

UNIVERSIDAD NACIONAL AUTÓNOMA DE NICARAGUA, UNAN-LEÓN

FACULTAD DE CIENCIAS Y TECNOLOGÍA



**UNAN-LEÓN
FUNDADA EN 1812**

Carrera: Ingeniería En Telemática

Componente: Software como un Servicio

Tarea: Guía 4.

Nombre:

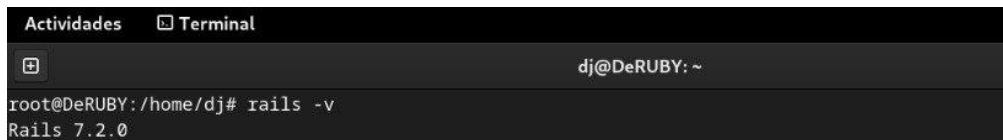
Luis Adolfo Díaz Mendoza

León, Nicaragua 28 de agosto del 2024.

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.

1.1. Verificar la versión de Rails.rails

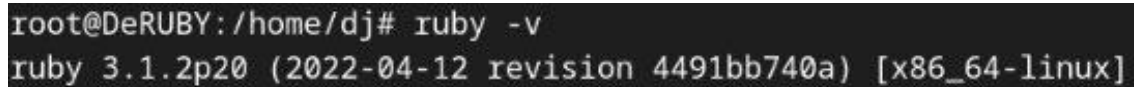
\$ rails -v: Con este comando verifica la versión de Rails instalada.



```
Actividades  Terminal
dj@DeRUBY: ~
root@DeRUBY:/home/dj# rails -v
Rails 7.2.0
```

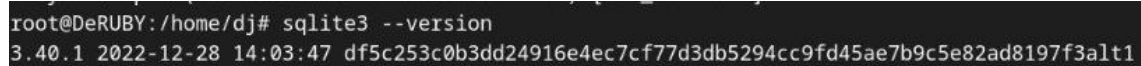
1.2. Verificar la versión de Ruby.

\$ ruby -v: Obtendrá una salida como se muestra en la siguiente figura.



```
root@DeRUBY:/home/dj# ruby -v
ruby 3.1.2p20 (2022-04-12 revision 4491bb740a) [x86_64-linux]
```

1.3. Verificar la versión de sqlite. \$ sqlite3 --version 2. Crear un nuevo proyecto en Rails.



```
root@DeRUBY:/home/dj# sqlite3 --version
3.40.1 2022-12-28 14:03:47 df5c253c0b3dd24916e4ec7cf77d3db5294cc9fd45ae7b9c5e82ad8197f3alt1
```

2.1. Haciendo uso del terminal, ubicarse en el directorio donde se va a almacenar el proyecto.

\$ cd /home/debian/Proyectos_RoR



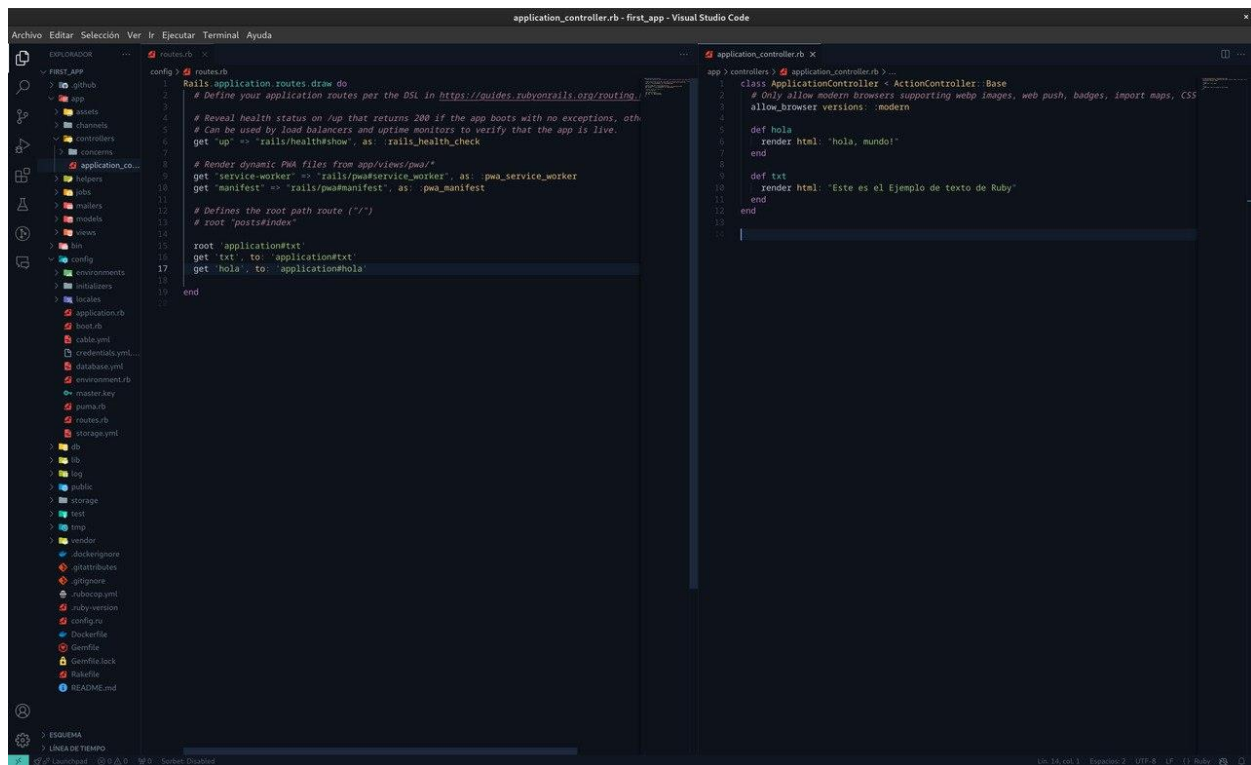
```
root@DeRUBY:/home/dj# cd Documentos/Ruby/P4
```

2.2. Generar un proyecto nuevo utilizando el siguiente comando:

```
$ rails new first_app
```

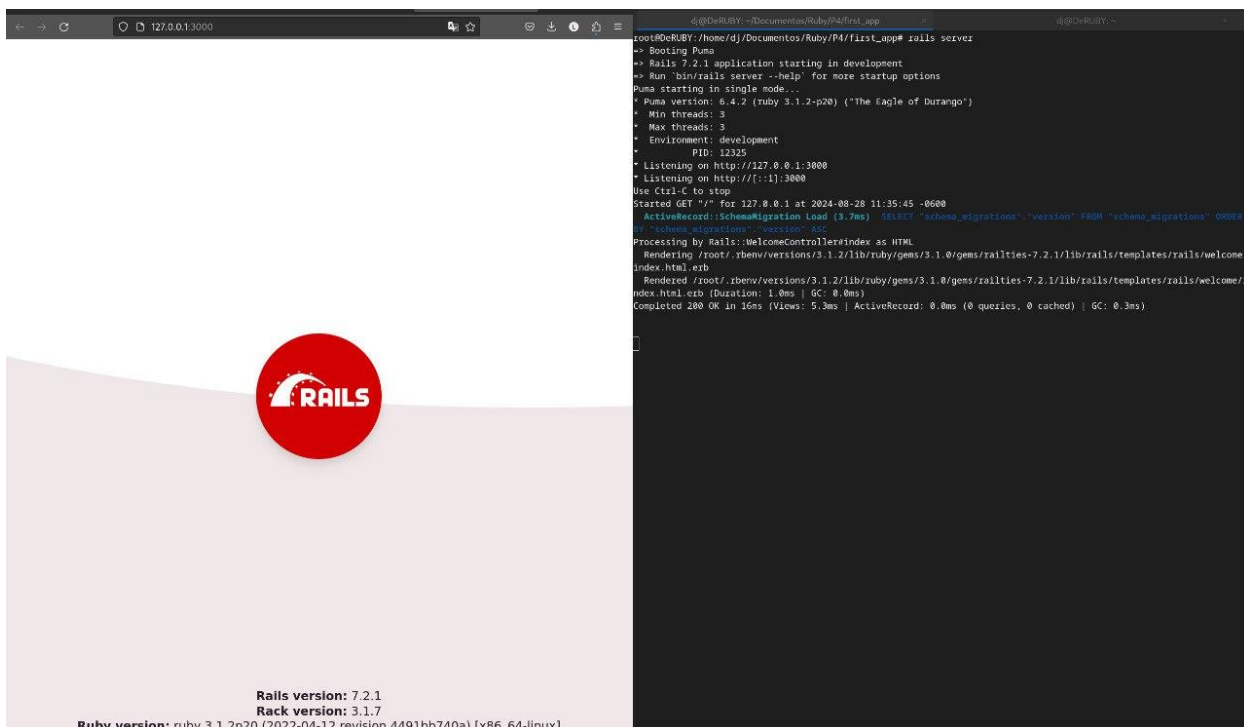
Al ejecutar el comando anterior se está creando un proyecto nuevo, mostrará unos mensajes de la creación de archivo.

```
root@DeRUBY:/home/dj/Documentos/Ruby/P4# rails new first_app
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/dj/Documentos/Ruby/P4/first_app/.git/
create  app
create  app/assets/config/manifest.js
create  app/assets/stylesheets/application.css
create  app/channels/application_cable/channel.rb
create  app/channels/application_cable/connection.rb
create  app/controllers/application_controller.rb
create  app/helpers/application_helper.rb
create  app/jobs/application_job.rb
create  app/mailers/application_mailer.rb
create  app/models/application_record.rb
create  app/views/layouts/application.html.erb
create  app/views/layouts/mailer.html.erb
create  app/views/layouts/mailer.text.erb
create  app/views/pwa/manifest.json.erb
create  app/views/pwa/service-worker.js
create  app/assets/images
create  app/assets/images/.keep
create  app/controllers/concerns/.keep
create  app/models/concerns/.keep
create  bin
create  bin/brakeman
create  bin/rails
create  bin/rake
create  bin/rubocop
create  bin/setup
create  Dockerfile
create  .dockerignore
create  bin/docker-entrypoint
create  .rubocop.yml
create  .github/workflows
create  .github/workflows/ci.yml
create  .github/dependabot.yml
create  config
create  config/routes.rb
create  config/application.rb
create  config/environment.rb
create  config/cable.yml
create  config/puma.rb
create  config/storage.yml
create  config/environments
```



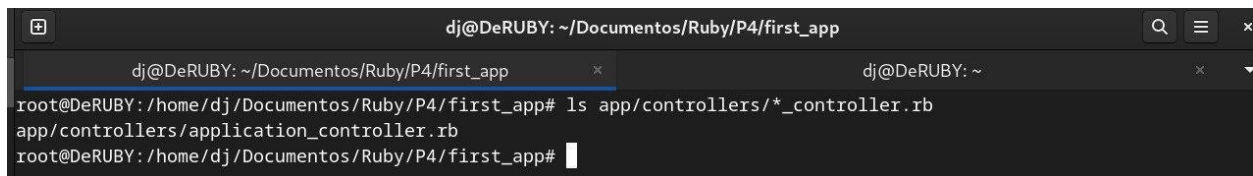
2.4. Una vez ubicado dentro del directorio del nuevo proyecto creado, iniciar el servidor para comprobar que arranca sin ningún problema.

\$ rails server



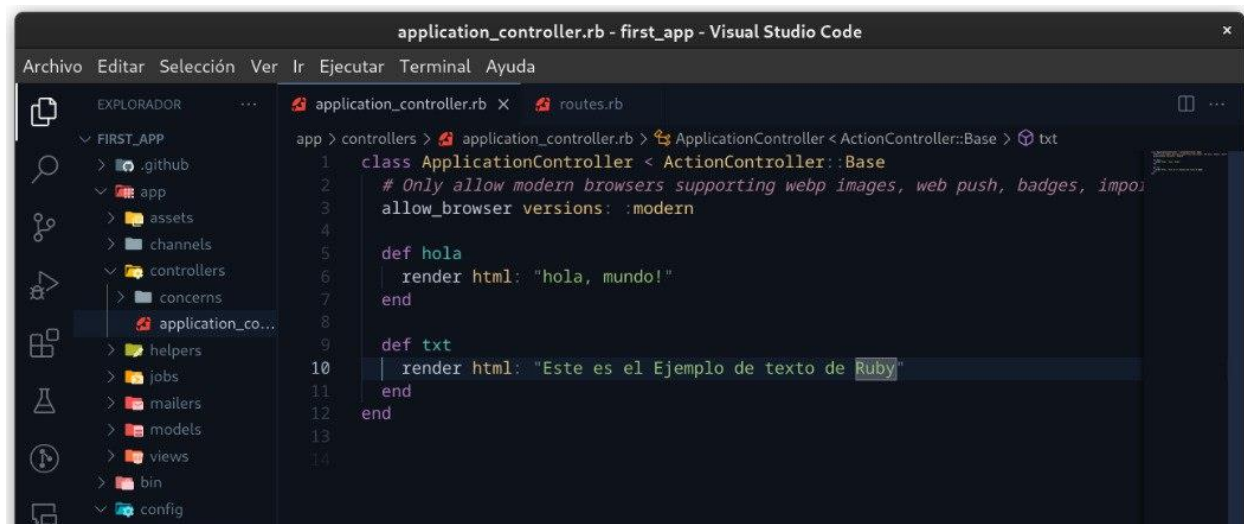
3. Agregar una acción al controlador. 3.1. Cuando se crea un proyecto nuevo en Rails, se crea por defecto un controlador llamado `Application_controller`. Para Verificar que existe el controlador, puede ejecutar el siguiente comando en el terminal.

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



```
terminal
dj@DeRUBY: ~/Documentos/Ruby/P4/first_app
root@DeRUBY: /home/dj/Documentos/Ruby/P4/first_app# ls app/controllers/*_controller.rb
app/controllers/application_controller.rb
root@DeRUBY: /home/dj/Documentos/Ruby/P4/first_app#
```

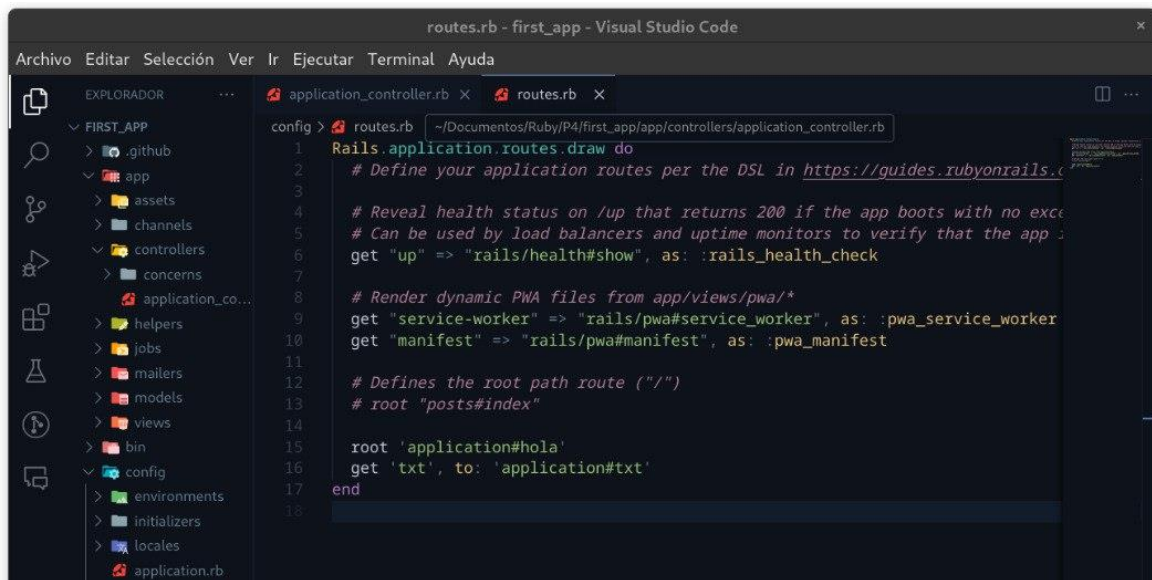
3.2. Abrir el archivo del controlador que se encuentra en la dirección `app/controllers/application_controller.rb` en el editor de texto, y definir la acción del controlador en este caso se escribe un hola mundo y también se emplea la parte del ejercicio propuesto 2. Crear una acción nueva que muestre un texto cualquiera en este caso se pone el texto “ESTE ES EL EJEMPLO DE TEXTO”



```
application_controller.rb - first_app - Visual Studio Code
EXPLORADOR
FIRST_APP
  .github
  app
    assets
    channels
    controllers
    concerns
    application_co...
  helpers
  jobs
  mailers
  models
  views
  bin
  config

app > controllers > application_controller.rb > ApplicationController < ActionController::Base > txt
1 class ApplicationController < ActionController::Base
2   # Only allow modern browsers supporting webp images, web push, badges, im...
3   allow_browser versions: :modern
4
5   def hola
6     render html: "hola, mundo!"
7   end
8
9   def txt
10    render html: "Este es el Ejemplo de texto de Ruby"
11  end
12 end
13
14
```

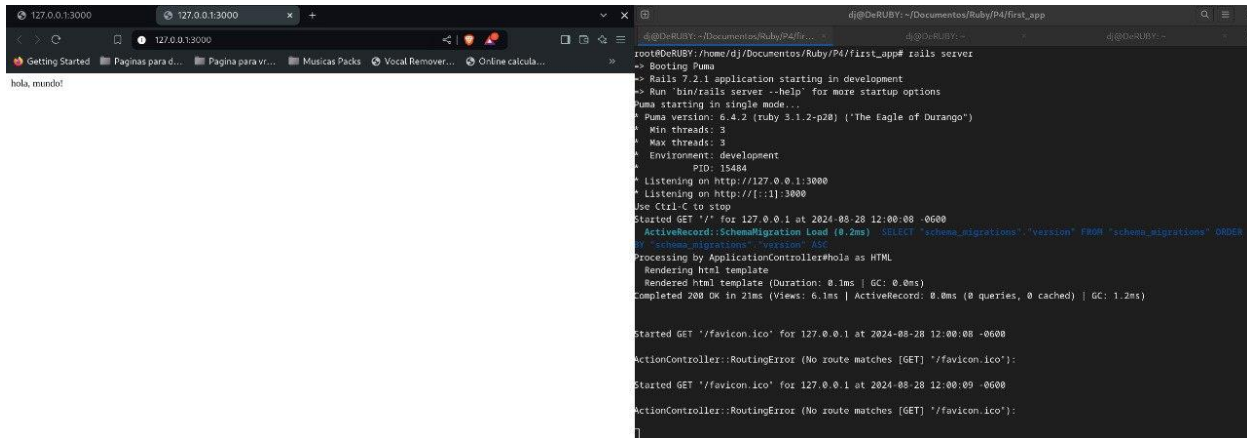
4. Configurar el archivo routes.rb. El archivo routes.rb es donde se definen las rutas de la aplicación en Rails, cada ruta está compuesta por: verbos HTTP (GET, POST, PATCH, PUT, DELETE), path, controlador y método (acción). Otra parte importante es el método root, este método es por medio del cual se establece la ruta raíz de la aplicación, esta ruta solicita get a una acción, es recomendable que esta ruta siempre esté en la parte superior del archivo routes. 4.1. 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 controlador y la acción hola que pertenece a ese controlador. También se logra ver como se hace la llamada de la definición “txt”



```
routes.rb - first_app - Visual Studio Code
Archivo  Editar  Selección  Ver  Ir  Ejecutar  Terminal  Ayuda
EXPLORADOR  application_controller.rb  routes.rb
FIRST_APP
  .github
  app
    assets
    channels
    controllers
    concerns
    application_co...
  helpers
  jobs
  mailers
  models
  views
  bin
  config
  environments
  initializers
  locales
  application.rb

config/routes.rb
1 Rails.application.routes.draw do
2   # Define your application routes per the DSL in https://guides.rubyonrails.org/routing.html
3
4   # Reveal health status on /up that returns 200 if the app boots with no exceptions
5   # Can be used by load balancers and uptime monitors to verify that the app is up
6   get "up" => "rails/health#show", as: :rails_health_check
7
8   # Render dynamic PWA files from app/views/pwa/*
9   get "service-worker" => "rails/pwa#service_worker", as: :pwa_service_worker
10  get "manifest" => "rails/pwa#manifest", as: :pwa_manifest
11
12  # Defines the root path route ("/")
13  # root "posts#index"
14
15  root 'application#hola'
16  get 'txt', to: 'application#txt'
17 end
```

En este punto se logra apreciar la ejecución del servidor y lo que muestra en pantalla una vez que se han configurado los archivos.

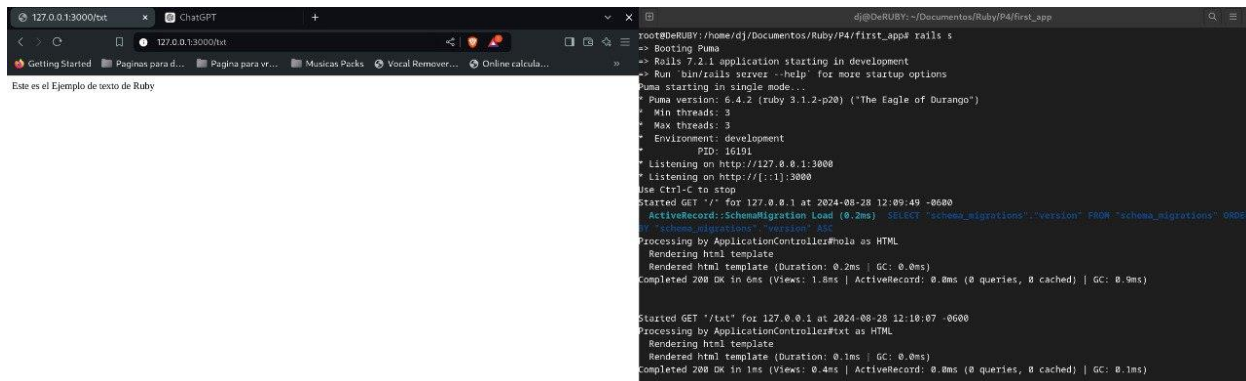


The screenshot shows a web browser window on the left and a terminal window on the right. The browser's address bar shows '127.0.0.1:3000' and the page content is 'hola, mundo!'. The terminal window shows the command 'rails server' being executed in the directory ~/Documents/Ruby/P4/first_app. The output indicates that the Rails 7.2.1 application is starting in development mode, using Puma version 6.4.2. It shows the application listening on http://127.0.0.1:3000. The first request is a GET request to '/', which returns a 200 OK status. The terminal output also shows the rendering of an HTML template and the completion of the request.

```
root@DeRUBY: ~/Documents/Ruby/P4/first_app# rails server
=> Booting Puma
=> Rails 7.2.1 application starting in development
=> Run 'bin/rails server --help' for more startup options
Puma starting in single mode...
Puma version: 6.4.2 (ruby 3.1.2-p20) ("The Eagle of Durango")
* Min threads: 3
* Max threads: 3
* Environment: development
* PID: 15484
* Listening on http://127.0.0.1:3000
* Listening on http://[::]:3000
Use Ctrl-C to stop
Started GET "/" for 127.0.0.1 at 2024-08-28 12:00:08 -0600
ActiveRecord::SchemaMigration Load (0.2ms) SELECT "schema_migrations"."version" FROM "schema_migrations" ORDER BY "schema_migrations"."version" ASC
Processing by ApplicationController#hola as HTML
Rendering html template
Rendered html template (Duration: 0.1ms | GC: 0.0ms)
Completed 200 OK in 2ms (Views: 6.1ms | ActiveRecord: 0.0ms (0 queries, 0 cached) | GC: 1.2ms)

Started GET "/favicon.ico" for 127.0.0.1 at 2024-08-28 12:00:08 -0600
ActionController::RoutingError (No route matches [GET] "/favicon.ico"):

Started GET "/favicon.ico" for 127.0.0.1 at 2024-08-28 12:00:09 -0600
ActionController::RoutingError (No route matches [GET] "/favicon.ico"):
```



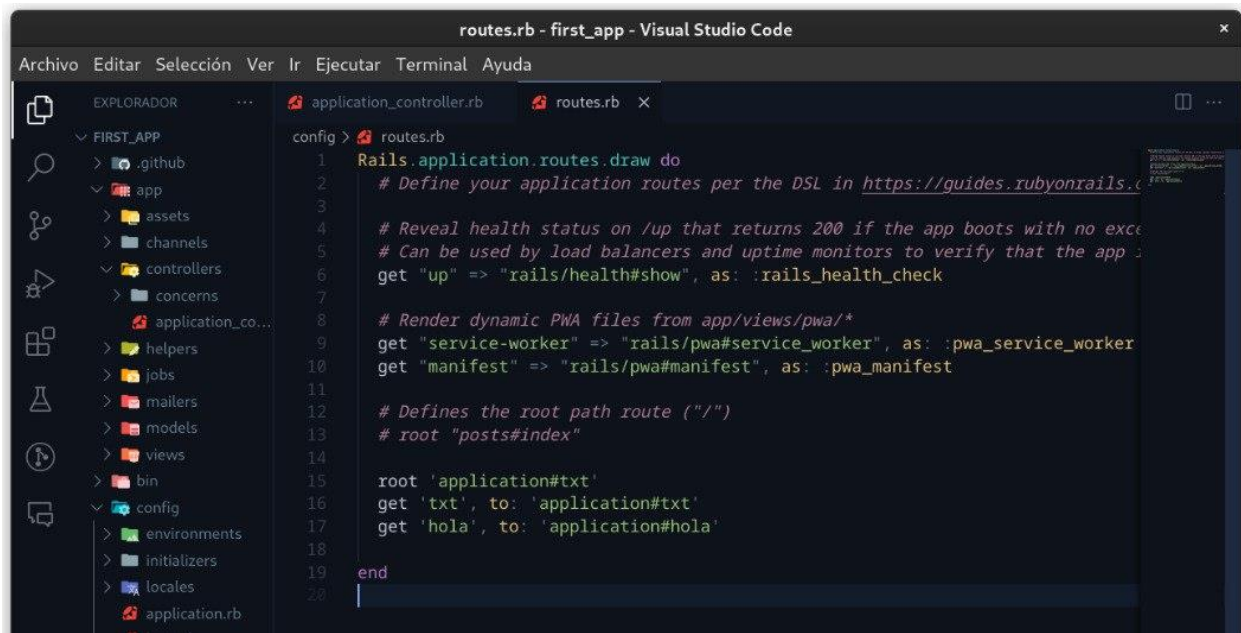
The screenshot shows a web browser window on the left and a terminal window on the right. The browser's address bar shows '127.0.0.1:3000/hola' and the page content is 'Este es el Ejemplo de texto de Ruby'. The terminal window shows the command 'rails s' being executed in the directory ~/Documents/Ruby/P4/first_app. The output indicates that the Rails 7.2.1 application is starting in development mode, using Puma version 6.4.2. It shows the application listening on http://127.0.0.1:3000. The first request is a GET request to '/hola', which returns a 200 OK status. The terminal output also shows the rendering of an HTML template and the completion of the request.

```
root@DeRUBY: ~/Documents/Ruby/P4/first_app# rails s
=> Booting Puma
=> Rails 7.2.1 application starting in development
=> Run 'bin/rails server --help' for more startup options
Puma starting in single mode...
Puma version: 6.4.2 (ruby 3.1.2-p20) ("The Eagle of Durango")
* Min threads: 3
* Max threads: 3
* Environment: development
* PID: 16191
* Listening on http://127.0.0.1:3000
* Listening on http://[::]:3000
Use Ctrl-C to stop
Started GET "/hola" for 127.0.0.1 at 2024-08-28 12:09:49 -0600
ActiveRecord::SchemaMigration Load (0.2ms) SELECT "schema_migrations"."version" FROM "schema_migrations" ORDER BY "schema_migrations"."version" ASC
Processing by ApplicationController#hola as HTML
Rendering html template
Rendered html template (Duration: 0.2ms | GC: 0.0ms)
Completed 200 OK in 6ms (Views: 1.8ms | ActiveRecord: 0.0ms (0 queries, 0 cached) | GC: 0.9ms)

Started GET "/txt" for 127.0.0.1 at 2024-08-28 12:10:07 -0600
Processing by ApplicationController#txt as HTML
Rendering html template
Rendered html template (Duration: 0.1ms | GC: 0.0ms)
Completed 200 OK in 1ms (Views: 0.4ms | ActiveRecord: 0.0ms (0 queries, 0 cached) | GC: 0.1ms)
```


En este punto se aplica el ejercicio propuesto 3 Editar el archivo routes.rb, para que la aplicación siempre inicie, por defecto, con la nueva acción creada en el inciso anterior.

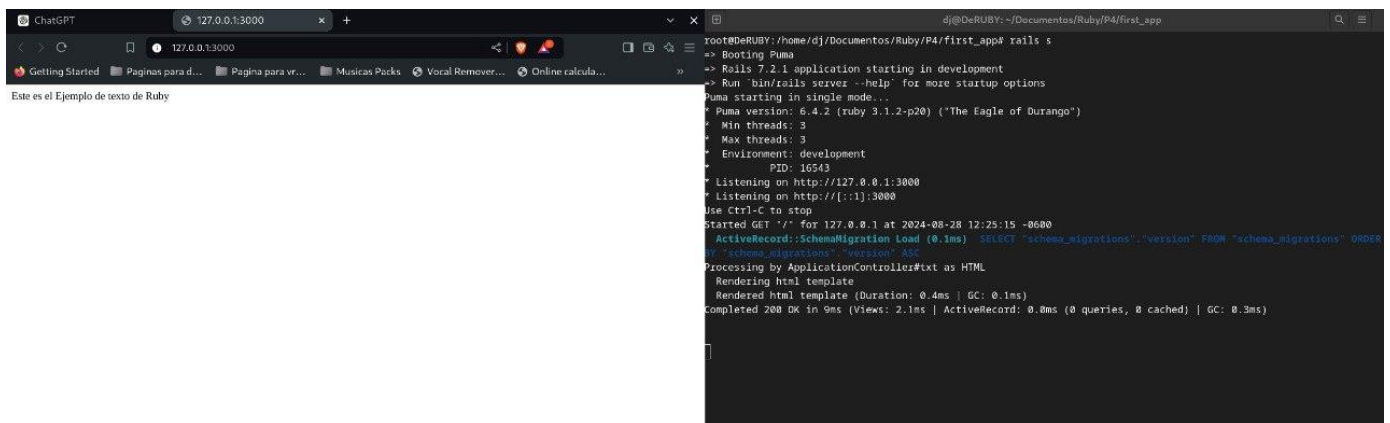
A



The screenshot shows the Visual Studio Code editor with the 'routes.rb' file open. The file is located in the 'config' directory of a project named 'first_app'. The code defines several routes: a health check route at '/up', dynamic PWA routes for 'service-worker' and 'manifest', a root path route at '/', and two GET routes at '/txt' and '/hola' that point to the 'application#txt' and 'application#hola' actions respectively.

```
1 Rails.application.routes.draw do
2   # Define your application routes per the DSL in https://guides.rubyonrails.org/routing.html
3
4   # Reveal health status on /up that returns 200 if the app boots with no exceptions
5   # Can be used by load balancers and uptime monitors to verify that the app is up
6   get "up" => "rails/health#show", as: :rails_health_check
7
8   # Render dynamic PWA files from app/views/pwa/*
9   get "service-worker" => "rails/pwa#service_worker", as: :pwa_service_worker
10  get "manifest" => "rails/pwa#manifest", as: :pwa_manifest
11
12  # Defines the root path route ("/")
13  # root "posts#index"
14
15  root 'application#txt'
16  get 'txt', to: 'application#txt'
17  get 'hola', to: 'application#hola'
18
19 end
```

Por acá se muestra una vez que abrimos el servidor <http://127.0.0.1:3000> nos abre el navegador y nos aparece el mensaje



The screenshot shows a web browser window at the address '127.0.0.1:3000' displaying the message 'Este es el Ejemplo de texto de Ruby'. To the right, a terminal window shows the output of the 'rails s' command, indicating that the Rails server is running on http://127.0.0.1:3000. The terminal output includes details about the Puma version, environment, and the successful rendering of the 'application#txt' action.

```
Root@DeRUBY: ~/Documents/Ruby/P4/first_app
$ rails s
=> Booting Puma
=> Rails 7.2.1 application starting in development
=> Run 'bin/rails server --help' for more startup options
Puma starting in single mode...
* Puma version: 6.4.2 (ruby 3.1.2-p20) ("The Eagle of Durango")
* Min threads: 3
* Max threads: 3
* Environment: development
* PID: 16543
* Listening on http://127.0.0.1:3000
* Listening on http://[::]:3000
Use Ctrl-C to stop

Started GET "/" for 127.0.0.1 at 2024-08-28 12:25:15 -0600
ActiveRecord::SchemaMigration Load (0.1ms) SELECT "schema_migrations"."version" FROM "schema_migrations" ORDER BY "schema_migrations"."version" ASC
Processing by ApplicationController#txt as HTML
Rendering html template
Rendered html template (Duration: 0.4ms | GC: 0.1ms)
Completed 200 OK in 9ms (Views: 2.1ms | ActiveRecord: 0.0ms (0 queries, 0 cached) | GC: 0.3ms)
```