

Setup

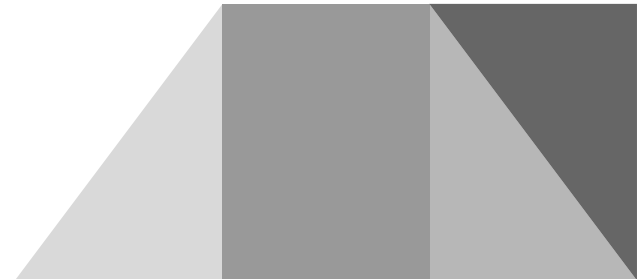
IIC2143 Ingeniería de Software - Ayudantía 1

¿Qué sistema operativo usas?

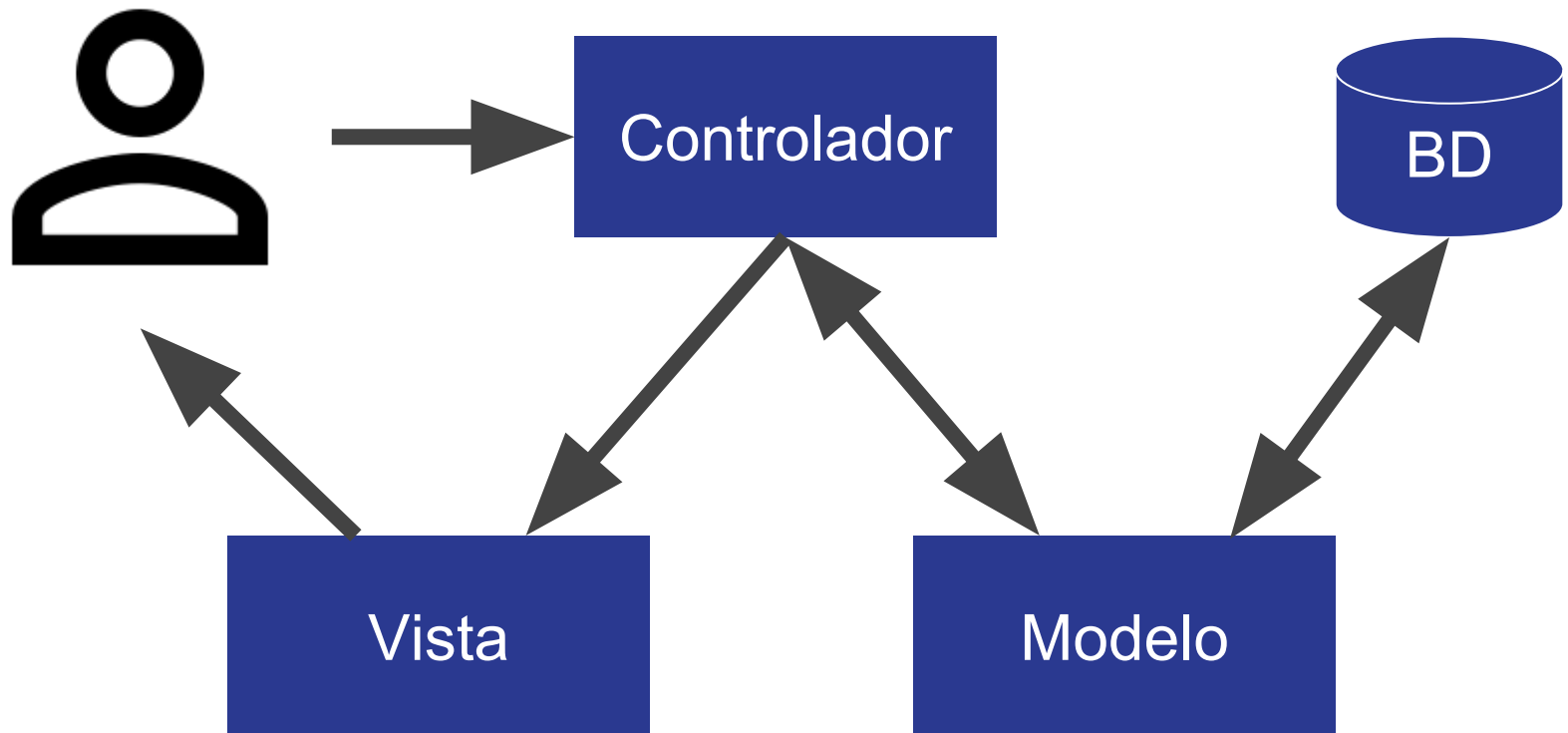
<http://bit.ly/SoftwareAY01>



¿En qué programaremos?

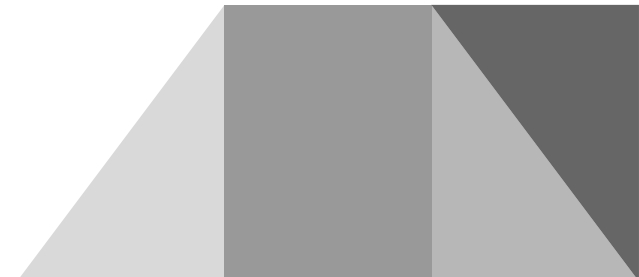


Modelo-Vista-Controlador (MVC)



¿Qué es un ambiente?

- Corresponde a la configuración del sistema en el que se ejecuta una aplicación (versiones de programas, sistema operativo, etc).
- Existen dos principales:
 - **Desarrollo:** normalmente, su computador
 - **Producción:** servidor de la aplicación real

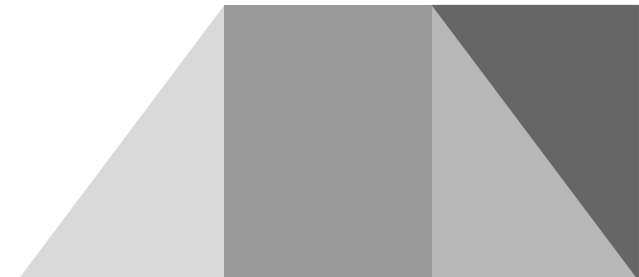


¿Por qué importan?

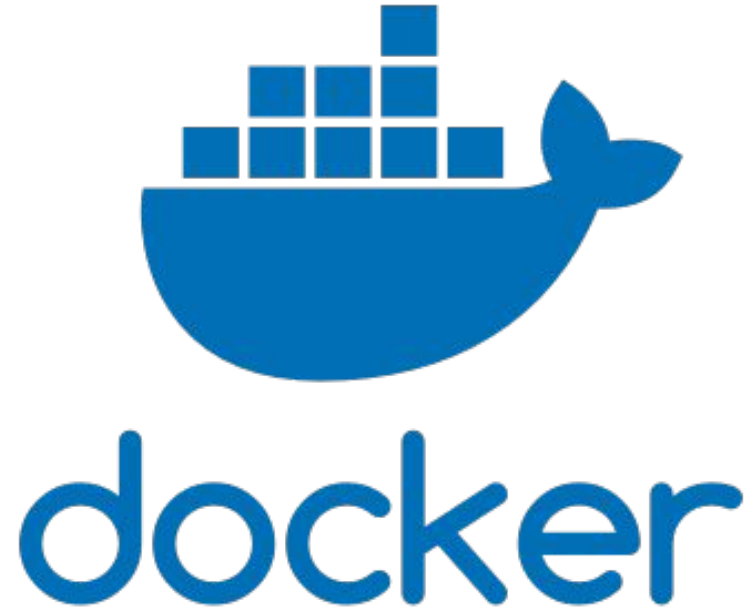
“¡Pero si funcionaba en mi computador!”



Buscamos que el ambiente de desarrollo
sea igual al de producción



¿Cómo?




Usaremos estos archivos de configuración







IIC2143-2018-1 / **project** Watch 10 Unstar 2 Fork 1

[Code](#) [Issues 0](#) [Pull requests 0](#) [Projects 0](#) [Wiki](#) [Insights](#)

Branch: master ▾ **project / example /** Create new file Upload files Find file History

 **rasaffie** Add example of docker + RoR Latest commit 1f90544 5 days ago


..

 Dockerfile	Add example of docker + RoR	5 days ago
 Gemfile	Add example of docker + RoR	5 days ago
 Gemfile.lock	Add example of docker + RoR	5 days ago
 README.md	Add example of docker + RoR	5 days ago
 docker-compose.yml	Add example of docker + RoR	5 days ago
 dockerignore	Add example of docker + RoR	5 days ago

<https://github.com/IIC2143-2018-1/project/tree/master/example>

Dockerfile

```
1. FROM ruby:2.5.0
2. ENV LANG C.UTF-8
3. RUN apt-get update -qq && apt-get install -y
   build-essential libpq-dev
   software-properties-common
4. [...]
5. ENV APP_HOME /example
6. RUN mkdir $APP_HOME
7. WORKDIR $APP_HOME
8. ADD Gemfile $APP_HOME/Gemfile
9. ADD Gemfile.lock $APP_HOME/Gemfile.lock
10. RUN bundle install --jobs=3 --retry=3
11. ADD . $APP_HOME
```



`docker-compose.yml`

```
1.  version: '3'
2.  services:
3.    postgres:
4.      image:
5.        postgres:10.3
6.      ports:
7.        - "5432"
8.    web:
9.      build: .
10.     env_file:
11.       - .env
```

```
11.     command: bash -c
12.       "(bundle check ||
13.        bundle install) &&
14.        bundle exec rails s -p
15.        3000 -b '0.0.0.0'"
16.     volumes:
17.       - ./example
18.     ports:
19.       - "3000:3000"
20.     depends_on:
21.       - postgres
```



¡Hagamos nuestra
aplicación!

Aplicación Rails en Docker

1. Abre Docker (daemon)
2. Muévete al directorio de tu proyecto
3. Copia los archivos del proyecto de ejemplo (<https://github.com/IIC2143-2018-1/project/tree/master/example>):
 - a. Dockerfile
 - b. docker-compose.yml
 - c. Gemfile
 - d. Gemfile.lock
 - e. dockerignore
4. Crea un archivo de variables de entorno vacío

```
touch .env
```

Aplicación Rails en Docker

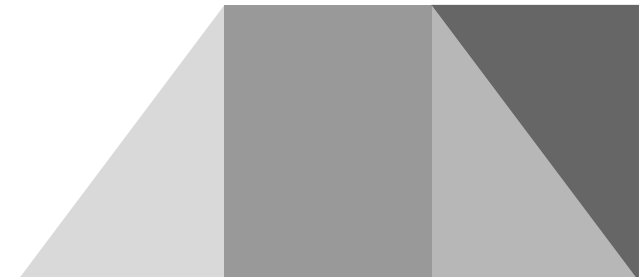
5. Crear proyecto de Ruby on Rails (RoR)

```
docker-compose run web rails new . --force  
--database=postgresql
```

```
docker-compose build
```

6. Agregar .env al .gitignore

```
echo ".env" >> .gitignore
```



Aplicación Rails en Docker

7. Conectar la app RoR a la base de datos Postgres

- a. Abrir `config/database.yml`
- b. Agregar en la sección `default` un `host` con el nombre de la imagen de Docker con Postgres

```
host: postgres
username: postgres
password:
```

- c. Levantar los contenedores de Docker

```
docker-compose up
```

- d. Crear base de datos

```
docker-compose run web rake db:create
```



Aplicación Rails en Docker

8. Entra a la aplicación en <http://localhost:3000>



Yay! You're on Rails!



Rails version: 5.1.5

Ruby version: 2.5.0 (x86_64-linux)

Aplicación Rails en Docker (futuro)

9. Crea un controlador de bienvenida (veremos esto en detalle en las siguientes ayudantías)

```
docker-compose run web rails generate controller welcome index
```

10. Escribe “Hello World” en la vista `app/views/welcome/index.html.erb`

```
echo "<h1>Hello World</h1>" >  
app/views/welcome/index.html.erb
```

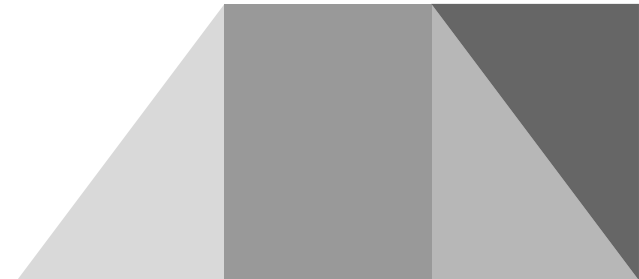
11. Verifica que el archivo `config/routes.rb` quede así

```
Rails.application.routes.draw do  
  # For details on the DSL available within this file, see  
  http://guides.rubyonrails.org/routing.html  
  root 'welcome#index'  
end
```


Aplicación Rails en Docker


12. Entra a la aplicación en <http://localhost:3000>

Hello World



¿Cuál será nuestro ambiente de producción?

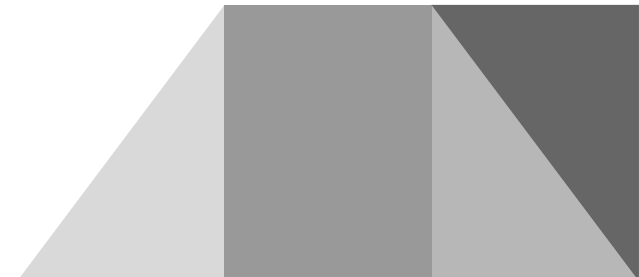




¡Despleguemos nuestra
aplicación!

Despliegue en Heroku

1. Crea una aplicación nueva desde tu dashboard
2. Una vez creada, en la pestaña “Deploy” (o despliegue), puedes conectar tu aplicación de dos formas
 - a. Conectar a un repositorio de GitHub (conectando tu cuenta)
 - b. Agregar el repositorio remoto de Heroku al repositorio de git de tu proyecto (recomendada)



Documentación útil

- <https://docs.docker.com/>
- <https://docs.docker.com/compose/rails/>
- <https://devcenter.heroku.com/articles/getting-started-with-rails4>
- Pequeño curso de Docker:
 - <https://www.codeschool.com/courses/try-docker>

