# Página web

## Instalación de python (CMD)

Bajamos la versión de python que queremos y la guardamos en la carpeta una carpeta, en este caso Python38

En mi ruta, por ejemplo

* cd C:\Entorno\_Desarollo\Python38

Entramos en la carpeta y vemos un ejecutable llamado "python", copiamos ese ejecutable y luego cambiamos el de la copia por el nombre del ejecutable por python3.8

Ejecutamos el python desde el cmd como python3.8

* python3.8

## Entorno virtual (CMD)

### Creación de un nuevo entorno virtual

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK
* python -m venv env\_BACK

### Activar entorno virtual

* + cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\env\_BACK\Scripts
* activate

## Git (BASH GIT, segundo botón una vez instalado git)

### Instalación de git

Me bajo git de su página web

Veo si está bien instalado

* git --version

### Creación nuevo proyecto git

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO
  1. Me voy a la página de git y me creo un repositorio, en este caso lo he llamado EY\_WEBPAGE

#### Comandos

Abriendo la terminal de bash con el segundo botón en la ruta de arriba:

1. echo "# EY\_WEBPAGE" >> README.md
2. git init
3. git add .
4. git commit -m "commit primero"
5. git branch -M master
6. git remote add origin https://github.com/slopezrap/EY\_AUTOMATIC\_AUDIT.git
7. git push -u origin master

### Actualización de proyectos git

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO

#### Comandos

Abriendo la terminal de bash con el segundo botón en la ruta de arriba:

1. git init
2. git add .
3. git commit -m "commit X"
4. git branch -M master
5. git remote add origin https://github.com/slopezrap/EY\_AUTOMATIC\_AUDIT.git
6. git push -u origin master

## Front (Angular)

### **Instalación de Angular (CMD)**

* 1. *Instalar NodeJS*
* Te vas a la página y te lo descargas
  1. *Ver si está bien instalado NodeJS y ver su versión*
* npm -v
* node -v
  1. *Instalar angularCLI*
* npm install -g @angular/cli
  1. *Ver si está bien instalado angularCLI*
* ng v

### **Creación del workspace (CMD)**

#### Explicación

Es el espacio donde colgaran todas las aplicaciones

#### Comandos

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT
* ng new WS-ANGULAR --createApplication="false"

**NOTA**: Borrar el fichero “.gitignore”, y la carpeta oculta “.git”

### **Creación de aplicacione (CMD)**

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR\

#### Aplicación Inicio

##### Explicación

Es una aplicación en la cual existe la página de inicio y tiene la navbar

##### Comandos usados

###### Generación de la aplicación (automatic audit proces)

* ng generate application APP-AAP

###### Componentes

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR\projects\app-aap\src\app
* ng g c components/home
* ng g c components/hero
* ng g c components/navbar
* ng g c components/page404

###### Lanzamiento servidor

* ng serve

### **Instalación de dependencias (CMD)**

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR

#### Dependencias

##### Bootstrap, jquery, Popper.js

(<https://www.youtube.com/watch?v=68XABZdEnwM&list=PL_9MDdjVuFjGJm9hBCwIb2nkLYIf83o2A&index=4>)

Es una aplicación en la cual existe la página de inicio y tiene la navbar

##### Comandos usados

###### Comando para instalar 3 dependencias

* npm install bootstrap jquery popper.js

###### Comando para usar las librerías de Electron

* npm install ngx-electron --save

### **Proceso en Angular**

Aquí se mostrará el proceso de angular

El paso 0, es que angular se va al fichero app.component.html para ver que muestra, aquí es donde estará la navbar y luego todas las rutas que estarán en el fichero app-routing.module.ts

* 1. Usuario escribe url, por ejemplo http://localhost:4200/
     + Lo que hace angular aquí es irse al fichero “app-routing.module.ts” que es el fichero donde están todas las rutas. En este caso se iría al path:''

|  |
| --- |
| { path:'', component: HomeComponent }, |

* 1. Fichero “app-routing.module.ts”
     + En este fichero están todas las rutas y cuando encuentra el primer positivo de una ruta lo renderiza, por eso es muy importante el orden, renderiza un componente que le decimos, en el caso anterior el componente a renderizar sería HomeComponent

|  |
| --- |
| { path:'', component: HomeComponent }, |

* 1. Fichero “XXX.component.ts”, aquí está la lógica del componente. Y como verás lo que hace es llamar al html.

## Back (Django)

### **Instalación de Django (CMD)**

Activamos el entorno virtual, en nuestro caso:

* + cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\env\_BACK\Scripts
* activate
  1. *Instalación de Django (uso esta versión de django porque ahora es estable con pyinstaller)*
* pip install django==2.1.8
  1. *Instalación de DjangoRestFramework*
* pip install djangorestframework==3.11

### **Creación del workspace (CMD)**

#### Explicación

Es el espacio donde colgaran todas las aplicaciones

#### Comandos

En mi ruta, por ejemplo

* + cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK
* django-admin startproject WS\_DJANGO

### **Creación de aplicaciones (CMD)**

En mi ruta, por ejemplo y con el entorno activado

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\WS\_DJANGO

#### Aplicación Inicio

##### Explicación

Es una aplicación en la cual existe la página de inicio y tiene la navbar

##### Comandos usados

###### Generación de la aplicación

* python manage.py startapp APP\_INICIO

###### Lanzamiento servidor

* python manage.py migrate
* python manage.py makemigrations
* python manage.py createsuperuser (pass 123456)
* python manage.py runserver

### **Empaquetado proyecto django y creación de .exe (CMD) , ABRIR CMD COMO ADMINISTRADOR**

En mi ruta, por ejemplo y con el entorno activado

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\WS\_DJANGO

#### Instalación de pyinstaller

##### Explicación

Programa que empaqueta Python en .exe

Pagina donde ver como se hace un ejecutable de django con pyinstaller: https://github.com/pyinstaller/pyinstaller/wiki/Recipe-Executable-From-Django

* + pip install pyinstaller
  + cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\WS\_DJANGO
  + pyinstaller --name=WS\_DJANGO C:/Users/slope/OneDrive/Escritorio/EY/EY\_ANGULAR-ELECTRON-DJANGO/PROYECTO\_WEB/BACK/WS\_DJANGO/manage.py

**NOTA**: Si da algún error y se para nos vamos al entorno y vamos a env\_BACK\Lib\site-packages\PyInstaller\utils\hooks\subproc y nos metemos en el fichero django\_import\_finder.py y cambiamos la varaible "hiddenimports"

-------------------------------------------------------------------------------------------------------------------

hiddenimports = list("settings.INSTALLED\_APPS") + \

list("settings.TEMPLATE\_CONTEXT\_PROCESSORS") + \

list("settings.TEMPLATE\_LOADERS") + \

["settings.ROOT\_URLCONF"]

-------------------------------------------------------------------------------------------------------------------

por

-------------------------------------------------------------------------------------------------------------------

hiddenimports = list(getattr(settings, "INSTALLED\_APPS", [])) + \

list(getattr(settings, "TEMPLATE\_CONTEXT\_PROCESSORS", [])) + \

list(getattr(settings, "TEMPLATE\_LOADERS", [])) + \

[getattr(settings, "ROOT\_URLCONF", "")]

-------------------------------------------------------------------------------------------------------------------

##### Para ver que está bien

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\BACK\WS\_DJANGO\dist\WS\_DJANGO
* WS\_DJANGO.exe runserver localhost:8000

## Electron

### **Instalación de Electron (CMD)**

(https://www.geeksforgeeks.org/integrate-angular-7-with-electronjs/)

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR
* npm install electron -save-dev
* npm install ngx-electron --save

### **Unión de Angular y Django con Electron (CMD)**

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR\

Creo un fichero:

* Fichero: main.js : fichero que crea un gui de electron por como está configurado

|  |
| --- |
| main.js |
| const electron = require('electron')      // Module to control application life.  const app = electron.app      // Module to create native browser window.  const BrowserWindow = electron.BrowserWindow  const path = require('path')  const url = require('url')      // Keep a global reference of the window object, if you don't, the window will      // be closed automatically when the JavaScript object is garbage collected.  let mainWindow  function createWindow() {      const path2 = "C:/Users/slope/OneDrive/Escritorio/EY/EY\_ANGULAR-ELECTRON-DJANGO/PROYECTO\_WEB/BACK/WS\_DJANGO/dist/WS\_DJANGO/"      console.log(path2)        var subpy = require('child\_process').spawn(path2+'WS\_DJANGO.exe', ['runserver', 'localhost:8000']);      var mainAddr = 'http://localhost:8000';      var openWindow = function() {          mainWindow = new BrowserWindow({              width: 800,              height: 600,              webPreferences: {                  nodeIntegration: false                }              })          //mainWindow.loadURL(          //    url.format({          //        pathname: "C:/Users/slope/OneDrive/Escritorio/EY/EY\_ANGULAR-ELECTRON-DJANGO/PROYECTO\_WEB/FRONT/WS-ANGULAR/projects/app-inicio/src/index.html",          //        protocol: "file:",          //        slashes: true          //    })          //);          mainWindow.loadFile('dist/ang-electron/index.html');          // Open the DevTools.          mainWindow.on('closed', function() {              mainWindow = null;              subpy.kill('SIGINT');          })      }      var startUp = function() {                  openWindow();      };      startUp();  }  // This method will be called when Electron has finished  // initialization and is ready to create browser windows.  // Some APIs can only be used after this event occurs.  app.on('ready', createWindow)      // Quit when all windows are closed.  app.on('window-all-closed', function() {      // On OS X it is common for applications and their menu bar      // to stay active until the user quits explicitly with Cmd + Q      if (process.platform !== 'darwin') {          app.quit()      }  })  app.on('activate', function() {          // On OS X it's common to re-create a window in the app when the          // dock icon is clicked and there are no other windows open.          if (mainWindow === null) {              createWindow()          }      }) |

Modificar un fichero:

* Fichero: package.json : fichero que se ejecuta al llamar a electron y ejecuta el main.ts

|  |
| --- |
| package.json |
| {    "name": "ws-angular",    "version": "0.0.0",    "main": "main.js",    "scripts": {      "ng": "ng",      "start": "ng serve",      "build": "ng build",      "test": "ng test",      "lint": "ng lint",      "e2e": "ng e2e",      "electron": "ng build && electron ."    },    "private": true,    "dependencies": {... |

En el fichero “app.module.ts” de angular hay que añadir esto

|  |
| --- |
| app.module.ts |
| import { NgxElectronModule } from 'ngx-electron';    imports: [      NgxElectronModule |

En el fichero “index.html” cambiar href

|  |
| --- |
| index.html |
| <base href="/">  por  <'base href="./"> |

### **Correr Electron (CMD)**

En mi ruta, por ejemplo

* cd C:\Users\slope\OneDrive\Escritorio\EY\EY\_ANGULAR-ELECTRON-DJANGO\PROYECTO\_WEB\FRONT\WS-ANGULAR\

Corro electron con este comando, usar el cmd como admin.

* npm run electron