# Guía 4: Introducción al framework Ruby on Rails

**Introducción**

Ruby on Rails es un framework de aplicaciones web de código abierto que sigue el paradigma del patrón de diseño Modelo Vista Controlador (MVC), dando la posibilidad de crear aplicaciones web con poco código y configuraciones mínimas.

En esta guía se tendrá un acercamiento al framework para conocer la base de su funcionamiento: versión, configuración, directorios, comandos básicos que se utilizan al momento de crear aplicaciones, etc.

# Objetivos

* Verificar las versiones de software instalados
* Conocer comandos básicos del framework RoR.
* Crear una primera aplicación básica en RoR.
* Conocer acerca de la jerarquía de directorios

# Tiempo

* Una sesión de clase

# Requerimientos

|  |  |
| --- | --- |
| **Software** | **Hardware** |
| Sistema Debian 9 virtualizado en Virtual box con:   * Ruby versión 2.4.1 * Rails 5.1.4 * Nodejs * Sublimetext | Computadora con características:   * Memoria ram mínimo 2GB * Procesador mínimo 2.1 GHz |

**Referencia**

* Michael Hartl. Ruby on Rails tutorial (Rail5), Learn Web Development with Rails. https:/[/www.railstutorial.org/book/be](http://www.railstutorial.org/book/beginning)g[inning](http://www.railstutorial.org/book/beginning)
* Ruby on Rails org, RailsGuides, The Rails command line. <http://guides.rubyonrails.org/command_line.html>

# Desarrollo

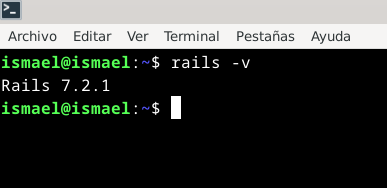
1. Verificar las versiones de software instaladas.

Es importante verificar las versiones de software instaladas, para así poder continuar con la realización de la guía sin ningún problema.

* 1. Verificar la versión de Rails.rails

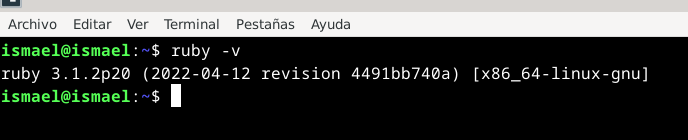
$ rails -v

Con este comando verifica la versión de Rails instalada.



**Figura 1 Versión de Rails**

* 1. Verificar la versión de Ruby.



Obtendrá una salida como se muestra en la siguiente figura.

* 1. Verificar la versión de sqlite.

$ sqlite3 --version

1. Crear un nuevo proyecto en Rails.
   1. Haciendo uso del terminal, ubicarse en el directorio donde se va a almacenar el proyecto.

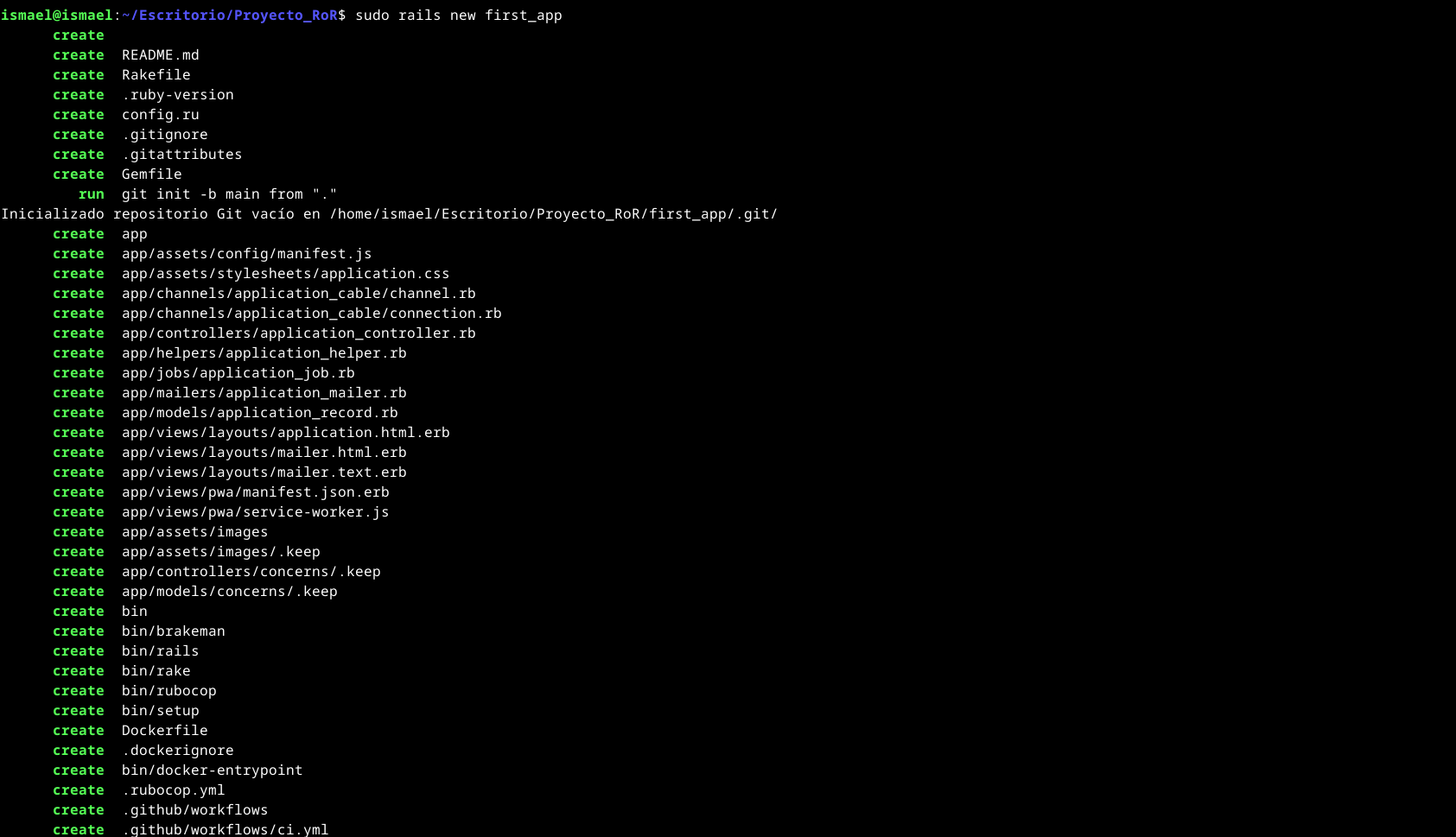


$ cd /home/debian/Proyectos\_RoR

* 1. Generar un proyecto nuevo utilizando el siguiente comando:

$ rails new first\_app

Al ejecutar el comando anterior se está creando un proyecto nuevo, mostrará unos mensajes de la creación de archivo y directorios a como se observa en la figura 40.



* 1. Entrar a la carpeta del proyecto creado.

$ cd first\_app/

Cada vez que se crea un proyecto en Rails, se debe ingresar al directorio que se ha generado con el comando rails new. Este directorio, es el que contiene la nueva aplicación en la que se va a trabajar y en todos los casos se debe estar ubicado dentro de él, para así hacer uso de los distintos comandos que se ejecutan en el desarrollo de aplicaciones con Ruby on Rails.

# Jerarquía de directorios

Al momento de crear un nuevo proyecto; Rails genera un conjunto de archivos y directorios necesarios para su funcionamiento, dentro de los más importantes y más utilizados están: el directorio **app** en este se encuentra información sobre las vistas, controladores, archivos css; el directorio **config** en este se encuentran archivos de configuración como **routes.rb** en el cual se configuran las rutas de la aplicación; el directorio **db** es el relacionado a la base de datos del proyecto, en este directorio se almacena el esquema actual de la base de datos y las migraciones que se realizan; el directorio **test** en este es donde se crean los archivos de los tests que se le aplican en la aplicación; y el archivo **Gemfile** este es el encargado de las gemas que se utilizan en el proyecto, a continuación se muestra una breve descripción de cada uno de ellos.

**Jerarquía de directorios de un proyecto en rails**

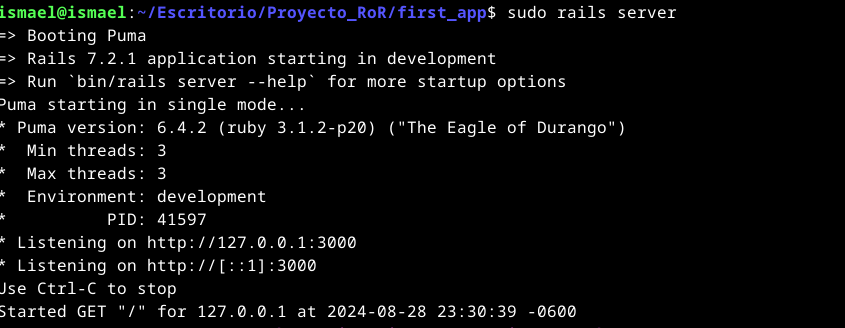
|  |  |
| --- | --- |
| **Directorio** | **Descripción** |
| app/ | Se encuentra todos los archivos y directorios relacionados con los **modelos**, **vistas**, **controladores** y **helpers** que son partes de la aplicación. También se encuentra en directorio assets el cual contiene el directorio **images** que es donde se ubican las **images** que contiene la aplicación, el directorios **stylesheets** que contienes los archivos de estilos y el directorio **javascripts** donde guardan los archivos javascripts (js) de la aplicación. |
| bin/ | Contiene un conjunto de scripts que son incluidos dentro del propio  framework. |
| config/ | Se encuentran los archivos de configuración de las rutas, de la base de datos etc. |
| db/ | Almacena el esquema actual de la base de datos y las migraciones. |
| Gemfile | Uno de los archivos más importantes del proyecto, este es el archivo de gemas encargado de especificar las dependencias de la aplicación Rails. |
| lib/ | Carpeta donde se incluyen módulos que extienden la aplicación. |
| log/ | Contiene los ficheros de registros de actividades del servidor. |
| public/ | Directorio de donde se sirven los archivos estáticos y los recursos compilados del framework. |
| Rakefile | Sirve para localizar las tareas que se pueden ejecutar desde la línea de comandos. |
| README.md | Contiene una pequeña descripción de la aplicación. |
| test/ | Test unitarios de la aplicación. |
| tmp/ | Archivos temporales como caché, archivos pid del sistema, etc. |
| vendor/ | Carpeta donde introducir el código de terceros a la aplicación. |

* 1. Una vez ubicado dentro del directorio del nuevo proyecto creado, iniciar el servidor para comprobar que arranca sin ningún problema.

$ rails server

Con la versión de Rails que se trabaja, viene instalado el servidor PUMA por defecto, que es una pequeña librería que proporciona un servidor HTTP muy rápido y concurrente para aplicaciones web de ruby, creado por Evan Phoenix a finales del 2011. A diferencia de otros servidores utilizados en un ambiente de desarrollo como WEBrick que es un servidor sencillo y que permite un solo proceso lo cual hace que las conexiones y solicitudes tarden más tiempo; PUMA utiliza hilos, además de procesos de trabajo para aprovechar de una mejor forma la CPU. En el proyecto creado en Rails se puede observar la configuración del servidor en el archivo **config/puma.rb**, este servidor se utiliza comúnmente en un ambiente de desarrollo, pero debido a su gran capacidad de procesamiento también puede ser utilizado en ambiente de producción.

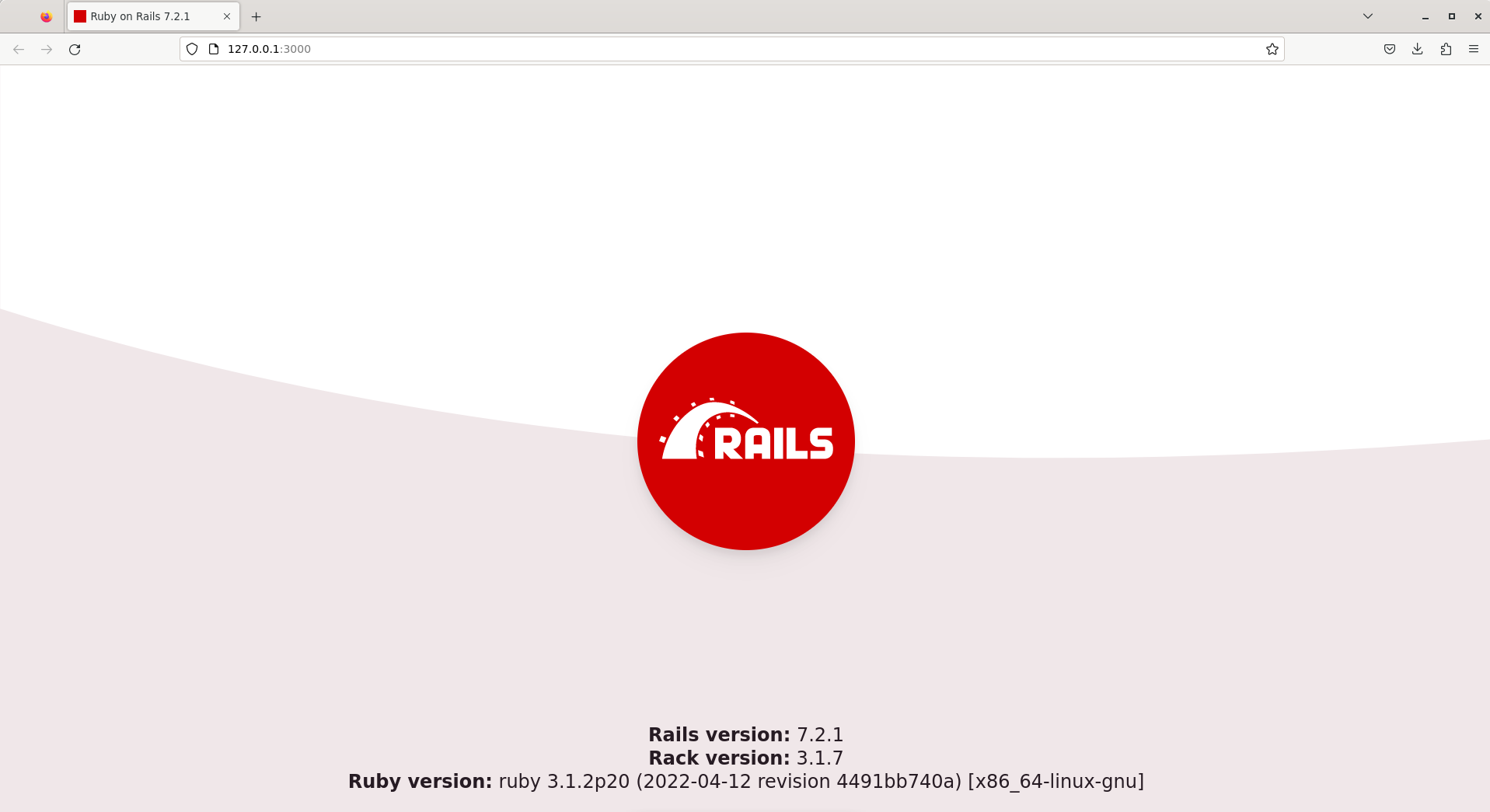
Si el servidor inicia de manera correcta, mostrará unos mensajes en el terminal donde se puede observar el nombre del servidor que se ejecuta, versión de Rails, versión de Ruby, cantidad de hilos o procesos configurados, ambiente en el que se está trabajando, la dirección y puerto en el que escucha el servidor, a como se muestra en la siguiente figura.



* 1. Abrir el navegador y escribir la siguiente dirección.

http://127.0.0.1:3000/

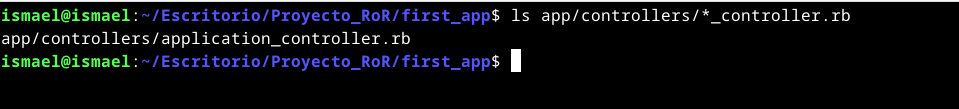
Navegando en la dirección anterior, tendrá acceso a la página por defecto del proyecto nuevo.



Detener el servidor presionando la combinación de teclas Ctrl+C en el terminal.

1. Agregar una acción al controlador.
   1. Cuando se crea un proyecto nuevo en Rails, se crea por defecto un controlador llamado Application\_controller. Para Verificar que existe el controlador, puede ejecutar el siguiente comando en el terminal.

$ ls app/controllers/\*\_controller.rb



Obtendrá una salida como la de la siguiente figura.

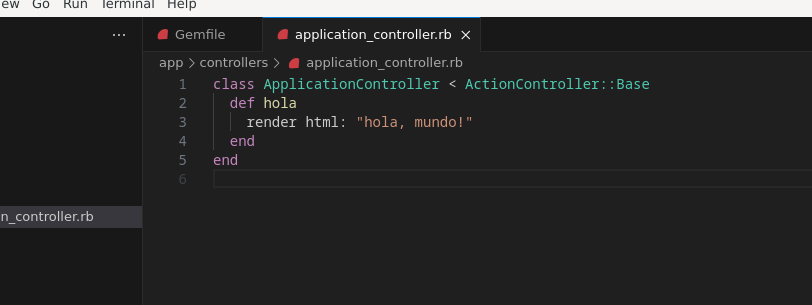
* 1. Abrir el archivo del controlador que se encuentra en la dirección app/controllers/**application\_controller.rb** en el editor de texto, y definir la acción del controlador.

def hola

render html: "hola, mundo!" end

Por defecto, cuando se ejecuta la acción de un controlador, Rails renderiza una vista que se llame igual a la acción (al método) y que se encuentre en una carpeta que se llame igual al controlador dentro del directorio **app/views.** Este comportamiento se puede cambiar haciendo uso del método **render,** con este método se puede renderizar código HTML o una vista diferente**,** como se puede observar en el código anterior en la acción hola se hace un render html del texto “hola mundo”

El archivo controller/**application\_controller.rb** deberá quedar como se muestra en la **figura 44.**

****

1. Configurar el archivo **routes.rb**.

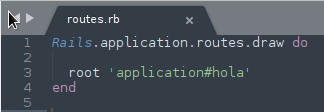
El archivo routes.rb es donde se definen las rutas de la aplicación en Rails, cada ruta está compuesta por: verbos HTTP (GET, POST, PATCH, PUT, DELETE), path, controlador y método (acción). Otra parte importante es el método root, este método es por medio del cual se establece la ruta raíz de la aplicación, esta ruta solicitudes get a una acción, es recomendable que esta ruta siempre este el parte superior del archivo routes.

* 1. Abrir el archivo routes (**config/routes.rb**) y agregar la siguiente línea.

root 'application#hola'

En este caso al método root se hace relación al controlado y la acción **hola** que pertenece a ese controlador.

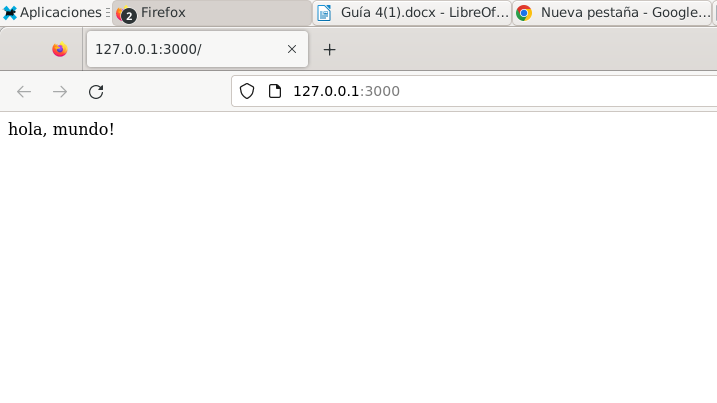
El archivo config/**routes.rb** quedará como se muestra a continuación.



* 1. Guardar los cambios realizados e iniciar el servidor en el terminal.

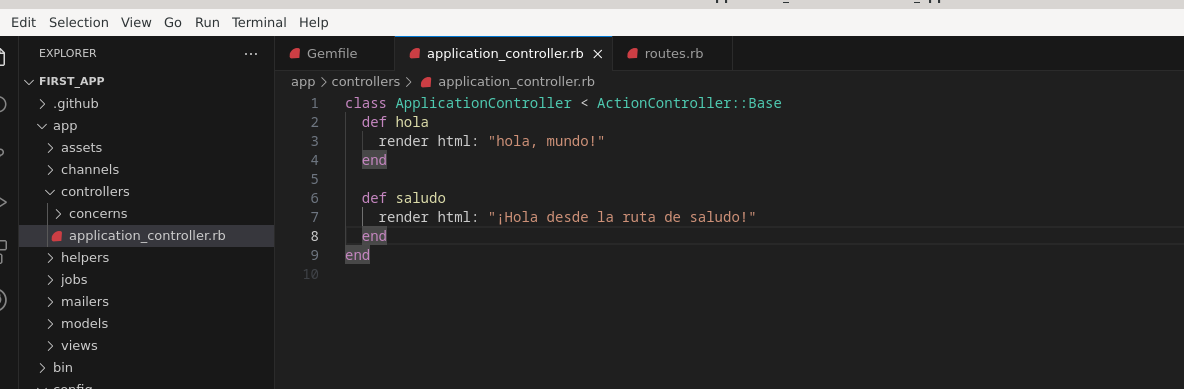
$ rails s

* 1. Abrir el navegador en la dirección 127.0.0.1:3000 y verificar que ahora, la página principal presenta la acción **hola** creada anteriormente.



**Ejercicios propuestos para ser entregados al docente**

1. Realizar cada uno de los enunciados de la guía y verificar el correcto funcionamiento.
2. Crear una acción nueva que muestre un texto cualquiera.



1. Editar el archivo routes.rb, para que la aplicación siempre inicie, por defecto, con la nueva acción creada en el inciso anterior.

