

# UNIVERSIDAD NACIONAL AUTONOMA DE NICARAGUA



**UNAN - León**

## **FACULTAD DE CIENCIAS Y TECNOLOGIA**

Carrera: Ingeniería en Telemática

Componente: Software como un servicio

Grupo:1

Docente: Erving Montes

Elaborado por:

- Alli Gissiell Herrera Chow.

Fecha: 02 de octubre de 2024

1. Realizar y analizar cada uno de los enunciados de la guía.
  1. Generar un nuevo proyecto.

---

```
gia@debian:~$ ls
Descargas  Escritorio  mi_prooo    Música  Plantillas  Proyectos_I
Documentos Imágenes    mi_proyecto mysql   prac3       Público
gia@debian:~$ cd Proyectos_RoR/
gia@debian:~/Proyectos_RoR$ ls
app  first_app  my_app  prac6
gia@debian:~/Proyectos_RoR$ su
Contraseña:
root@debian:/home/gia/Proyectos_RoR# rails new App_RoR -T
   create
   create  README.md
   create  Rakefile
   create  .ruby-version
   create  config.ru
   create  .gitignore
   create  .gitattributes
   create  Gemfile
   run  git init -b main from "."
Inicializado repositorio Git vacío en /home/gia/Proyectos_RoR/App_RoR
   create  app
   create  app/assets/config/manifest.js
   create  app/assets/stylesheets/application.css
   create  app/channels/application_cable/channel.rb
```

- 1.1. Agregar las gemas que se utilizarán en el proyecto, dentro grupo development, test.
  - 1.2. El grupo development, test quedará como se muestra en la siguiente figura.

```

# gem 'image_processing', '~> 1.2

group :development, :test do
  # See https://guides.rubyonrails.org/debugging_rails.html
  gem "debug", platforms: %i[ mri mswin mswin64 mingw ]
  gem "rspec-rails"
  gem 'capybara'
  gem 'database_cleaner'
  # Static analysis for security vulnerabilities [https://brakeman.org/]
  gem "brakeman", require: false

  # Omakase Ruby styling [https://github.com/rails/rubocop-omakase]
  gem "rubocop-rails-omakase", require: false
end

group :development do
  # Use console on exceptions pages [https://github.com/rails/rails]

```

1.3. Ejecutar el siguiente comando en el terminal para instalar las gemas.

Contraseña:

```

root@debian:/home/gia/Proyectos_RoR/App_RoR#
root@debian:/home/gia/Proyectos_RoR/App_RoR# bundle install
Don't run Bundler as root. Bundler can ask for sudo if it is needed, and
installing your bundle as root will break this application for all non-root
users on this machine.
Fetching gem metadata from https://rubygems.org/.....
Resolving dependencies...
Using rake 13.2.1
Using base64 0.2.0
Using bigdecimal 3.1.8
Using concurrent-ruby 1.3.4
Using connection_pool 2.4.1
Using drb 2.2.1
Using i18n 1.14.6
Using logger 1.6.1
Using minitest 5.25.1
Using securerandom 0.3.1
Using tzinfo 2.0.6

```

1.4. Ejecutar el siguiente comando para crear los directorios que rspec necesita para trabajar.

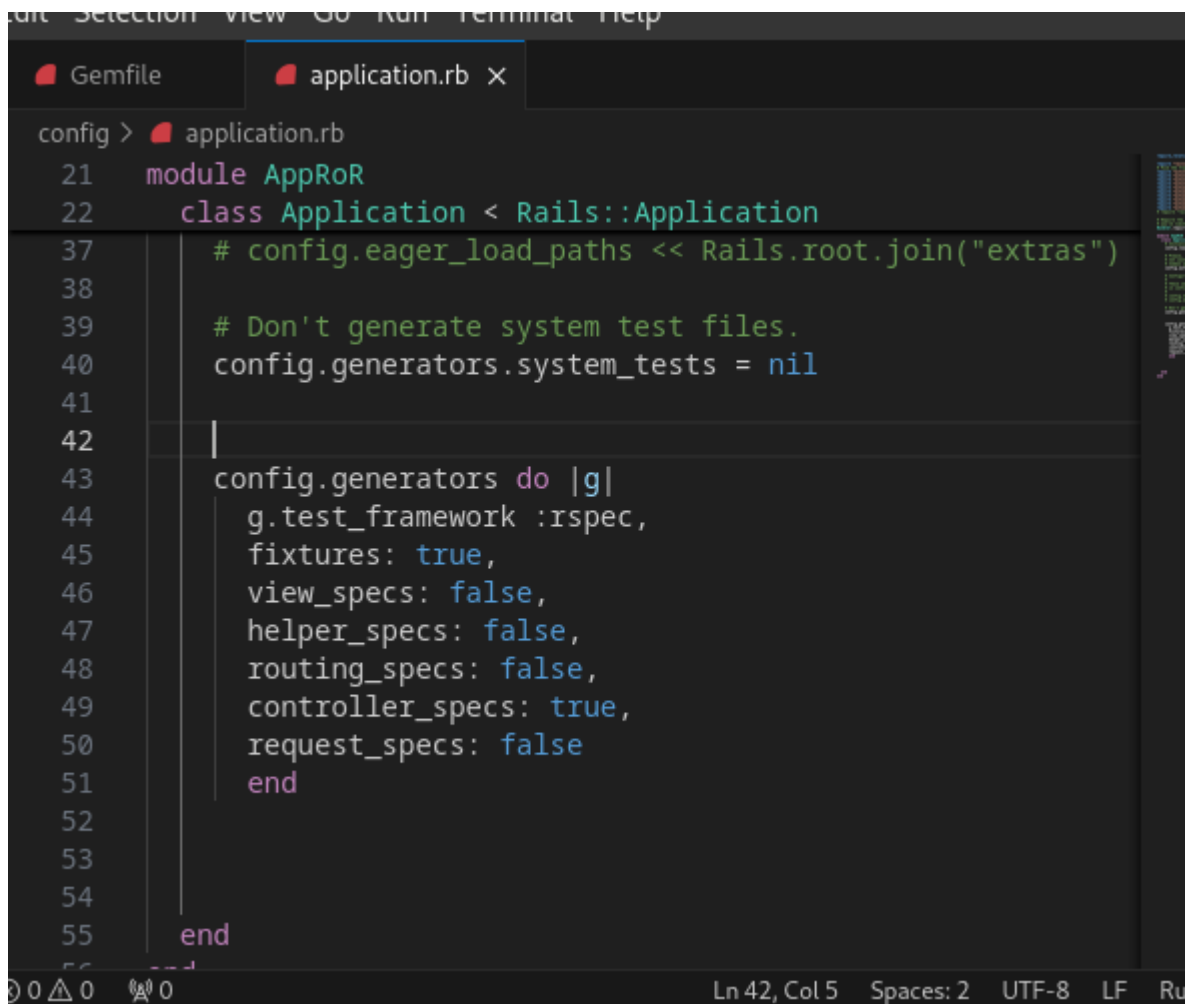
```

use bundle into [gemname] to see where a bundled gem is installed.
root@debian:/home/gia/Proyectos_RoR/App_RoR#
root@debian:/home/gia/Proyectos_RoR/App_RoR# rails g rspec:install
  create  .rspec
  create  spec
  create  spec/spec_helper.rb
  create  spec/rails_helper.rb
root@debian:/home/gia/Proyectos_RoR/App_RoR#

```

## 2. Configuración de Rspec.

2.1. Abrir el archivo config/application.rb y agregar el siguiente código dentro de la clase, en el mismo archivo.

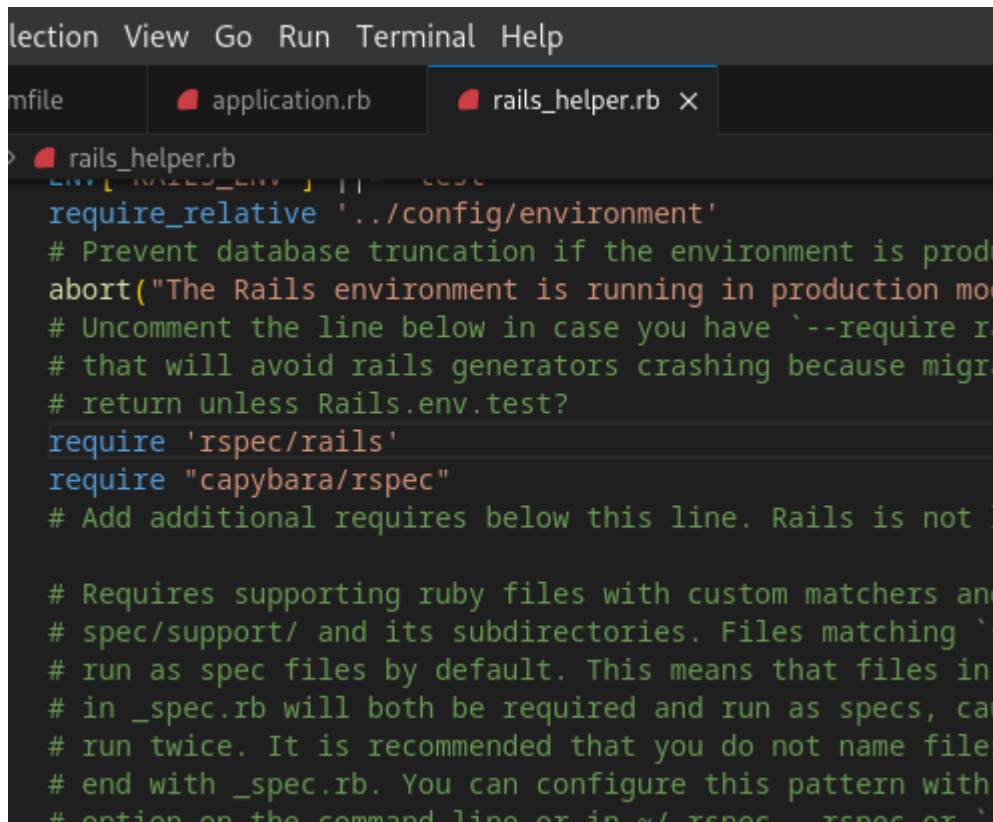


```

config > application.rb
21  module AppRoR
22    class Application < Rails::Application
37      # config.eager_load_paths << Rails.root.join("extras")
38
39      # Don't generate system test files.
40      config.generators.system_tests = nil
41
42      |
43      config.generators do |g|
44        g.test_framework :rspec,
45          fixtures: true,
46          view_specs: false,
47          helper_specs: false,
48          routing_specs: false,
49          controller_specs: true,
50          request_specs: false
51      end
52
53
54
55  end

```

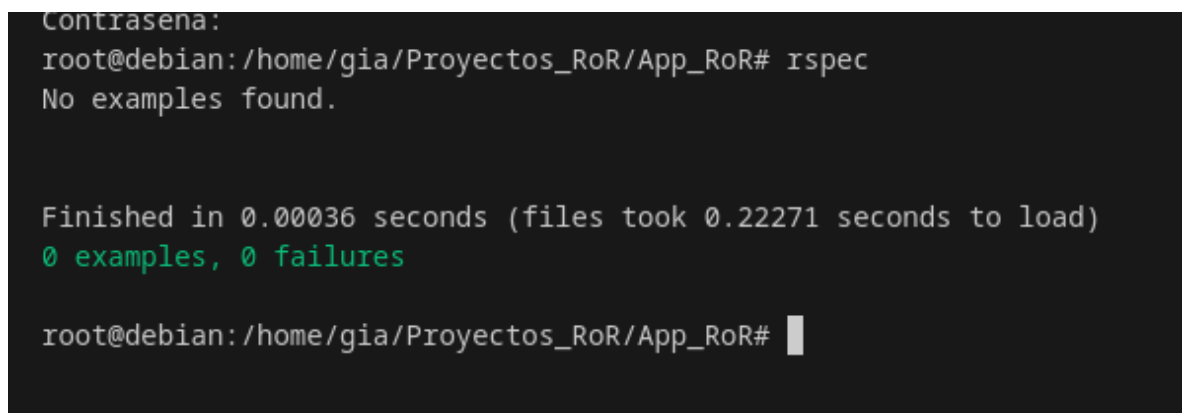
2.2. Abrir el archivo rails\_helper.rb y agregar debajo de la línea require 'rspec/rails'.



```
lection View Go Run Terminal Help
application.rb rails_helper.rb x
> rails_helper.rb
ENV['RAILS_ENV'] ||= 'test'
require_relative '../config/environment'
# Prevent database truncation if the environment is production
abort("The Rails environment is running in production mode!")
# Uncomment the line below in case you have `--require rails_helper`
# that will avoid rails generators crashing because migration
# return unless Rails.env.test?
require 'rspec/rails'
require "capybara/rspec"
# Add additional requires below this line. Rails is not loaded until this point.

# Requires supporting ruby files with custom matchers and
# spec/support/ and its subdirectories. Files matching `spec/**/*_spec.*`
# run as spec files by default. This means that files in
# in _spec.rb will both be required and run as specs, causing
# run twice. It is recommended that you do not name files
# end with _spec.rb. You can configure this pattern with
# option on the command line or in ~/.rspec or `rspec --`
```

2.3. Ejecutar el comando y verificar que no muestra ningún mensaje en rojo.



```
Contraseña:
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec
No examples found.

Finished in 0.00036 seconds (files took 0.22271 seconds to load)
0 examples, 0 failures

root@debian:/home/gia/Proyectos_RoR/App_RoR#
```

### 3. Static pages

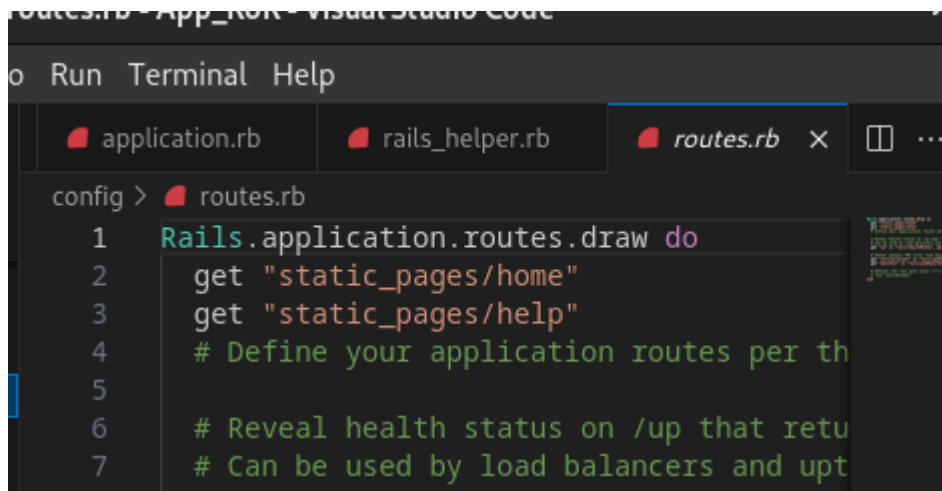
3.1. Para crear páginas estáticas escribir lo siguiente en el terminal.

```

rspec-support (3.13.1)
root@debian:/home/gia/Proyectos_RoR/App_RoR# rails generate controller Static
es home help
  create  app/controllers/static_pages_controller.rb
  route   get "static_pages/home"
         get "static_pages/help"
  invoke  erb
  create  app/views/static_pages
  create  app/views/static_pages/home.html.erb
  create  app/views/static_pages/help.html.erb
  invoke  rspec
  create  spec/controllers/static_pages_controller_spec.rb
  invoke  helper
  create  app/helpers/static_pages_helper.rb
  invoke  rspec
root@debian:/home/gia/Proyectos_RoR/App_RoR#

```

3.2. Verificar la configuración del archivo routes.rb.



```

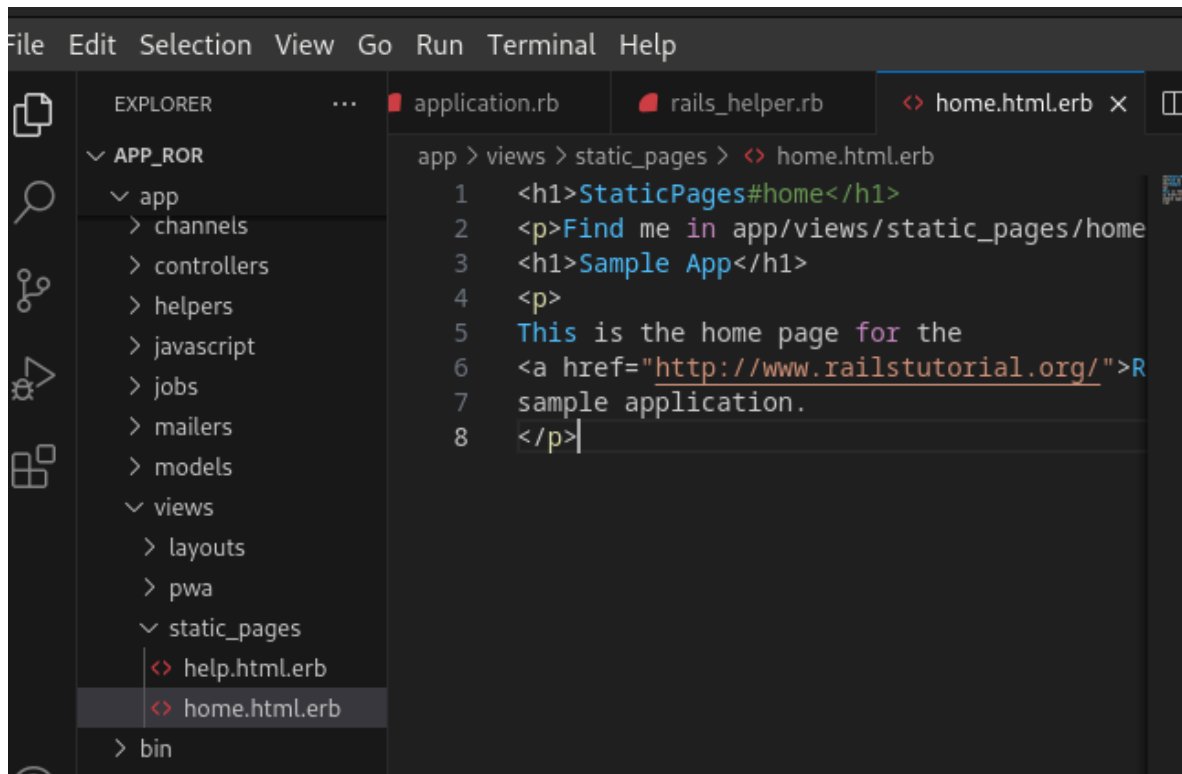
routes.rb - App_RoR - Visual Studio Code
o Run Terminal Help
  application.rb  rails_helper.rb  routes.rb x
config > routes.rb
1 Rails.application.routes.draw do
2   get "static_pages/home"
3   get "static_pages/help"
4   # Define your application routes per th
5
6   # Reveal health status on /up that retu
7   # Can be used by load balancers and upt

```

3.3. Verificar que se han creado las vistas en app/views/static\_pages/.

#### 4. Modificar las vistas.

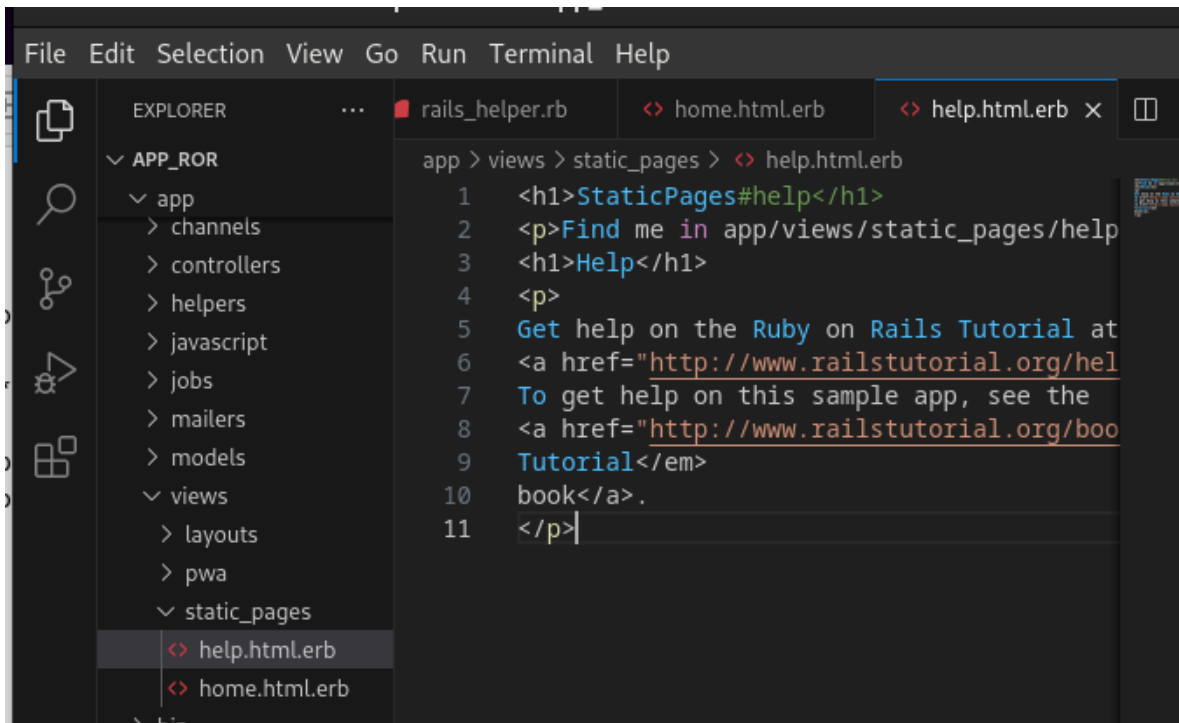
##### 4.1. Agregar al archivo home.html.erb.



The screenshot shows a code editor with a dark theme. On the left, the 'EXPLORER' sidebar is open, showing a file tree for a project named 'APP\_ROR'. The tree is expanded to show the 'views' directory, which contains 'layouts', 'pwa', and 'static\_pages'. The 'static\_pages' directory is further expanded, showing 'help.html.erb' and 'home.html.erb'. The 'home.html.erb' file is selected and highlighted. The main editor area shows the content of 'home.html.erb' with the following code:

```
app > views > static_pages > <> home.html.erb
1  <h1>StaticPages#home</h1>
2  <p>Find me in app/views/static_pages/home
3  <h1>Sample App</h1>
4  <p>
5  This is the home page for the
6  <a href="http://www.railstutorial.org/">R
7  sample application.
8  </p>
```

##### 4.2. Modificar el archivo help.html.erb.



5. Hacer testing con rspec.

5.1. El primer paso será ejecutar el comando db:migrate en el terminal para crear el esquema de pruebas con el que se ejecutarán los tests.

5.2. Probar hacer testing escribiendo en el terminal el comando.

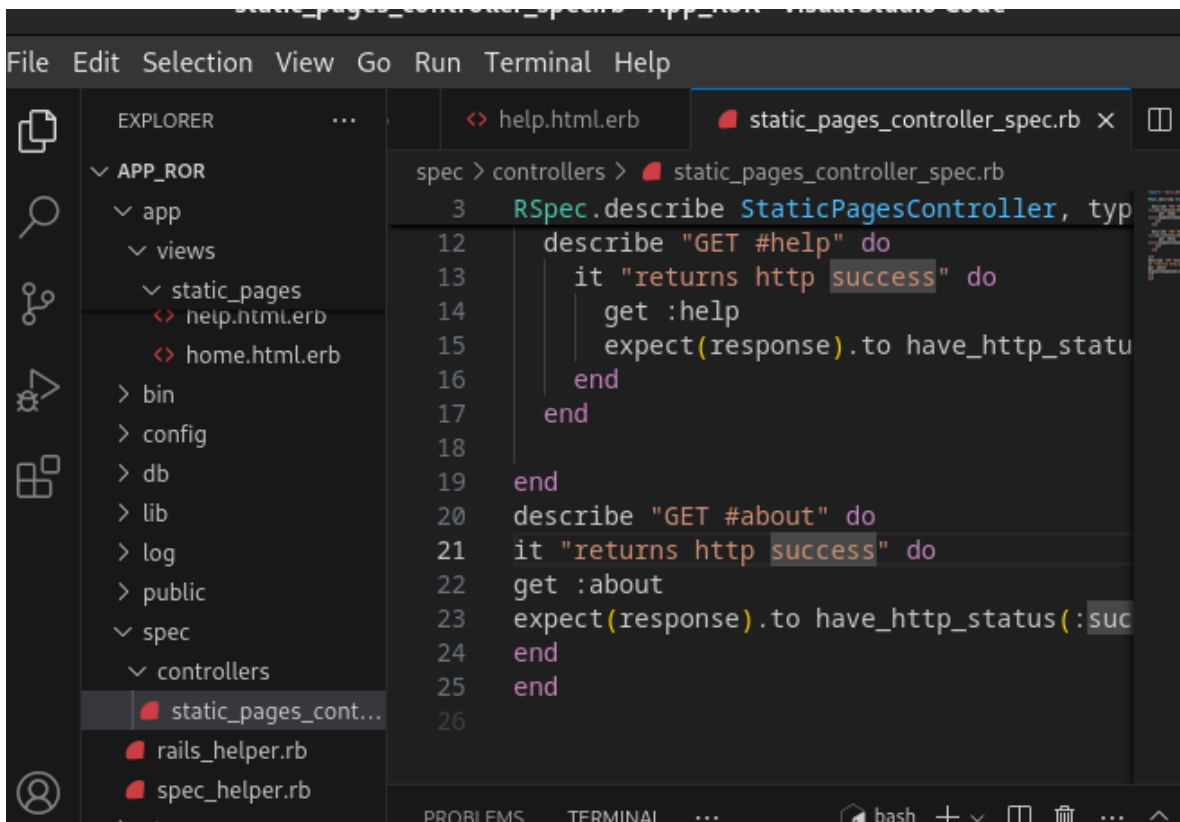
```
root@debian:/home/gia/Proyectos_RoR/App_RoR# rails db:migrate
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_pages
_controller_spec.rb
..

Finished in 0.24743 seconds (files took 4.63 seconds to load)
2 examples, 0 failures

root@debian:/home/gia/Proyectos_RoR/App_RoR#
```



5.3. Agregar siguiente código dentro del archivo `static_pages_controller_spec.rb` del directorio `spec/controllers/`.



The screenshot shows a code editor with a dark theme. On the left, the 'EXPLORER' sidebar shows a project structure for 'APP\_ROR'. The 'spec' directory is expanded, showing 'controllers'. The file 'static\_pages\_controller\_spec.rb' is selected. The main editor area shows the content of this file, which is a RSpec spec for the 'StaticPagesController'. The code includes two test blocks: one for the 'GET #help' action and another for the 'GET #about' action. The 'GET #help' block is currently selected, and the text 'it "returns http success" do' is highlighted. The bottom status bar shows 'bash' and some icons.

```
File Edit Selection View Go Run Terminal Help

EXPLORER
  APP_ROR
    app
    views
      static_pages
        help.html.erb
        home.html.erb
    bin
    config
    db
    lib
    log
    public
    spec
      controllers
        static_pages_controller_spec.rb
      rails_helper.rb
      spec_helper.rb

spec > controllers > static_pages_controller_spec.rb
3  RSpec.describe StaticPagesController, type: :controller do
12  describe "GET #help" do
13    it "returns http success" do
14      get :help
15      expect(response).to have_http_status(:success)
16    end
17  end
18
19 end
20 describe "GET #about" do
21  it "returns http success" do
22    get :about
23    expect(response).to have_http_status(:success)
24  end
25 end
26
```

5.4. Ejecutar el comando `rspec` y ver que muestra el test en rojo.

```
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_pages_controller_spec.rb
```

```
..F
```

```
Failures:
```

```
1) GET #about returns http success
```

```
Failure/Error: get :about
```

```
RuntimeError:
```

```
@controller is nil: make sure you set it in your test's setup method.
```

```
# ./spec/controllers/static_pages_controller_spec.rb:22:in `block (2 levels) in <top (required)>'
```

```
Finished in 0.03594 seconds (files took 1.02 seconds to load)
```

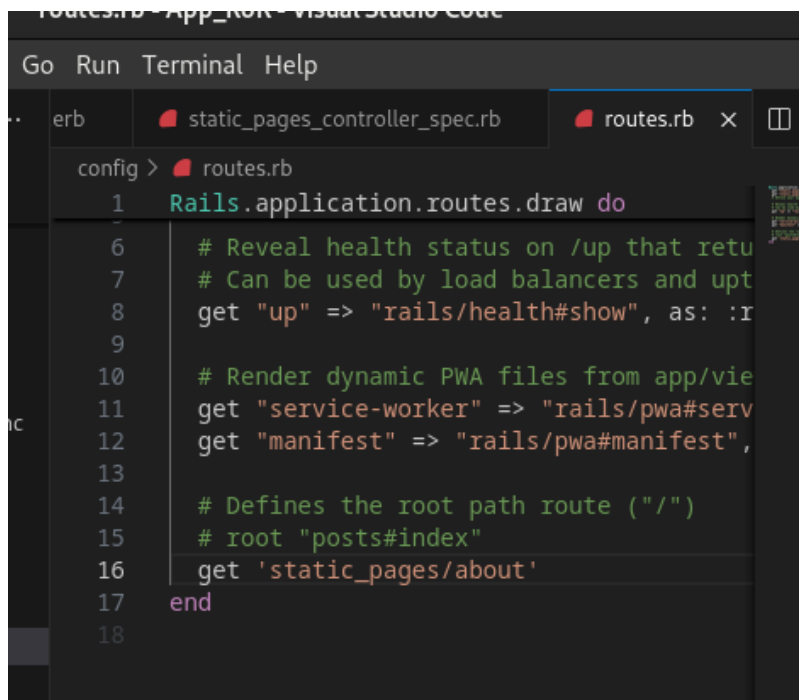
```
3 examples, 1 failure
```

```
Failed examples:
```

```
rspec ./spec/controllers/static_pages_controller_spec.rb:21 # GET #about returns http success
```

```
root@debian:/home/gia/Proyectos_RoR/App_RoR#
```

5.5. Editar el archivo routes.rb y agregar lo siguiente.



```
routes.rb - App_RoR - Visual Studio Code
Go Run Terminal Help
erb static_pages_controller_spec.rb routes.rb x
config > routes.rb
1 Rails.application.routes.draw do
2
3
4
5
6 # Reveal health status on /up that returns 200 OK
7 # Can be used by load balancers and upstream proxies
8 get "up" => "rails/health#show", as: :rails_health_show
9
10 # Render dynamic PWA files from app/views/pwa
11 get "service-worker" => "rails/pwa#service_worker"
12 get "manifest" => "rails/pwa#manifest", defaults: { format: :json }
13
14 # Defines the root path route ("/")
15 # root "posts#index"
16 get 'static_pages/about'
17 end
18
```

5.6. Ejecutar el test de nuevo. El mensaje que muestra, es debido a que no existe la acción about que intenta alcanzar el método get.

```
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_
_controller_spec.rb
..F
```

Failures:

1) GET #about returns http success

Failure/Error: get :about

RuntimeError:

@controller is nil: make sure you set it in your test's setup method  
# ./spec/controllers/static\_pages\_controller\_spec.rb:22:in `block (2 levels) in <top (required)>'

Finished in 0.04882 seconds (files took 1.74 seconds to load)

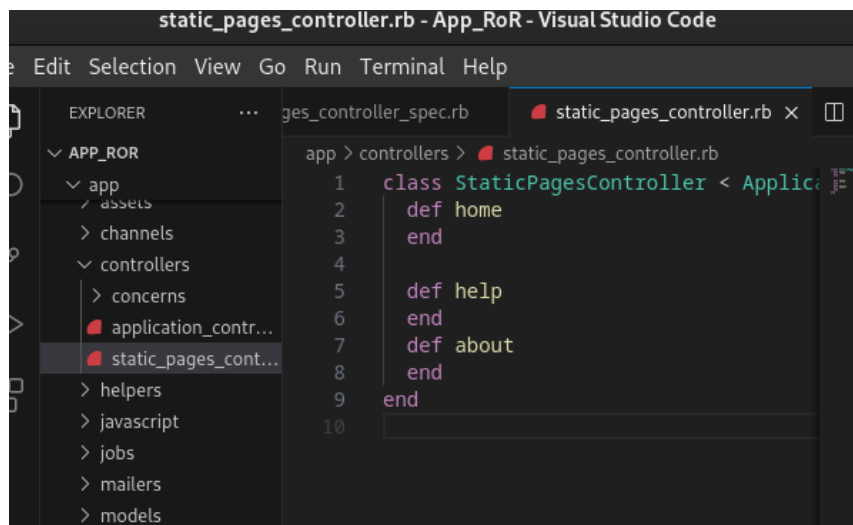
3 examples, 1 failure

Failed examples:

rspec ./spec/controllers/static\_pages\_controller\_spec.rb:21 # GET #about returns http success

```
root@debian:/home/gia/Proyectos_RoR/App_RoR#
```

5.7. Agregar el método en el archivo controllers/static\_pages\_controller.rb y luego ejecutar el test.



```
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_pages_controller_spec.rb
..F
```

Failures:

```
1) StaticPagesController GET #about returns http success
   Failure/Error: get :about

   ActionController::MissingExactTemplate:
     StaticPagesController#about is missing a template for request formats: text/html
   # ./spec/controllers/static_pages_controller_spec.rb:20:in `block (3 levels) in <top (required)>'
```

```
Finished in 0.11109 seconds (files took 1.98 seconds to load)
3 examples, 1 failure
```

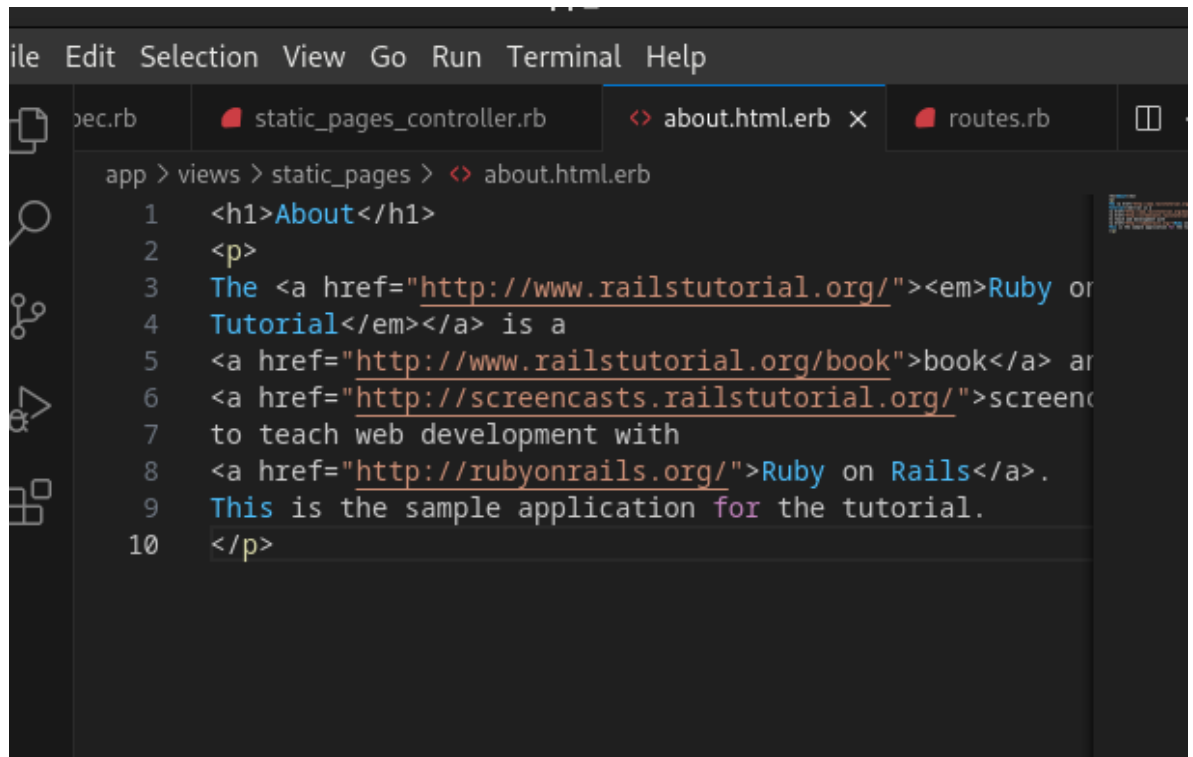
Failed examples:

```
rspec ./spec/controllers/static_pages_controller_spec.rb:19 # StaticPagesController GET #about returns http success
```

```
root@debian:/home/gia/Proyectos_RoR/App_RoR#
```

5.8. Crear una vista para ese método. Crear el archivo about.html.erb en el directorio app/views/static\_pages/ y agregar el código html.

```
root@debian:/home/gia/Proyectos_RoR/App_RoR# ls
app      config.ru  Gemfile    log         README.md  tmp
bin      db         Gemfile.lock  public      spec        vendor
config  Dockerfile  lib        Rakefile    storage
root@debian:/home/gia/Proyectos_RoR/App_RoR# cd app/views/static_pages/
root@debian:/home/gia/Proyectos_RoR/App_RoR/app/views/static_pages# ls
help.html.erb  home.html.erb
root@debian:/home/gia/Proyectos_RoR/App_RoR/app/views/static_pages# touch about.html.erb
root@debian:/home/gia/Proyectos_RoR/App_RoR/app/views/static_pages# ls
about.html.erb  help.html.erb  home.html.erb
root@debian:/home/gia/Proyectos_RoR/App_RoR/app/views/static_pages#
```

A screenshot of a code editor with a dark theme. The top menu bar includes 'File', 'Edit', 'Selection', 'View', 'Go', 'Run', 'Terminal', and 'Help'. Below the menu, there are tabs for 'spec.rb', 'static\_pages\_controller.rb', 'about.html.erb' (which is active and has a close button), and 'routes.rb'. The main editor area shows the content of 'about.html.erb' with the following code:

```
app > views > static_pages > <> about.html.erb
1  <h1>About</h1>
2  <p>
3  The <a href="http://www.railstutorial.org/"><em>Ruby or
4  Tutorial</em></a> is a
5  <a href="http://www.railstutorial.org/book">book</a> an
6  <a href="http://screencasts.railstutorial.org/">screenc
7  to teach web development with
8  <a href="http://rubyonrails.org/">Ruby on Rails</a>.
9  This is the sample application for the tutorial.
10 </p>
```

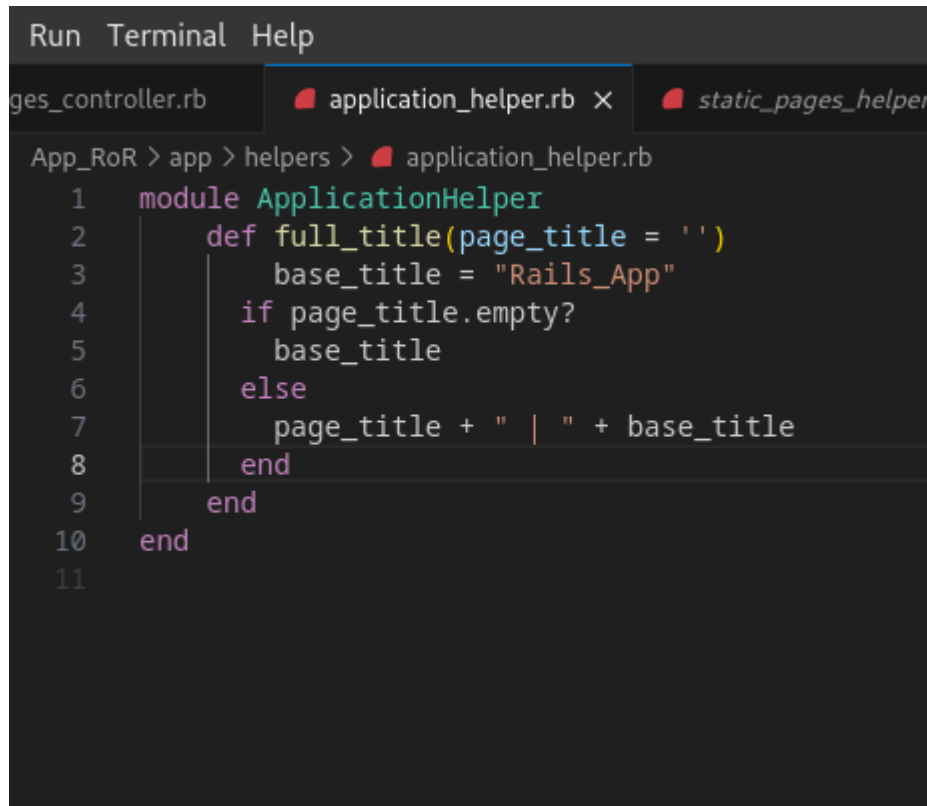
5.9. Para comprobar que todo funciona, ejecutar el test nuevamente.

```
root@debian:/home/gia/Proyectos_RoR/App_RoR/app# cd ..
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_pages
_controller_spec.rb
...

Finished in 0.08837 seconds (files took 1.33 seconds to load)
3 examples, 0 failures

root@debian:/home/gia/Proyectos_RoR/App_RoR#
```

6. Configurar el título de la página. 6.1. Para que el título de la página cambie por cada una de las vistas a la que se accede en la aplicación, copiar el código en el archivo `app/helpers/application_helper.rb` dentro del módulo.

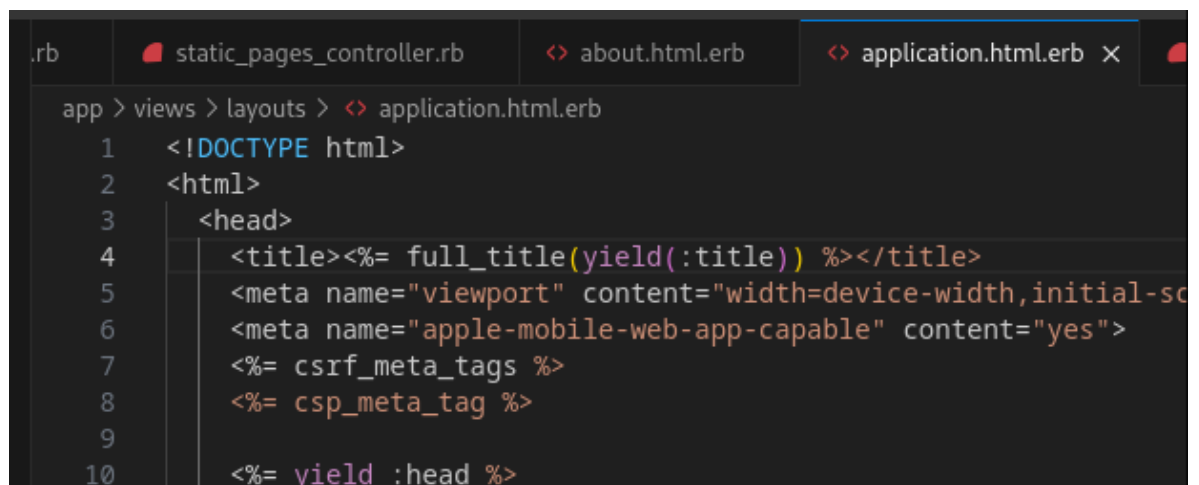


```
Run Terminal Help
ges_controller.rb application_helper.rb x static_pages_helper

App_RoR > app > helpers > application_helper.rb
1 module ApplicationHelper
2   def full_title(page_title = '')
3     base_title = "Rails_App"
4     if page_title.empty?
5       base_title
6     else
7       page_title + " | " + base_title
8     end
9   end
10 end
11
```

6.2. En el archivo `app/views/layouts/application.html.erb` modificar la línea de la etiqueta `title` por la siguiente.

Deberá quedar a como se muestra en la figura .



```
.rb static_pages_controller.rb about.html.erb application.html.erb x

app > views > layouts > application.html.erb
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title><%= full_title(yield(:title)) %></title>
5     <meta name="viewport" content="width=device-width,initial-sc
6     <meta name="apple-mobile-web-app-capable" content="yes">
7     <%= csrf_meta_tags %>
8     <%= csp_meta_tag %>
9
10    <%= yield :head %>
```

## 7. Modificar los archivos de las vistas.

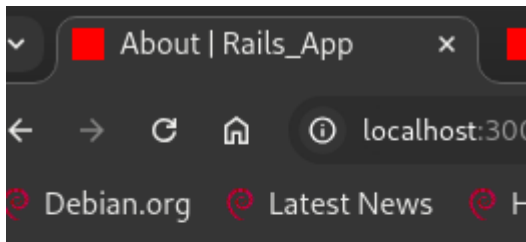
7.1. En el archivo help.html.erb, agregar la siguiente línea en la parte superior del código.

```
App_RoR > app > views > static_pages > <> help.html.erb
1  <%provide(:title,"Help") %>
2  <h1>Help</h1>
3  <p>Find me in app/views/static_pages/help.html.erb
4
5  <h1>Help</h1>
6  <p>
7  Get help on the Ruby on Rails Tutorial at the
8  <a href="http://www.railstutorial.org/help">Rails
9  To get help on this sample app, see the
10 <a href="http://www.railstutorial.org/book"><em>R
11 </p>
```

7.2. En el archivo about.html.erb, agregar la siguiente línea en la parte superior de todo el código.

```
App_RoR > app > views > static_pages > <> about.html.erb
1  <%provide(:title,"About") %>
2  <h1>About</h1>
3  <p>
4  The <a href="http://www.railstutorial.org/"><em>
5  Tutorial</em></a> is a
6  <a href="http://www.railstutorial.org/book">book
7  <a href="http://screencasts.railstutorial.org/">
8  to teach web development with
9  <a href="http://rubyonrails.org/">Ruby on Rails<
10 This is the sample application for the tutorial.
11 </p>
```

7.3. Navegar en la dirección [http://localhost:3000/static\\_pages/about](http://localhost:3000/static_pages/about) y verificar que cambia el título de la página.



## About

The [Ruby on Rails Tutorial](#) is a [book](#) and [Ruby on Rails](#). This is the sample applicat

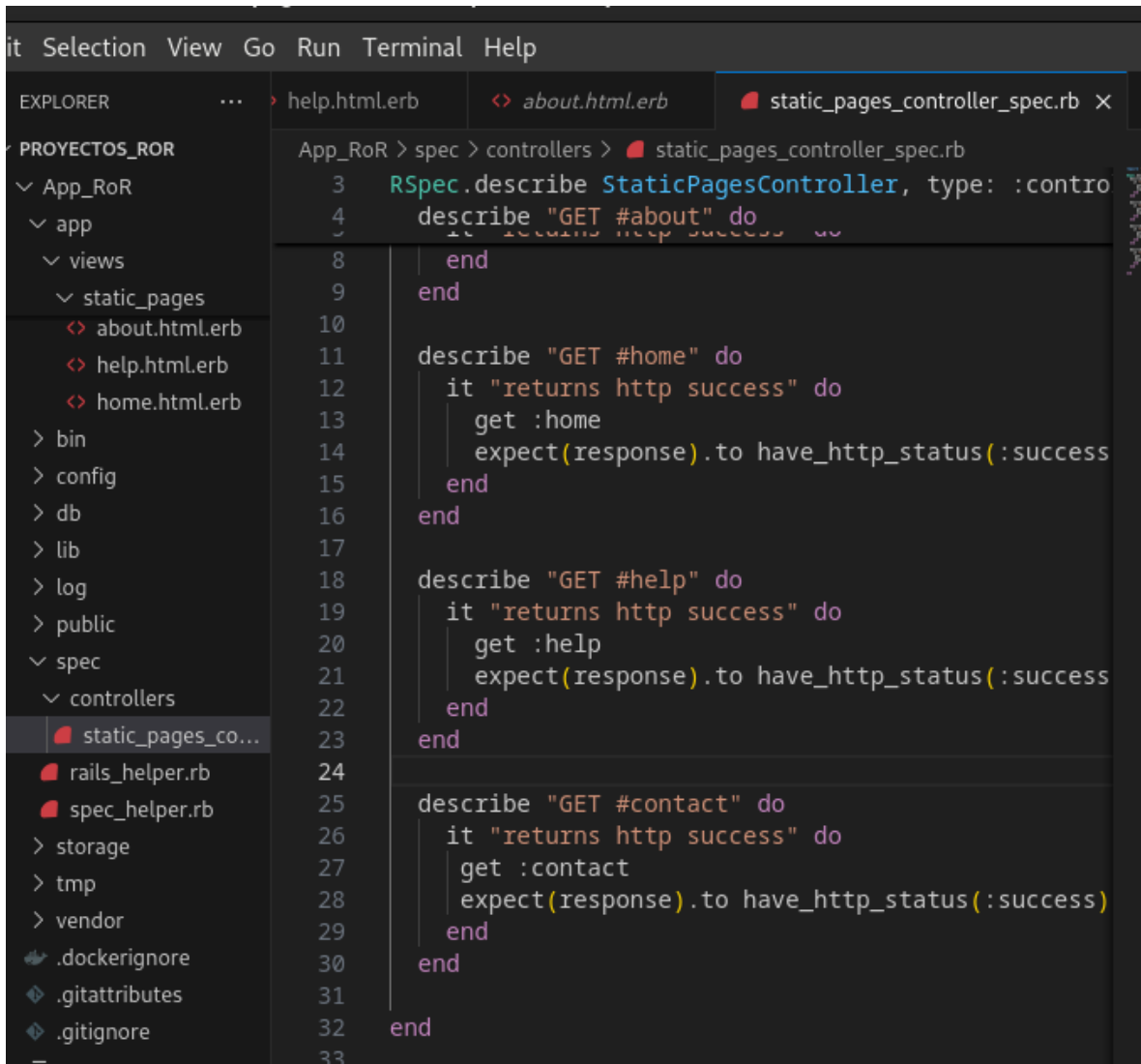
7.4. Navegar en la dirección [http://localhost:3000/static\\_pages/help](http://localhost:3000/static_pages/help) y verificar que cambia el título de la página.

## Help

Get help on the Ruby on Rails Tutorial at the [Rails Tutorial help page](#) To get help c  
sample app, see the [Ruby on RailsTutorialbook](#)



2. Copiar el código del recuadro, en el archivo `static_pages_controller_spec.rb` del directorio `spec/controllers/`.



The screenshot shows a code editor with a dark theme. On the left is the 'EXPLORER' sidebar showing a project structure for 'PROYECTOS\_ROR'. The 'spec' directory is expanded, showing 'controllers' and 'static\_pages\_controller\_spec.rb' (highlighted). The main editor area shows the content of 'static\_pages\_controller\_spec.rb' with line numbers 3 through 33. The code is a RSpec spec for the 'StaticPagesController'.

```
3  RSpec.describe StaticPagesController, type: :controller do
4    describe "GET #about" do
5      it "returns http success" do
6        get :about
7        expect(response).to have_http_status(:success)
8      end
9    end
10
11   describe "GET #home" do
12     it "returns http success" do
13       get :home
14       expect(response).to have_http_status(:success)
15     end
16   end
17
18   describe "GET #help" do
19     it "returns http success" do
20       get :help
21       expect(response).to have_http_status(:success)
22     end
23   end
24
25   describe "GET #contact" do
26     it "returns http success" do
27       get :contact
28       expect(response).to have_http_status(:success)
29     end
30   end
31 end
32
33
```

3. Ejecutar el test y deberá mostrar los siguientes mensajes.

```

Exiting
root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_
pages_controller_spec.rb
...F

Failures:

  1) StaticPagesController GET #contact returns http success
     Failure/Error: get :contact

     ActionController::UrlGenerationError:
       No route matches {:action=>"contact", :controller=>"static_pages"}
     # ./spec/controllers/static_pages_controller_spec.rb:27:in `block (3 :
ls) in <top (required)>'

Finished in 0.03932 seconds (files took 0.92837 seconds to load)
4 examples, 1 failure

Failed examples:

rspec ./spec/controllers/static_pages_controller_spec.rb:26 # StaticPagesCo
nroller GET #contact returns http success

root@debian:/home/gia/Proyectos_RoR/App_RoR#

```

4. Realizar los mismos pasos que realizo con la página about y hacer que el test pase.

```

root@debian:/home/gia/Proyectos_RoR/App_RoR# rspec spec/controllers/static_
pages_controller_spec.rb
....

Finished in 0.08941 seconds (files took 0.858 seconds to load)
4 examples, 0 failures

root@debian:/home/gia/Proyectos_RoR/App_RoR#

```

```

static_pages_controller_spec.rb  contac.html.erb  routes.rb
App_RoR > app > views > static_pages > <> contac.html.erb
1  <%provide(:title,"contac") %>
2  <h1>contac</h1>
3  <p>
4  The <a href="http://www.railstutorial.org/"><em>Rub
5  Tutorial</em></a> is a
6  <a href="http://www.railstutorial.org/book">book</a>
7  <a href="http://screencasts.railstutorial.org/">sca
8  to teach web development with
9  <a href="http://rubyonrails.org/">Ruby on Rails</a>
10 This is the sample application for the tutorial.
11 </p>

```

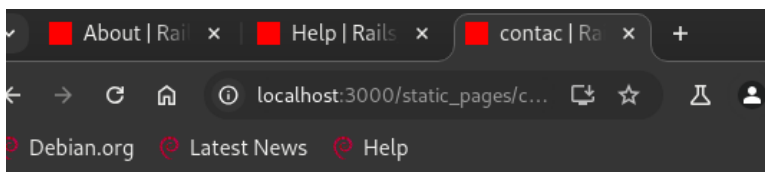
```

static_pages_controller_spec.rb  contac.html.erb
App_RoR > config > routes.rb
1  Rails.application.routes.draw do
2    get "static_pages/home"
3    get "static_pages/help"
4    get 'static_pages/about'
5    get 'static_pages/contac'
6    # Define your application routes
7
8    # Reveal health status on /up that
9    # Can be used by load balancers at

```

```
pages_helper.rb static_pages_controller.rb x application_helper.rb
App_RoR > app > controllers > static_pages_controller.rb
1 class StaticPagesController < ApplicationController
2   def home
3   end
4
5   def help
6   end
7
8   def about
9   end
10
11  def contac
12  end
13 end
14
```

5. Navegar en la dirección [http://localhost:3000/static\\_pages/contact](http://localhost:3000/static_pages/contact).



**contac**

he [Ruby on Rails Tutorial](#) is a [book](#) and [screencast series](#) to teach web development [uby on Rails](#). This is the sample application for the tutorial.

