Controller Filters

Filters are methods that run *before*, *after* or *around* a controller action. To authenticate a user, w
methods before our controller actions.

Let's look at a quick example of using filters before a controller action.

```
class HellosController < ApplicationController
  before_action :say_hello

  def welcome
  end

  private

  def say_hello
    puts "Hello!"
  end
end
```

Before every action in the Hello Controller, the `say_hello` method will run, which will output "h
how many actions we add to this controller, say_hello will run before all of them. We have set th
because it will only act as a filter within this controller.

What if we only want our filter to run before some actions and not others?

```
class HellosController < ApplicationController
  before_action :say_hello, only: [:welcome]

  def welcome
  end

  def goodbye
  end

  def denied
  end

  private

  def say_hello
      puts "Hello!"
  end
end
```

By passing in `only: [:welcome]` to our before_action filter, we made it so that the `say_hello`
welcome action of our HellosController. We could have instead passed in `except: [:goodbye, :`
effect. In that case, the say_hello method would run on all of the actions except goodbye and d

What if I have multiple controllers and I want `say_hello` to run before every actio

We could add the same `say_hello` method to each one of our controllers, but our code wouldn
assignments that all of the controllers inherit from the ApplicationController. So we can add our

```
class ApplicationController < ActionController::Base
  before_action :say_hello

  private

  def say_hello
      puts "Hello Everybody!"
  end
end
```

Whenever any controller action is invoked, the say_hello method will run. The same way that ou
attributes from the ApplicationController, it will inherit filters as well.

Ok, but what if I want the say_hello method to run on every action, except for twc controllers?

Let's go back to the HellosController and see if we can make that happen.

```
class HellosController < ApplicationController
  skip_before_action :say_hello, except: [:welcome]

  def welcome
  end

  def goodbye
  end

  def denied
  end
end
```

`skip_before_action` allows me to remove a before filter that has been inherited from another c skipped on every action within this controller except for on the welcome action. This way, we er possible. An excellent example of this and the subject of the next tab, ensuring that a User is lo throughout our website.