

One Piece Love



Tecnologie Web A.A 2022/2023

11/04/2023

Studente: Bollini Simone (simone.bollini@studio.unibo.it)

Matricola: 0000975149

Introduzione:

- Ho deciso di realizzare un Social Network per appassionati di One Piece, manga giapponese che racconta le vicende di una ciurma di pirati. Le funzionalità sono quelle base di un social network, ovvero creare un account, loggarsi, inserire nuovi post, seguire nuovi amici, mettere mi piace e commentare i post, ricevere notifiche.

Elementi distintivi di questo social network sono:

- in fase di registrazione la possibilità di scegliere il proprio personaggio preferito di One Piece, questa scelta andrà ad impostare l'immagine profilo dell'utente.
- Una volta loggati nella home sono mostrati in automatico i post creati da utenti che amano lo stesso personaggio da noi scelto.

Strumenti utilizzati:

- Lato server **PHP**
- Lato client: **Javascript, Axios**
- Css: **Bootstrap**
- Per creare database **DB-Main**
- **MySQL**

Ho preso spunto dal codice fatto in laboratorio, vedi bootstrap.php oppure la funzione uploadImage(), l'utilizzo di template base e dei templateParams.php che estraggono dati direttamente dal server tramite \$dbh->

Design:




One Piece Love



Login

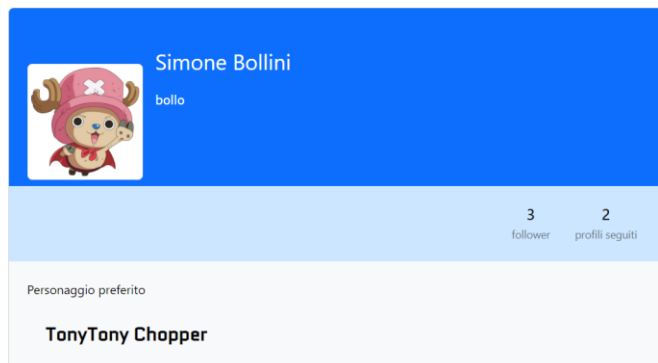
Accedi

Ricordami ☐

Non hai un account? [iscriviti](#)

[Home](#) [Profilo](#) [Logout](#)



Per realizzare il design sono partito dalla bandiera della ciurma principale del Manga, ho scelto questa immagine anche come logo del sito e utilizzando i colori presenti nella bandiera ho cercato di riprenderla nelle varie sezioni.

Fin da subito mi sono posto come obiettivo una struttura del sito e un design semplice ma funzionante.

Sono partito sempre da due template: base.php e login.php per andare a costruirci sopra tutto il sito.

Base.php ha al suo interno il codice per realizzare l'header con il menù, il bottone per visualizzare le notifiche e ricercare utenti.

Registrazione e login:

Entrambe le pagine tramite php vanno a costruire sul template **login.php** con i rispettivi file js le proprie form sfruttando una promise get di axios per estrapolare i dati dal server passando per le rispettive api-.php.

Sempre tramite javascript vengono controllati gli eventi di submit che inviano i dati tramite axios.post.

In fase di login se fleggato "Ricordami" vengono settati i cookie nel browser. Se i dati indicati in fase di login sono corretti vengono settati tramite la funzione php registraLoginUser in sessione Nickname e IdUser e viene reindirizzato nella home altrimenti viene dato un messaggio d'errore nella form.

Nella pagina di registrazione 'iscriviti' viene controllato tramite javascript, prima di passare i dati al server, che tutti i campi siano compilati e i campi password e conferma password corrispondano. Inoltre tramite una chiamata ajax mentre si digita nel form il nickname viene verificato che nel database non ne sia presente uno uguale altrimenti viene dato un messaggio d'errore sulla form.

Home con feed di post di utenti seguiti:

Nella home vengono visualizzati i post degli utenti seguiti e di quelli che amano lo stesso personaggio di One Piece dell'utente. La pagina viene creata partendo da Php che richiama i file js easo riempie i \$templateParams["notifiche"] e ["notificheDescr"] con i dati che andranno a inserire le notifiche per l'utente. In home.js ho inserito la costruzione del bottone per richiama la possibilità di inserire nuovi post, mettere una sezione a comparsa mi

sembrava stare bene con la mia idea. La pagina inoltre tramite `axios.get api-post.php`, passando la pagina come parametro, riesce a distinguere i casi in cui si è nella pagina profilo da quando si è nella home per utilizzare la corretta query per generare i post da richiamare.

Commenti di post:

Vengono inseriti tramite la funzione javascript `inserisciCommento()`

che tramite `axios.post` passa i dati tramite php al server e se l'inserimento del commento va a buon fine richiama la funzione `js mostracommenti()` che ricarica i commenti. Inoltre viene verificato se il commento è dell'utente loggato nella sessione corrente in modo di mostrare solo in questo caso la possibilità di cancellare il commento.

Post con testo e/o foto:

E' possibile inserirli dalla form creata dal bottone `Racconta()` descritto in precedenza, in fase di submit utilizzando `axios.post` inserisce il post per poi usando js viene ricaricata la pagina dell'homepage.

Follow di utente:

Per seguire un nuovo utente è possibile, aprendo il suo profilo seguirlo o non seguirlo più. In fase di caricamento della pagina profilo, se è verificato non essere quella dell'utente della sessione corrente, viene passato tramite `axios.get` dal server se l'utente è già seguito o meno dall'utente in questo modo in base al risultato tramite js viene creato il bottone corretto.

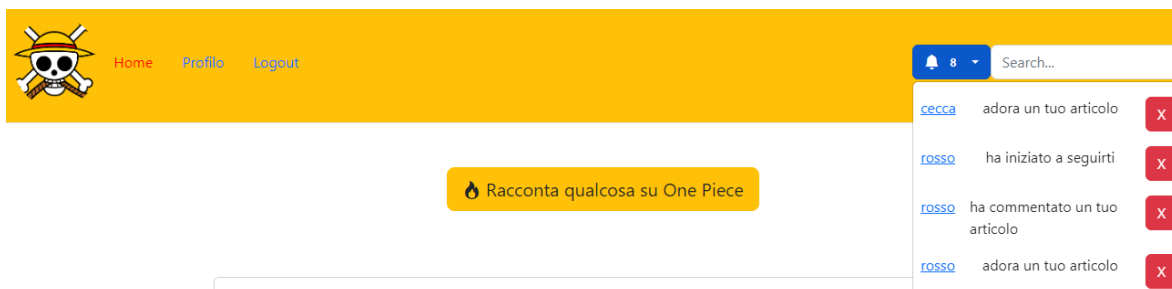
Una volta caricata la pagina se viene eseguita l'azione di

follow/unfollow tramite una chiamata ajax vengono aggiornati i dati nel database e nella pagina.

Profilo utente con post, follow e follower:

Sfruttando la chiamata `axios.get` descritta sopra per follow utente vengono passate le informazioni del profilo **compresi** follow e follower. Per indirizzare la pagina sul profilo corretto in fase di ricerca dell'utente viene passato all'url il parametro `?user=` che viene utilizzato dalla chiamata `axios.get('api-profilo.php'+location.search)` in modo da verificare la differenza tra il `$_GET["user"]` e `$_SESSION["iduser"]`. Per inserire i post pubblicati dall'utente del profilo ho sfruttato la stessa funzione `showProfilo()` usata per la homepage ma ricevendo come parametro una diversa pagina utilizzerà una query differente.

Notifiche:



Una volta loggato l'utente dovrà vedere le notifiche in tutte le pagine del sito, per questo motivo essendo elemento ricorrente le ho inserite direttamente dentro il template `base.php` sfruttando la classe dropdown di Bootstrap al suo interno vengono innestate tramite php foreach tutte le notifiche aperte presenti per l'utente sfruttando il `templateParams["notificheDescr"]` e vengono chiuse con una chiamata ajax passando per la funzione `js closeNotifica()` che esegue una richiesta `axios.post`.

Per gestire tutti i tipi di notifica ho dovuto fare in modo che all'inserimento di un commento, di un mi piace e di una nuova amicizia venisse creata una nuova notifica. Le notifiche sono state distinte per idtipo e da un boolean `notificaAperta` per capire se da visualizzare o meno.

