12.4 Email in production (take two)

Now that we've got password resets working in development, in this section we'll get them working in production as well. The steps are exactly the same as for account activations, so if you already followed Section 11.4 you can skip right to Listing 12.24.

To send email in production, we'll use SendGrid, which is available as an add-on at Heroku for verified accounts. (This requires adding credit card information to your Heroku account, but there is no charge when verifying an account.) For our purposes, the "starter" tier (which as of this writing is limited to 400 emails a day but costs nothing) is the best fit. We can add it to our app as follows:

```
$ heroku addons:create sendgrid:starter
```

(This might fail on systems with older version of Heroku's command-line interface. In this case, either upgrade to the latest Heroku toolbelt or try the older syntax **`heroku addons:add sendgrid:starter`**.)

To configure our application to use SendGrid, we need to fill out the SMTP settings for our production environment. As shown in Listing 12.23, you will also have to define a **`host`** variable with the address of your production website.

Listing 12.23: Configuring Rails to use SendGrid in production.
config/environments/production.rb

```
Rails.application.configure do
  .
  .
  .
  config.action_mailer.raise_delivery_errors = true
  config.action_mailer.delivery_method = :smtp
  host = '<your heroku app>.herokuapp.com'
  config.action_mailer.default_url_options = { host: host }
  ActionMailer::Base.smtp_settings = {
    :address        => 'smtp.sendgrid.net',
    :port           => '587',
    :authentication => :plain,
    :user_name      => ENV['SENDGRID_USERNAME'],
```

```
    :password       => ENV['SENDGRID_PASSWORD'],
    :domain         => 'heroku.com',
    :enable_starttls_auto => true
  }
  .
  .
  .
end
```

The email configuration in Listing 11.41 includes the **user_name** and **password** of the SendGrid account, but note that they are accessed via the **ENV** environment variable instead of being hard-coded. This is a best practice for production applications, which for security reasons should never expose sensitive information such as raw passwords in source code. In the present case, these variables are configured automatically via the SendGrid add-on, but we'll see an example in Section 13.4.4 where we'll have to define them ourselves.

At this point, you should merge the topic branch into master (Listing 12.24).

Listing 12.24: Merging the **password-reset** branch into **master**.

```
$ rails test
$ git add -A
$ git commit -m "Add password reset"
$ git checkout master
$ git merge password-reset
```

Then push up to the remote repository and deploy to Heroku:

```
$ rails test
$ git push && git push heroku
$ heroku run rails db:migrate
```

Once the Heroku deploy has finished, you can reset your password by clicking the "(forgot password)" link (Figure 12.4). The result should be a reset email as shown in Figure 12.14. Following the link and making invalid or valid submissions should work as it did in development (Figure 12.12 and Figure 12.13). Likewise, upon successfully changing the password, the user should be redirected to the profile page (Figure 12.15).

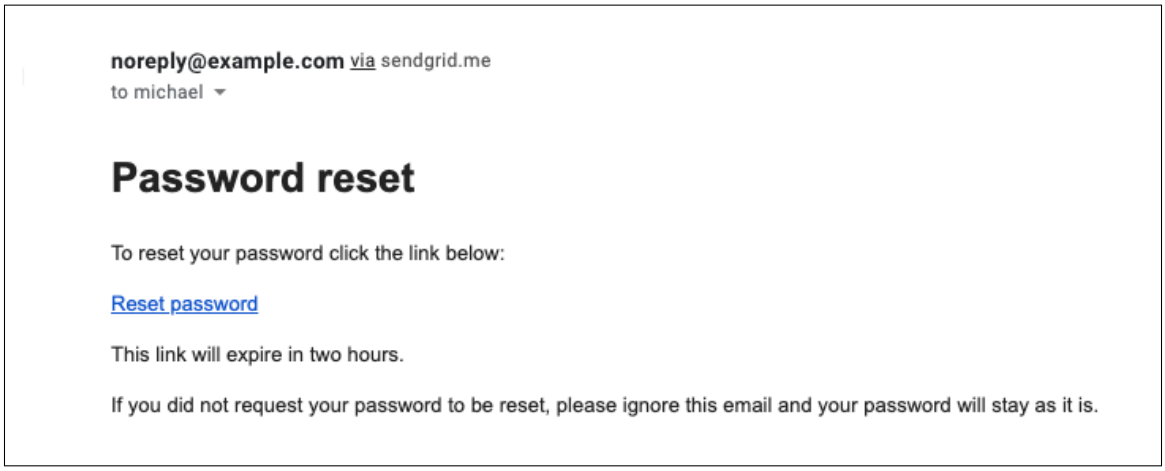

Figure 12.14: A password reset email sent in production.

Exercises

Solutions to the exercises are available to all Rails Tutorial purchasers here.

To see other people's answers and to record your own, subscribe to the Rails Tutorial course or to the Learn Enough All Access Bundle.

1. Sign up for a new account in production. Did you get the email?
2. Click on the link in the activation email to confirm that it works. What is the corresponding entry in the server log? *Hint*: Run `heroku logs` at the command line.
3. Are you able to successfully update your password?

12.5 Conclusion

With the added password resets, our sample application's sign up, log in, and log out machinery is complete and professional-grade. The rest of the *Ruby on Rails Tutorial* builds on this foundation to make a site with Twitter-like microposts (Chapter 13) and a status feed of posts from followed users (Chapter 14).