Tecnologie Web

Relazione di Progetto

Stefano Gismano

Matr. 917723

1. Tema del sito

Bloggify è un sito che permette agli utenti di pubblicare post, in modo simile a un social network come Twitter o Reddit. Gli utenti possono anche mettere like e commentare i post.

Per poter utilizzare il sito è necessario possedere un account, creabile tramite la pagina Register.

La homepage del sito permette di visualizzare i post pubblicati dagli utenti. Essi possono essere visualizzati in ordine cronologico, di numero di like o di numero di commenti.

La pagina di un post permette di interagire con esso, aggiungendo o rimuovendo un commento e dando la possibilità di cancellare il post, nel caso in cui l'utente loggato sia l'autore del post o un amministratore.

La pagina Submit permette di pubblicare un post tramite un form, composto da un titolo e un testo.

La pagina Search permette di cercare il nome di un utente e, se esso esiste, di visualizzare i post che ha pubblicato.

La pagina dell'utente permette di visualizzare tutti i post pubblicati dall'utente attualmente loggato.

2. Funzionalità

- Register: la creazione di un account è necessaria per l'utilizzo del sito. Per creare un account è necessario inserire un nome utente e una password. Il nome utente deve non essere già presente tra quelli del database, e deve essere lungo tra i 3 e i 20 caratteri. La password deve essere di una lunghezza superiore a 8 caratteri, e deve essere confermata in un ulteriore campo del form. Se entrambi i campi sono uguali, allora la password viene cifrata tramite hash con password_hash e viene inserita nel database insieme al nome utente.

-Login: il form per il login contiene due campi, per l'username e per la password. Una volta inseriti i dati e inviato il form, viene controllata la presenza della tupla all'interno del database. Se il login va a buon fine, viene creata una sessione per l'utente e viene caricata la homepage.

-Contenuto generato dall'utente: l'utente può effettuare alcune operazioni: creare o cancellare un post, aggiungere o rimuovere un commento a un post, e mettere like a commenti e post.

3. Caratteristiche

- Usabilità: Il sito utilizza un font semplice e un testo di colore scuro su sfondo bianco per rendere le scritte facilmente leggibili. Le operazioni più importanti, quali il submit dei post e la ricerca degli utenti, sono reperibili nella parte superiore del sito, per essere sempre facilmente trovabili.

- Interazione: l'utente può aggiungere like ai post e ai commenti tramite un pulsante.

- Sessioni: il sito prevede un sistema di sessioni, che vengono gestite tramite la session_start(). Nel caso in cui non vi sia una sessione attiva, l'utente viene reindirizzato verso la pagina di login. La sessione viene terminata quando l'utente effettua il logout dal sito.

-Interrogazione del database: in ogni caso in cui sia necessario interrogare il database, vengono preparati degli statements per ogni query da effettuare, e vengono effettuati dei controlli per gestire eventuali eccezioni.

- Validazione dei dati di input: vengono effettuati controlli su tutti i dati inseriti dall'utente, ad esempio per il nome utente e la password inseriti durante la registrazione e il login. In particolare, vengono utilizzate due funzioni javascript, isvalid_user(username) e isvalid_password(password), che controllano rispettivamente che l'username abbia solo caratteri alfanumerici e che sia di lunghezza compresa tra 3 e 20, e che la password sia di lunghezza maggiore a 8. Inoltre avviene la sanitizzazione degli input tramite il binding dinamico dei parametri, usando bindParam.

- Sicurezza: la sicurezza è garantita dalla validazione degli input e dall'utilizzo di un algoritmo di hashing per salvare le password.

4. Front-end

- Separazione presentazione/contenuto/comportamento: la presentazione è separata dal contenuto in quanto vengono utilizzati dei fogli css che vengono inclusi solo quando necessario. Il contenuto è separato dal comportamento in quanto i file Javascript utilizzati sono separati dai file html, e vengono inclusi solo quando necessario, creando un approccio unobtrusive.

- Soluzioni cross platform: viene utilizzato un metatag che rendono il sito compatibile con browser come Internet Explorer

- Organizzazione file e cartelle di progetto: il sito è organizzato in più cartelle:

css contiene i fogli di stile.

img contiene la favicon del sito

html contiene la presentazione del sito.

js contiene tutti gli script JS per la gestione del front-end.

php contiene i file PHP per la gestione del back-end.

sql contiene la struttura del database.

5.Back-end e comunicazione front/back-end

- Architettura generale classi/funzioni PHP: Il front-end comunica con il back-end tramite chiamate Ajax di JQuery sfruttando oggetti JSON. Gli script PHP utilizzano la funzione json_decode() per trasformare tali oggetti da JSON a PHP.

-Schema del database:

COMMENT (commentid, postid, time, text, userid)

• COMMENT(postid) referenzia POST(postid)

• COMMENT(userid) referenzia USER(userid)

COMMENTLIKES (commentid, userid)

• COMMENTLIKES(commentid, userid) referenzia COMMENT(commentid, userid)

LIKES (postid, userid)

• COMMENTLIKES(postid, userid) referenzia POST(postid, userid)

POST (postid, timestamp, title, userid, text)

• POST(userid) referenzia USER(userid)

USER (userid, password, admin)

- Descrizione delle funzioni remote: un esempio adeguato riguarda la creazione di un post da parte di un utente. In un caso del genere, viene effettuata una chiamata Ajax di tipo POST per il submit del contenuto verso il server.

```
$.ajax({
  url: "/progetto/php/post/submit.php",
  type: "POST",
  data: formData,
  contentType: false,
  cache: false,
  processData: false,
```

```
 success: function (data) {
   $(window.location).attr("href", '/progetto/html/post/comments.php?postid=${data.message}');
 },
 error: function (responseJSON) {
   return showError("Error submitting post.");
 }
```

Il codice in "/progetto/php/post/submit.php" serve ad accogliere la richiesta inviata.

Viene verificato che i campi necessari siano presenti, e poi il post viene inserito nel database:

```
$db = dbConnect ();
$insertpost = $db->prepare (
  "INSERT INTO post (title, userid, text) VALUES (:title, :username, :text) ; "
) ;
$insertpost->bindParam (':title', $POST["title"], PDO::PARAMSTR) ;
$insertpost->bindParam (':username', $SESSION["username"], PDO::PARAMSTR) ;
$insertpost->bindParam (':text', $POST["text"], PDO::PARAMSTR) ;
try {
$insertpost->execute() ;
} catch (PDOException $ ) {
respond (500 , "Error inserting post . " ) ;
}
```

La funzione di callback specificata in caso di successo porta al caricamento della pagina di presentazione del post.