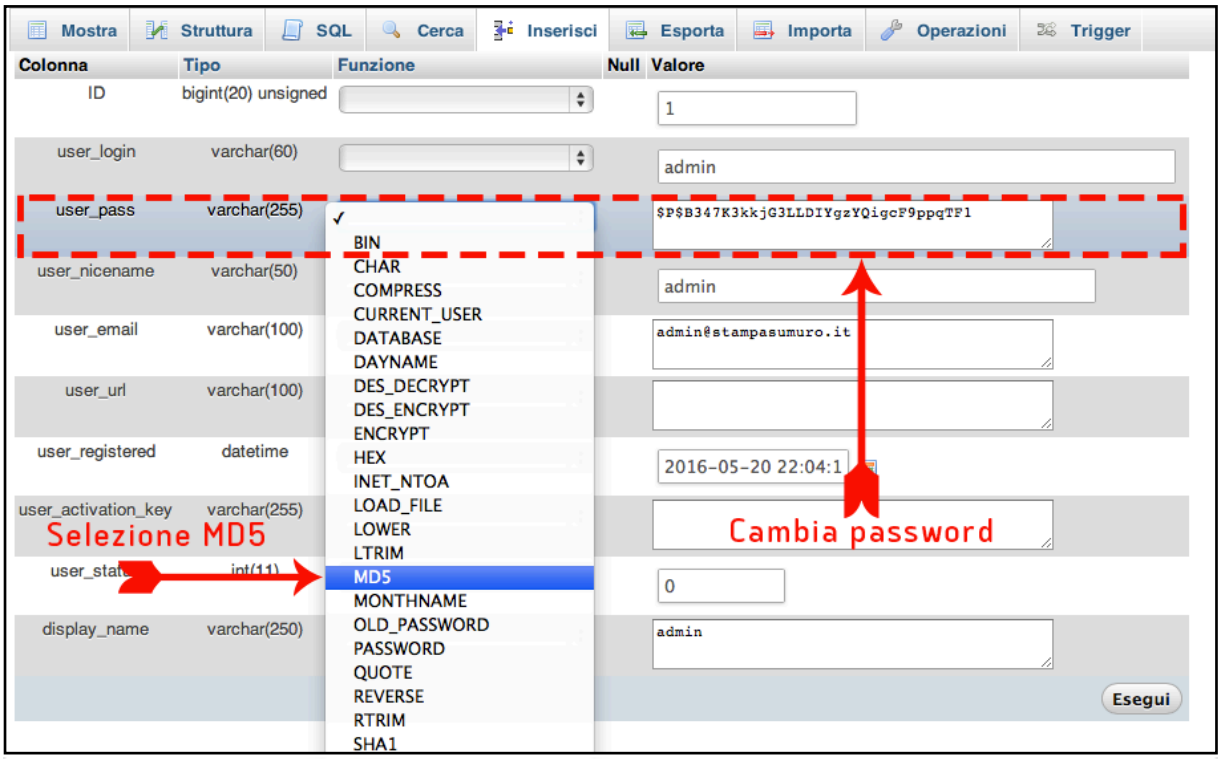
Cerca e clicca su PhpMyAdmin



Sull'explorer di destra seleziona "user" e poi clicca su "modifica" come indicato nell'immagine sottostante.



Cerca la riga "user_pass" e nella colonna "Funzione" inserisci "MD5", mentre nella colonna "Null Valore" imposta la nuova password.



Password resettata senza alcun problema.

3_Bloccare un indirizzo ip con il file haccess

Più il tuo sito diventerà famoso e più rischierai di subire attacchi di forza bruta. Io ne ho subiti 2 solo nell'anno corrente.

Tuttavia esistono molti metodi per scongiurare questi attacchi prima che avvengano, ad esempio utilizzando un servizio di hacking allert, come quello fornito da Siteground.

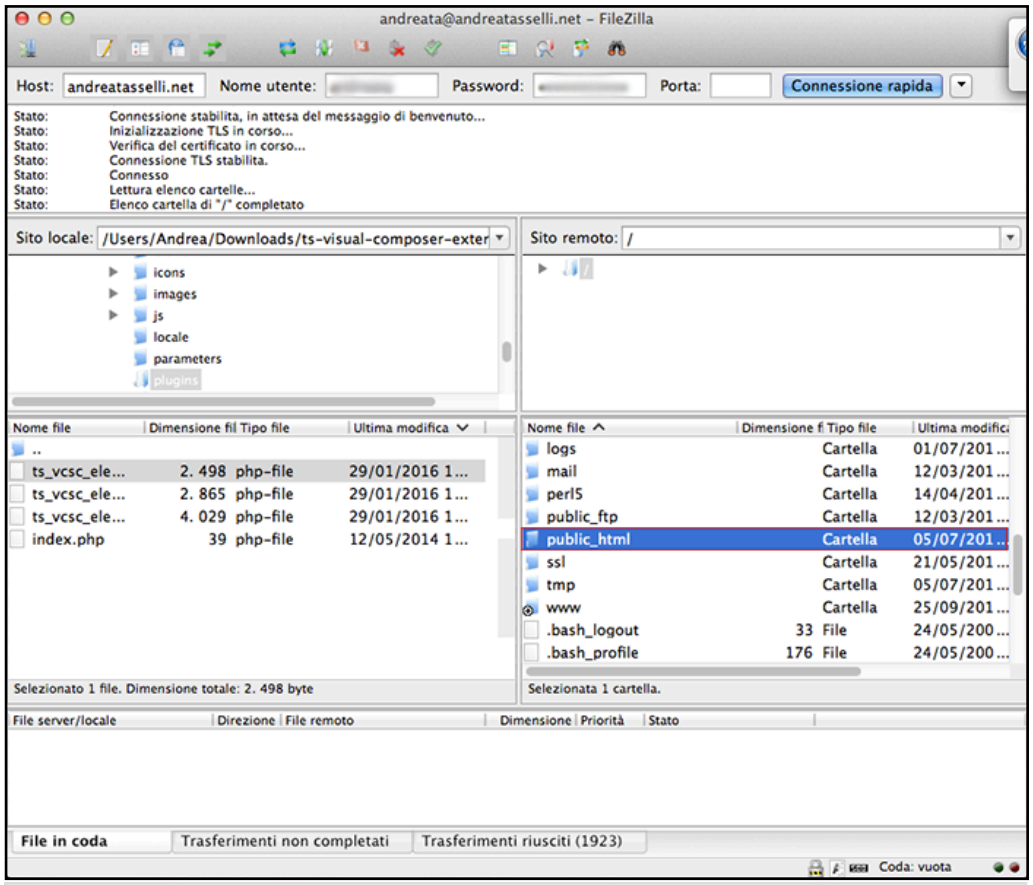
In alternativa, se risultassimo sprovveduti, prima che un attacco di forza bruta abbia successo, avvengono dei segnali che possono salvarci la pelle.

Il primo è un sovrautilizzo della memoria disponibile sul sito. Il propio hosting di solito ci informa su questo utilizzo incontrollato, e addirittura minaccia di chiudere il sito per almeno un giorno in caso il problema non sia risolto.

Tutto questo è assolutamente normale, poiché il fornitore non ha modo di stabilire automaticamente che il sovrautilizzo sia determinato da un attacco di forza bruta, ma potrebbe pensare che il problema venga da un numero eccessivo di utenti che vista il sito (quindi di conseguenza dovremo pagare un servizio premium).

1 Utilizza FIleZilla come spiegato nel Kit Fragola.

2 Inserisci le credenziali che ti saranno state consegnate dal tuo fornitore di hosting.



3 Vai su public_HTML > “cerca il tuo sito se ne hai più di uno” > tasto destro sul file .haccess > clicca su “visualizza/modifica”

5 Inserisci la seguente stringa di codice all’interno del file, dove 222.222.222.222.222 sta per l’indirizzo IP che non vuoi più far accedere al sito.

```
#Blocco IP
order allow,deny
Deny from 222.222.222.222.222
allow from all
```

Fantastico no? Adesso sai anche come difenderti dagli attacchi Hacker utilizzando metodi avanzati.

Buon coding!