

Quicknote AB-151-04 Joel Liechti

Zusammenfassung:

In diesem AB ging es um Flashmessages, Modale Dialoge und Gravatar. Das Konzept von Gravatar ist interessant allerdings nur semi-praktisch.

Die Modalen Dialoge und Flashmessages sind sehr praktisch und schön und werden heutzutage viel verwendet. Die Flashmessages wurden mit der Hilfe von Toastr implementiert, welches schönere Popups erlaubt.

Wie und wo können die vorgestellten Techniken, Methoden und Konzepte in einer Rails-App angewandt werden?

Flash/Toastr:

- **Anwendung:** Mithilfe solche Messages kann man den User auf eine augenfreundliche Art informieren.
- **Vorteile:** Sehr schön gestaltet und einfach zu verwenden.
- **Nachteile:** -

Bootstrap Modal/Popup Dialogs:

- **Anwendung:** Ein einfacher und schöner Weg den Benutzer zu informieren oder Eingaben zu lesen.
- **Vorteile:** Man kann häufig weniger falsch machen als bei einem normalen Formular. Häufig auch schöner.
- **Nachteile:** Pops sind als nervig bekannt und wenn man zu viele hat, leidet die Userexperience.

Gravatar:

- **Anwendung:** Man kann ein Bild für alle möglichen Accounts (unter derselben E-Mail) verwenden.
- **Vorteile:** Praktisch als Einzelperson da es einfach und schnell ist.
- **Nachteile:** Man wird abhängig von diesem Dienst und in einem Projekt wie diesem macht es nicht wirklich Sinn da jeder Benutzer ein solches Profil benötigte.

Routing :users:

- **Anwendung:** Man kann mit einer Zeile alle nötigen Routen für die angegebene Klasse hinzufügen.
- **Vorteile:** Eine einzelne Zeile für alle Routen -> Weniger Code.
- **Nachteile:** Teilweise fügt man überflüssige Routen hinzu.

Aufgaben:

Schreiben Sie in die Datei `app/views/layouts/application.html.erb` folgenden Code:

```
<script type="text/javascript">
  toastr.success("Wow, a success")
</script>
```

Was passiert da?

Ein Script wird eingefügt welches die JS-Methode success auf dem JS-Objekt toastr ausführt. Dies zeigt eine schöne Pop-up Meldung an.

Partial-View für Registration:

```
<% if resource.errors.any? %>
  <script type="text/javascript">
    <% resource.errors.full_messages.each do |msg| %>
      toastr.error('<%= msg %>');
    <% end %>
  </script>
<% end %>
```

Command für Migration hinzufügen:

`rails g migration AddFieldsToUser`

Inhalt:

```
class AddFieldsToUser < ActiveRecord::Migration[5.2]
  def change
    add_column :users, :website, :string
    add_column :users, :bio, :text
    add_column :users, :provider, :string
    add_column :users, :uid, :string
    add_column :users, :image, :string
  end
end
```

Helper hinzugefügt mit `resources :users`

`users_path`

`new_user_path`

`users_path`

`user_path(:id)`

`edit_user_path(:id)`

user_path(:id)

user_path(:id)

User Show Methode:

```
def show
  @user = User.find(params[:id])
end
```

```
1 module ApplicationHelper
2   def avatar_url user
3     gravatar_id = Digest::MD5::hexdigest(user.email).downcase
4     "https://www.gravatar.com/avatar/#{gravatar_id}.jpg"
5   end
6 end
```

Erklären Sie in groben Zügen, was das bedeutet:


Es wird ein MD5 Hash aus der Email gemacht. Dieser wird in die URL getan und so kann das Bild abgerufen werden.

Erklären Sie kurz folgende Codeausschnitte:

```
...
<%= image_tag avatar_url(@user), width: '152', height: '152',
class: "round-img" %>
...
<% if @user == current_user %>
  <%= link_to "Edit Profile", edit_user_registration_path, class:
"btn btn-outline-dark common-btn edit-profile-btn" %>
  <button type="button" class="core-sprite setting" data-
toggle="modal" data-target="#exampleModal"></button>
<% end %>
...
<%= link_to "Log out", destroy_user_session_path, method: :delete,
class: "list-group-item list-group-item-action" %>
...
```

Es wird ein Image Tag mit dem korrekten Bild erstellt. Dabei wird der neu erstellte Helper verwendet. Danach wird geschaut ob der Benutzer der angezeigt wird, auch derjenige ist dem das Profil gehört. Wenn ja kann er das Profil bearbeiten.

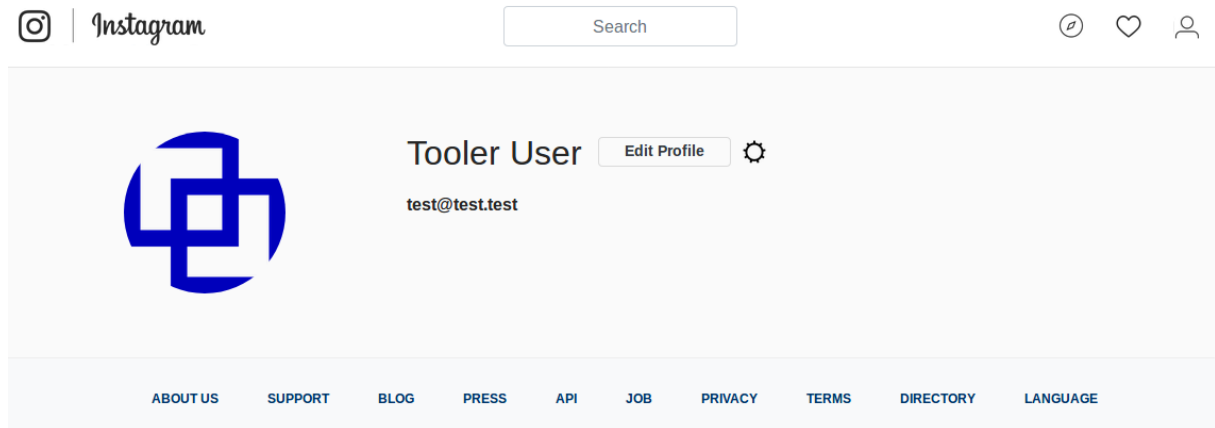
Versuchen Sie nun etwas zu verändern. Was stellen Sie fest?

Es kann nichts verändert werden, da das Passwort nicht angegeben wurde. 

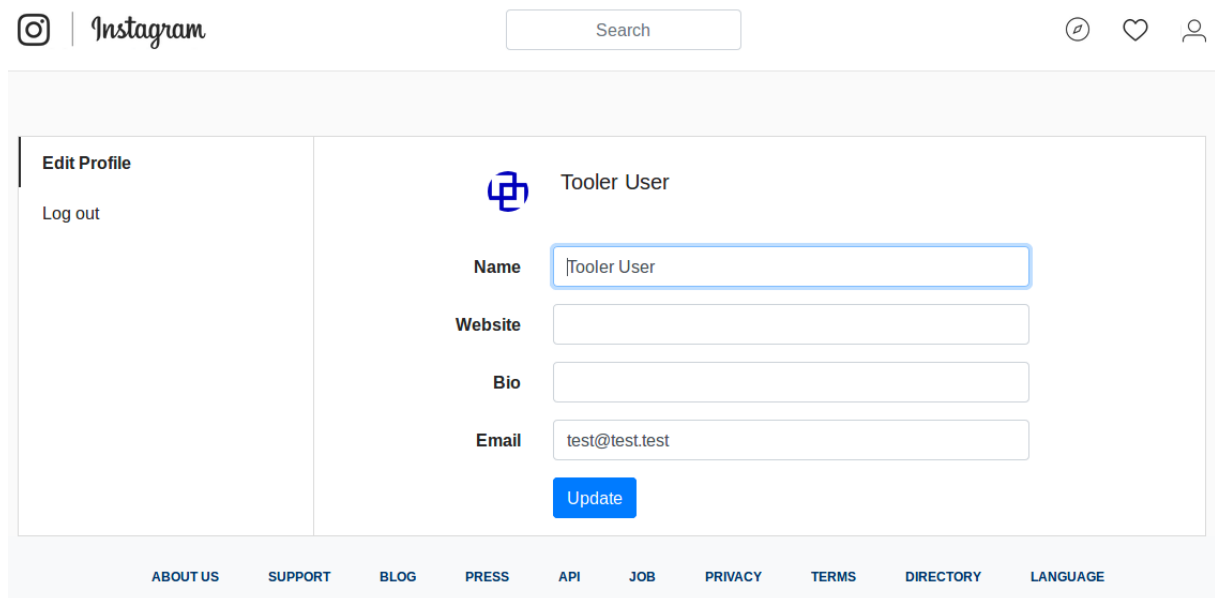
Das Problem kann zwar mit `update_without_password` behoben werden. Allerdings kann der Benutzer dann sein eigenes Passwort nicht mehr verändern.

Screenshots:

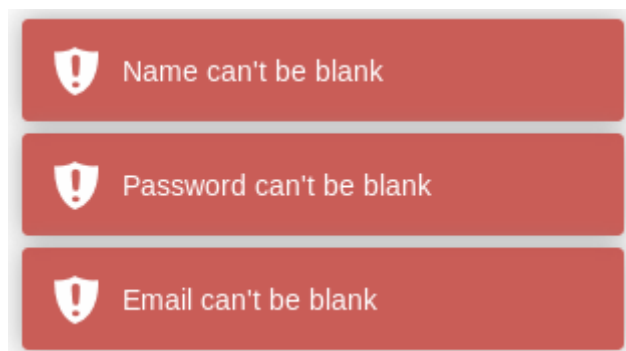
Show User:



Edit User:



Leeres Registrationsformular:



Selbstreflexion:

Was habe ich gelernt?

In diesem Arbeitsblatt war einiges neu für mich. Ich kannte das Prinzip von Flash-Messages/Toastr unter Rails noch nicht und verwendete diese vorher auch nicht. Das Prinzip von Modals kannte ich allerdings habe ich es noch nie unter Rails verwendet. Das Routing kannte ich bereits und Gravatar war neu für mich. Ich war jedoch nicht sehr davon begeistert.

Was hat mich behindert?

Im Arbeitsblatt hat es einige Dinge, die nicht aufgehen. Man verhindert, dass der Benutzer das Passwort ändern kann aber lässt die Möglichkeit offen. Das ist sehr verwirrend und meiner Meinung nach ein Fehler.

Gravatar hat mich nicht sonderlich überzeugt, da es für ein solches Projekt sicherlich nicht das richtige ist.

Was habe ich nicht verstanden?

Wieso man bei diesem Projekt Gravatar verwenden sollte.

Was kann ich beim Studium besser machen?

-

Schlussfolgerung

In diesem Arbeitsblatt lernte ich einiges über wichtige Themen wie Flash-Messages und Modal. Diese wendete ich in Rails an. Da ich eine Woche noch weg war, musste ich etwas schneller arbeiten als sonst, konnte aber trotzdem alles abschliessen.