02NPYPD - LINGUAGGI E AMBIENTI MULTIMEDIALI A

FOLLOWING USERS

Seguire ed essere seguiti da altri utenti

Lo scopo di questa esercitazione è duplice: da una parte porterà a compimento il nostro social network di base, dall'altra permetterà di affrontare alcuni concetti avanzati che riguardano (nuovamente) la modellazione. Nella soluzione sarà presente anche un primo esempio di utilizzo di Ajax in Rails (che sarà anche discusso in aula). Prerequisito è il set di slide intitolato "*Lab 7 - Warm Up*" e la realizzazione delle precedenti esercitazioni.

ESERCIZIO 1

Basandosi su quanto mostrato nel set di slide presentate in aula, generare un modello chiamato *Relationship* che:

- abbia due attributi, `follower_id` e `followed_id` (entrambi di tipo integer);
- tra gli attributi accessibili del modello figuri solo `followed_id`;
- entrambi siano essere sempre presente nella creazione di un nuovo post;
- preveda, nella sua migrazione, *tre* indici: uno per `follower_id`, uno per `followed_id` e uno che definisca l'unicità dell'esistenza della coppia `follower_id` e `followed_id`;
- utilizzare la funzionalità `belongs_to` e `has_many` per collegare relationships e utenti; tali metodi vanno usati nei modelli delle relationship e degli utenti;
- nel modello degli utenti, has_many dovrà definire `follower_id` come chiave esterna (`foreign_key: "follower_id"`), altrimenti Rails cercherà un campo chiamato *relationship_id* che non esiste nel nostro database;
- assicurarsi che, quando un utente viene cancellato dal social network, anche tutte le sue relazioni vengano eliminate (aggiungere a `has_many` nel modello degli utenti la proprietà `dependent: :destroy`).

ESERCIZIO 2

Gestire i meccanismi di following e follower. Nel modello degli utenti, collegare gli utenti ai `followed_users` come mostrato nelle slide e spiegato in aula. Allo stesso modo, collegare gli utenti ai followers. Definire, sempre nel modello degli utenti, tre metodi: `following?(other_user)`, `follow!(other_user)` e `unfollow!(other_user)` che permetteranno, rispettivamente, di sapere se un utente ne sta seguendo un altro, di seguire un altro utente (creando una nuova relationship) o di smettere di seguire un utente.

Modificare il task di Rake che permette di creare utenti e post "finti" in modo tale che possa generare anche followers finti:

```
users = User.all
user = users.first
```

```
followed_users = users[2..50]
followers = users[3..40]
followed_users.each { |followed| user.follow!(followed) }
followers.each { |follower| follower.follow!(user) }
```

Resettare e popolare il database utilizzando i soliti task di Rake.

Aggiornare le route dell'utente come mostrato nelle slide (e spiegato in aula) e aggiungere le route "standard" per le relationships. Aggiungere, poi, sotto le informazioni sull'utente loggato nel sito (in *show.html.erb* e *home.html.erb*) due dati: il numero degli utenti seguiti e il numero di utenti da cui si è seguiti.

ESERCIZIO 3

Realizzare nella pagina del profilo utente (*show.html.erb*) un bottone etichettato come "Follow". Questo bottone sarà, in realtà, un form che permetterà di costruire una nuova relazione). A questo form sarà associata l'azione `create` del controllore delle Relationship. Crearla. Se l'utente, invece, è già seguito, cambiare il testo del bottone in "Unfollow", prevedendo il corrispondente form che sarà associato all'azione `destroy` del controllore delle Relationship. Creare anche questa azione.

Creare due nuove azioni nel controllore degli utenti: `following` e `followers`. Queste azioni faranno il rendering di una vista chiamata "`show_follow`" (da creare) che permetterà di visualizzare gli utenti seguiti o gli utenti da cui si è seguiti.

ESERCIZIO 4 (FACOLTATIVO)

Gli utenti loggati al sito, nella home page (home.html.erb) dovranno vedere i loro post più tutti quelli delle persone a cui seguono. Realizzare questa funzionalità, agendo sul modello dell'utente, su quello dei post (facendo una query al database) e modificando la vista relativa alla home page.