```
$ bundle exec rake db:migrate:reset
$ bundle exec rake db:seed
```

Заполнение базы может протекать медленно и в некоторых системах занимает несколько минут. Кроме того, некоторые читатели сообщали, что не могут запустить команду `reset`, когда работает сервер Rails, поэтому вам, быть может, придется сначала остановить его.

После запуска задачи `db:seed` в приложении появилось 100 образцов пользователей. Как видно на рис. 9.10, я взял на себя смелость связать первые несколько образцов электронных адресов с граватарами, поэтому не все изображения совпадают с картинками по умолчанию. (Вам может понадобиться перезапустить вебсервер в этой точке.)

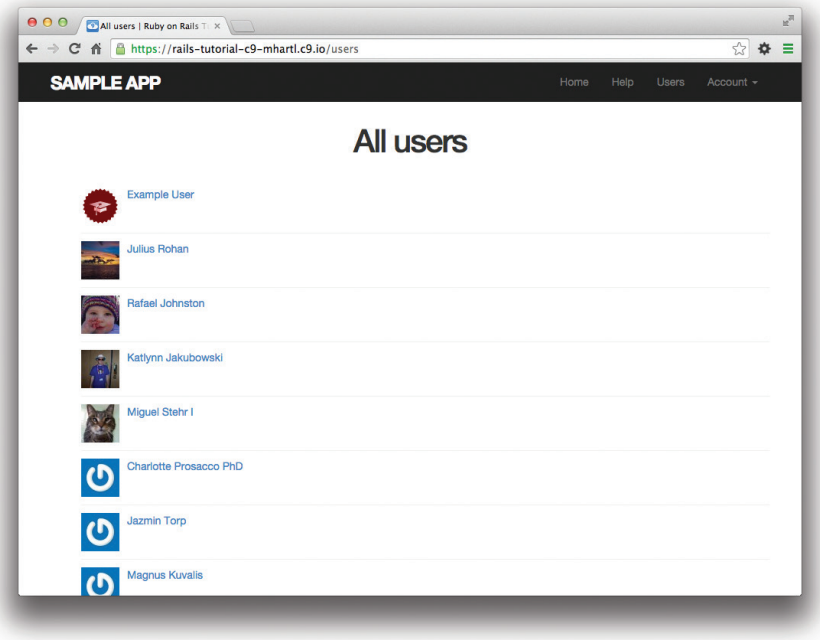

Рис. 9.10 ❖ Страница с сотней пользователей

9.3.3. Постраничный просмотр

Наш исходный пользователь более не страдает от одиночества, но теперь появилась другая проблема: у него *слишком большая компания*, и вся она расположилась на одной странице. Сейчас это сотня, что уже довольно много, а на реальном сайте это могут быть и тысячи. Решение заключается в *разбиении списка пользователей на страницы*, чтобы показывать единовременно только (например) 30 записей на странице.

В Rails поддерживается несколько способов разбиения на страницы, и мы будем использовать один из самых простых и надежных, `will_paginate` (https://github.

com/mislav/will_paginate/wiki). Чтобы воспользоваться им, нам потребуются два гема: `will_paginate` и `bootstrap-will_paginate`, который настраивает `will_paginate` на использование стилей из Bootstrap. Дополненный `Gemfile` показан в листинге 9.40.

Листинг 9.40 ❖ Добавление will_paginate в Gemfile

```
source 'https://rubygems.org'

gem 'rails',                   '4.2.0'
gem 'bcrypt',                  '3.1.7'
gem 'faker',                   '1.4.2'
gem 'will_paginate',           '3.0.7'
gem 'bootstrap-will_paginate', '0.0.10'
.
.
.
```

Выполним `bundle install`:

```
$ bundle install
```

Также следует перезапустить веб-сервер, чтобы корректно загрузить новые гемы.

Чтобы разбиение на страницы заработало, нужно добавить немного кода в представление `index`, а также заменить вызов `User.all` в методе `index` на объект, реализующий разбиение на страницы. Для начала добавим в представление специальный метод `will_paginate` (листинг 9.41); мы вскоре увидим, почему код был добавлен выше и ниже списка пользователей.

Листинг 9.41 ❖ Список пользователей с разбиением на страницы (app/views/users/index.html.erb)

```
<% provide(:title, 'All users') %>
<h1>All users</h1>

<%= will_paginate %>

<ul class="users">
  <% @users.each do |user| %>
    <li>
      <%= gravatar_for user, size: 50 %>
      <%= link_to user.name, user %>
    </li>
  <% end %>
</ul>

<%= will_paginate %>
```

Метод `will_paginate` немного необычный; внутри представления `users` он автоматически находит объект `@users`, а затем отображает ссылки для доступа к страницам. И все же представление пока не работает, так как в данный момент `@users` содержит результаты вызова `User.all` (листинг 9.33), в то время как `will_paginate` требует постраничного вывода результатов с явным применением метода `paginate`:

```
$ rails console
>> User.paginate(page: 1)
  User Load (1.5ms) SELECT "users".* FROM "users" LIMIT 30 OFFSET 0
    (1.7ms) SELECT COUNT(*) FROM "users"
=> #<ActiveRecord::Relation [#<User id: 1,...
```

Обратите внимание, что `paginate` принимает в качестве аргумента хэш с ключом `:page` и номером запрашиваемой страницы. `User.paginate` извлекает пользователей из базы данных блоками (30 по умолчанию), основываясь на параметре `:page`. Так, например, страница 1 содержит пользователей с 1 по 30, страница 2 – с 31 по 60 и т. д. Если значение ключа `:page` равно `nil`, `paginate` просто вернет первую страницу.

Используя `paginate` вместо `all` в методе `index` (листинг 9.42), мы можем организовать постраничный вывод пользователей в учебном приложении. Параметр `page` берется из `params[:page]`, который автоматически генерируется `will_paginate`.

Листинг 9.42 ❖ Постраничный вывод списка пользователей в методе index (app/controllers/users_controller.rb)

```
class UsersController < ApplicationController
  before_action :logged_in_user, only: [:index, :edit, :update]
.
.
.
  def index
    @users = User.paginate(page: params[:page])
  end
.
.
.
end
```

Страница со списком пользователей теперь должна выглядеть, как показано на рис. 9.11. (На некоторых системах может потребоваться перезапустить сервер Rails.) Так как мы включили `will_paginate` выше и ниже списка пользователей, ссылки на страницы появились в обоих местах.

Если сейчас щелкнуть на ссылке **2** или **Next** (Следующая), будет выполнен переход на вторую страницу списка, как показано на рис. 9.12.

9.3.4. Тестирование страницы со списком пользователей

Теперь, когда страница со списком пользователей заработала, напишем упрощенные тесты для нее, в том числе тесты для проверки разбиения на страницы. Суть в том, чтобы войти на сайт, открыть страницу со списком, проверить наличие первой страницы и ссылок на другие страницы. Для этого нам необходимо иметь в тестовой базе данных достаточное количество пользователей, то есть их должно быть больше 30.