UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

COLEGIO DE CIENCIAS E INGENIERÍAS

Aplicación de visualización de datos académicos

Joel Nestor Males Morales

Ingeniería en Ciencias de la Computación

Trabajo de integración curricular presentado como requisito
Para la obtención del titulo de
Ingeniero en Ciencias de la Computación

Quito, 15 de diciembre de 2021

UNIVERSIDAD SAN FRANCISCO DE QUITO USQF

COLEGIO DE CIENCIAS E INGENIERÍAS

HOJA DE CALIFICACIÓN
DE TRABAJO DE INTEGRACIÓN CURRICULAR

Aplicación de visualización de datos académicos

Joel Nestor Males Morales

Nombre del profesor, Título académico Daniel Riofrío, PhD

Quito, 15 de diciembre de 2021

© DERECHOS DE AUTOR

Nombres y apellidos:	Joel Nestor Males Morales
Código:	00124959
Cédula de Identidad:	1718902735
Lugar y fecha:	Quito, 15 de diciembre de 2021

ACLARACIÓN PARA PUBLICACIÓN

Nota: El presente trabajo, en su totalidad o cualquiera de sus partes, no debe ser considerado como una publicación, incluso a pesar de estar disponible sin restricciones a través de un repositorio institucional. Esta declaración se alinea con las prácticas y recomendaciones presentadas por el Committee on Publication Ethics COPE descritas por Barbour et al. (2017) Discussion document on best practice for issues around theses publishing, disponible en http://bit.ly/COPETheses.

UNPUBLISHED DOCUMENT

Note: The following capstone project is available through Universidad San Francisco de Quito USFQ institutional repository. Nonetheless, this project – in whole or in part – should not be considered a publication. This statement follows the recommendations presented by the Committee on Publication Ethics COPE described by Barbour et al. (2017) Discussion document on best practice for issues around theses publishing available on http://bit.ly/COPETheses.

Resumen

Este proyecto nace como resultado de brindar una solución al sistema HUBI manejado por el personal administrativo de la Universidad San Francisco de Quito USFQ mediante una aplicación web. En el sistema HUBI existe una sección para el seguimiento del progreso en la carrera de los estudiantes e información descriptiva de los aspirantes nuevos que han logrado ingresar a la universidad. Estos reportes se ofrecen en forma de tablas que contienen la información mencionada anteriormente, por lo que el uso de estos datos para producir información útil se torna en un trabajo arduo y muy susceptible a errores. El presente trabajo busca reducir esta complejidad de dar un uso sencillo a estos datos mediante visualizadores gráficos implementados en una aplicación web que permita navegar esta información de una manera más interactiva y con mayor facilidad de uso para el personal docente de la universidad.

Palabras clave: Django, Python, Web, Graphviz, Grafo, Visualización de datos, Link.

Abstract

This project was born as a result of providing a solution to the HUBI system managed by the administrative staff from the Universidad San Francisco de Quito USFQ through a web application. In the HUBI system there is a section for monitoring student progress in their careers and descriptive information of new applicants who have being admitted into the university. These reports are offered in the form of tables that contain the information mentioned above, so using this data to produce useful information becomes arduous work and very susceptible to error. The present work seeks to reduce this complexity by giving a simple use to this data through plots implemented in a web application that allow to navigate this information in a more interactive way and with greater ease of use for the university teaching staff.

Key words: Django, Python, Web, Graphviz, Graph, Data viewers, Link.

TABLA DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

INTRODUCCIÓN

Descripción del problema

En la Universidad San Francisco de Quito USFQ existe actualmente la plataforma HUBI. Este sistema consta de una sección que permite ver al personal administrativo y docente, el progreso en la carrera de los estudiantes y los aspirantes nuevos que han sido admitidos a la universidad. Esta información sobre los estudiantes se maneja en tablas cuadriculadas semejantes a las de Excel, por lo que al tener que manejar una gran cantidad de datos, este trabajo se torna arduo y exhaustivo, pues esta información se utiliza en parte para poder determinar la demanda de las clases para el siguiente semestre y tener control sobre los nuevos cupos de estudiantes. Esto ha causado que, en ocasiones, se calcule mal el número de clases que se deben abrir para el nuevo semestre y varios estudiantes tengan estos inconvenientes a último momento; además, al manejar tablas enormes se corre el riesgo de asignar información incorrecta hacia los estudiantes.

En base a la problemática mencionada anteriormente, se ha analizado la solución de poder crear una aplicación web que sea capaz de visualizar esta información de una manera mucho más fácil. Esto con la implementación de visualizadores gráficos que principalmente serán grafos y diagramas de pastel, los cuales estarán presentados con varios atributos para mejorar la navegación de estos datos hacia el usuario. Se ha escogido esta opción debido a que una aplicación web se puede anclar fácilmente al sistema HUBI, el cual, de igual manera está conformado por una plataforma web. Al querer trabajar de esta manera, se facilita el uso de la información ya existente para la nueva aplicación web, pues las mallas académicas pueden ser extraídas de forma dinámica y los datos de los estudiantes pueden ser ingresados en tiempo

real a través de ficheros, sin necesidad de usar ninguna base datos que guarde la información, pues todos los datos se manipularán solo en memoria RAM.

Esto también se alinea con las normas de confidencialidad que se mantienen dentro de la institución para poder garantizar el uso correcto de los datos de los estudiantes. Se manejará la anonimización de datos de los individuos, de tal manera que, todos los datos que se manipulen en la aplicación durante la creación de la misma serán datos ficticios, con el fin de que solo el personal autorizado pueda ingresar los datos reales una vez creada la plataforma. Por último, se estará dando una actualización a la interfaz gráfica de esta sección, lo cual mejora la experiencia del usuario y facilitará el uso a la hora de trabajar con esta herramienta.

Esta aplicación web consta principalmente de dos módulos, uno para ver el progreso de los estudiantes y otro para tener información sobre los nuevos estudiantes de la universidad. Para el desarrollo de la plataforma se usará el framework Django escrito en Python con una estructura independiente que no manipula datos del sistema HUBI, es decir, no hará uso de la base de datos de la universidad y el proceso de manejo de la información será solamente usado al momento de ejecución. Posterior a esto, la información será eliminada y se lo podrá corroborar en el código fuente que será publico a todos, esto con el fin de garantizar la privacidad y confidencialidad de la información de los estudiantes.

Objetivo General

Crear una manera interactiva y con mayor facilidad de visualización de los datos académicos de los estudiantes de la universidad San Francisco de Quito USFQ.

Objetivos Específicos

- Crear un reporte de cumplimiento de materias, en el cual se muestre el progreso de la carrera de cada estudiante, a través de la ayuda de un grafo.

- Crear un reporte de aspirantes aceptados, en el cual se muestre los datos más relevantes sobre el estudiante por medio de diagramas que faciliten la visualización de esta información.

DESARROLLO DEL TEMA

Análisis y diseño

Reporte progreso de estudiantes

En este reporte se muestra principalmente las mallas académicas de cada carrera, para la creación de este primer reporte se implementa la estructura de datos denominada grafo. Principalmente un grafo se compone de una colección de vértices o nodos, acompañado por un grupo de aristas o bordes que se encargan de conectar los vértices entre sí. Para un grafo dirigido, los bordes apuntan con dirección hacia un nodo (Fernández y Fernández, 2016). En base a esto principalmente se pretende estructurar la malla académica de forma en que las clases sean los nodos y los vértices representen la dependencia de una clase a otra, es decir si una materia tiene como prerrequisito alguna otra clase, esto se representa por medio del vértice que los une.

Las mallas académicas en la Universidad San Francisco de Quito USFQ están estructuradas por año académico y cada año consta de dos semestres, con excepción del último año donde en ocasiones solo hay un semestre y cuando un año tiene verano, por lo que el diseño general será con un grafo principal que represente el año y dentro de él habrá sub-grafos dirigidos que representen los semestres y verano en caso de existir. De esta manera se trata de mantener el diseño actual de las mallas que están en la página de la universidad.

Los nodos que representan las clases ofertadas por semestre se muestran dentro de los sub-grafos, estos contienen el código y nombre de la clase, además de un contador que muestra el número de estudiantes que han cursado la materia. Estos nodos se conectan por medio de los vértices y crean también las conexiones entre los sub-grafos, de esta manera se enlazan de igual

forma los años y todo en conjunto crea la malla académica dando como resultado un grafo dirigido.

Reporte de aspirantes aceptados

Para el segundo reporte se muestra un seguimiento de los aspirantes aceptados en la universidad. Para generar este reporte se utiliza principalmente un diagrama de pastel que se muestra por capas. La estructura académica de la Universidad San Francisco de Quito USFQ esta formada por dos niveles: colegios académicos y carreras. Estos niveles se muestran por capas en el diagrama de pastel, es decir se muestra desde el nivel más general como la primera capa del diagrama, hasta el nivel más específico donde se muestra información descriptiva agrupada por carrera.

Para agrupar las diferentes carreras en la Universidad San Francisco de Quito USFQ se empieza con el primer el nivel, el cual se refiere a los colegios académicos, que abarcan las distintas áreas de estudio. En el segundo nivel se tienen las carreras que se ofertan, es decir que por cada colegio existen un grupo de carreras que pertenecen a esa rama de estudio. Continuando con el tercer nivel se tienen dos diagramas de pastel donde en el primero se muestra la distribución de estudiantes por género y en el segundo se muestra la distribución de estudiantes por provincias.

Este modelo permite tener un fácil acceso hacia los datos de los estudiantes nuevos y mantener de manera estructurada esta información. Esto da como resultado una aplicación que puede ser escalable y mucho más fácil para darle mantenimiento en caso de ser necesario. Este reporte, además, representa una actualización al sistema manejado anteriormente para el usuario, permitiéndole tener una mejor experiencia y visión sobre la demanda académica por cada ciclo de inscripciones.

Implementación

Arquitectura

Para la creación del programa se utiliza una arquitectura denominada cliente-servidor, donde por un lado se tiene el servidor, el cual se encarga de actuar como un gestor de base de datos, y por otro lado se tiene el lado del cliente que solicita algunos servicios al servidor y este le devuelve una respuesta (Schiaffarino, 2019). En base a lo mencionado anteriormente se puede ver un diagrama de flujo sobre este esquema para una aplicación web en la Figura 1.

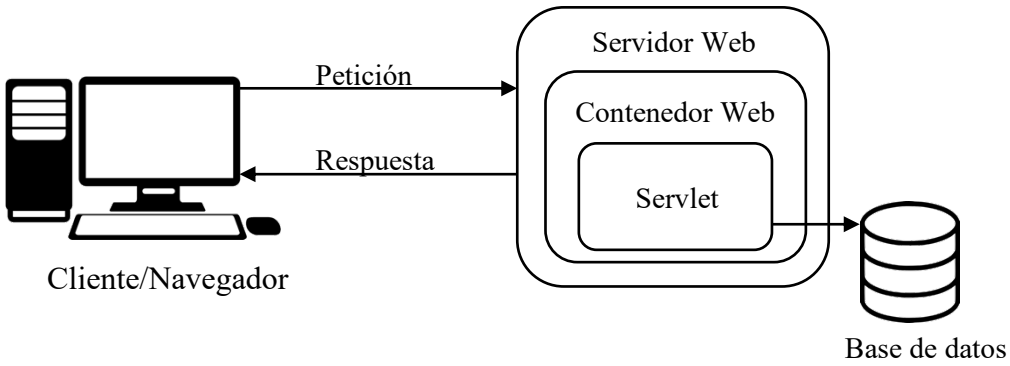

Figura 1. Diagrama de flujo del cliente-servidor web

Para este proyecto se utiliza como servidor a Django, el cual es un framework gratuito y de código abierto de alto nivel que permite construir sitios web de manera rápida y eficaz. (Mozilla, 2021). Para estos reportes no se trabaja con base de datos ya que los datos sobre los estudiantes se ingresan a la plataforma por medio de un archivo Excel, todo se trabaja en memoria RAM y una vez cerrada la página web, todos los datos usados son destruidos.

Para el primer reporte se utiliza esencialmente el código fuente HTML establecido para generar las mallas académicas, pues al tener una estructura ya establecida, se leen estos datos directamente en la aplicación creada y esto da la estructura al grafo resultante, para tener la

conexión de los vértices entre los nodos, se debe de igual manera ingresar un Excel con los datos de las clases y sus clases prerrequisito correspondientes a la carrera elegida.

En el segundo reporte se utilizan los archivos Excel que disponen los usuarios de la plataforma HUBI, de este archivo se extraen los colegios académicos, las carreras de cada colegio y la información de las provincias y géneros. De esta manera se crea el diagrama de pastel completo sin necesidad de ninguna base de datos.

Herramientas y Librerías

Reporte progreso de estudiantes

Nombre Librería	Versión
Graphviz	0.17
Django	3.2.7
Pandas	1.3.4
Requests	2.26
Numpy	1.21.2
Openpyxl	3.0.9

Tabla 1. Librerías usadas en reporte progreso de estudiantes

Al usar Django se trabaja con el lenguaje de programación Python para toda la implementación y desarrollo del código, esto se refiere al backend de la aplicación, por otro lado, para la parte visual, el frontend, se utiliza HTML como se visualiza en la figura 2. En base a esto, para la creación del primer reporte se usará la librería denominada Graphviz, la cual sirve para trabajar específicamente con grafos. La Tabla 1 muestra las librerías y las versiones utilizadas en la implementación de este reporte.

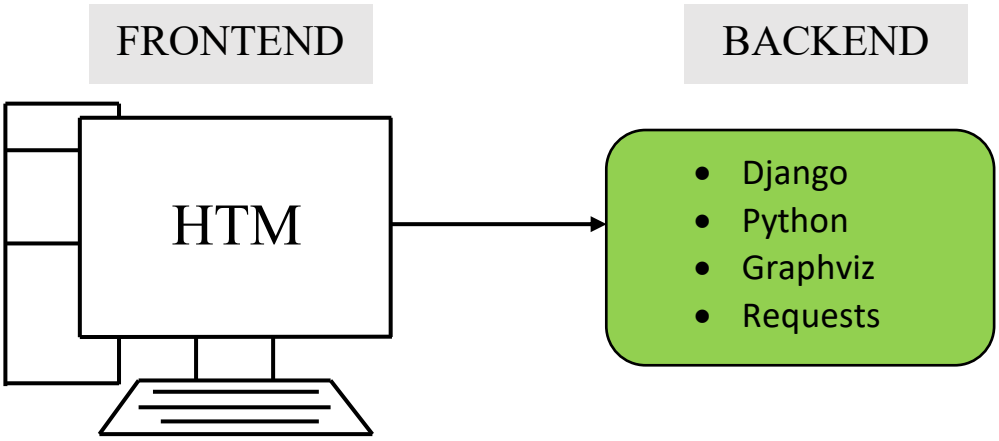

Figura 2. Estructura frontend y backend reporte 1

El primer reporte se basa en la estructura de un grafo, por lo que la lógica y estructura que se usan están basados en la malla original con formato HTML. Para esto se usa principalmente la librería requests de Python, la cual permite obtener el código fuente HTML de una página web a través su URL, de manera que dependiendo de la malla que se quiera visualizar, se ingresa su URL. Esto permite obtener las clases de la malla por secciones, es decir, agrupadas por semestres y por años. Para esto se manipula el código como una cadena string, este texto obtenido pasa por varios filtros para recuperar solo la información necesaria y excluir las demás partes del código.

Esta información obtenida se almacena en un diccionario de la siguiente manera: la clave consiste en el año académico y el valor es un arreglo, este arreglo contiene dentro más arreglos. Cada uno de estos representan los semestres por año y dentro de estos arreglos, cada elemento es una clase perteneciente a ese semestre. En caso de existir clases de verano, se lo interpreta como un semestre más. Este diccionario se utiliza como dato de entrada para así crear el grafo.

Por otro lado, en cada materia se tiene un número contador, este número se refiere a la cantidad de estudiantes que han cursado esa materia, esta variable proviene de un archivo Excel que contiene la información del progreso académico de los estudiantes. Este archivo contiene columnas con los nombres de las materias, por lo que en este caso se utiliza la librería pandas para extraer toda esta información y de igual manera se utiliza un diccionario para almacenar la información. El diccionario está estructurado de la siguiente manera: la clave es el nombre de la materia y el valor es un contador que representa los estudiantes que han cursado esa clase.

Con los datos obtenidos de estos dos diccionarios se procede a utilizar la librería Graphviz para armar el grafo. Se crea un grafo padre con la información del año académico, es decir la clave del primer diccionario, luego se crean subgrafos dentro del primer grafo. Estos subgrafos son armados a partir de los arreglos que se encuentran dentro de los valores del primer diccionario. Por último, utilizando los elementos de estos últimos arreglos, se crean los nodos dentro de cada subgrafo, en este mismo momento también se utiliza el segundo diccionario y se comparan las claves con los elementos de los arreglos del primer diccionario Con esto se comparan las materias y si estas coinciden, entonces se concatena el valor de esa clave al nodo y se obtiene el total de estudiantes que han cursado esa clase.

Graphviz devuelve una imagen del grafo con los datos ingresados anteriormente. Esta imagen resultante se la pasa a la vista armada con código HTML, para esto se decodifica la imagen a bytes en el backend y se lo envía al frontend, el cual se encarga de renderizar la imagen y mostrarla en la aplicación web. Esta estructura no requiere de mucha manipulación con código para la parte visual.

Reporte de aspirantes aceptados

Nombre Librería	Versión
Chart js	2.9.3
Django	3.2.7
Pandas	1.3.4
Sweet alert	2.26
Openpyxl	3.0.9

Tabla 2. Librerías usadas en reporte de aspirantes aceptados

El segundo reporte usa de igual manera código Python para el backend de la aplicación y HTML con JavaScript para el frontend como se ve en la figura 3. En la primera capa del diagrama de pastel, se extraen del archivo Excel los nombres de los colegios pertenecientes a la universidad y se los almacenan en un diccionario donde la clave es el nombre del colegio y el valor es el número de estudiantes registrados en ese colegio.

Para la segunda capa se utiliza nuevamente el archivo Excel y se extraen los datos en dos diccionarios: el primero tiene como clave el nombre del colegio y el valor es un arreglo que contiene el nombre de las carreras pertenecientes a ese colegio, el segundo tiene como valor el nombre de la carrera y el valor es el número de estudiantes registrados en esa carrera. La Tabla 2 muestra las librerías y las versiones utilizadas en la implementación de este reporte.

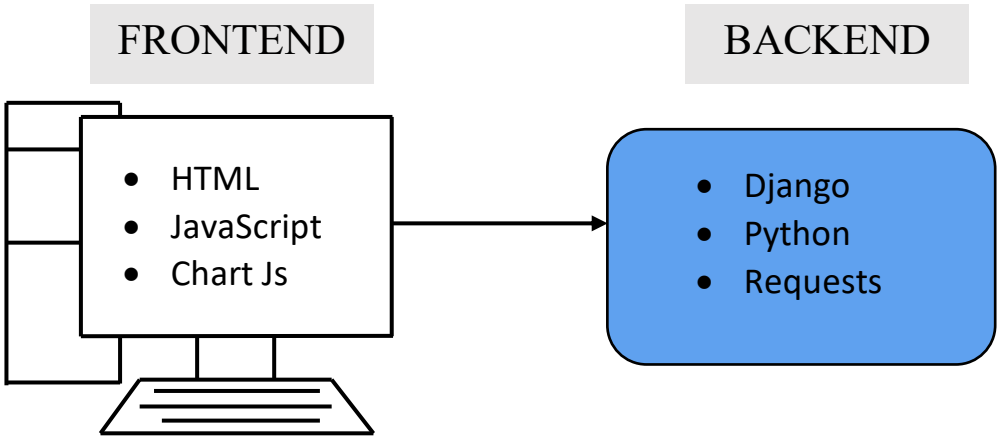

Figura 3. Estructura frontend y backend reporte 2

El uso de los diccionarios facilita la estructura de los datos en los diagramas de pastel, pues sus valores y sus calves se pueden extraer como arreglos y al hacer clic en algún colegio de la primera capa del diagrama de pastel, se visualizan las carreras pertenecientes a ese colegio en la segunda capa. Este proceso se logra a través de los índices de los arreglos principales y también por medio de las claves de cada diccionario.

En la tercera y última capa se utiliza también el archivo Excel con la información de los estudiantes, aquí se crean dos diccionarios principales donde el primero pertenece a los géneros de los estudiantes y el segundo a las provincias de cada estudiante. En el primer diccionario se almacena como clave la carrera y como valor un arreglo que contiene dos diccionarios más, estos diccionarios adicionales contienen como clave el género de los estudiantes y como valor el número de estudiantes pertenecientes a ese género. Para el segundo diccionario principal se utiliza como clave el nombre de la provincia y como valor el número de estudiantes registrados como en esa provincia.

Para la parte visual del diagrama de pastel se utiliza la librería chart.js, la cuál permite generar diferentes tipos de gráficos. Por otra parte, también se utiliza HTML para posicionar los elementos en la vista de la página, los cuales principalmente son un contenedor y una tabla con la información de los estudiantes. En el contenedor se encuentra el gráfico y para escribir el código que genera dicho gráfico se utiliza javascript. En este lado del frontend se reciben los diccionarios mencionados anteriormente para así generar el reporte completo.

Pruebas y Resultados

Página principal de los reportes

Para usar la aplicación se tiene como inicio la pagina que se visualiza en la Figura 4, en esta sección se tiene la opción de elegir cual reporte se quiere utilizar. La primera opción es el reporte de progreso de estudiantes donde se debe ingresar dos archivos Excel: el archivo que proviene del sistema HUBI y el archivo de las clases prerrequisito, además, se debe ingresar el link de la malla académica de la carrera que se quiere visualizar. La segunda opción es el reporte de aspirantes aceptados, aquí se debe ingresar un archivo Excel que proviene del sistema HUBI.

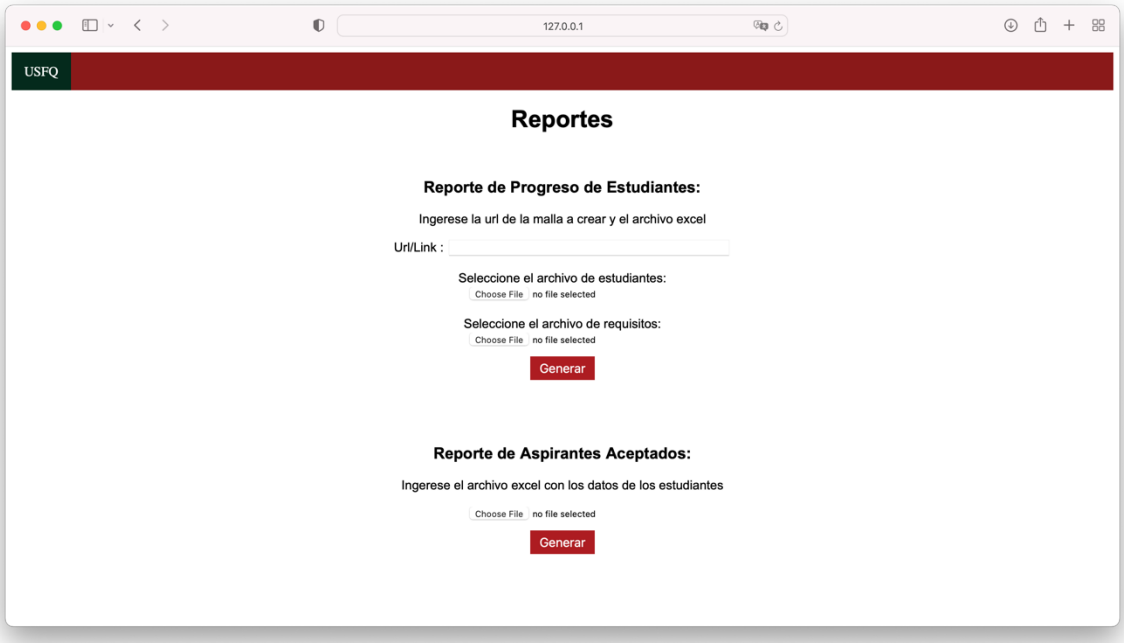

Figura 4. Página de inicio de la aplicación web

Reporte de progreso de estudiantes

Para el reporte de progreso de estudiantes se tiene el grafo resultante tras haber subido los dos archivos Excel junto con el link. En este grafo se visualiza la malla académica dividida por los años, dentro de los años académicos están los semestres y en cada semestre se visualiza el nombre de las clases junto con un contador que muestra el número de estudiantes que han cursado esa clase. Al ser este un grafo dirigido los vértices muestran que una clase es prerrequisito de la clase a la que va dirigida, tal como se muestra en la Figura 5.

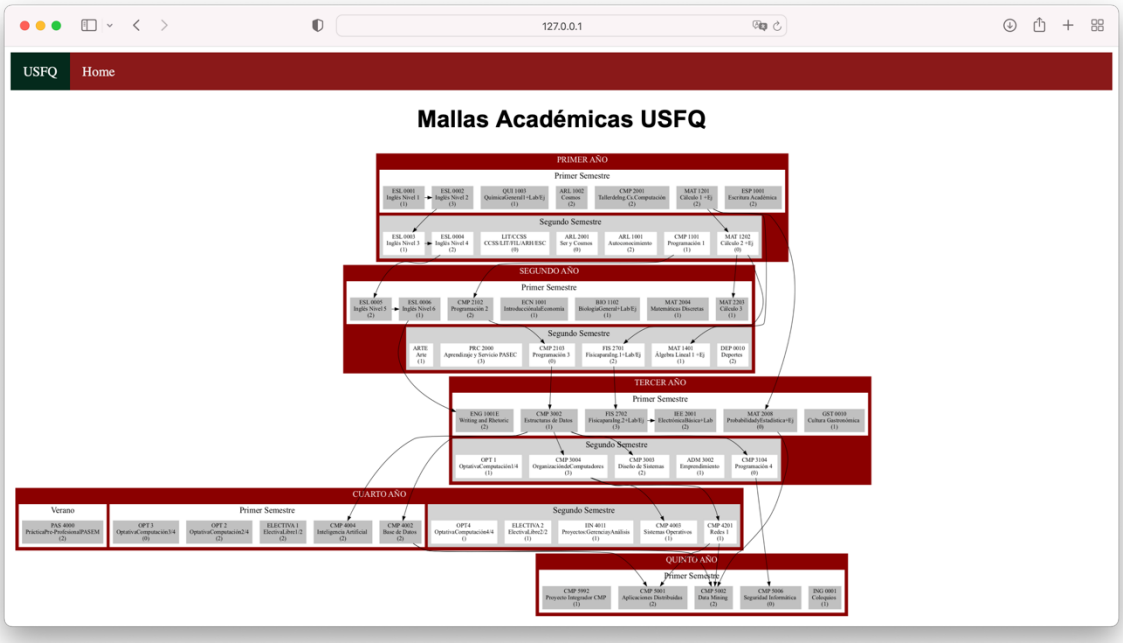

Figura 5. Grafo de la malla académica

Reporte de aspirantes aceptados

Para el reporte de aspirantes aceptados se tiene el primer escenario tras haber subido los archivos Excel correspondientes y el link de la malla académica en la página principal. En esta sección se visualiza la primera capa del diagrama de pastel que pertenece a los colegios académicos de la Universidad San Francisco de Quito USFQ. Estos se distinguen por los colores y en la parte derecha se encuentra una leyenda con la información del nombre y color de cada colegio, tal como se muestra en la Figura 6.

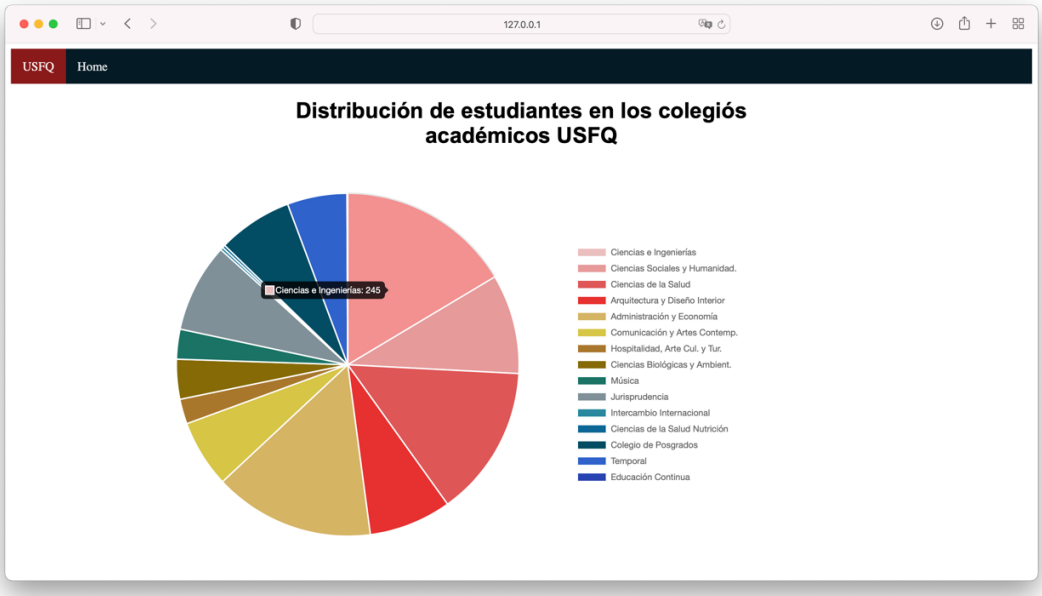

Figura 6. Primer nivel del diagrama de pastel

A partir de aquí se puede seleccionar cualquier sección del diagrama. Esto conlleva a que se muestre la segunda capa. En este nivel se visualizan las carreras pertenecientes al colegio seleccionado anteriormente. De igual manera, las carreras se muestran a la derecha y se diferencian por su color en la leyenda del diagrama. Tal como se muestra en la Figura 7.

Figura 7. Segundo nivel del diagrama de pastel

En el tercer y último nivel se muestra la distribución de los aspirantes aceptados por género y provincias. En esta sección se visualizan dos diagramas de pastel donde el primero del lado izquierdo pertenece a los géneros de los aspirantes y el segundo del lado derecho pertenece a las provincias a las que pertenecen los aspirantes. Cada uno de los diagramas de pastel tienen a su derecha la leyenda con sus etiquetas ordenadas por su color. Tal como se muestra en la Figura 8.

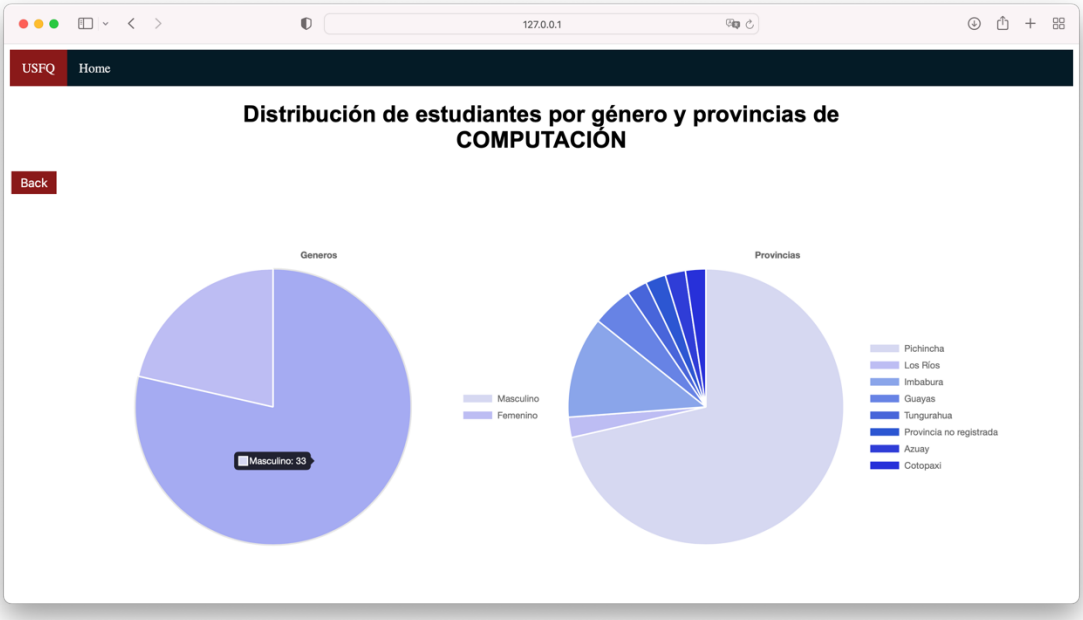

Figura 8. Tercer nivel del diagrama de pastel

Conclusiones, Recomendaciones y Trabajo futuro

Este proyecto se llevó a cabo usando principalmente Django como contenedor web lo que permite el uso del lenguaje Python para las partes de lógica en el backend. El uso de Python facilitó muchas de estas partes ya que este lenguaje es uno de los más sencillos de aprender y tiene la posibilidad de agregar una gran variedad de librerías que evitan desarrollar algoritmos desde cero.

Chart js es una librería diseñada para usarse con javascript y que de igual manera resulta muy útil para realizar este tipo de reportes, tiene varias extensiones que permiten personalizar el gráfico y logra crearlos visualmente atractivos e intuitivos para el público. En este proyecto resultó de gran ayuda para poder alcanzar los objetivos planteados al inicio. Además, si se quisiera agregar más clasificaciones en un trabajo a futuro, resulta muy fácil integrar al proyecto actual debido a las facilidades que ofrece esta librería.

En cuanto a la librería Graphviz para generar el grafo en el primer reporte, se tuvo varios inconvenientes para poder tener un orden visual atractivo en el grafo resultante, esta librería usa un lenguaje intérprete llamado dot que Python lo genera al usar su librería, sin embargo, esto causa limitaciones debido a que no se puede manipular por completo el lenguaje dot y como resultado no se pueda personalizar completamente el grafo. Por lo que, en este caso, para una nueva versión de la aplicación se recomendaría usar otra librería que no use un interprete de por medio como por ejemplo la librería Networkx o alguna otra de la variedad que existen disponibles.

Para este reporte también se tuvo otro inconveniente principal para generar los vértices. Debido a que la mayor parte de la información se obtiene del código fuente de la página de la universidad, este código no permitía el acceso hacia la estructura de los prerrequisitos de las materias ya que se necesitaba accesos hacia la base de datos de la universidad con credenciales. Por lo que en base a esto se planteó la solución de implementar un archivo Excel más que contenga las clases de prerrequisitos de la malla; sin embargo, esto limita la generación dinámica del grafo por medio de su link, por lo que en trabajos a futuro se puede plantear una solución para poder acceder a estos recursos y generar toda la malla de manera dinámica.

En general en el proyecto presentado se logró alcanzar los objetivos planteados al inicio de la creación de este, estos reportes serán de gran utilidad para el personal administrativo de la Universidad debido a las limitaciones actuales que se tenía con los reportes que genera el sistema HUBI. Además, al tener como base este proyecto, se puede ampliarlo a mayor escala y generar reportes más detallados que abarquen temas que no se encuentran actualmente.

Bibliografía.

Recursos para desarrolladores, por desarrolladores. (2021). *Introducción a Django*. Obtenido 18 de octubre del 2021 de https://developer.mozilla.org/es/docs/Learn/Server-side/Django/Introduction

Fernández, P. y Fernández, J. (2016). El lenguaje de los grafos. *Departamento de matemáticas*, (Nov 16). Obtenido 25 de octubre del 2021 de http://verso.mat.uam.es/~pablo.fernandez/entrega5-EDEM-MD16-17.pdf

Schiaffarino, A. (2019). Modelo cliente servidor. *Infranetworking*, (Mar 12). Obtenido 27 de octubre del 2021 de https://blog.infranetworking.com/modelo-cliente-servidor/