02NPYPD - LINGUAGGI E AMBIENTI MULTIMEDIALI A

HANDLING USERS

Login, logout e gestione degli utenti

Lo scopo di questa esercitazione è quello di approfondire alcuni metodi utilizzati per l'accesso di un utente registrato al nostro social network. In questa esercitazione, infatti: (a) si aggiungerà al sito la possibilità di eseguire il login/logout e (b) si implementerà la gestione avanzata degli utenti (visualizzazione dei profili, modifica, ecc.).

SIGN IN, SIGN OUT

Si vuole realizzare il meccanismo di login/logout e permettere il login permanente nel nostro sito. Per "login permanente" si intende il fatto che l'utente rimarrà loggato fintanto che non farà esplicitamente un sign out. Per far ciò utilizzeremo una sessione, intesa come una connessione semi-permamente tra un client e un server, modellandola come usa risorsa REST: il login corrisponderà alla creazione di una nuova sessione e il logout alla sua distruzione. Innazitutto, nel modello (e nel database) creare un campo `remember_token` di tipo `string`; tale campo è da rendere un indice nel database tramite la rispettiva migrazione. Sempre nel modello, occorre aggiungere l'istruzione `before_save :create_remember_token`, in modo da creare il token prima del salvataggio su DB dell'utente e realizzare il metodo `create_remember_token` utilizzando `self.remember_token = SecureRandom.urlsafe_base64`: il token sarà così un numero casuale difficilmente replicabile. Tale token va poi memorizzato nell'helper di sessione, salvato in un cookie che scadrà 20 anni dopo la sua creazione, grazie all'istruzione: `cookies.permanent[:remember_token] = user.remember_token`. Si possono ora aggiungere al social network le funzionalità di login/logout per gli utenti registrati:

- generare un controllore di sessione che definisca le azioni `new`, `create`, `destroy`;
- inserire una route per tale risorsa, così come fatto nella precedente esercitazione per la risorsa "utenti";
- inserire una named route che colleghi l'azione `new` a `/signin`;
- inserire una named route che colleghi l'azione `destroy` a `/signout` (attraverso una richiesta HTTP DELETE), in questo modo: `match '/signout', to: 'sessions#destroy', via: :delete ;`
- l'azione `new` sarà l'unica ad avere associata una vista, che presenterà il form per il sign in, con un campo per l'indirizzo e-mail e uno per la password; l'azione associata all'invio di tale form è da esplicitare: `form_for(:session, url: sessions_path) do |f|` in quanto la sessione non è un modello;
- l'azione `create` servirà per il sign in vero e proprio e presenterà un messaggio di errore nel caso in cui i dati inseriti per il login non siano corretti;
- l'azione `destroy` servirà per il logout.

MODIFICARE, VISUALIZZARE E RIMUOVERE UTENTI

Aggiungere, al social network di base, le seguenti funzionalità.

1) La possibilità di aggiornare il *proprio* profilo, *dopo aver eseguito il login**, utilizzando l'azione `edit` nel controllore degli utenti. Utilizzare un form e il servizio Gravatar per modificare la propria immagine del profilo (basta un link a http://gravatar.com/emails). Gestire, con l'azione `update`, la buona o cattiva riuscita di tale aggiornamento, visualizzando gli eventuali errori.
2) La possibilità di vedere, una volta eseguito il login, tutti gli utenti registrati al sito, definendo un'azione `index` nel controllore degli utenti. Per generare automaticamente un grande numero di utenti "finti" è consigliato aggiungere la gemma `faker` al `Gemfile` del progetto (vedi *suggerimento* al fondo dell'esercizio).
3) Superato un certo numero di utenti nella pagina `index`, dividerli in pagine utilizzando la gemma `will_paginate`[1] (da installare nella versione 3.0.3, insieme alla gemma `bootstrap-will_paginate` – versione 0.0.10 – se si sta usando Bootstrap).
4) La possibilità, da parte di utenti "amministratori" di rimuovere altri utenti. Si consiglia di modificare il modello (e il database) in modo da includere un campo `admin` (che potrà contenere vero/falso) e di utilizzare un'azione `destroy` all'interno del controllore degli utenti. Mostrare la possibilità di rimuovere utenti nella pagina che elenca tutti gli utenti del sito.

* si suggerisce di utilizzare `before_filter :signed_in_user, only: [:edit, :update]` nel controllore degli utenti, realizzando il metodo `signed_in_user` in modo tale che, se l'utente non è loggato, visualizzi un avviso e faccia il redirect alla pagina di login. `before_filter` imposta dei filtri (uno, in questo caso, realizzato dal metodo `signed_in_user`) che verranno eseguiti prima dell'azione chiamata dal controllore. Utilizzando un altro `before_filter`, controllare che l'utente a cui si vuole modificare il profilo sia il proprio.

Suggerimento: per creare un numero di finti utenti, su può utilizzare la gemma `faker` (da installare nella versione 1.0.1) insieme a un task rake simile al seguente (da inserire in un file con estensione `.rake` nella cartella `lib/tasks`):

```
namespace :db do
  desc "Fill database with sample data"
  task populate: :environment do
    User.create!(name: "Example User",
                 email: "example@lam.it",
                 password: "pwd123",
                 password_confirmation: "pwd123")
    99.times do |n|
      name  = Faker::Name.name
      email = "example-#{n+1}@lam.it"
      password  = "password"
      User.create!(name: name,
                   email: email,
                   password: password,
                   password_confirmation: password)
    end
  end
end
```

[1] https://github.com/mislav/will_paginate/wiki