PROYECTO 2: Chet Gaming

201731241 – Daniel Albizurez Alpirez
2991470610101

Resumen

La empresa dedicada a la compra y venta de videojuegos Chet maneja su cartera comercial a través de documentos CSV, debido a su crecimiento exponencial, desean implementar una página web, expandir su alcance y estandarizar el manejo de sus clientes por medio de un lenguaje de programación entendible para cualquier persona.

Utilizando la arquitectura cliente-servidor en un ambiente web el proyecto realiza la carga de archivos CSV y realiza una conversión a archivos XML los cuales se generan gráficas que muestran las condiciones e información de la empresa de una manera amigable para el usuario.

Se utilizó el lenguaje de programación Python, y las librerías Flask y Django para generar la dualidad cliente-servidor y la librería Matplotlib para la generación de las gráficas necesarias.

Utilizando protocolos http y ftp para la transmisión de datos y archivos en el proyecto.

El proyecto se desarrolló siguiendo los criterios establecidos por el paradigma de Programación Orientada a Objetos (POO).

Palabras clave

Python, Flask, Django, Cliente-Servidor, XML, CSV, Web

Abstract

The video games selling company, Chet, uses CSV documents to keep its records, given their exponential growth, the company has decided to use a web service, to expand their reach and standardize their clients stored information, using a programming language understandable to anyone.

Using a Client-Server model for a web environment, the project is able to do the uploading of CSV file, converting them to a XML file, with standardized information afterwards used to generate graphs, showing important business information in an easy and friendly way.

In order to accomplish this, Python was used as a programming language, relying on it's libraries Django, Flask and Matplotlib. Using HTTPS and FTP protocols for web communication.
Finally using an Object Oriented Programming approach.

Keywords

Python, Flask, Django, Cliente-Server, XML, CSV, Web

Introducción

Una aplicación web con una arquitectura cliente-servidor utilizando Django como cliente y Flask como servidor (Ambas librerías de Python), que realiza la carga y posterior transformación de archivos CSV en archivos XML y gráficas en base a la información contenida.

Desarrollo del tema

El proyecto consiste en una aplicación web modelada sobre una arquitectura cliente-servidor, la cual utiliza Django, librería de Python para la creación de aplicaciones web, como cliente y utiliza Flask, librería de Python para el manejo de solicitudes web, como servidor.

Del lado del cliente el proyecto realiza la carga de archivos CSV los cuales contienen la información pertinente de la empresa, seguidamente realiza la conversión de los archivos CSV a una estructura XML.

Del lado del servidor el proyecto se encarga de la recepción de la estructura XML para poder ser almacenada, luego procesa esa misma estructura, la interpreta y en base a los datos interpretados generar una serie de gráficas y listados que muestran la información más importante de una manera sencilla y amistosa para los usuarios de la empresa.

La aplicación también hace uso de la librería MatplotLib para realizar la realización de las gráficas en base a los datos recibidos.

1. CSV
   Los archivos CSV (Del inglés comma-separated values) son un tipo de documento que utiliza un formato sencillo para representar datos almacenados en forma de tabla, en las que las columnas se encuentran separadas por un carácter especial conocido como delimitador, el cual debe ser la coma, y las filas por saltos de línea.

   Debido al uso de delimitadores alternativos a la coma, a pesar de ser este el carácter estipulado para el formato, las aplicaciones que utilizan el formato permiten indicar el delimitador deseado.

2. XML
   XML (Del inglés eXtensible Markup Language), lenguaje de marcado extensible es un lenguaje que permite definir lenguajes de marcas, utilizado para almacenar datos de una manera legible. Permite definir la gramática de lenguajes específicos para estructurar documentos grandes.

   Gracias a la versatilidad y estandarización del lenguaje, XML es sumamente utilizado como medio de comunicación para información entre diferentes aplicaciones de diferentes plataformas.

3. Python
   Python es un lenguaje de programación cuya filosofía se centra en la legibilidad del código. Se trata de un lenguaje de programación multiparadigma, ya que soporta la orientación a objetos, la programación imperativa, la programación modular y la programación funcional. Es un lenguaje interpretado lo cual le permite ser un lenguaje multiplataforma (Lo cual

significa que es posible ejecutar programas desarrollados en este lenguaje en cualquier sistema operativo).
Es administrado por la "Python Software Foundation". Cuenta con una licencia de código abierto, denominada Python Software Foundation License.

Python cuenta con varios principios que definen su Filosofía, descritos por el desarrollador de Python, Tim Peters:

a. Bello es mejor que feo.
b. Explícito es mejor que implícito.
c. Simple es mejor que complejo.
d. Complejo es mejor que complicado.
e. Plano es mejor que anidado.
f. Disperso es mejor que denso.
g. La legibilidad cuenta.
h. Los casos especiales no son tan especiales como para quebrantar las reglas.
i. Lo práctico gana a lo puro.
j. Los errores nunca deberían dejarse pasar silenciosamente.
k. A menos que hayan sido silenciados explícitamente.
l. Frente a la ambigüedad, rechaza la tentación de adivinar.
m. Debería haber una —y preferiblemente solo una— manera obvia de hacerlo.
n. Aunque esa manera puede no ser obvia al principio a menos que usted sea holandés.[23]
o. Ahora es mejor que nunca.
p. Aunque *nunca* es a menudo mejor que *ya mismo*.
q. Si la implementación es difícil de explicar, es una mala idea.
r. Si la implementación es fácil de explicar, puede que sea una buena idea.
s. Los espacios de nombres (*namespaces*) son una gran idea ¡Hagamos más de esas cosas!

4. Flask
   Es un framework web minimalista desarrollado en Python. Se encuentra clasificado como un microframework por que no requiere de herramientas o librerías particulares. No cuenta con una capa de abstracción, validación o cualquier otro componente cuando otras librerías proveen funcionalidades comunes.

   Aún así Flask soporta extensiones que permiten agregar funcionalidades de aplicaciones como si fueran implementadas en Flask.

5. Django
   Es un framework de desarrollo web de código abierto, desarrollado en Python, el cual utiliza el patrón de diseño conocido como Modelo-Vista-Controlador (MVC).

   Está a cargo de la Django Software Foundation.

   Su meta fundamental es facilitar la creación de sitios web complejos. Tienen un énfasis en la reutilización , la conectividad y extensibilidad de componentes, el desarrollo rápido y el principio "No te repitas".
   Python es utilizado en todas las partes del framework, incluso en configuraciones,

archivos y modelos de datos.

6. MatplotLib
   Es una librería de graficado diseñada para Python. Provee una API para insertar gráficos en las aplicaciones

7. MVC
   El Modelo-Vista-Controlador es un patrón de arquitectura de software que busca la separación de los datos y la lógica del negocio de una aplicación de su representación y de los elementos encargados de gestionar los eventos y comunicaciones.

   Para lograrlo, el modelo propone la construcción de tres componentes distintos:
   a. El modelo
   b. La visa
   c. El controlador

   Definiendo componentes para el manejo de la información, la visualización de la información y la interacción de ambas partes.

   Este patrón se basa en las ideas de reutilización de código y la separación de conceptos, características que buscan facilitar el desarrollo y el mantenimiento de aplicaciones

8. Arquitectura Cliente-Servidor
   Es un modelo de diseño de software en el que las tareas del sistema son repartidas entre los proveedores de los recursos, servidores, y los demandantes, conocidos como clientes.

   Siguiendo un flujo similar a:
   Un cliente realiza diversas peticiones al servidor, quien le proporciona una respuesta.

   La idea puede ser aplicada a programas que se ejecutan sobre una sola computadora, aunque es más ventajoso aplicado a un sistema de software distribuido a través de una red de varias computadoras.

   En esta arquitectura la capacidad de proceso está repartida entre los clientes y los servidores. Cuenta con grandes ventajas de tipo organizativo, debidas a la centralización en la gestión de la información y la separación de las responsabilidades, facilitando y clarificando el diseño del sistema.

   Una disposición muy común en estos modelos son los sistemas “multicapa” en los que el servidor se encuentra descompuesto en diferentes programas que pueden ser ejecutados en diferentes computadoras aumentando así el grado de distribución del sistema.

Durante el desarrollo del proyecto se utilizaron los conceptos desarrollados anteriormente para generar una página web, que permite al usuario la carga de 4 archivos CSV, los cuales se espera contengan la información pertinente a la empresa.

Estos archivos son leídos y evaluados por parte del cliente (Programado utilizando Django), si se encuentran errores en la estructura se notifica por medio de un mensaje en pantalla; una vez evaluados

los datos son transformados en un solo archivo XML, presentado al usuario para que pueda realizar las modificaciones que considere necesarias.

Si el usuario se encuentra satisfecho con el XML presentado, tiene la opción de enviarlo para su correcto procesamiento y almacenamiento en el servidor (Implementado utilizando Flask), el cual toma los aspectos considerados más importantes del XML para generar una serie de reportes (Consistente de 3 gráficas generadas por medio de MatplotLib y dos listados), estos aspectos son posteriormente almacenados en un nuevo archivo XML.

Los 5 reportes son entonces presentados al usuario para que pueda trabajar con ellos.