SPAW
“Progetto teledidattico”

Introduzione

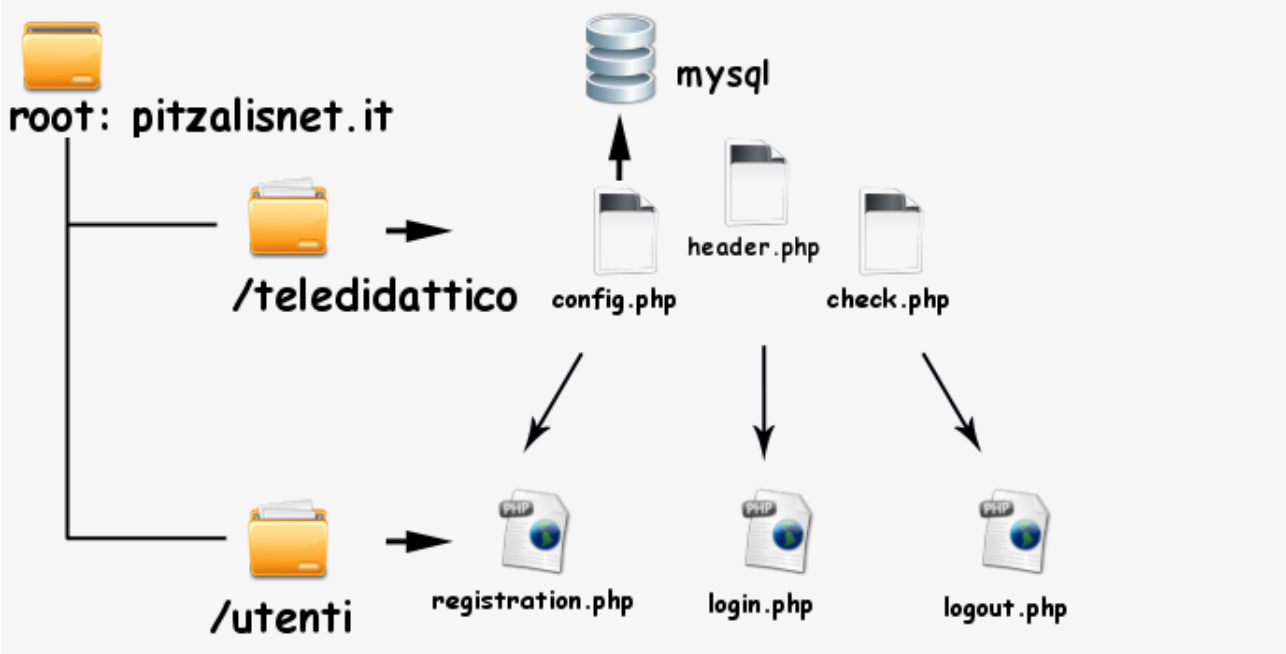
Il progetto nasce con l'idea di creare un supporto per gli studenti iscritti al corso di laurea di Ingegneria Informatica e Ingegneria Meccanica con “didattica a distanza”. L'invito a portare avanti questa idea è stata dello stesso prof. Calabrese che dava disponibilità a seguire un gruppetto di studenti che fosse interessato a sviluppare appunto un progetto utile a compendio del già presente CorsiOnLine (my.unipr.it). Il “gruppetto” avrebbe dato certamente un'impronta meno personale e senza dubbio più funzionale rispetto all'attuale risultato, ma per vari motivi mi sono ritrovato a portare avanti in solitudine un'idea che già comunque era partita, sia pure con un forte e importante sostegno da parte dei colleghi iscritti.
La trama del progetto è sviluppata in modo tale da poter essere facilmente ampliata e implementata anche per estenderne in futuro i servizi, col supporto dei docenti e dei tutori, a tutti gli studenti, con particolare attenzione a quegli studenti lavoratori che intendessero iscriversi all'Università di Parma malgrado la sospensione della modalità “teledidattica”.

Struttura generale

Partendo da nozioni minime di linguaggio HTML e con l'ausilio di strumenti quali Dreamweaver e Namo Web Editor in versione trial, noti come programmi “tuttofare” per neofiti, ho allestito, nel dominio da me acquistato presso il provider Aruba, alcune pagine relative a corsi di cui avevo già conseguito l'esame, mettendo a disposizione suggerimenti, appunti, materiali e quant'altro in mio possesso, oltre a link utili. Malgrado l'elementarità del sito web, gli accessi allo stesso, non solo da parte dei cosiddetti teledidattici, cominciavano ad essere considerevoli.
Con l'inizio dello studio di strumenti per Applicazioni Web, ho abbandonato i suddetti strumenti utilizxzando piuttosto un text editor potente, come Notepad++. Mi sono quindi inizialmente dedicato a una sorta di pulizia (non ancora terminata) delle pagine web già in uso, riunendo in due pagine di stile (style.css e style2.css) posizionate nella root, la maggior parte degli aspetti fondamentali che identificano le pagine web da me costruite. In tal modo, nel rispetto delle direttive del W3C, a cui mi sono appoggiato per cercare eventuali errori, ho notevolmente alleggerito le pagine originarie costruite con gli strumenti prima menzionati, separando nettamente il contenuto dalla presentazione, come richiesto dalle specifiche CSS. Un passo alla volta sono passato dall'HTML alla sua recente evoluzione XHTML.
L'utilizzo di piccoli codici javascript, mi ha permesso di aprire finestre separate, successivamente di creare effetti tipo dock del Mac, modificando e adattando alle mie esigenze più script trovati in rete, e di crearne personalmente di semplici, come il calcolo della media universitaria sia per informatici che per meccanici.
In seguito mi sono avvicinato al linguaggio di scripting PHP, che mi ha permesso di sviluppare pagine dinamiche dando più “vita” al progetto e permettendomi il collegamento con il database.
Per agevolare l'utente nelle ricerche, la pagina principale contiene, un piccolo form di ricerca all'interno del sito, costituito da un servizio offerto da Google e da un file in formato xml, da aggiornare saltuariamente, detto `sitemap`, che include le informazioni riguardanti il contenuto del sito e delle pagine che lo costituiscono.
Tra le altre cose, forse ciò che rende più interessante il progetto è la possibilità da parte dell'utente, che preferibilmente potrebbe essere un tutore piuttosto che uno studente, di fare l'upload di materiale didattico specifico per ogni materia.

Accesso al sito

Con il seguente schema intendo riassumere il lavoro svolto.



Il file **registration.php,** se l'utente non è già loggato, all'invio del modulo compilato si occupa di gestire alcuni controlli, tra cui essenzialmente, se l'utente, la cui username di accesso deve coincidere con un indirizzo e-mail, è inserito o meno nella lista degli iscritti alla mailing list del Nettuno, oltre, naturalmente, ad inserimenti a mia discrezione.
Questa parte di controllo si può riassumere nelle seguenti righe:

```
<?php session_start();
switch ($_REQUEST["campoNome"]) {
case 'user1@studenti.unipr.it':
case 'user2@studenti.unipr.it':
....
....
break;
default:
die("...
```

Dove viene confrontato il nome inserito nel campo “campoNome” con la lunga lista che segue. Questo purtroppo mi obbliga a verificare continuamente eventuali nuove iscrizioni alla lista del Nettuno e aggiornare il file **registration.php** manualmente, ma più avanti esporrò una possibile soluzione.
La struttura della tabella del database è definita dal seguente file di testo utilizzato per la query:

```
CREATE TABLE `utenti` (
`user_id` mediumint(8) NOT NULL auto_increment,
`user_email` varchar(255) NOT NULL default '',
`user_password` varchar(32) NOT NULL default '',
PRIMARY KEY (`user_id`)
);
```

La password verrà inserita attraverso l'algoritmo di criptazione **md5**.
Portata a buon fine la registrazione e chiusa automaticamente la finestra ci si trova nuovamente nella pagina **login.php**, dove l'utente può finalmente richiedere l'accesso ai servizi offerti.
Lo script **login.php** controllerà innanzitutto se siamo già loggati, in tal caso ci reindirizzerà direttamente verso la homepage del teledidattico. Da notare, che la stessa pagina html del login verrà inviata dal server solo dopo aver valutato alcune condizioni:

```
<?php
session_start();
…
if(.....)
{
...
} else {
//visualizza il form login
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="it" lang="it">
<head>
...
<form...
...
</body>
</html>
<?
}
?>
```

Nel caso della spunta della casella 'ricordami', si provvederà a creare il cookie attraverso la riga di codice `setcookie($nome_cookie, $joined, time()+$scadenza, $percorso, $dominio);`
dove le variabili `$nome_cookie`, `$scadenza`, `$percorso`, `$dominio` sono state precedentemente definite nel file **header.php** mentre `$joined` non è altro che il “valore” del cookie, definito da: `$joined =''.$_POST['email'].'[]'.md5($_POST['password']).'';`
Dopo di che si passa al recupero delle informazioni con il relativo confronto nel database e in caso di esito positivo verranno impostate le variabili di sessione nel seguente modo:

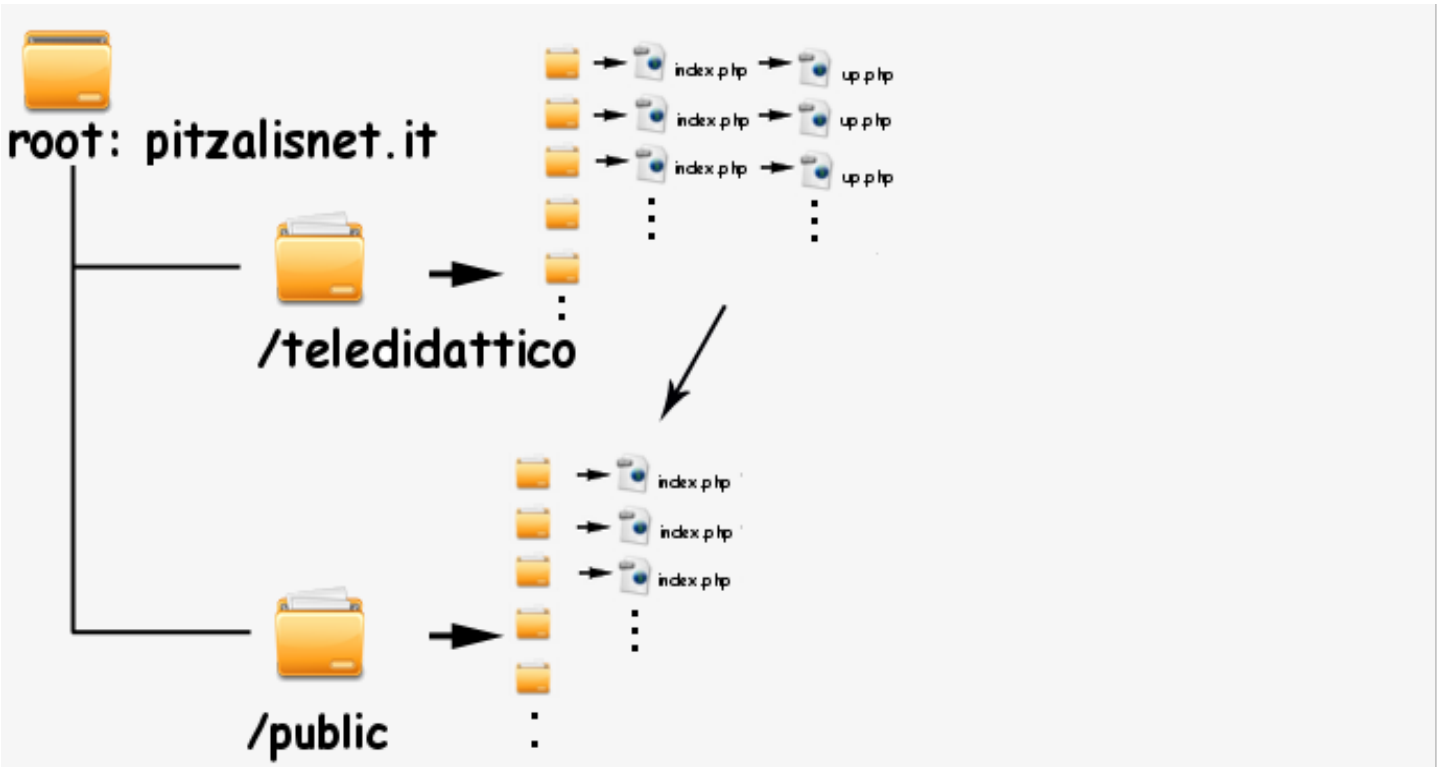
```
$_SESSION['logged_in'] = 1;
$_SESSION['email'] = $_POST['email'];
$_SESSION['password'] = $_POST['password'];
session_write_close();
```

A questo punto saremo loggati e reindirizzati verso la homepage del teledidattico:
Infine, il file **header.php**, che viene incluso sia in **registration.php** come in login.php, si occupa, oltre che a definire alcuni parametri, di controllare se nella nostra macchina è presente un cookie valido per il dominio, in caso di riscontro positivo preleva le informazioni necessarie (user_email e password) e imposta la sessione con il valore '1': `$_SESSION['logged_in']=1`.
Tale controllo non viene effettuato se la sessione è già attiva.
ob_start, presente nel codice, dovrebbe servire per ottimizzare l'esecuzione dello script con un discreto miglioramento della velocità nel caricamento della pagina. Ob sta per Output Buffering, e il suo scopo è di inviare ogni dato dello script in una sorta di memoria temporanea, il buffer appunto.
Il piccolo file **check.php**, che verrà incluso in tutte le pagine web che richiedono protezione, verifica in sostanza, col seguente codice: `if ($_SESSION['logged_in'] ==0) {...`
se l'utente è autorizzato o meno. In caso positivo la pagina verrà aperta, altrimenti viene creata una pagina html di avviso che reindirizza automaticamente nella pagina di login.
Le pagine che necessitano di protezione avranno quindi in testa semplicemente il richiamo a check.php:

```
<? include("check.php");?>
<!DOCTYPE html PUBLIC '-//W3C//DTD XHTML 1.0 Transitional//EN'
...
```

Upload

Come accennato nell'introduzione, le pagine relative a ogni materia contengono un form contenente la possibilità di inviare in una cartella apposita (anch'essa una per ogni materia) posizionata a sua volta in una cartella `public` (pubblica appunto), del materiale utile al corso stesso.
Schematizzo in questo modo il lavoro:



In pratica nel file **index.php,** relativo a ogni specifica materia, vi è un form in cui viene richiamato il file **up.php.** La riga chiave del form è: `<form action="up.php" method="post" enctype="multipart/form-data">...` dove, oltre al richiamo del file **up.php**, specificando il valore "multipart/form-data" per enctype, il contenuto di ciascun file sarà 'impacchettato' per l'inoltro in una sezione separata di un documento multi-parte, cioè per inviare grandi quantità di dati (file, immagini ecc.). Il file **up.php** si occupa in definitiva di dirottare il file temporaneo, copia del file dell'utente, in una cartella pubblica apposita:

```
move_uploaded_file($_FILES["file"]["tmp_name"],
$PercorsoDominio. $public . $_FILES["file"]["name"]);
```

Sempre all'interno della pagina **index.php** relativa al corso, vi è un link che, tramite una finestra separata, porta direttamente l'utente alla cartella di upload, permettendone a sua volta il download, tramite un ciclo di scrittura dove, davanti a ogni riga creata viene posizionato il tag `"<a href=..."`

Un capitolo a parte meriterebbero l'essenziale form presente nella pagina di login, anch'esso scritto in linguaggio PHP, che consente anche all'utente non registrato di inviare un'email all'amministratore del sito, e il guestbook necessario all'utente per lasciare un suo contributo scritto.

Appendice

Possibili miglioramenti

Certamente un simile sistema con un 'eccesso' di interattività avrebbe bisogno di una modalità più efficace di controllo da parte dell'amministratore, senza dover continuamente accedere al database. Pertanto prevedo di aggiungere nel sito un modulo, con accesso esclusivo da parte del webmaster, che gli consenta di verificare velocemente sia la presenza di nuovi contributi da parte degli utenti nel guestbook, sia la presenza di nuovi file inseriti tramite apposita interfaccia di upload, consentendone eventualmente l'immediata cancellazione per scorrettezze o altro.

Sarebbe interessante in futuro, in accordo col CEDI di Parma, implementare la possibilità di inserire, da parte del docente, lezioni precedentemente videoregistrate, di modo che lo studente possa visionarle in streaming senza necessariamente dover scaricare il file. Naturalmente l'accesso a questa particolare area di upload sarebbe consentita solo ed esclusivamente al personale docente o comunque di servizio all'Università.

Possibile soluzione per l'accesso agli iscritti alla mailing list

Per ciò che concerne il problema d'accesso al sito ai nuovi iscritti nella mailing list del Polo Tecnologico, di cui si è accennato nella parte relativa alla registrazione, si potrebbe automatizzare il processo di lettura delle presenze nella lista in questo modo: il CEDI potrebbe mettere in una cartella privata, quindi non accessibile direttamente dall'utente, un file di testo con l'elenco aggiornato degli iscritti e nel file **registratio.php**, eliminando la lunga lista degli user_mail, si potrebbe andare a leggere il suddetto file di testo confrontandolo, riga per riga, col dato inserito dall'utente.

In pratica: il CEDI produce in automatico in una cartella privata, un file di testo chiamato, ad esempio, **iscritti.txt** dove vi è solo una lista di indirizzi e-mail degli iscritti. Dall'altra parte il file **registration.php**, può mantenere la seguenti righe di codice:

```
switch ($_REQUEST["campoNome"]) {
case 'user1@studenti.unipr.it':
case 'user2@studenti.unipr.it':
...
```

coi soli indirizzi a discrezione dell'amministratore, ma in coda vi si dovrebbe inserire l'ulteriore codice che approssimativamente potrebbe risultare come segue:

```
if (isset($_REQUEST["campoNome"])) {
  $puntatore = fopen("http://unipr../private/iscritti.txt", "r");
  $trovato = 0;
  while ((!feof($puntatore)) && (!$trovato)) {
    $linea = fgets($puntatore);
    $trovato = stristr($linea, $_POST["email"]);
    $puntatore++;
  }
  fclose($puntatore);
  if (($trovato)
...
ecc.
```

la cui funzione è di leggere i dati dal file di testo e confrontarli con quello immesso dall'utente.

Progetto realizzato da Claudio Pitzalis, iscritto al terzo anno in Ingegneria Informatica presso l'Università degli Studi di Parma. Presentato all'esame di Strumenti per Applicazioni Web il 24 giugno 2009. Docente ed esaminatore prof. Eduardo Calabrese.