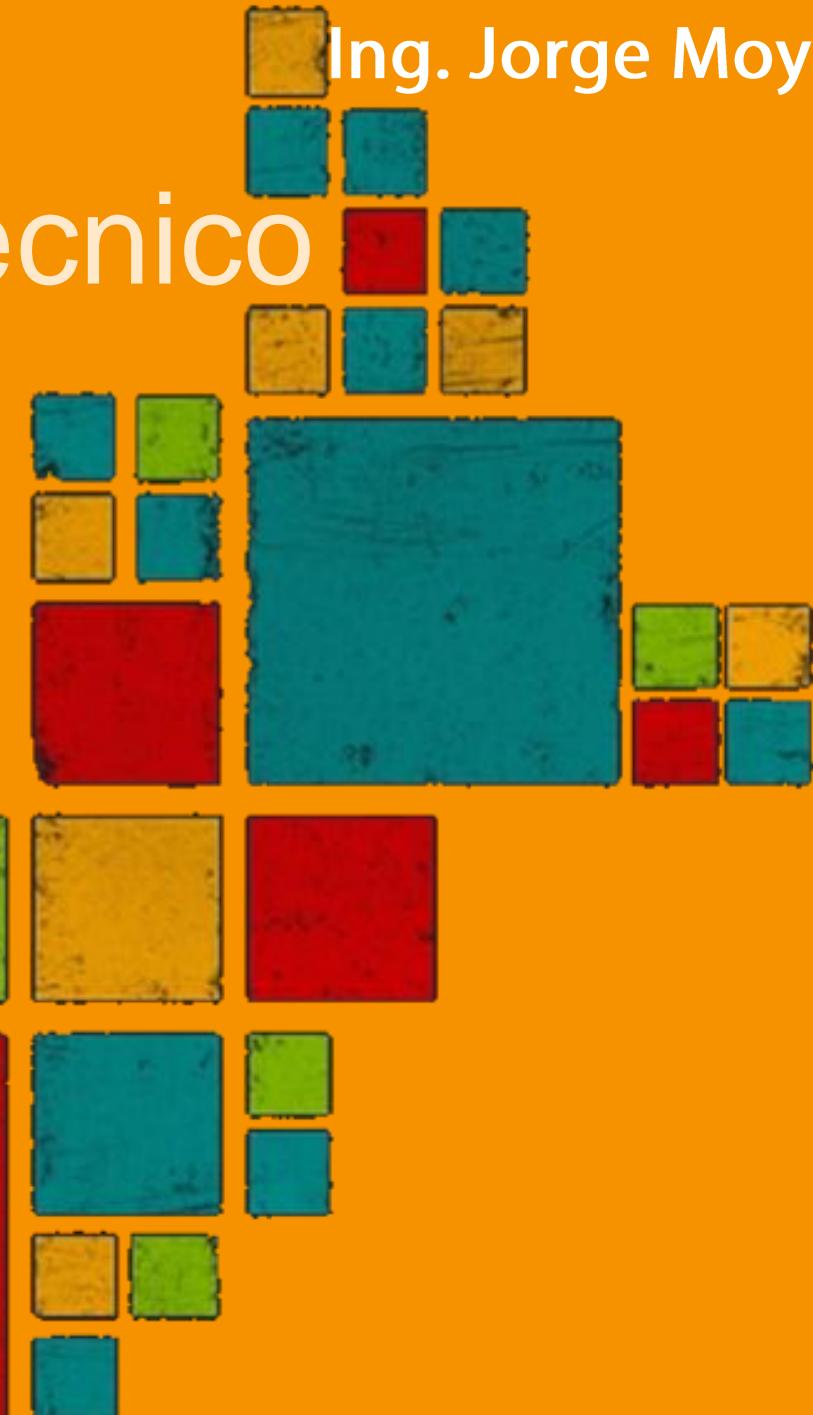


Técnicas de Simulación

Ing. Jorge Moya

Manual Técnico



Sánchez Cobeña Cristóbal Andrés

Octavo Nivel “A”



Manual Técnico

Contenido

Manual técnico.....	3
Librerías y Versiones Utilizadas:	3
Descargas e instalación:	4
Python 2.7.13.....	4
Librerías:	4
Herramientas.....	5
CSS3.....	5
Javascript.....	6
Herramientas utilizadas y como usarlas.....	7
¿Cómo Iniciar este proyecto en su equipo con PyCharm?	8
Primeros pasos con HEROKU	14
¿Como subirlo a un Servidor (Heroku)?	15
¿Como modificar nuestro proyecto subido en Heroku?	19



Manual técnico

El presente manual tiene como finalidad describir la instalación y la ejecución del proyecto realizado para el sistema web de simulación, la siguiente aplicación web fue desarrollada con Python 2.7.13, la librería principal es Django.

Front-end: Se utilizó HTML:5, Librerías CSS3, Librerías JavaScript y plot.ly

Back-end: Python y Django

Librerías y Versiones Utilizadas:

- Python 2.7.13
 - Librerías de Python:
 - Django 1.11.2
 - Numpy 1.31.1
 - Scipy 0.19.1
 - PIP
 - whitenoise
- CSS 3
 - Librerías CSS
 - Bootstrap 4.0
 - ValidationEngine.jquery.css
 - Styles Personalizados
- JavaScript
 - Librerías JavaScript
 - Bootstrap 4.0
 - Bootstrap Validate
 - JQuery.validate
 - JQuery-3.2.1
 - Plotly-latest
 - validaciones
 - Funciones Personalizadas
- HTML 5
- Heroku CI (servidor Web)



Descargas e instalación:

Python 2.7.13

```
https://www.python.org/downloads/release/python-2713/
```

Librerías:

PIP:

Ejecutar en Consola: `python get-pip.py`

DJANGO: <https://www.djangoproject.com/download/>

Opcion1:

```
pip install Django==1.11.2
```

Opción 2:

```
git clone https://github.com/django/django.git
```

NUMPY:

Opcion1:

```
pip install numpy
```

Opcion2:

```
git clone https://github.com/numpy/numpy.git numpy
```

SCIPY:

Opcion1:

```
pip install scipy
```



Opcion2:

```
git clone https://github.com/scipy/scipy.git scipy
```

WHITENOISE:

```
pip install whitenoise
```

PIPENV:

```
pip install pipenv
```

Herramientas.

PYCHARM 2017.2 IDE

<https://www.jetbrains.com/pycharm/download/#section=windows>

HEROKU CLI

<https://devcenter.heroku.com/articles/heroku-cli#download-and-install>

CSS3

Bootstrap 4.0

<http://getbootstrap.com/docs/4.0/getting-started/download/>

TIPO DE LETRA

<https://fonts.googleapis.com/css?family=Montserrat>

Styles Personalizados

Los archivos de estilos (css) personalizados se encuentran en el proyecto:

- * bootstrap.css
- * bootstrap-theme.css
- * estilo.css
- * multi-step-form.css

```
* style.css  
* validationEngine.jquery.css
```

Javascript

Bootstrap 4.0

<http://getbootstrap.com/docs/4.0/getting-started/download/>

JQuery-3.2.1

<https://jquery.com/download/>

Plotly-latest

<https://cdn.plot.ly/plotly-latest.min.js>

Funciones Personalizadas

Los archivos de JavaScript (.js) personalizados y externos se encuentran en el proyecto:

```
* Bootstrap Validate  
* funciones.js  
* jquery.validate.min.js  
* jquery.validationEngine-2.6.2.js  
* knockout-min.js  
* multi-step-form.js  
* npm.js  
* valid.js
```



Herramientas utilizadas y como usarlas

PYCHARM 2017.2 IDE

Instrucciones de instalación

WINDOWS:

1. Ejecute el archivo pycharm-2017.2.3.exe que inicia el Asistente para la instalación.
2. Siga todos los pasos sugeridos por el asistente. Preste especial atención a las opciones de instalación correspondientes.

MAC OS:

1. Descargar el archivo de imagen de disco pycharm-2017.2.3.dmg macos
2. Montarlo como otro disco en su sistema.
3. Copie PyCharm en su carpeta de aplicaciones.

LINUX:

1. Copie el pycharm-2017.2.3.tar.gz en la ubicación de instalación deseada (asegúrese de tener permisos rw para ese directorio)
2. Descomprima el pycharm-2017.2.3.tar.gz utilizando el siguiente comando:
`tar -xzf pycharm-2017.2.3.tar.gz`
3. Quitar el pycharm-2017.2.3.tar.gz para ahorrar espacio en disco (opcional).
4. Ejecutar pycharm.sh desde el subdirectorio bin.

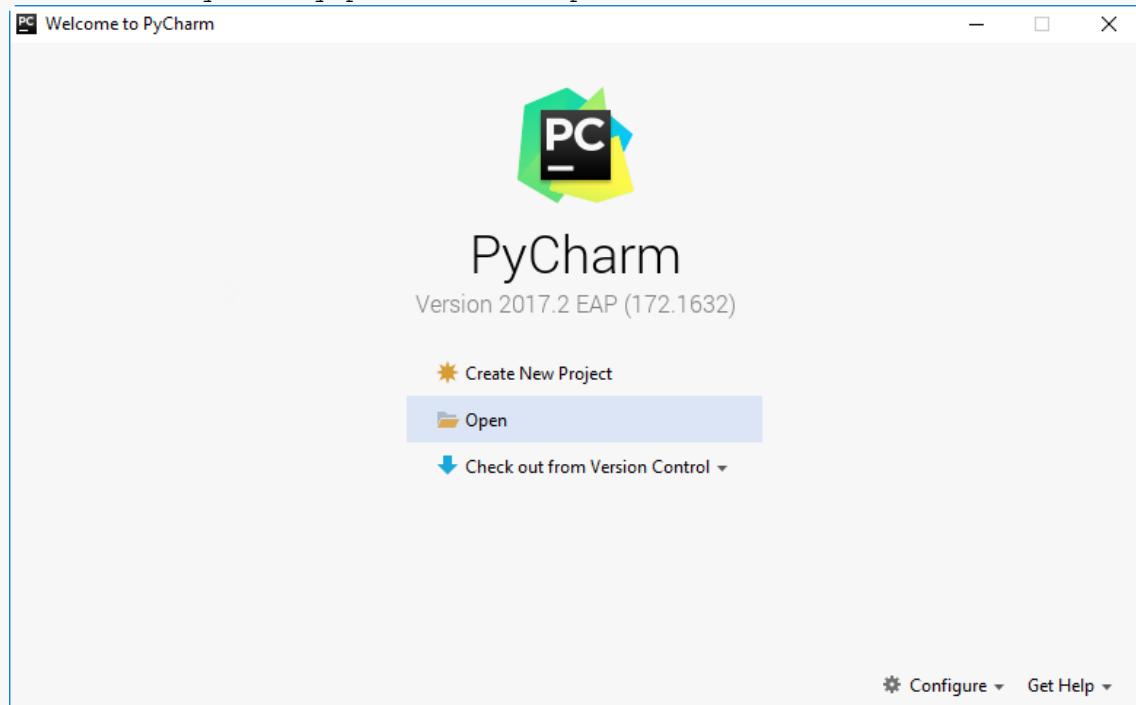
PARA REALIZAR UN PROYECTO POR FAVOR REVISAR LA DOCUMENTACION OFICIAL DE PyCharm

<https://www.jetbrains.com/pycharm/documentation/>

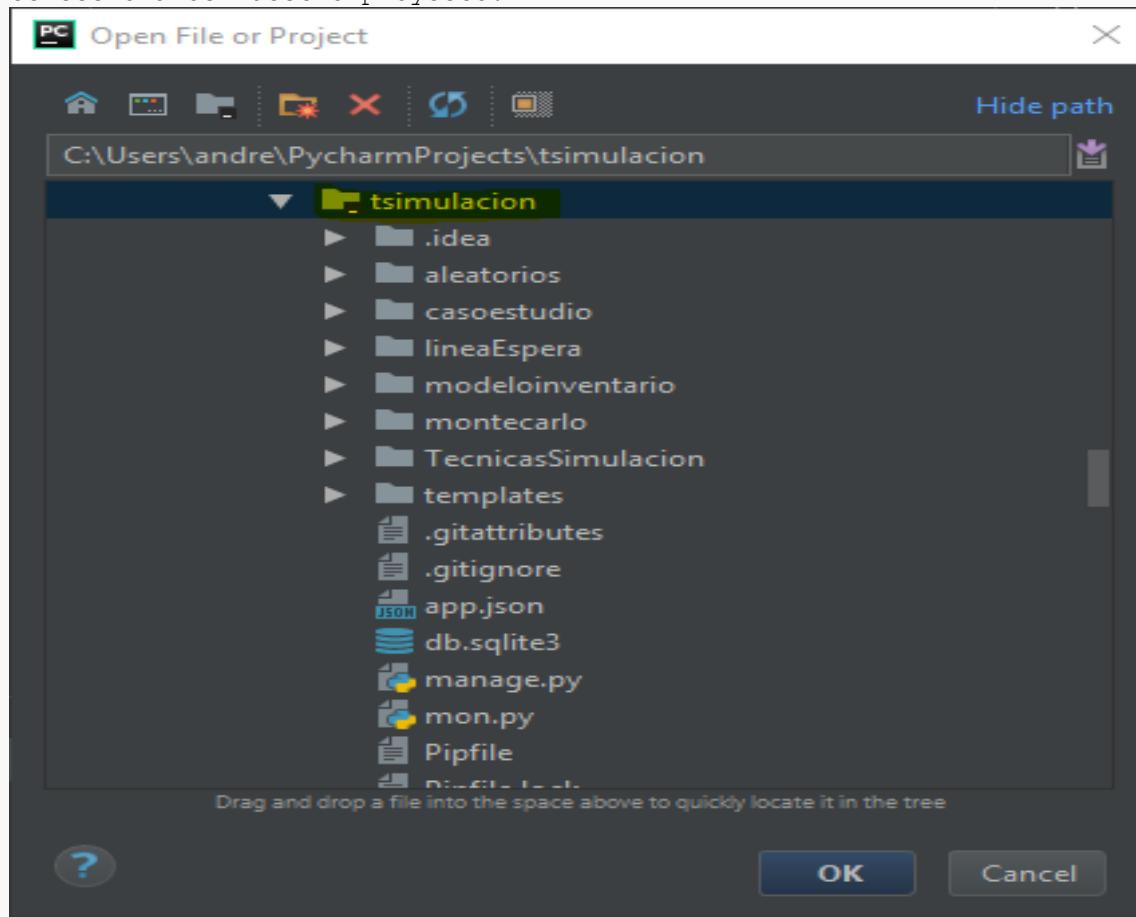


¿Cómo Iniciar este proyecto en su equipo con PyCharm?

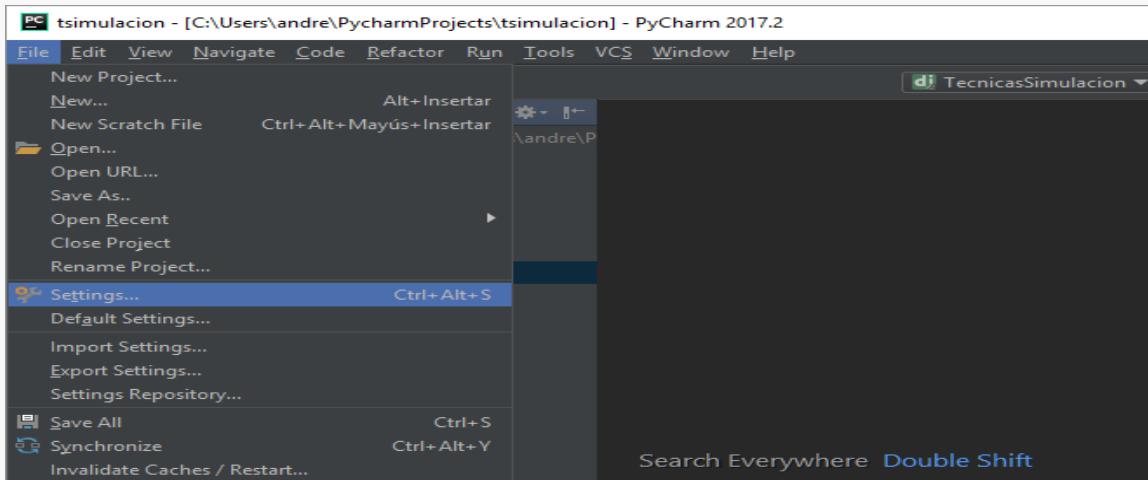
1. Iniciar PyCharm y presionar en Open.



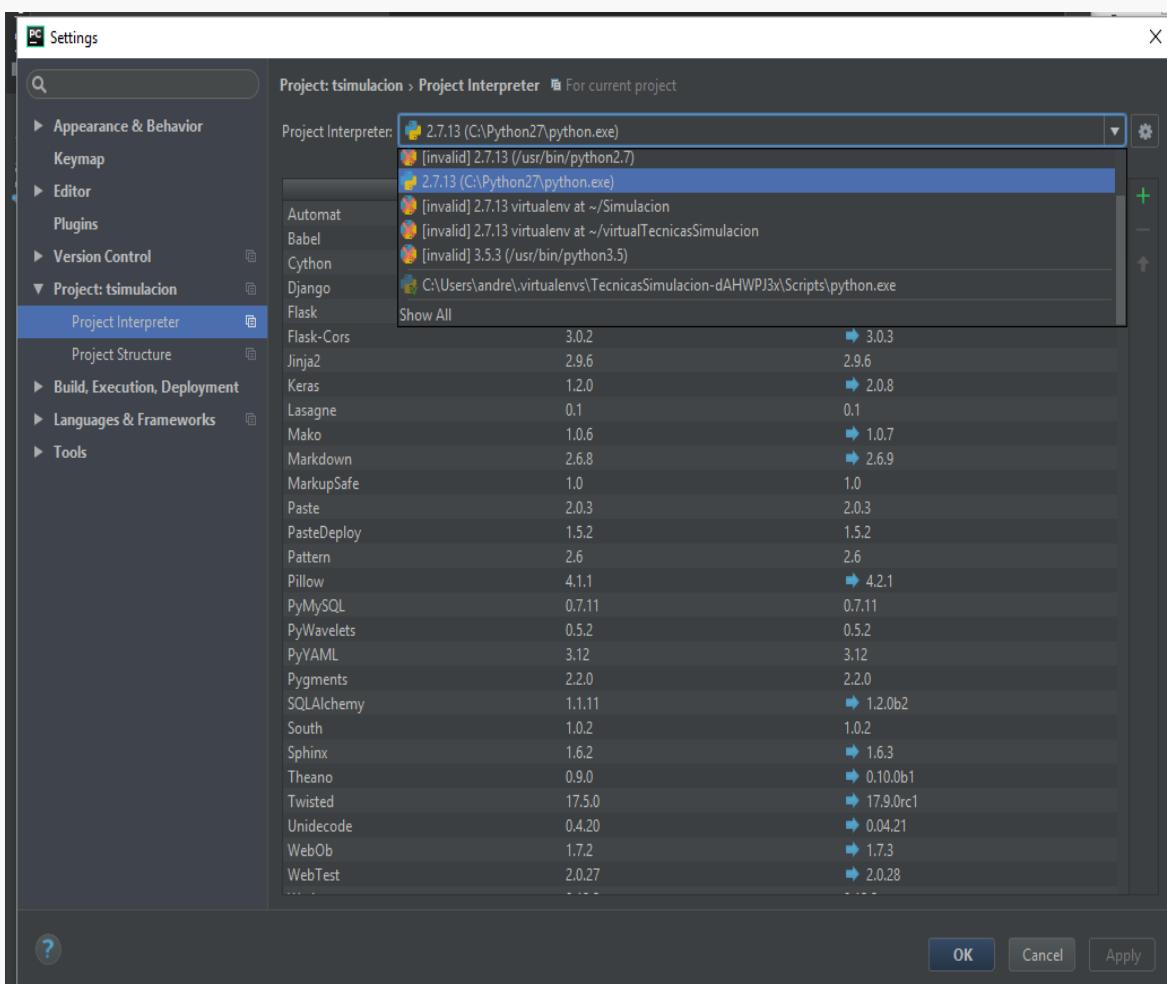
2. Seleccionamos nuestro proyecto.



3. Presionamos Ctrl + Alt + S o en File - Settings



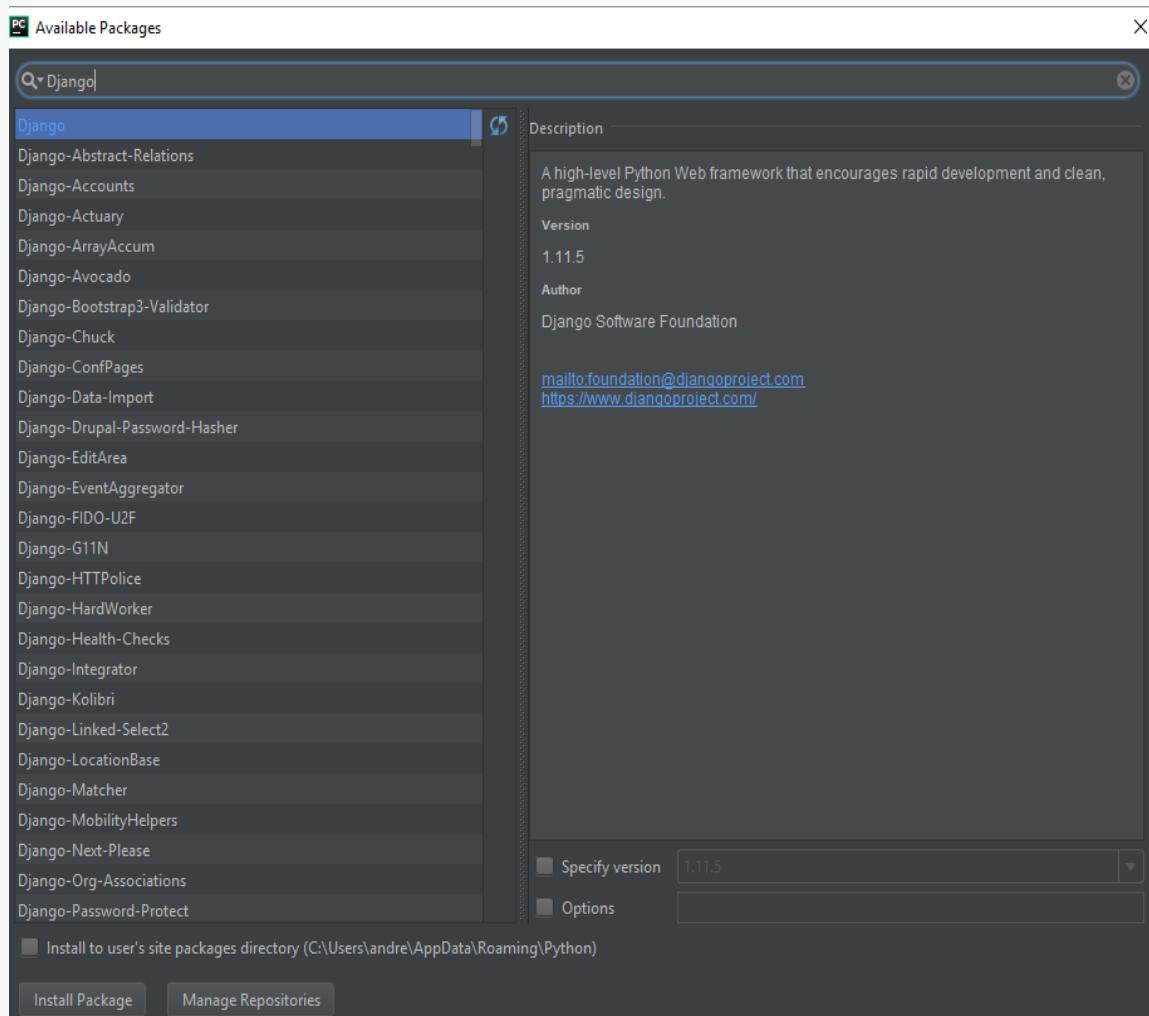
4. Nos ubicamos en Project > Project Interpreter y seleccionamos nuestro interprete de Python o creamos un Entorno Virtual.



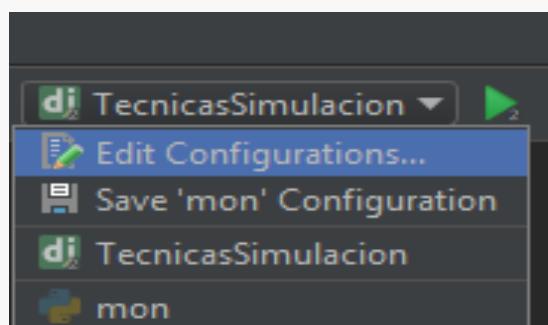
5. Presionamos en el Signo de + para agregar librerías.

6. Buscamos y Seleccionamos todas las Librerías necesarias en nuestro proyecto y presionamos en Install Package.

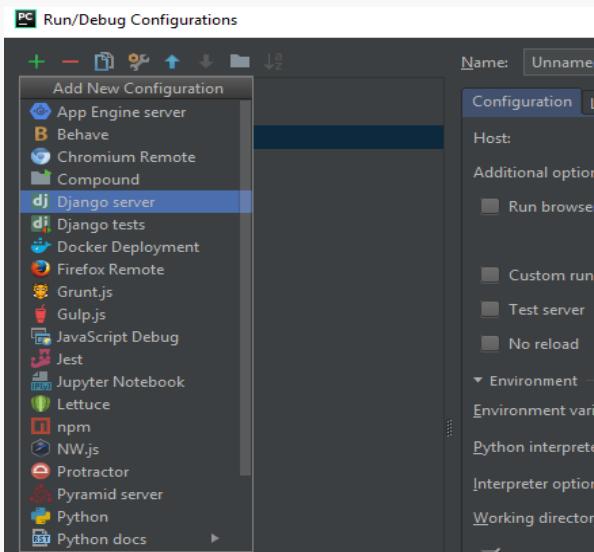
Django - Numpy - Scipy - Whitenoise - Pipenv



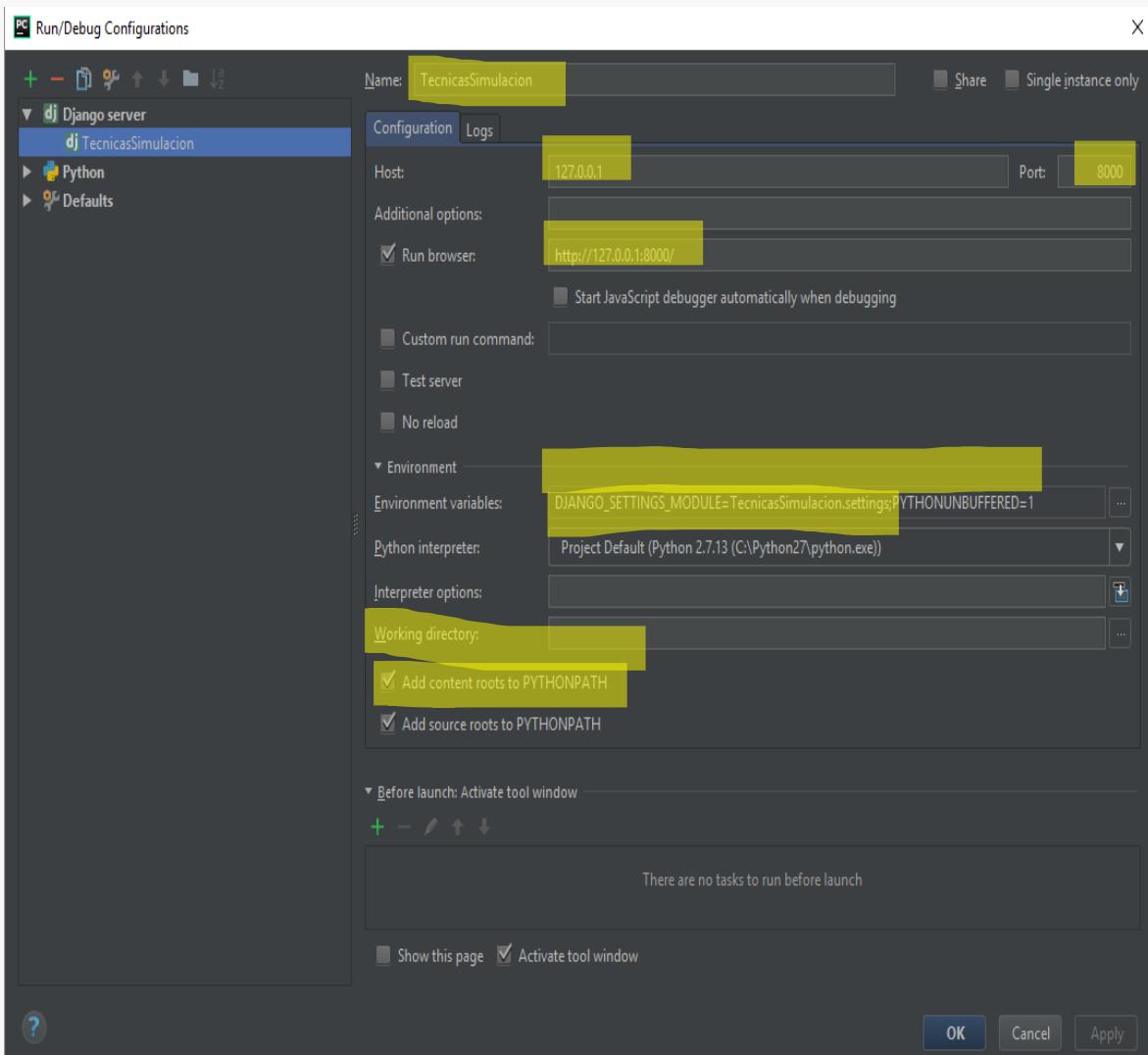
7. Cerramos el instalador de librerias y editamos las Configuraciones



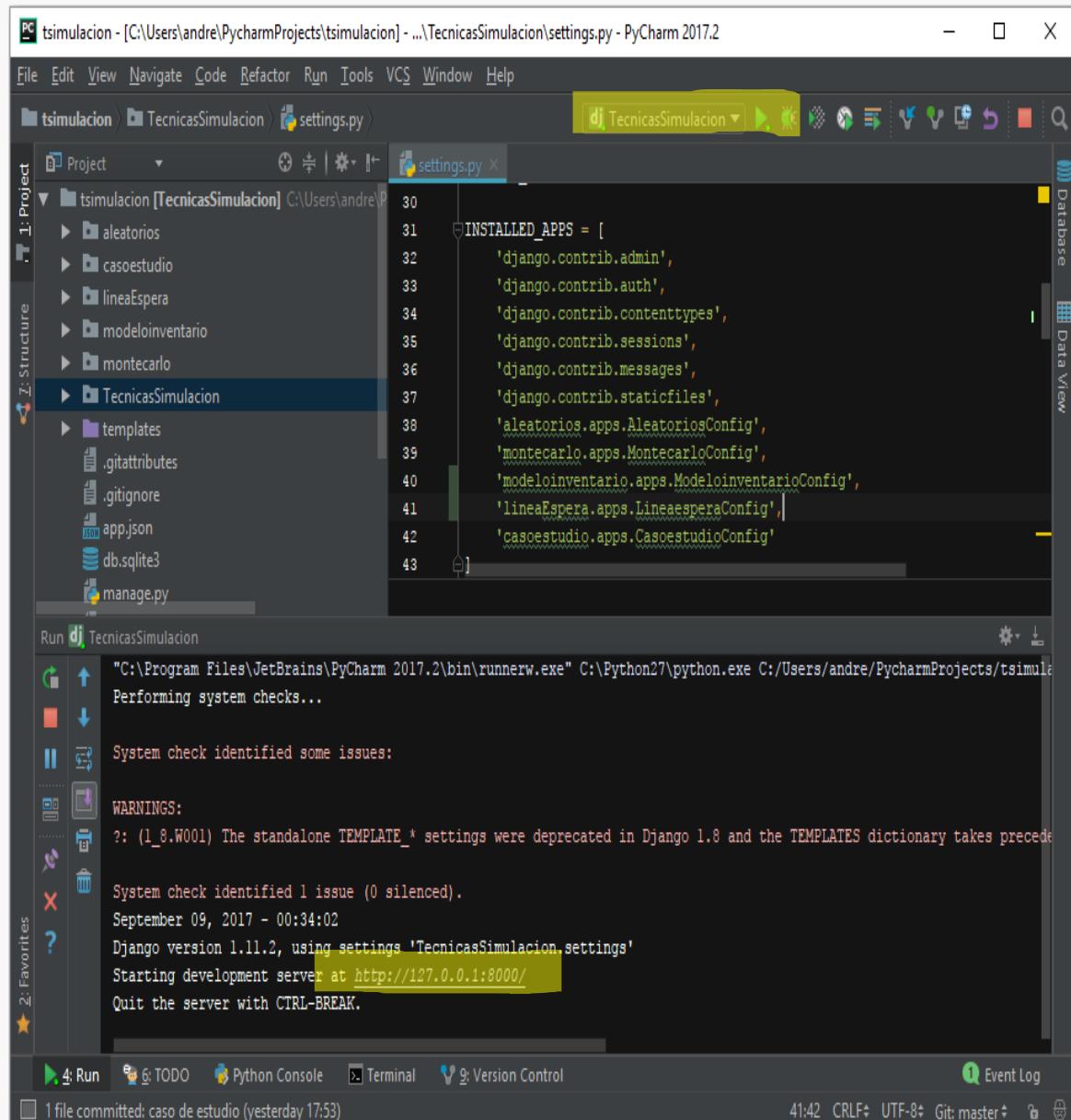
8. Si no hay una configuración de Django Server la agregamos y configuramos



8.1 Configuración de Django Server



9. Guardamos y ejecutamos el Proyecto.



```
PC tsimulacion - [C:\Users\andre\PycharmProjects\tsimulacion] - ...\\TecnicasSimulacion\\settings.py - PyCharm 2017.2
File Edit View Navigate Code Refactor Run Tools VCS Window Help
tsimulacion TecnicasSimulacion settings.py
Project Structure Database Data View
1: Project 2: Favorites
tsimulacion [TecnicasSimulacion] C:\Users\andre\PycharmProjects\tsimulacion
  aleatorios
  casoestudio
  lineaEspera
  modeloInventario
  montecarlo
  TecnicasSimulacion
  templates
  .gitattributes
  .gitignore
  app.json
  db.sqlite3
  manage.py
settings.py
30
31     INSTALLED_APPS = [
32         'django.contrib.admin',
33         'django.contrib.auth',
34         'django.contrib.contenttypes',
35         'django.contrib.sessions',
36         'django.contrib.messages',
37         'django.contrib.staticfiles',
38         'aleatorios.apps.AleatoriosConfig',
39         'montecarlo.apps.MontecarloConfig',
40         'modeloInventario.apps.ModeloInventarioConfig',
41         'lineaEspera.apps.LineaEsperaConfig',
42         'casoestudio.apps.CasoestudioConfig'
43     ]
Run dJ TecnicasSimulacion
"C:\Program Files\JetBrains\PyCharm 2017.2\bin\runnerw.exe" C:\Python27\python.exe C:/Users/andre/PycharmProjects/tsimulacion
Performing system checks...
System check identified some issues:
WARNINGS:
?: (1_8.W001) The standalone TEMPLATE_* settings were deprecated in Django 1.8 and the TEMPLATES dictionary takes precedence.
System check identified 1 issue (0 silenced).
September 09, 2017 - 00:34:02
Django version 1.11.2, using settings 'TecnicasSimulacion.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.
Event Log
4: Run TODO Python Console Terminal Version Control
1 file committed: caso de estudio (yesterday 17:53)
41:42 CRLF+ UTF-8+ Git: master+ ?
```

10. Visualización del Proyecto.

Técnicas de Simulación

- X Inicio
- Montecarlo
- Números Aleatorios
- Simulación Inventory
- Simulación Línea de Espera
- Caso de Estudio Cell Tuning

Caso de Estudio Línea de Espera en Cell Tuning

[Información De caso de estudio](#)

Generar Modelo Línea de Espera

Tiempo entre Llegadas de clientes
¿Cuantos Datos va a ingresar?

Tiempo del Soporte Técnico
¿Cuantos Datos va a ingresar?

Continuar

Desarrollado por:
Sánchez Cobeña
Cristóbal Andrés
8vo Nivel "A"

Docente:
Ing. Jorge Moya



Primeros pasos con HEROKU

HEROKU CLI

Instrucciones Descargar e instalar:

<https://devcenter.heroku.com/articles/heroku-cli#macos>

MAC OS - HOMEBREW:

```
$ brew install heroku/brew/heroku
```

Compatible con 10.7 o versiones posteriores

MAC OS - INSTALADOR:

1. Descargar y ejecutar el [instalador de MacOS](#).

Compatible con 10.7 o versiones posteriores

WINDOWS:

Descargar y ejecutar el instalador de Windows [de 32 bits y de 64 bits](#).

1. Ejecute el archivo .exe que inicia el Asistente para la instalación.

2. Siga todos los pasos sugeridos por el asistente. Preste especial atención a las opciones de instalación correspondientes.

[Debian / Ubuntu](#)

Esta versión no se actualiza automática. Vas a tener que actualizar manualmente Heroku CLI con apt-get. Utilice la instalación independiente para una versión autoupdating de la CLI.

Ejecute el siguiente para agregar nuestro repositorio apt e instalar la CLI:

```
$ wget -qO- https://cli-assets.heroku.com/install-ubuntu.sh | sh
```



¿Como subirlo a un Servidor (Heroku)?

Documentación Oficial:

<https://devcenter.heroku.com/articles/getting-started-with-python#introduction>

Requisitos:

1. Crear una Cuenta en HEROKU

<https://signup.heroku.com/dc>

2. Instalar Python

3. Instalar PipEnv

```
pip install pipenv
```

4. Tener instalado HEROKU CLI

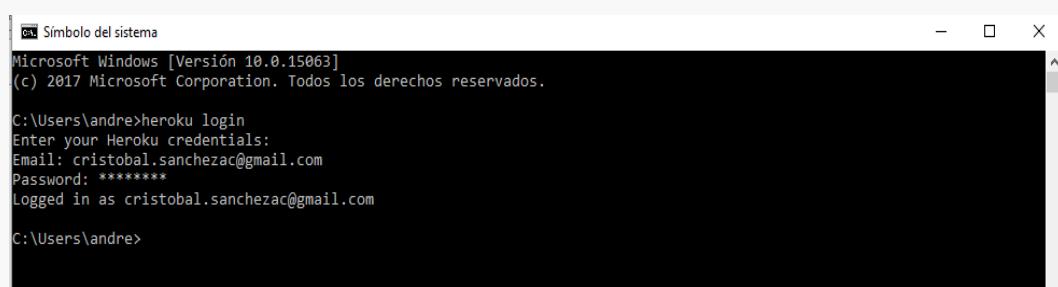
Preparar:

Una vez instalado, puede utilizar heroku con su shell de comandos.

1. Iniciamos nuestra consola e iniciamos sesión con el comando:

```
Heroku login
```

y ponemos nuestras credenciales creadas en heroku.



The screenshot shows a Windows Command Prompt window titled "Símbolo del sistema". The text inside the window is as follows:

```
C:\Símbolo del sistema
Microsoft Windows [Versión 10.0.15063]
(c) 2017 Microsoft Corporation. Todos los derechos reservados.

C:\Users\andre>heroku login
Enter your Heroku credentials:
Email: cristobal.sanchezac@gmail.com
Password: *****
Logged in as cristobal.sanchezac@gmail.com

C:\Users\andre>
```

La autenticación es necesaria para permitir que los comandos heroku y git funcionen.



2. Nos ubicamos en la carpeta donde está el proyecto.

```
C:\ Símbolo del sistema
Microsoft Windows [Versión 10.0.15063]
(c) 2017 Microsoft Corporation. Todos los derechos reservados.

C:\Users\andre>heroku login
Enter your Heroku credentials:
Email: cristobal.sanchezac@gmail.com
Password: *****
Logged in as cristobal.sanchezac@gmail.com

C:\Users\andre>cd C:\Users\andre\PycharmProjects\tsimulacion

C:\Users\andre\PycharmProjects\tsimulacion>
```

3. verificamos nuestros archivos Pipfile y Procfile

En el archivo PipFile van los nombres de las librerías utilizadas ya que de aquí toma la información para instalar las librerías necesarias para nuestro proyecto.

 Pipfile: Bloc de notas Archivo Edición Formato Ver Ayuda [[source]] url = "https://pypi.python.org/simple" verify_ssl = true [packages] dj-database-url = "*" django = "*" gunicorn = "*" psycopg2 = "*" whitenoise = "*" numpy = "*" scipy = "*" [requires] python_version = "2.7"	 Procfile: Bloc de notas Archivo Edición Formato Ver Ayuda web: gunicorn TecnicasSimulacion.wsgi --log-file -
---	--

4. Creamos la aplicación, y también se nos crea un git.

`Heroku create nombredeAplicacion`

```
C:\ Símbolo del sistema
Microsoft Windows [Versión 10.0.15063]
(c) 2017 Microsoft Corporation. Todos los derechos reservados.

C:\Users\andre>heroku login
Enter your Heroku credentials:
Email: cristobal.sanchezac@gmail.com
Password: *****
Logged in as cristobal.sanchezac@gmail.com

C:\Users\andre>cd C:\Users\andre\PycharmProjects\tsimulacion

C:\Users\andre\PycharmProjects\tsimulacion>heroku create tsimulacion1
Creating tsimulacion1... done
https://tsimulacion1.herokuapp.com/ | https://git.heroku.com/tsimulacion1.git

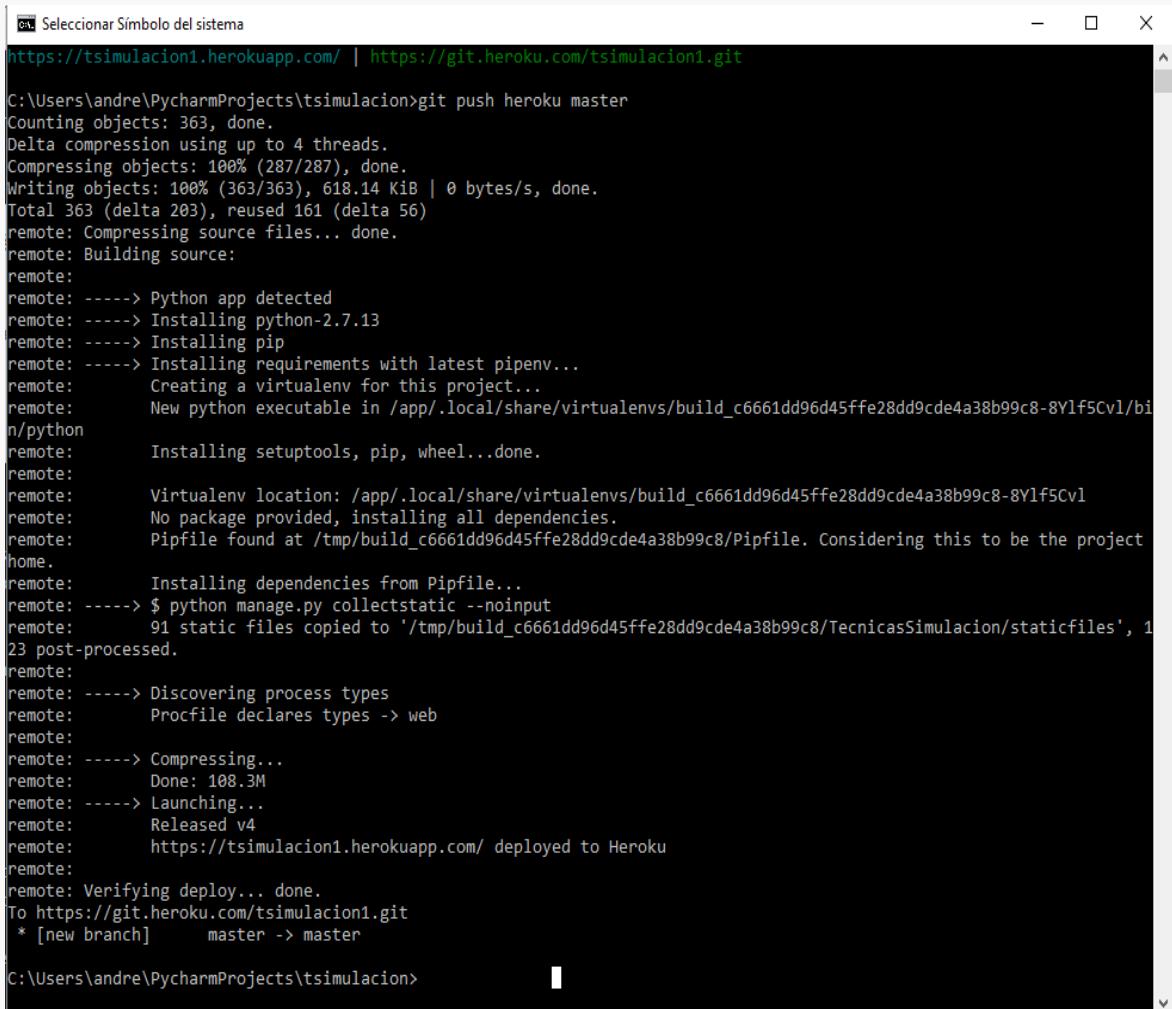
C:\Users\andre\PycharmProjects\tsimulacion>
```



5. Subimos los archivos de nuestro proyecto con git push.

```
git push heroku master
```

Se instalan las librerías y se suben todos los archivos al servidor de heroku.



```
git push heroku master
Counting objects: 363, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (287/287), done.
Writing objects: 100% (363/363), 618.14 KiB | 0 bytes/s, done.
Total 363 (delta 203), reused 161 (delta 56)
remote: Compressing source files... done.
remote: Building source:
remote:
remote: -----> Python app detected
remote: -----> Installing python-2.7.13
remote: -----> Installing pip
remote: -----> Installing requirements with latest pipenv...
remote: Creating a virtualenv for this project...
remote: New python executable in /app/.local/share/virtualenvs/build_c6661dd96d45ffe28dd9cde4a38b99c8-8Ylf5Cvl/bin/python
remote: Installing setuptools, pip, wheel...done.
remote:
remote: Virtualenv location: /app/.local/share/virtualenvs/build_c6661dd96d45ffe28dd9cde4a38b99c8-8Ylf5Cvl
remote: No package provided, installing all dependencies.
remote: Pipfile found at /tmp/build_c6661dd96d45ffe28dd9cde4a38b99c8/Pipfile. Considering this to be the project home.
remote:     Installing dependencies from Pipfile...
remote: -----> $ python manage.py collectstatic --noinput
remote:         91 static files copied to '/tmp/build_c6661dd96d45ffe28dd9cde4a38b99c8/TecnicasSimulacion/staticfiles', 123 post-processed.
remote:
remote: -----> Discovering process types
remote:     Procfile declares types -> web
remote:
remote: -----> Compressing...
remote:     Done: 108.3M
remote: -----> Launching...
remote:     Released v4
remote:     https://tsimulacion1.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/tsimulacion1.git
 * [new branch]      master -> master
```



6. Verificamos que la página funcione correctamente y la abrimos con:

Heroku open

7. Si queremos ver los logs utilizamos el comando:

`heroku logs --app tsimulacion1`

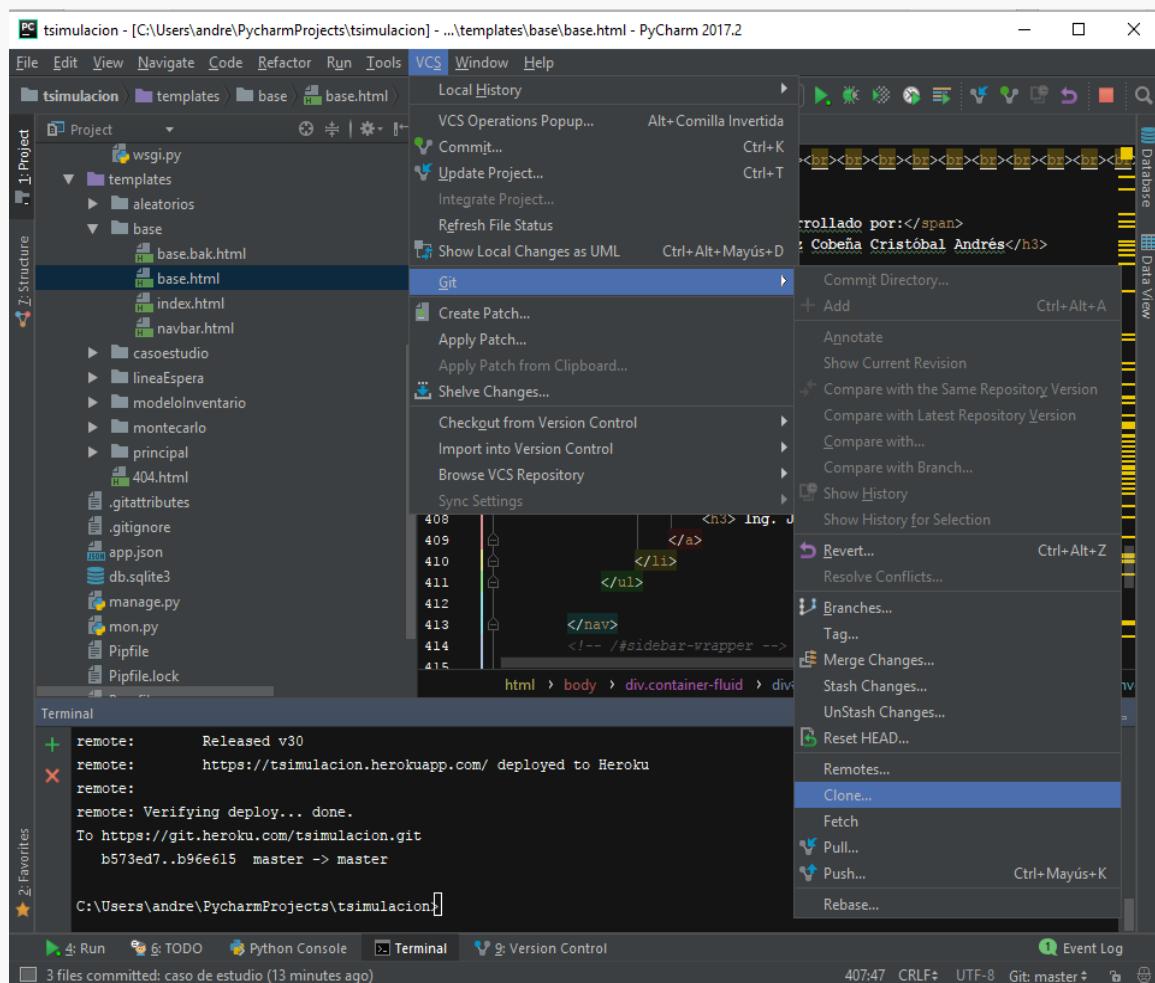
```
C:\Users\andre\PycharmProjects\tsimulacion>heroku logs --app tsimulacion1
2017-09-09T06:22:28.000000+00:00 app[api]: Build succeeded
2017-09-09T06:23:37.030519+00:00 heroku[web.1]: Starting process with command `gunicorn TecnicasSimulacion.wsgi --log-file -` 
2017-09-09T06:23:39.556898+00:00 app[web.1]: [2017-09-09 06:23:39 +0000] [4] [INFO] Starting gunicorn 19.7.1
2017-09-09T06:23:39.558552+00:00 app[web.1]: [2017-09-09 06:23:39 +0000] [4] [INFO] Using worker: sync
2017-09-09T06:23:39.558338+00:00 app[web.1]: [2017-09-09 06:23:39 +0000] [4] [INFO] Listening at: http://0.0.0.0:23066 (4)
2017-09-09T06:23:39.566910+00:00 app[web.1]: [2017-09-09 06:23:39 +0000] [9] [INFO] Booting worker with pid: 9
2017-09-09T06:23:39.583876+00:00 app[web.1]: [2017-09-09 06:23:39 +0000] [10] [INFO] Booting worker with pid: 10
2017-09-09T06:23:40.613741+00:00 heroku[web.1]: State changed from starting to up
2017-09-09T06:25:25.535812+00:00 heroku[router]: at=info method=GET path="/" host=tsimulacion1.herokuapp.com request_id=ad438487-38f2-4f18-acfa-984a04d64cc7 fwd="190.214.154.197" dyno=web.1 connect=1ms service=321ms status=200 bytes=12282 protocol=https
2017-09-09T06:25:25.783510+00:00 heroku[router]: at=info method=GET path="/static/css/estilo.css" host=tsimulacion1.herokuapp.com request_id=179ba008-a293-4618-b455-b0dc11a353ae fwd="190.214.154.197" dyno=web.1 connect=2ms service=3ms status=200 bytes=4038 protocol=https
2017-09-09T06:25:25.707092+00:00 heroku[router]: at=info method=GET path="/static/css/bootstrap.css" host=tsimulacion1.herokuapp.com request_id=530b6162-8414-4cf1-9f39-0bc724cecae8 fwd="190.214.154.197" dyno=web.1 connect=1ms service=14ms status=200 bytes=154853 protocol=https
2017-09-09T06:25:26.046943+00:00 heroku[router]: at=info method=GET path="/static/js/bootstrap.js" host=tsimulacion1.herokuapp.com request_id=250aad67-94da-4999-b2d8-3d4a4c147bab fwd="190.214.154.197" dyno=web.1 connect=1ms service=8ms status=200 bytes=76836 protocol=https
2017-09-09T06:25:25.898233+00:00 heroku[router]: at=info method=GET path="/static/css/style.css" host=tsimulacion1.herokuapp.com request_id=ff7733dc-aba3-4676-a921-5c240ef7023d fwd="190.214.154.197" dyno=web.1 connect=1ms service=3ms status=200 bytes=35075 protocol=https
2017-09-09T06:25:26.077546+00:00 heroku[router]: at=info method=GET path="/static/js/valid.js" host=tsimulacion1.herokuapp.com request_id=f8fe899e-7c77-4ad0-9d5a-fc5c4fec611f fwd="190.214.154.197" dyno=web.1 connect=1ms service=4ms status=200 bytes=21724 protocol=https
2017-09-09T06:25:26.065092+00:00 heroku[router]: at=info method=GET path="/static/js/bootstrap.min.js" host=tsimulacion1.herokuapp.com request_id=e7824ba8-92c2-4148-96db-42e4d61857d7 fwd="190.214.154.197" dyno=web.1 connect=1ms service=5ms status=200 bytes=37364 protocol=https
2017-09-09T06:25:26.044676+00:00 heroku[router]: at=info method=GET path="/static/js/funciones.js" host=tsimulacion1.herokuapp.com request_id=93333333-3333-4333-8333-333333333333 fwd="190.214.154.197" dyno=web.1 connect=1ms service=2ms status=200 bytes=1136 protocol=https
```



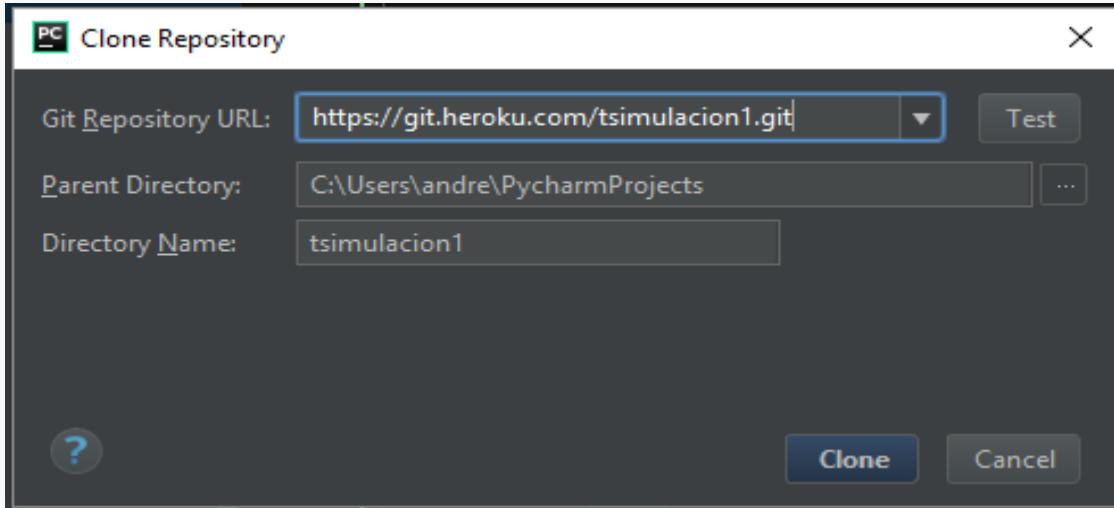
¿Cómo modificar nuestro proyecto subido en Heroku?

1. Iniciamos pycharm
2. Nos dirigimos a VCS y hacemos click en enable repository
3. Nuevamente vamos a VCS - git - y clone, clonaremos nuestro proyecto, en este caso nuestro repositorio que nos creó fue:

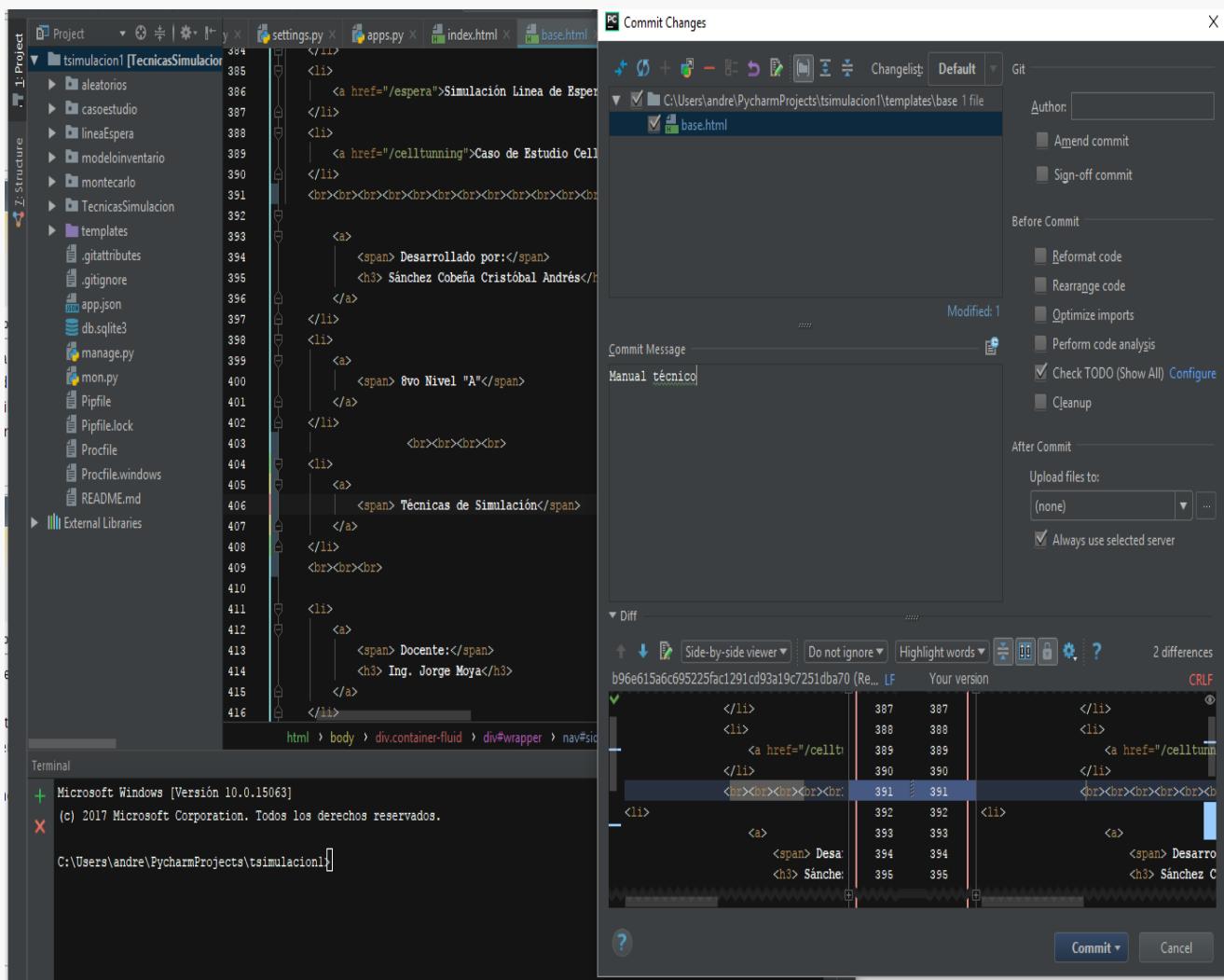
<https://git.heroku.com/tsimulacion1.git>



4. Clonamos nuestro proyecto pegando el enlace aquí.



5. Una vez copiado nuestros archivos de git en nuestros archivos locales podemos editarlos, para guardar los cambios hacemos un commit, y escribimos algún comentario.



6. hacemos un git push para actualizar los archivos al git y se actualice nuestra página.

```
git push
```

```
Terminal
remote: -----> Discovering process types
remote:         Procfile declares types > web
remote:
remote: -----> Compressing...
remote:         Done: 108.1M
remote: -----> Launching...
remote:         Released v6
remote:         https://tsimulacion1.herokuapp.com/ deployed to Heroku
remote:
remote: Verifying deploy... done.
To https://git.heroku.com/tsimulacion1.git
  b96e615..72a5bac master -> master

C:\Users\andre\PycharmProjects\tsimulacion1>
```

Event Log

1 file committed: Manual técnico (7 minutes ago)

406:54 CRIE UTF-8 Git: master

7. Verificamos los cambios en nuestra página.

Caso de Estudio Linea de Espera en Cell Tuning

Información De caso de estudio

Generar Modelo Línea de Espera

Tiempo entre Llegadas de clientes
¿Cuántos Datos va a ingresar?

Tiempo del Soporte Técnico
¿Cuántos Datos va a ingresar?

Continuar

Desarrollado por:
Sánchez Cobeña
Cristóbal Andrés

8vo Nivel "A"

Técnicas de Simulación

Docente:
Ing. Jorge Moya