Ayudantía 1

Rails

1. Explicar MVC (Modelo-Vista-Controlador)
2. Introducción a Web (?)

¿Qué es un ambiente?

-
- Configuración general en el que se corre una aplicación
    - Versiones
    - Programas
    - Sistema operativo
    - Variables de entorno (ambiente)

Docker

1. Abrir Docker (aplicación)
2. Configurar el proyecto
    a. `Dockerfile`

```
FROM ruby:2.5.0
ENV LANG C.UTF-8
RUN apt-get update -qq && apt-get install -y build-essential libpq-dev
software-properties-common
# Node.js
RUN curl -sL https://deb.nodesource.com/setup_8.x | bash - \
   && apt-get install -y nodejs
# yarn
RUN curl -sS https://dl.yarnpkg.com/debian/pubkey.gpg | apt-key add -\
   && echo "deb https://dl.yarnpkg.com/debian/ stable main" | tee
/etc/apt/sources.list.d/yarn.list \
   && apt-get update \
   && apt-get install -y yarn
ENV APP_HOME /example
RUN mkdir $APP_HOME
WORKDIR $APP_HOME
ADD Gemfile $APP_HOME/Gemfile
ADD Gemfile.lock $APP_HOME/Gemfile.lock
RUN bundle install --jobs=3 --retry=3
ADD . $APP_HOME
```

    b. `docker-compose.yml`

```
version: '3'
services:
 postgres:
   image: postgres:10.3
   ports:
     - "5432"
 web:
   build: .
   env_file:
     - .env
```

```
    command: bash -c "(bundle check || bundle install) && bundle exec rails s -p 3000 -b
'0.0.0.0'"
    volumes:
      - .:/example
    ports:
      - "3000:3000"
    depends_on:
      - postgres
```

c. `Gemfile`

```
source 'https://rubygems.org'
gem 'rails', '5.1.5'
```

d. `Gemfile.lock` (Archivo vacío)

e. Crear archivo de variables de entorno

```
touch .env
```

3. Crear proyecto de Rails

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

```
docker-compose build
```

4. Agregar `.env` al `.gitignore`
5. Conectar base de datos
   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:
```

Ver imágenes con `docker-compose ps`

   c. Levantar los servicios de docker

```
docker-compose up -d
```

   d. Crear base de datos

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

6. “Yay! You’re on Rails” (`http://localhost:3000`)
7. Apagar aplicación

```
docker-compose stop
```

Heroku

1. Agregar modelo de prueba (se explicará más adelante)

   ```
   docker-compose exec web rails generate scaffold Article title:string content:text
   ```

2. Correr migración

   ```
   docker-compose exec web rake db:migrate
   ```

3. Crear aplicación de Heroku
   a. Diferencia entre app y pipeline: Pipeline es un conjunto de aplicaciones de heroku independientes, pero con el mismo codebase (código base). Sirve para agrupar aplicaciones de producción y staging.
   b. Pueden crear un pipeline, pero que lo piensen
4. Elegir deployment method:
   a. GitHub apuntando a su rama de producción => simple, pero que se preocupen de revisar que los deploys se estén ejecutando
   b. Desde Heroku CLI => seguir instrucciones desde heroku

Docker

8. Abrir Docker (aplicación)
9. Configurar el proyecto
    a. Crear archivo de variables de entorno

       ```
       touch .env
       ```

10. Crear proyecto de Rails

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

    ```
    docker-compose build
    ```

11. Agregar `.env` al `.gitignore`
12. Conectar base de datos
    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:
       ```

       Ver imágenes con `docker-compose ps`

    c. Levantar los servicios de docker

       ```
       docker-compose up
       ```

    d. Crear base de datos

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

13. “Yay! You’re on Rails” (`http://localhost:3000`)