Chapter 11

In this chapter, we'll implement an account activation step to verify that the user controls the email address they used to sign up. This will involve:

- associating an activation token and digest with a user
- sending the user an email with a link including the token
- activating the user upon clicking the link

11.1

Because we'll be treating account activations as a resource, we'll interact with them via a standard REST URL. The activation link will be modifying the user's activation status, and for such modifications the standard REST practice is to issue a PATCH request to the update action

We'll remember tokens by pairing a publicly exposed virtual attribute with a secure hash digest saved to the database. This way we can access the activation token using

```
user.activation_token
```

and authenticate the user with code like

```
user.authenticated?(:activation, token)
```

before_create callback

- before_create :create_activation_digest
    - *method reference*, arranges for Rails to look for a method called create_activation_digest and run it before creating the user
- purpose of the before_create callback is to assign the token and corresponding digest

11.2

With the data modeling complete, we're now ready to add the code needed to send an account activation email.
The method is to add a User *mailer* using the Action Mailer library, which we'll use in the Users controller create action to send an email with an activation link.

Action Mailer supports both plain-text and HTML mail.

To make a working activation email:

- first customize the generated template
- create an instance variable containing the user (for use in the view)
- mail the result to user.email

Account activations use a generated token to create a unique URL for activating users.

Account activations use a hashed activation digest to securely identify valid activation requests.

Both mailer tests and integration tests are useful for verifying the behavior of the User mailer.

11.3

We need to write the edit action in the Account Activations controller that actually activates the user

Metaprogramming: a program that writes a program. Metaprogramming is one of Ruby's strongest suits, and many of the "magic" features of Rails are due to its use of Ruby metaprogramming.

remember_digest is an attribute on the User model, and inside the model we can rewrite it as follows:

```
Self.remember_digest
```

Somehow, we want to be able to make this *variable*, so we can call

```
self.activation_token
```

We do this by metaprogramming with the powerful send method, which lets us call a method with a name of our choice by "sending a message" to a given object

11.4

We'll configure our application so that it can actually send email in production.

To send email in production, we'll use SendGrid, which is available as an add-on at Heroku for verified accounts.

- `$ heroku addons:create sendgrid:starter`