

PYTHON

Adrián Jiménez Nieto

PROYECTO FINAL



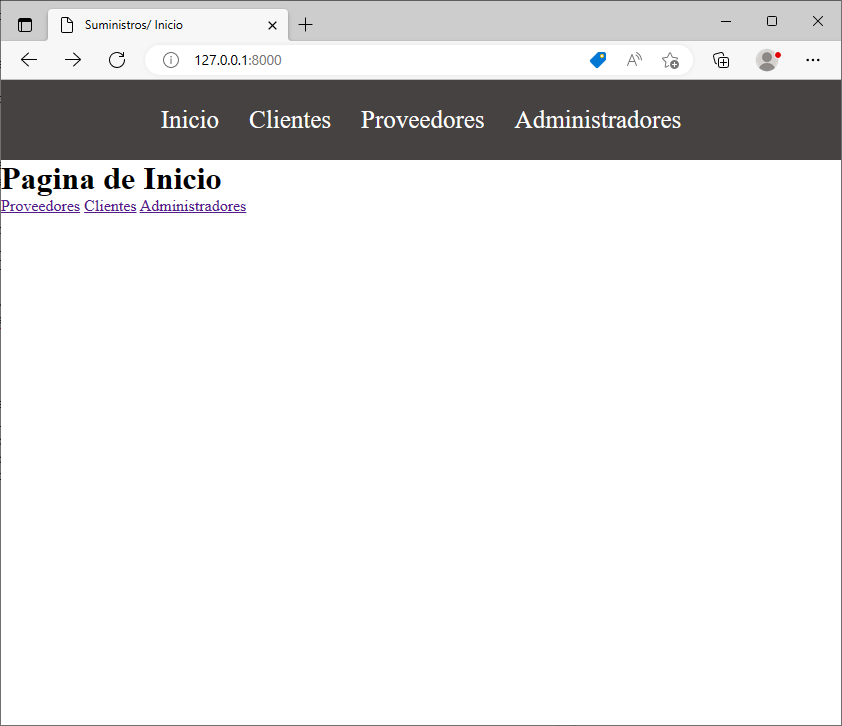
ADRIÁN JIMÉNEZ NIETO

RESPONSABLE DEL DESARROLLO DE LA APLICACIÓN

**Fecha:** 09/03/2023

**DESARROLLO**

1. Se nos pide desarrollar una aplicación web con Python en la que haremos uso de algún framework. La aplicación consiste en un gestor para una empresa de suministros informáticos.
2. Para ello definiremos una serie de clases básicas que necesitaremos para la aplicación web. Una clase de Productos, una clase de Proveedores que relacionaremos con la anterior y una clase de Clientes.
3. Crearemos también varias App, una para gestionar los clientes, otra para gestionar los Proveedores, una para gestionar los administradores que llamaremos Admin, y otra para los Productos (para poder visualizarlos y demás).
4. Una vez generadas las app de nuestro proyecto, las instalaremos en nuestro ‘manage.py’. Después y poniendo el broche a la configuración general inicial definiremos las URLs de las app en el general(modularizando con unos ‘insert’), y las URLs de cada app encargadas de llevarnos a las vistas donde residirá nuestra lógica y renderizaremos los templates oportunos.
5. Antes de empezar a construir los modelos de cada app crearemos los directorios y archivos dedicados a la parte visual de nuestra aplicación. Crearemos los templates donde estarán las plantillas HTML que modularizaremos a través del método ‘extends’ para que quede un código mucho más limpio. Y crearemos un pequeño menú de navegación para tener algo visual con lo que empezar a trabajar. Dejo aquí abajo un primer vistazo de la aplicación.



Primera vista de la App

1. Una vez todo listo es hora de preparar los modelos que van a tener cada app. En un principio desarrollaremos la aplicación web en SQLite3 pero más tarde podremos configurar la aplicación para que trabaje con postgreSQL para tener algo un poco más profesional y potente.
2. Una vez creados los modelos pasamos a crear los formularios que usaremos para registrar tanto clientes, proveedores, productos, y administradores (en el código lo llamamos Staff para no crear conflictos con la ruta ‘admin/’ propia de django de la que hablaremos más adelante).

**CONCLUSIONES**