Service Modules

A minor refactoring

RAILS

Service Classes

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

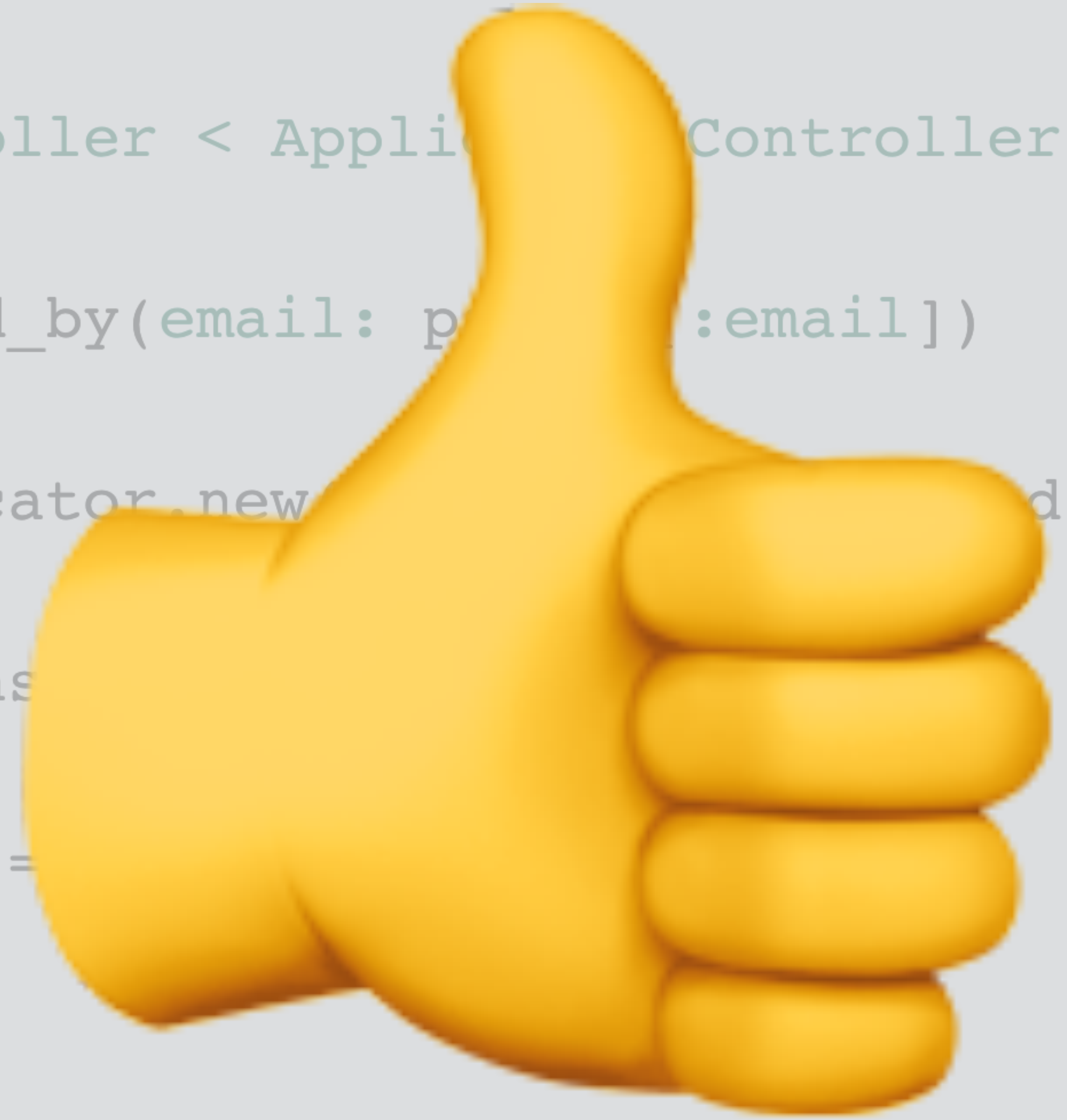
```
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if user &&
      BCrypt::Password.new(user.password_digest) == params[:password]
      # ...
    else
      # ...
    end
  end
end
```

```ruby
class UserAuthenticator
  def initialize(user)
    @user = user
  end

  def authenticated?(unencrypted_password)
    return false unless @user

    BCrypt::Password.new(@user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.new(user).authenticated?(params[:password])
      # ...
    else
      # ...
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.new(user).authenticated?(params[:password])
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```
class SessionsController < Appli        Controller
  def create
    user = User.find_by(email: p       :email])

    if UserAuthenticator.new                   d?(params[:password])
      sign_in user
      redirect_to das
    else
      flash[:alert] =
      render "new"
    end
  end
end
```



BUT

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.new(user).authenticated?(params[:password])
      # ...
    else
      # ...
    end
  end
end
```

```
authenticator = UserAuthenticator.new(user)
authenticator.authenticated?
```

```ruby
authenticator = UserAuthenticator.new(user)
authenticator.authenticated?
authenticator.two_factor_enabled?
```

```
authenticator = UserAuthenticator.new(user)
authenticator.authenticated?
authenticator.two_factor_enabled?
authenticator.authenticate_with_facebook
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.new(user).authenticated?(params[:password])
      # ...
    else
      # ...
    end
  end
end
```

```
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.authenticated?(user, params[:password])
      # ...
    else
      # ...
    end
  end
end
```

```ruby
class UserAuthenticator
  def initialize(user)
    @user = user
  end

  def authenticated?(unencrypted_password)
    return false unless @user

    BCrypt::Password.new(@user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class UserAuthenticator
  def authenticated?(unencrypted_password)
    return false unless @user

    BCrypt::Password.new(@user.password_digest) ==
      unencrypted_password
  end
end
```

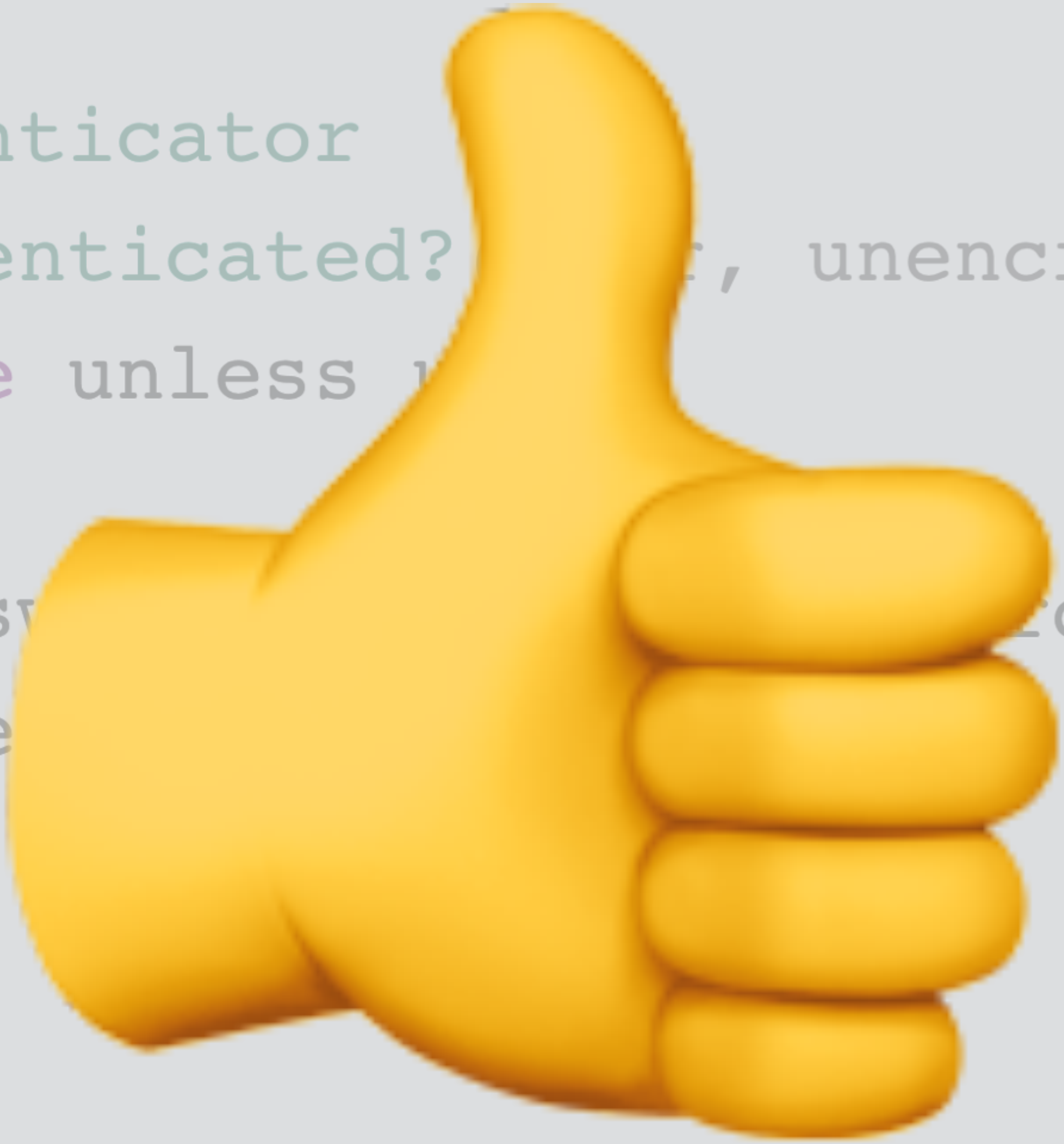
```
class UserAuthenticator
  def authenticated?(user, unencrypted_password)
    return false unless @user

    BCrypt::Password.new(@user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class UserAuthenticator
  def authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```
class UserAuthenticator
  def self.authenticated?          , unencrypted_password)
    return false unless u

    BCrypt::Passw                   rd_digest) ==
      unencrypte
  end
end
```



```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.authenticated?(user, params[:password])
      # ...
    else
      # ...
    end
  end
end
```

BUT

```ruby
class UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```
class UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
module UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
module UserAuthenticator
  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```
module UserAuthenticator
  module_function

  def self.authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
module UserAuthenticator
  module_function

  def authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
module UserAuthenticator
  module_function

  def authenticated?(user, unencrypted_password)
    return false unless user

    BCrypt::Password.new(user.password_digest) ==
      unencrypted_password
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.authenticated?(user, params[:password])
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

```ruby
class SessionsController < ApplicationController
  def create
    user = User.find_by(email: params[:email])

    if UserAuthenticator.authenticated?(user, params[:password])
      sign_in user
      redirect_to dashboard_path
    else
      flash[:alert] = "Login failed."
      render "new"
    end
  end
end
```

Do you

REALLY

need that OBJECT?

if not, go

module

Thanks!

@paulfioravanti