Curso 2017/2018

Proyecto web con Django

[Subtítulo del documento]

Carlos Rubiales Ordóñez

**Introducción**

Este proyecto está enfocado en aprender el uso del Framework de Django, el lenguaje de programación Python, y otras tecnologías y software que serán expuestas más adelante.

Para esto vi adecuado desarrollar una web donde se almacenen alumnos, profesores, etc

**Objetivos**

El objetivo de este proyecto es implementar un sistema de gestión del alumnado y profesorado en una web, basándose en el framework Django.

Los principales sub-objetivos que han guiado el desarrollo del presente programa son:

-Aplicación universal: este es uno de los objetivos más importantes; desarrollar un programa que pueda utilizarse desde prácticamente cualquier plataforma

-Creación de alumnos y profesores con las particularidades de cada uno

-Poder listarlos, editarlos y borrarlos.

-Sencillez de uso para que todo el mundo pueda entender la interfaz

En las siguientes páginas se abordará la explicación de forma más detallada y pormenorizada.

**Lenguajes de programación utilizados**

**Python**

Python es un lenguaje de programación interpretado, de alto nivel, orientado a objetos y con semántica dinámica. Sus estructuras de alto nivel, combinada con un tipado dinámico, lo hacen muy atractivo para el desarrollo ágil y rápido de aplicaciones y software, también para su uso como un lenguaje de script o lenguaje para conectar componentes existentes. Simple y muy fácil de aprender, la sintaxis de Python, hace hincapié en la lectura y a consecuencia de ello reduce el costo de mantenimiento del programa. Python soporta módulos y paquetes, lo que fomenta mucho la modularidad de los programas y la reutilización de su código. El intérprete de Python y la amplia librería estándar están disponibles en código fuente y/o binario sin coste ninguno para todas las plataformas que se deseen y pueden ser distribuidos libremente.

Comúnmente, los desarrolladores deciden utilizar Python debido al gran aumento de la productividad que proporciona. Al no haber paso de compilación, el ciclo de edición-testear-depuración es sorprendentemente rápido. La depuración de fallos en Python es fácil: un fallo no tiene porque causar un fallo de segmentación. En cambio, cuando el intérprete descubre un error, genera una excepción. Cuando el programa no captura la excepción, el intérprete imprime un seguimiento de la pila. Un depurador a nivel de fuente permite la inspección de las variables globales y locales, la evaluación de expresiones arbitrarias, el establecimiento de puntos de ruptura, paso a paso a través del código de una línea al mismo tiempo, y asi continuamente. El depurador está escrito en Pyhon también, lo que deja en evidencia la capacidad de instrospeccion de Python.

Elegí Python para elaborar este proyecto para aprender este lenguaje que está en auge y porque el Framework de Django funciona bajo Python.

Las principales características de este lenguaje son:

- Orientación a objetos.

- Manejo de excepciones.

- Cuatro niveles de ámbito de variable: global, clase, instancia y local.

- Similitud con Perl en expresiones regulares nativas.

- Fácil portabilidad.

- Amplia librería.

- En tiempo de ejecución soporta alteración de objetos.

La versión de Python utilizada para este proyecto es la 3.6.3

**HTML**

HTML (HyperText Markup Language) es un lenguaje de marcado, utilizado para la creación de páginas web.

Los documentos HTML comienzan con una declaración para indicar la versión de HTML en la que se ha escrito la página. Este lenguaje describe la estructura de un sitio web semántica e incluye las señas para la presentación y la apariencia de la web. Esto lo convierte en un lenguaje de marcas, no de programación

HTML5 es la quinta y última versión de HTML, todavía en desarrollo actualmente no es una versión cerrada, y es compatible con las versiones anteriores. La principal causa de su aparición es la gran evolución de la web en los últimos años y el cambio en la forma de interactuar con el usuario, pasando de páginas estáticas a páginas dinámicas.

**CSS3**

CSS (Cascading Style Sheets) u hojas de estilo en cascada es el lenguaje que se utiliza para describir la presentación de documentos HTML o XML, aunque también es utilizado para modificar la interfaz de usuario de algunos programas.

**Bootstrap**

No es un lenguaje de programación ni de marcado, pero esta relacionado con los dos anteriores. Bootstrap es un framework de software libre para desarrollar aplicaciones web, aportando, principalmente, funcionalidades para diseñar el estilo de una página web usando HTML, CSS y JavaScript. Fue creado en su momento por trabajadores de Twitter para lograr una homogeneidad en el estilo de las paginas creadas por la empresa, aunque fue publicada de forma libre y su uso se ha popularizado, siendo actualmente una herramienta muy utilizada y el proyecto más popular de GitHub, pagina para alojar proyectos utilizando el sistema de control de versiones Git.

Una de las principales características de Bootstrap es la existencia de clases HTML que asignadas a un elemento permiten indicar el ancho de la pantalla que ocuparan, pudiendo asignar diferentes anchuras según el tamaño de la pantalla en la que se visualiza, añadiendo varias clases al elemento.

Otras características muy utilizadas son sus iconos, el diseño de los botones y el de los formularios y las barras de navegación (las cuales pueden plegarse en dispositivos peque nos y tener menús desplegables gracias al empleo de JavaScript).

La versión de bootstrap utilizada para este proyecto es la 3.3.6

**Javascript**

JavaScript es un lenguaje ligero, interpretado, orientado a objetos, con funciones de primera clase, y es mejor conocido como el lenguaje de script para páginas web, aunque se puede usar en muchos mas entornos sin navegador también. Es un lenguaje de programación dinámico multiparadigma basado en prototipos. Es compatible con estilos de programación funcional, imperativo y orientado a objetos

Centrándonos en el lado del cliente en páginas web, que es el ámbito que nos ocupa en este proyecto, la función principal de estos scripts es la modificación de lo mostrado en la ventana del navegador (ocultar elementos, mostrar elementos que estén ocultos o mostrar nuevos elementos generados o solicitados). Para poder modificarlo se desarrolló y estandarizó una interfaz denominada árbol DOM (Document Object Model), que permite representar y manipular tanto documentos HTML como documentos XML (otro lenguaje de marcado). Los conjuntos de elementos del documento HTML forman el árbol DOM, con una estructura similar a un árbol genealógico, en el cual la raíz es el documento HTML del que cuelga el elemento HTML que generalmente tiene como hijos al elemento head y al elemento body y se va extendiendo con los descendientes (o elementos anidados) de ambos

JavaScript es un lenguaje bastante orientado a la gestión de eventos. Sobre algunos elementos o variables, por ejemplo, elementos HTML, se crea un listener o manejador de eventos, por ejemplo, se programa el código JavaScript de tal manera que si se realiza un click de ratón sobre cierto elemento se dispara el evento click sobre dicho elemento (el lanzamiento del evento está integrado de forma nativa por el navegador) y el manejador captura el evento y realiza la función que se le hubiese encomendado o programado

**jQuery**

jQuery es una librería multiplataforma escrita en JavaScript, diseñada para simplificar los scripts de lado del cliente HTML. A día de hoy es la librería de JavaScript más popular, y está instalada en más del 65% de los 10 millones de lugares con más alto tráfico en la Web. jQuery es un software gratuito de código abierto bajo la licencia MIT.

La sintaxis de jQuery está diseñada para que sea más fácil la navegación del documento, seleccionar los elementos DOM, crear animaciones, el control de eventos, desarrollo de aplicaciones AJAX. jQuery también proporciona facilidades para que los desarrolladores creen plugins para la librería. Esto permite a los desarrolladores crear abstracciones para a interacción de bajo nivel y animaciones, efectos más avanzados y widgets de alto nivel.

**El IDE: PyCharm**

PyCharm es un IDE (Entorno de desarrollo integrado), desarrollado por la compañía Jetbrains para programar Python

Algunas de las características de PyCharm son:

– Autocompletado, resaltador de sintaxis, herramienta de análisis y refactorización.

– Integración con frameworks web como: Django, Flask, Pyramid, Web2Py.

– Frameworks javascripts: jQuery, AngularJS.

– Debugger avanzado de Python y Javascript.

– Integración con lenguajes de plantillas: Mako, Jinja2, Django Template.

– Soporta entornos virtuales e intérpretes de Python 2.x, 3.x, PyPy, Iron Python y Jython.

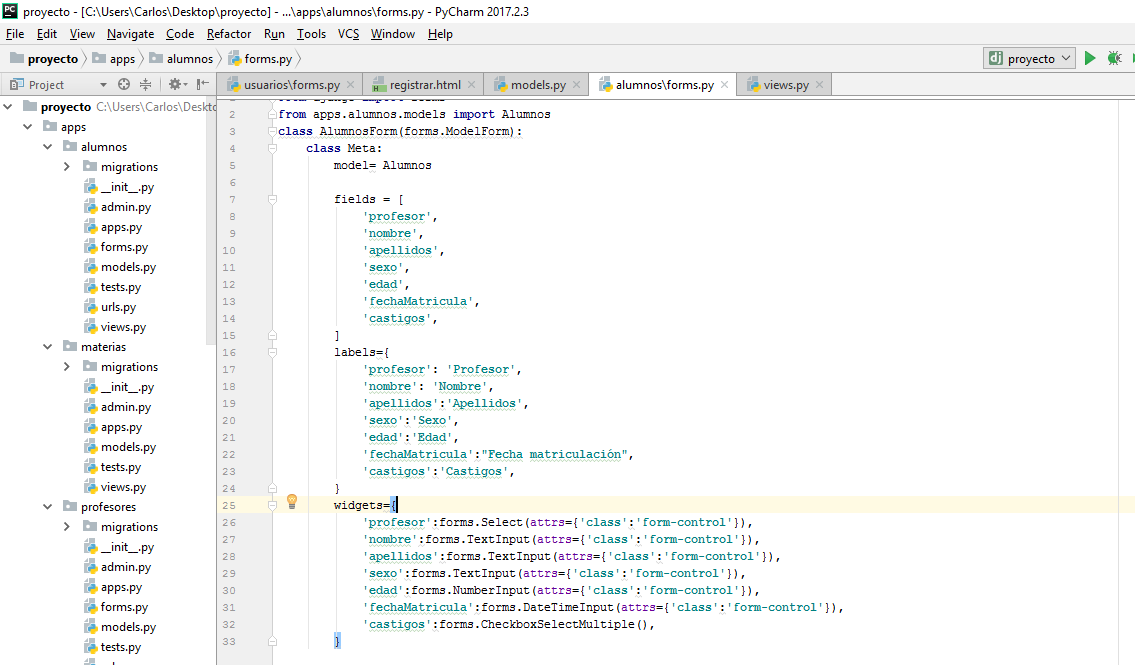
– Compatibilidad con SQLAlchemy (ORM), Google App Engine, Cython.

– Soporte para modo VIM (Con plugin)

– Sistemas de control de versiones: Git, CVS, Mercurial.

PyCharm es multiplataforma, hay binarios para: WIndows, Linux y Mac OS X. Existen dos versiones de PyCharm, una comunitaria y otra profesional con diferencias en características más que todo de en qué se integra una versión y otra.

La versión utilizada para el proyecto es la 2017.2.3



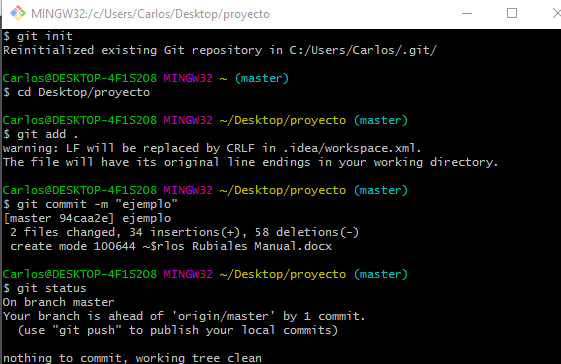
**Control de versiones: Git**

Git es un sistema de control de versiones libre y abierto, creado por Linus Torvalds. Está diseñado para controlar desde pequeños hasta grandes proyectos de forma rápida y eficiente.

Un sistema de control de versiones es un programa para la gestión de los cambios en un proyecto. Git permite registrar los cambios que se van realizando, asignándoles una breve descripción para saber que incluye cada cambio, para poder llevar un seguimiento de la evolución del programa, pero también para saber el estado de cada versión por si se quiere retroceder a una versión anterior, ya que esta herramienta permite además cambiar de una versión a otra (tener guardadas por separado cada una de las versiones implicaría mayor tiempo al tener que copiar todo el proyecto y un mayor espacio ocupado, ya que Git lo optimiza guardando solo las diferencias entre una versión y otra).

Otra característica importante es la posibilidad de crear ramificaciones y trabajar en cada una de ellas sin afectar al resto, permitiendo estar trabajando con varios aspectos complejos de un programa de forma simultánea y completamente aislada, o dando la posibilidad de dividir un proyecto para desarrollar partes específicas para un determinado servicio o un determinado cliente, e incluso facilitar el desarrollo de proyectos en los que trabajen varias personas.

Git Bash es una consola desde la cual se interpretan los comandos de Git. Además, en esta consola se pueden ejecutar los comandos más habituales de Linux (ls, cd, mkdir, rm, pwd, etc.).



**¿Qué es un Framework?**

Un framework es un esquema para el desarrollo y/o implementación de una aplicación. Es decir, un framework nos facilita la programación debido a que muchas funciones que deberían ser escritas a mano en muchas líneas ya vienen implementadas en sus librerías. Nos evita escribir grandes cantidades de código y facilita la interpretación del código (siempre y cuándo se tengan nociones del framework que se está utilizando).

Generalmente, los frameworks están ligados a un lenguaje en concreto, aunque no siempre es así.

Utilizar un framework tiene algunas ventajas, como:

- Facilidad para el programador a la hora de diseñar la estructura global de la aplicación. El framework proporciona esta estructura, la cual hay que ir programando.

- Facilita la colaboración, ya que está todo más estandarizado y definido según el framework. Es un código menos personal y más automatizado.

- Hay un mayor número de utilidades y liberías adaptadas a frameworks en concreto.

- Aunque al desarrollar una aplicación, se necesita un cierto tiempo y costes iniciales de aprendizaje, a largo plazo se facilita el desarrollo de la aplicación y el mantenimiento de la misma.

**Django**

Django es un framework o conjunto de herramientas para desarrollo web, de código abierto y escrito en su totalidad en Python. Fue desarrollado inicialmente para gestionar paginas de noticias de la World Company de Lawrence, pero en 2005 fue liberada al público bajo una licencia BSD.

Se basa en el paradigma Modelo-Vista-Controlador, consistente en separar los datos y la lógica de negocio de una aplicación (modelo), de la interfaz de usuario (vista) y del módulo encargado de gestionar los eventos y las comunicaciones (controlador).

Su filosofía general consiste en la creación de sitios web complejos de forma rápida, facilitar la reutilización de elementos (realizando módulos independientes que no necesiten conocer la existencia del resto, en la medida de lo posible), disminuir la cantidad de código, dar prioridad a lo explicito frente a lo implícito (cualidad heredada de Python), ser consistente en todos los niveles y evitar repetir código o información, siendo una de sus principales citas ´ ‘Don’t repeat yourself (DRY)’ o ‘no te repitas’ en español.

Algunas de sus características extra son:

-Admin: Django es el único framework que “por defecto” viene con un sistema de administración activo, listo para ser utilizado sin ningún tipo de configuración

-ORM: una herramienta que te permite realizar consultas SQL a la Base de Datos, sin usar SQL

Ejemplo:

Sin ORM:

SELECT \* FROM “autores\_autor” INNER JOIN “autor\_libro” ON (“autores\_autor”.”id” = “autores\_libro”.”autor\_id”)

INNER JOIN “autores\_libro\_librerias” ON (“autores\_libro”.”id” = “autores\_libro\_librerias”.”libro\_id”)

INNER JOIN “autores\_libreria” ON (“autores\_libro\_librerias”.”libreria\_id” = “autores\_libreria”.”id”) WHERE “autores\_libreria”.”nombre” = “La Cultura”

Con ORM:

Autor.objects.filter(libros\_\_libreria\_\_nombre = “La Cultura”)

-Cuenta con sus propios sistemas de enrutamiento, que nos permite definir las url o apartados de la aplicación de una forma muy flexible

-Validación automática de formularios, lo que simplifica el desarrollo de complejos paneles de gestión y control, entrada de datos, etc.

Además de los objetivos globales, Django puede dividirse en cinco grandes bloques para una mejor comprensión:

* Los **models** o modelos: es la definición de los elementos que se almacenan en la base de datos; generalmente, cada modelo se mapea o equivale a una tabla en la base de datos. De nuevo, lo explicito prima sobre lo implícito, sin asumir comportamientos en función de los nombres de los campos de un modelo, siendo necesario indicar el tipo de dato que almacenaran.

Los modelos deberían incluir toda la lógica sobre ellos mismos, desde el tipo de datos que almacenara hasta los campos para ser interpretados por los humanos, para una mejor comprensión de un determinado modelo u objeto y una mayor organización.

- La **API** de la base de datos: Django aporta abstracciones para interactuar con la base de datos de una forma sencilla. Sus objetivos principales consisten en lograr eficiencia en el acceso a la base de datos, intentando reducir el número de accesos a la misma, una sintaxis rica y concisa, que permita declaraciones expresivas, y permitir el acceso desde un objeto a otro relacionado en ambos sentidos, pero permitiendo además la posibilidad de realizar declaraciones de más bajo nivel para acceder a la base de datos, ofreciendo así una gran versatilidad.

- Las **URLs**: es una parte muy importante en el desarrollo de una aplicación en Django, es la interfaz entre el cliente y el servidor, son las referencias que permiten al cliente solicitar, crear o modificar información. Su filosofía defiende desemparejar las URLs de las funciones para una mayor reutilización, ya que diferentes sitios web podrían tener diferentes URLs para la misma función. Además, deben ser flexibles, permitiendo cualquier URLs concebible y permitir que tengan una estética cuidada, sin obligar al empleo de caracteres o características específicas

-. Las **templates** o plantillas: Django presenta un sistema para renderizar o generar dinámicamente la respuesta a una petición a partir de unas plantillas previamente creadas por el desarrollador y un contexto, que puede variar según determinadas circunstancias como pueden ser la URL de la petición, la información disponible en la base de datos o el usuario que realiza la petición. Este sistema pretende separar la lógica de la presentación (introduciendo esta ´ ultima en la propia plantilla). También busca evitar redundancia, basándose en la premisa de que la mayoría de páginas web dinámicas comparten una ´ gran parte del diseño, como puede ser la cabecera, la barra de navegación o el pie de página. Las plantillas no pretenden inventarse un nuevo lenguaje de programación por lo que solo tienen funcionalidades básicas, pensando además de la separación con la lógica, en la posibilidad de ser creadas por diseñadores en lugar de programadores; y otra razón es la de evitar la inclusión de código malicioso, el cual es a su vez otro objetivo de los desarrolladores.

* Las **views** o vistas: son funciones Python que reciben una petición web y devuelven una respuesta web. Esta respuesta puede ser un HTML, una redirección, una imagen... La vista contiene la lógica para devolver esa respuesta. Se busca la simplicidad, que la creación de una vista sea igual de fácil que escribir una función de Python. Las vistas deben tener acceso a un objeto petición que contenga la información de la solicitud actual; para ello siempre reciben este objeto como parámetro, en lugar de tener que acceder a dicha información desde una variable global. Además, debe ser capaz de poder diferenciar peticiones GET y POST, ya que son diferentes y, por tanto, deben ser gestionadas de forma diferente (generalmente una petición GET es realizada para solicitar un recurso y una petición POST para crearlo o modificarlo).

La versión de Django utilizada es la 1.11

**MVC**

MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado y donde se separe la lógica de negocio de la interfaz de usuario, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos.

Su fundamento es la separación del código en tres capas diferentes, acotadas por su responsabilidad, en lo que se llaman Modelos, Vistas y Controladores. En Django Se denomina Modelo-Template-Vista pero a la hora dela verdad su comportamiento es casi indistinguible.



El flujo de control que sigue el MVC es el siguiente:

- Se produce la interacción con la interfaz de usuario (click en botón por ejemplo).

- El controlador recibe la notificación de la acción solicitada y gestiona el evento (trata el evento de entrada) a través, generalmente, de un gestor de eventos (handler).

- El controlador accede al modelo, actualizándolo o modificándolo si fuera necesario según la acción solicitada por el usuario. –

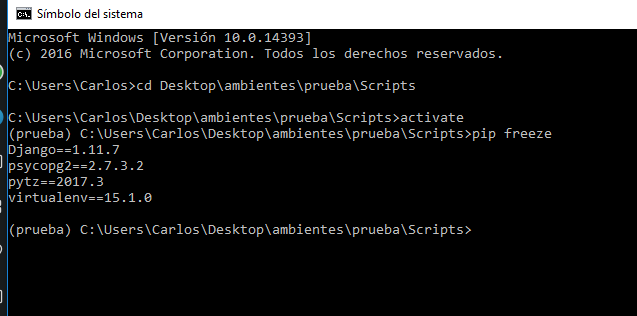
-La vista obtiene sus datos del modelo y genera la interfaz apropiada.

- La interfaz de usuario espera otra interacción del usuario, que comenzará otro nuevo ciclo

**Virtual enviroment o VENV**

Un virtualenv o entorno virtual de Python es un ambiente creado con el objetivo de aislar recursos como librerías y entorno de ejecución, del sistema principal o de otros entornos virtuales.

Lo anterior significa que, en el mismo sistema, maquina o computadora, es posible tener instaladas múltiples versiones de una misma librería sin crear ningún tipo de conflicto.



**Base de datos.**

**PostgreSQL**

PostgreSQL es un sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia PostgreSQL,1​ similar a la BSD o la MIT.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre o apoyados por organizaciones comerciales. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group).

Una característica interesante de PostgreSQL es el control de concurrencias multiversión, o MVCC, este método agrega una imagen del estado de la base de datos a cada transacción. Esto permite hacer transacciones eventualmente consistentes, ofreciendo grandes ventajas en el rendimiento, ejemplo de esto, es que no se requiere usar bloqueos de lectura al realizar una transacción, lo cual aporta una mayor escalabilidad.

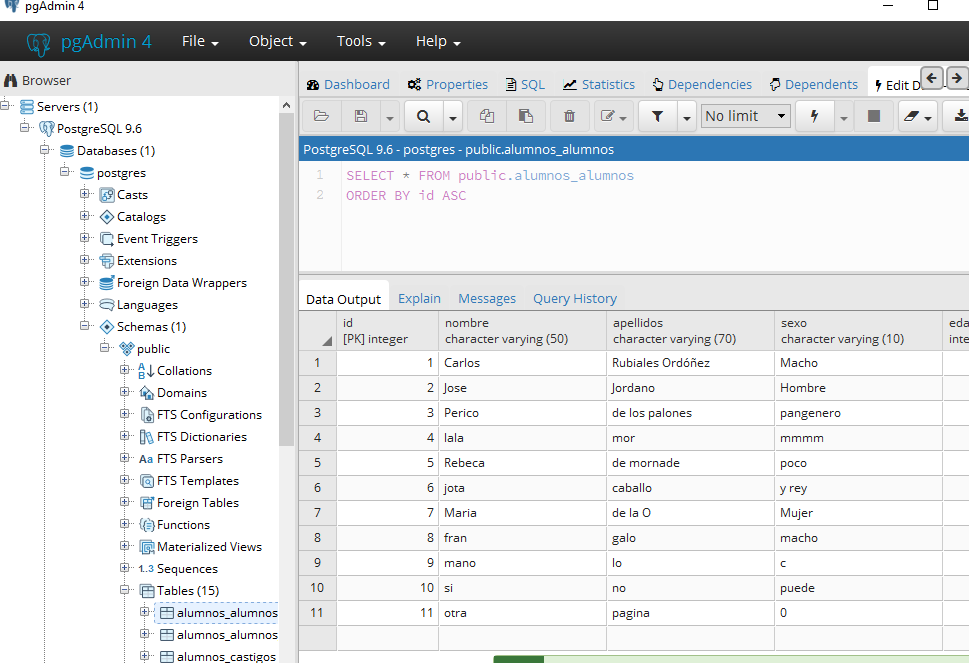
Otra característica relevante, es el Hot-Standby, que permite que los clientes hagan búsquedas (sólo de lectura) en los servidores, mientras están en modo de recuperación o espera. De esta manera, se pueden ejecutar tareas de mantenimiento o recuperación, sin bloquear completamente el sistema.

En la nube elástica de Facilcloud se puede trabajar con esta herramienta, que aporta mucha flexibilidad a los proyectos, y permite definir funciones personalizadas por medio de lenguajes, como: PL/pgSQL, PL/Tcl, PL/Perl, PL/Python, PL/PHP, PL/Ruby, PL/Java.

Otra de sus ventajas es que está disponible para muchas plataformas y ofrece el código fuente desde el sitio oficial. Entre sus builds oficiales, destacan: Mac OS X, Windows, Solaris, Red Hat, Debian, Ubuntu.

Asimismo, pgAdmin es la herramienta oficial para administrar bases de datos en PostgreSQL, y esta posibilita, desde hacer búsquedas SQL, hasta desarrollar toda una base de datos de manera fácil e intuitiva; directamente desde la interfaz gráfica.

Una herramienta muy útil de pgAdmin es el Query Tool. Este te permite ejecutar comandos SQL y además nos da la opción de analizar nuestra base de datos de forma gráfica.



**PSYCOPG2**

Psycopg es el adaptador de base de datos PostgreSQL más popular para el lenguaje de programación Python. Sus principales características son la implementación completa de la especificación Python DB API 2.0 y la seguridad de subprocesos (varios subprocesos pueden compartir la misma conexión). Fue diseñado para aplicaciones con muchos subprocesos múltiples que crean y destruyen muchos cursores y hacen una gran cantidad de INSERT o ACTUALIZACIONES simultáneas.

Psycopg 2 esta principalmente implementado en C como un contenedor libpq, lo que resulta en un uso eficiente y seguro. Cuenta con cursores del lado del cliente y del servidor, comunicación asincrónica y notificaciones, compatibilidad con COPY. Muchos tipos de versiones de Python son compatibles desde el primer momento y están adaptados para hacer coincidir los tipos de datos de PostgreSQL, la adaptación puede ampliarse y personalizarse gracias a un sistema flexible de adaptación de objetos.

Psycopg 2 es amigable tanto para Unicode como para Python 3. La versión utilizada es la 2.7