

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

**Diseño e implementación de una guía curricular basada en
inteligencia artificial**

Eduardo Humberto Guerrero Altamirano

Ingeniería en Ciencias de la Computación

Trabajo de fin de carrera presentado como requisito
para la obtención del título de
Ingeniero en Ciencias de la Computación

Quito, 13 de diciembre de 2024

UNIVERSIDAD SAN FRANCISCO DE QUITO USFQ

Colegio de Ciencias e Ingenierías

HOJA DE CALIFICACIÓN DE TRABAJO DE FIN DE CARRERA

**Diseño e implementación de una guía curricular basada en inteligencia
artificial**

Eduardo Humberto Guerrero Altamirano

Nombre del profesor, Título académico

Ricardo Flores Moyano, PhD

Quito, 13 de diciembre de 2024

© DERECHOS DE AUTOR

Por medio del presente documento certifico que he leído todas las Políticas y Manuales de la Universidad San Francisco de Quito USFQ, incluyendo la Política de Propiedad Intelectual USFQ, y estoy de acuerdo con su contenido, por lo que los derechos de propiedad intelectual del presente trabajo quedan sujetos a lo dispuesto en esas Políticas.

Asimismo, autorizo a la USFQ para que realice la digitalización y publicación de este trabajo en el repositorio virtual, de conformidad a lo dispuesto en la Ley Orgánica de Educación Superior del Ecuador.

Nombres y apellidos: Eduardo Humberto Guerrero Altamirano

Código: 00326712

Cédula de identidad: 1718867946

Lugar y fecha: Quito, 13 de diciembre de 2024

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UNPUBLISHED DOCUMENT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>.

AGRADECIMIENTOS

Primero quiero agradecer a Ricardo Flores, mi tutor de este proyecto, por su guía y apoyo. Su experiencia y acompañamiento fueron fundamentales para que pudiera completar este trabajo de la mejor manera posible.

A mi familia, especialmente a mis padres y hermano, su apoyo emocional, su amor incondicional y la confianza que han depositado en mí. Ustedes han sido mi mayor motivación y la razón por la cual dediqué tanto esfuerzo a este trabajo. A mis abuelos, quienes han sido como unos segundos padres para mí; gracias por ser ese pilar que no sabía que necesitaba, gracias por siempre buscar lo mejor para mí.

A la universidad San Francisco de Quito, por abrirme las puertas y darme a oportunidad de aprender, no solo acerca de computación, pero también sobre la vida, me han permitido crecer profesional y personalmente. A todos los profesores que formaron parte de mi educación, gracias por brindarme conocimiento de calidad, por su dedicación y por fomentar un ambiente de aprendizaje que siempre me motivó a seguir creciendo.

A mis amigos, tanto dentro y fuera de la universidad, por hacer que este último semestre estuviera lleno de alegrías, risas y recuerdos. Su energía positiva me ayudó a seguir adelante, a completar este trabajo y terminar el semestre de la mejor manera.

A todos, gracias.

DEDICATORIA

A mi madre, Mónica; mi padre, Jairo; y mi hermano, Nicolás.
Por estar siempre ahí para mí de manera incondicional, por confiar en mí incluso cuando yo no lo hacía, y por ayudarme a convertirme en la persona que soy hoy. Gracias por inspirarme a siempre ser mi mejor versión.

Este trabajo es suyo.

RESUMEN

Este proyecto presenta el diseño e implementación de una guía curricular orientada a mejorar el proceso de selección de carreras de los estudiantes de la Universidad San Francisco de Quito (USFQ). La guía curricular se implementa como un aplicativo web que aprovecha los sistemas de bases de datos relacionales, el marco Django para el backend, Bootstrap para el frontend y la inteligencia artificial para las recomendaciones. Aborda las ineficiencias en el sistema de búsqueda de cursos actual al filtrar cursos irrelevantes y proporcionar una visualización personalizada de materias obligatorias, optativas y opcionales. Las características clave incluyen autenticación de usuario, sistemas de recomendación basados en técnicas de procesamiento de lenguaje natural como TF-IDF y similitud del coseno, y una interfaz fácil de usar para agilizar la selección de cursos.

Palabras clave: Aplicación Web, Base de Datos, BERT, Bootstrap, Buscador de Cursos, Django, Guía curricular, Inteligencia Artificial, MySQL, Sistema de recomendación, TF-IDF.

ABSTRACT

This project presents the design and implementation of a curriculum guide aimed at improving the course selection process for students at Universidad San Francisco de Quito (USFQ). The curriculum guide is implemented as an application which leverages relational database systems, Django framework for backend, Bootstrap for frontend, and artificial intelligence for recommendations. It addresses inefficiencies in the current course search system by filtering irrelevant courses and providing a personalized display of mandatory, elective, and optional subjects. Key features include user authentication, recommendation systems based on natural language processing techniques like TF-IDF and cosine similarity, and a user-friendly interface to streamline course selection.

Key words: Artificial Intelligence, BERT, Bootstrap, Course Search Engine, Curriculum Guide, Database, Django, MySQL, Recommendation System, TF-IDF, Web Application.

TABLA DE CONTENIDO

Introducción	14
Objetivos.....	16
Objetivo general.....	16
Objetivos específicos	16
Desarrollo del Tema	18
Estado del arte.....	18
Desarrollo del proyecto	24
Base de Datos	25
Diseño del diagrama de entidades y modelo relacional.....	25
Implementación de la base de datos.....	27
Inserción de tablas	27
Inserción de datos	28
Creación de Índices.....	29
Creación de triggers.....	30
Creación de Stored Procedures.....	30
Back end con Django	31
Creación del proyecto	31
Vinculación con la base de datos.....	32
Módulo de autenticación del usuario	34
Buscador de cursos	35
Front end con bootstrap.....	38
Sistema de recomendación	39
Palabras clave.....	39
Creación de los vectores.....	42
Búsqueda de la similitud	42
Implementación en Django	44
Pruebas de funcionalidad.....	45
Base de datos	45
Proyecto Django	50
Sistema de Recomendación	53
Conclusiones.....	55
Trabajo futuro.....	55
Referencias bibliográficas.....	57
Anexo A: Modelo relacional de la base de datos	61
Anexo B: Inserción de tablas para la base de datos.....	62
Anexo C: Detalle de las inserciones en base de datos.....	70
Anexo D: Script de la vista table_user.....	71
Anexo E: Resultados de tf_idf.....	72

Anexo F: Pruebas de la base de datos	76
Anexo G: Pantallas de la página web	81

ÍNDICE DE TABLAS

Tabla I Procedimientos almacenados.....	30
Tabla II Métodos POST de la vista table_user	37
Tabla III Variantes de tf_idf	41
Tabla IV Modelos usados para Keybert	42
Tabla V Métricas de distancia y similitud	43
Tabla VI Inserción de la tabla Colegio.....	62
Tabla VII Inserción de la tabla KeyWords.....	62
Tabla VIII Inserción de la tabla Semestre	62
Tabla IX Inserción de la tabla Notas.....	62
Tabla X Inserción de la tabla Categoría	62
Tabla XI Inserción de la tabla Profesor.....	62
Tabla XII Inserción de la tabla Dias	63
Tabla XIII Inserción de la tabla Horario	63
Tabla XIV Inserción de la tabla Edificio.....	63
Tabla XV Inserción de la tabla Modalidad.....	63
Tabla XVI Inserción de la tabla Carrera	63
Tabla XVII Inserción de la tabla Subespecialización	64
Tabla XVIII Inserción de la tabla Estudiante	64
Tabla XIX Inserción de la tabla Palabra_Estudiante	64
Tabla XX Inserción de la tabla Asignatura	65
Tabla XXI Inserción de la tabla Prerrequisito	65
Tabla XXII Inserción de la tabla Subespecialización	65
Tabla XXIII Inserción de la tabla MallaSubEsp_Asignatura.....	66
Tabla XXIV Inserción de la tabla Malla_Carrera.....	66
Tabla XXV Inserción de la tabla MallaCarreraAsignatura	66
Tabla XXVI Inserción de la tabla Campus.....	66
Tabla XXVII Inserción de la tabla Kardex.....	67
Tabla XXVIII Inserción de la tabla Kardex_Asignatura	67
Tabla XXIX Inserción de la tabla Curso	67
Tabla XXX Inserción de la tabla Dias_Asignatura.....	68
Tabla XXXI Inserción de la tabla Correquisito	68
Tabla XXXII Inserción de la tabla Restricciones	68
Tabla XXXIII Inserción de la tabla Audit_Log.....	69
Tabla XXXIV Inserción de la tabla Malla_Extras.....	69
Tabla XXXV Cantidad de Inserción de datos por tabla.....	70
Tabla XXXVI Funciones de la vista table_user	71
Tabla XXXVII Variables globales de la vista table_user	71
Tabla XXXVIII EventListeners de la vista table_user	71
Tabla XXXIX Variantes utilizadas de tf_idf.....	72
Tabla XL Palabras Clave con tf_idf.....	72

ÍNDICE DE FIGURAS

Fig. 1 Arquitectura de la aplicación	25
Fig. 2. Diagrama de entidades	26
Fig. 3 Creación de la base de datos.....	27
Fig. 4 Inserción de la tabla Estudiantes.....	28
Fig. 5 Creación de Índices	30
Fig. 6 Creación de trigger para la eliminación de un elemento de la tabla Colegio	30
Fig. 7 Comando para crear proyecto en Django	32
Fig. 8 Comando para crear aplicaciones en Django.....	32
Fig. 9 Conexión a la base de datos con Django	32
Fig. 10 Nuevas tablas creadas por Django	33
Fig. 11 Información de los usuarios en la tabla auth_user	33
Fig. 12 Modelo de la tabla Estudiante.....	34
Fig. 13 Vista del login	35
Fig. 14 Vinculación del path en views.py	35
Fig. 15 Template del login.....	35
Fig. 16 Desarrollo de la vista table_user	36
Fig. 17 Llamada a store procedures en la vista table_user	36
Fig. 18 Llamada al URL de Bootstrap en el código html	39
Fig. 19 Script para guardar la descripción de las asignaturas.....	40
Fig. 20 Actualización de las palabras clave.....	42
Fig. 21 Matriz de Acuerdo entre las métricas de distancia	44
Fig. 22 Datos del estudiante de prueba	45
Fig. 23 Información del usuario de prueba.....	50
Fig. 24 Flujo para escoger una nueva subespecialización.....	50
Fig. 25 Visualización de las palabras clave.....	51
Fig. 26 Flujo para cambiar las palabras clave	51
Fig. 27 Asignaturas disponibles para el usuario de prueba	51
Fig. 28 Flujo para elegir nuevas asignaturas que desplegar	52
Fig. 29 Asignaturas ya tomadas por el usuario	52
Fig. 30 Verificaciones con otros usuario.....	53
Fig. 31 Prueba de funcionalidad sistema de recomendación.....	54
Fig. 32 Modelo Relacional	61
Fig. 33 GetMallaCarrera para Ing. en Ciencias de la Computacion	76
Fig. 34 GetKardexEst para el estudiante de prueba	76
Fig. 35 ObtenerAsignaturasRestantes para el estudiante de prueba	77
Fig. 36 ObtenerDetallesdeAsignatura para ver prerrequisito de una asignatura	77
Fig. 37 VerificarPrerrequisitos para el estudiando de prueba y la materia Sistemas Lean	77
Fig. 38 ObtenerCorrequisito	77
Fig. 39 ObtenerInformacionAsignatura para ver las restricciones	78
Fig. 40 VerificarRestriccionesEstudiante para el estudiando de prueba y la asignatura de Programación Avanzada de Apps	78
Fig. 41 GetOptativas para el estudiante de prueba	78
Fig. 42 GetElectivas para el estudiante de prueba	79
Fig. 43 VerCursos	79
Fig. 44 getSubespecializacion para el minor de Cine	79
Fig. 45 verSubTomadas para un estudiante que toma el minor de Cine	80
Fig. 46 verSubRestantes para un estudiante que toma el minor de Cine	80

Fig. 47 Inicio de Sesión.....	81
Fig. 48 Buscador de Cursos.....	82
Fig. 49 Historial de Asignaturas	83

INTRODUCCIÓN

La Universidad San Francisco de Quito (USFQ) enseña bajo la filosofía de Artes Liberales. Cada estudiante debe escoger su horario de clases para un nuevo semestre mediante la revisión de la oferta de cursos existente, que presenta diferentes opciones de horario, días, profesores y modalidades. Además, la USFQ también requiere que los estudiantes tomen clases de colegio general, optativas, electivas, al menos una clase en inglés y clases de diferentes categorías, como artes, humanidades, ciencias sociales o ciencias exactas.

El estudiante puede revisar esta oferta en el buscador de cursos, una página web que presenta todos los cursos disponibles para un semestre. Si bien es requerido que un alumno ingrese con su cuenta institucional, no hay cambio alguno en la página web, la cual sigue presentando una gran cantidad de asignaturas que se ofrecen en la universidad. Estas incluyen, en su mayoría, clases que no son de utilidad para el estudiante, como aquellas que ya ha cursado y aprobado y aquellas que no puede cursar, debido a prerrequisitos o restricciones.

Usualmente, al buscar un curso, cada usuario tiene un listado entre seis y cuatro clases en las que debe inscribirse. Sin embargo, la página no permite verlas todas juntas y se requiere realizar una búsqueda individual de cada una, forzando al estudiante a anotar la información en algún lugar aparte, abrir varias pestañas o constantemente realizar las mismas búsquedas.

Específicamente con las optativas y electivas, los estudiantes enfrentan dificultades para encontrar asignaturas que se ajusten a su preferencia. En el buscador actual, la oferta de cursos es tan alta que, buscar una asignatura representa un gran desafío. Esto termina en un gasto de tiempo, el cual no siempre deja satisfecho al usuario.

En este contexto, el proyecto busca mejorar el proceso de revisión y búsqueda de cursos, para entregar información personalizada. A través de los datos académicos del estudiante se propone diseñar e implementar una aplicación que proporcione una guía curricular para

resolver los problemas antes mencionados, tomando ventaja del hecho de que el estudiante requiere iniciar sesión con su cuenta académica. Con esto tan solo se presentarán las materias que el estudiante tiene pendiente por cursar, según la malla académica de su carrera y subespecialización que tome. Además, para materias no específicas, como optativas, electivas, o de alguna categoría, se tomará en cuenta si el estudiante cumple o no con los requisitos para tomarla.

La guía curricular consta de tres componentes principales. Primero, el diseño de la estructura académica en una base de datos relacional; segundo, el diseño e implementación de la página web mediante la programación de su front y back end; y, por último, la implementación de un sistema de recomendaciones para la búsqueda de materias optativas y electivas.

Debido a restricciones y políticas de privacidad, no es posible acceder a información real sobre la estructura académica de los estudiantes de la USFQ. Por esta razón, es necesario emular dicha estructura en una base de datos externa. Este diseño busca ser lo más parecido al actual de la universidad, incluyendo información de cada carrera, colegio, malla curricular, entre otros. La base de datos permitirá establecer relaciones entre cada componente, para crear una estructura flexible. Igualmente, se espera que este modelo pueda servir para proyectos futuros. Reutilizar la estructura, permitirá a los estudiantes presentar diseños y propuestas para la universidad, con una estructura y datos similares. Esto ahorra tiempo en el diseño de una nueva estructura desde cero y permite enfocar los recursos en nuevas implementaciones o mejoras.

Segundo, el desarrollo de la página web busca ser una nueva propuesta para la forma en la que el estudiante puede visualizar los cursos disponibles, dando prioridad a aquellos cursos en los que debe inscribirse el próximo semestre, considerando que cumpla con los prerrequisitos y restricciones. Para esto se obtendrá información como la carrera, semestre y

kardex académico del usuario, y comparar aquellas asignaturas que ya ha aprobado, con aquellas en la malla académica de su carrera que le faltan por cursar.

Finalmente, la aplicación web contará con un sistema de recomendación, basado en el filtrado de contenido, el cual, mediante un máximo de diez palabras clave elegidas por el usuario de una lista, entregará un ranking de materias recomendadas, verificando que se cumplan requisitos y prerrequisitos. Este procedimiento se realizará, mediante inteligencia artificial y minería de texto, se implementarán técnicas de procesamiento de lenguaje natural (NLP por sus siglas en inglés), frecuencia de texto y frecuencia inversa de documento (TF-IDF por sus siglas en inglés), y similitud del coseno.

Objetivos

Objetivo general

- Desarrollar una guía curricular como una aplicación web que permita la visualización de los cursos obligatorios, electivos y optativos de los estudiantes, se proporcionará un sistema de recomendaciones para simplificar el proceso de selección de cursos.

Objetivos específicos

- Diseñar una base de datos relacional que emule de forma precisa la estructura académica de la universidad.
- Implementar un sistema de autenticación que permita leer la información académica de un estudiante, según su usuario y contraseña.
- Desarrollar un sistema de verificación que identifique los cursos obligatorios para cada estudiante, considerando su información académica y los requisitos de cada materia.

- Desarrollar un sistema de recomendaciones basado en inteligencia artificial que presente materias optativas y electivas al estudiante, según sus preferencias.
- Desarrollar una aplicación web con interfaz intuitiva, que permite visualizar los cursos obligatorios y las recomendaciones de materias optativas y electivas.

DESARROLLO DEL TEMA

Estado del arte

Todo el diseño de la estructura académica de la universidad será implementado en una base de datos relacional, con el objetivo de organizar y normalizar los datos de manera eficiente. Las bases de datos son conjuntos estructurados de información almacenada en sistemas informáticos, cuyo propósito es facilitar el acceso, la modificación y la gestión de la información [1]. Estas bases de datos permiten realizar operaciones de creación, lectura, actualización y eliminación de datos, conocidas como operaciones CRUD (Create, Read, Update, Delete) [2].

Existen dos tipos principales de bases de datos utilizadas en la actualidad: las relacionales y las no relacionales. Para este proyecto, se optará por las bases de datos relacionales, que organizan la información en tablas relacionadas mediante tuplas [1]. Este tipo de organización facilita la ejecución de consultas más complejas y permite una presentación uniforme y comprensible de los datos, lo que resulta ideal para modelar sistemas estructurados como el académico. Además, las bases de datos relacionales siguen el principio ACID (Atomicidad, Consistencia, Aislamiento y Durabilidad), que presentan confiabilidad y seguridad en las transacciones. Una transacción es un conjunto de operaciones CRUD [3].

La atomicidad garantiza la totalidad de una transacción, es decir, asegura que se realicen todas las operaciones CRUD, o no se realice ninguna. La consistencia protege la integridad de los datos, y evita que existan errores o datos corruptos. El aislamiento asegura que una transacción no afecte a otra, haciendo que cada transacción se realice independiente del resto. La durabilidad permite que cualquier cambio hecho por una transacción, perdure de forma permanente [4].

Si bien existe la opción de realizar la estructura académica universitaria por medio de

una base de datos no relacional, no es recomendable para el tipo de proyecto que se está llevando a cabo. Por un lado, las bases de datos no relacionales no siguen el principio de ACID, se guían por CAP, (Consistencia, Disponibilidad y Tolerancia al Particionamiento) [4]. Al no tener el principio ACID, las transacciones de la base de datos no están completamente protegidas. Esto puede traer problemas considerando que se trabaja con información académica, la cual, si tiene errores de inserción, actualización o eliminación de datos, generaría un problema de inconsistencia de datos que traería complicaciones al estudiante y la institución.

Por último, al no ser organizada por relaciones, una base de datos no relacional es ineficiente cuando se realizan consultas de múltiples criterios [5]. Esto es crítico, debido a que la búsqueda de cursos se basa en varios criterios de información. En conclusión, una base de datos no relacional, al no tener propiedades ACID, pone en riesgo la consistencia de datos y disminuye la fiabilidad de la información presentada. Además, al ser necesario realizar consultas de varios criterios, el tiempo de consulta puede ser muy alto en comparación con una base de datos relacional.

Para administrar la base de datos, se requiere de un sistema gestor de base de datos (SGBD). En este caso, se optó por MySQL, un SGBD reconocido por su simplicidad y rendimiento [6]. Debido a que se trata de un sistema de libre uso, que puede ser implementado tanto en sistemas operativos de Windows y Linux [6], se convierte en una herramienta flexible y adaptable que permite alta escalabilidad y desarrollo. Este permite el uso de diferentes variables que serán necesarias en el diseño, como variables numéricas, booleanas y caracteres de texto; además, permite colocar restricciones a la inserción de datos.

Junto con estas características, MySQL permite la declaración de claves foráneas (FK por sus siglas en inglés), necesarias para asociar entidades entre sí [7]. Asimismo, se pueden programar acciones que permitan modificar una variable dependiente de otra. Por ejemplo, si se eliminara una asignatura, todos los cursos de la misma también se eliminarían. El sistema

también permite crear triggers, un procedimiento que ocurre automáticamente tras ejecutar un evento DML (cualquier acción que modifique un objeto de una tabla, incluyendo eliminar, insertar o modificarlo), para modificar algún dato o tabla [8].

Por último, también se permite crear índices y procedimientos almacenados. Los índices permiten optimizar el rendimiento de una búsqueda [6]. Los procedimientos almacenados son similares a funciones, se define uno o más atributos y se realiza un evento CRUD [8]. Esto permite llamar procesos de búsqueda complejos de forma rápida y asegurándose que siempre se cumplan los mismos parámetros.

Para el desarrollo de la aplicación, se utilizará el lenguaje de programación Python. Python es un lenguaje de programación de propósito general y código abierto. Se lo considera amigable para el usuario debido a su sintaxis y lectura de alto nivel [9]. Sin embargo, debido a que es de propósito general, es utilizado en varios campos de la computación, incluyendo el análisis de datos, la inteligencia artificial y el desarrollo web, tanto en front end como en back end [10].

En el desarrollo web se requiere de frameworks para facilitar la implementación de algunas funcionalidades, se puede definir como “una aplicación genérica incompleta y configurable, con directrices arquitectónicas ofreciendo al desarrollador un conjunto de herramientas para agilizar el proceso de construir una aplicación web concreta” [11]. Para este proyecto, se requieren dos frameworks, uno que trabaje en el front end de la página, y otro en el back end. El front end gestiona la interacción de los usuarios, mientras que el back end maneja la lógica del programa; ambos se comunican entre ellos mediante llamadas a una API [12].

En el front end, un framework crea la arquitectura, el comportamiento y el diseño de todo lo que es visible en la pantalla [12]. Existen varios frameworks utilizados para el desarrollo de aplicaciones web, como Angular JS, React JS, Vue o Next JS [12]. Sin embargo,

debido a que se decidió que en el proyecto se trabajará con Python, se optó por utilizar Bootstrap. Bootstrap, si bien no es un framework específico para Python, permite una integración más sencilla con frameworks de back end que requieran Python [12]. Además, Bootstrap es conocido por sus componentes prediseñados y por sus aplicaciones web con respuesta rápida [12].

Para el back end, un framework facilita el desarrollo de la lógica del programa, mediante el uso de herramientas preestablecidas [9]. Dado que se va a trabajar con Python, existen opciones como Django, Pyramid, Turbogear o WebPy. Sin embargo, tras una investigación, se encuentra que, al comparar los cuatro frameworks en una prueba de calidad basada en la norma ISO/IEC 9126, Django superó en puntaje a los otros tres en los aspectos de “funcionalidad”, “fiabilidad”, “usabilidad”, “eficiencia”, “mantenibilidad” y “portabilidad” [9]. Por esta razón, se opta por usar Django, un framework de código abierto, que soporta el uso de bases de datos con SQL [12].

Django tiene un modelo inspirado en el Vista Controlador (MVC), conocido como el Modelo Vista Template (MVT). “Modelo” hace referencia a la capa que accede a la información de los datos, incluyendo sus relaciones. “Vista” es la capa lógica, que une el modelo con la plantilla (template). “Template” es la capa de presentación, donde se determina que información es mostrada al usuario y cual se mantiene oculta [13],

Bootstrap en front end y Django en back end han sido utilizados comúnmente en aplicaciones web, debido a sus características combinadas de consistencia, escalabilidad y desarrollo rápido [14]. Se ha usado esta combinación para el desarrollo de aplicaciones de varios propósitos, como páginas de turismo [14], mantenimientos del cuerpo de bomberos [15], evaluación de código [16], procesos de venta [17], entre otros. Esta variedad resalta la flexibilidad y adaptabilidad de ambos frameworks, haciéndolos una opción ideal para el desarrollo de aplicaciones web con requisitos variados y complejos.

Con el fin de poder presentar al estudiante diferentes materias optativas y electivas que se acomoden a sus gustos, se implementará un sistema de recomendaciones. Un sistema de recomendación es un sistema que filtra información para predecir elementos que sean de preferencia del usuario [18]. Actualmente, este tipo de sistemas se realizan mediante la implementación de inteligencia artificial. Se encuentran en plataformas de servicio de streaming, como Netflix, Spotify o Amazon Prime. En plataformas de educación en línea como Coursera, Edx y Udemey, o en redes sociales, como Facebook, Instagram, Tiktok. La intención de estos sistemas es proveer contenido personalizado al usuario, adaptándolo a sus preferencias, disminuyendo el tiempo de búsqueda del consumidor y facilitando la toma de decisiones.

Para implementar estas herramientas, las dos técnicas más comunes son, filtrado basado en contenido y filtrado colaborativo. El primero analiza la información del usuario, con sus gustos y preferencias, para recomendar elementos [19]. Esta información se puede obtener de manera implícita, aprendiendo de los comportamientos y elección previas del consumidor, o de manera explícita, por medio de una encuesta, cuestionario u otro método de recolección de datos [20]. Una vez que se obtiene esta información, se realiza un análisis en el contenido, buscando coincidencias en los textos, etiquetas, nombres, entre otros. Sin embargo, una limitación de esta técnica es que elementos no textuales pueden tener mayor problema a la hora de ser analizados [21].

El otro método es el filtrado colaborativo. Como su nombre lo indica, este sistema basa sus recomendaciones en coincidencias que otros usuarios hacen sobre ciertos elementos [19]. La recomendación se puede basar en el usuario, donde se busca perfiles similares y analiza los productos que han escogido, para recomendarlos entre ellos. La otra forma es por medio de elementos, donde se toman usuarios que han valorado o utilizado el mismo producto, para recomendarse entre sí, productos que hayan tomado previamente [20]. No obstante, esta forma

de recomendación puede tener problemas de escasez de datos, escalabilidad y diversidad, dependiendo de las características del proyecto en el cual se desea implementar el sistema [19].

Existe igual el filtrado híbrido, el cual busca combinar ambas técnicas (u otras técnicas menos conocidas) para limitar los problemas que pueden tener los métodos individuales [20]. Este enfoque es el más común en empresas tecnológicas como Netflix, el cual analiza las preferencias del usuario de forma explícita, al pedir que den una calificación a una serie o película, de forma implícita al observar el contenido que observa, y por medio de comparar con otros usuarios, con perfiles y gustos similares [20]. Si bien este modelo podría considerarse el más práctico, también se trata del más costoso [21].

Para el proyecto, se optó por usar el modelo filtrado por contenido, obteniendo información del usuario de forma explícita a través de la elección de keywords, para luego analizar la descripción y nombre de los cursos, y generar recomendaciones. Se decidió no usar el filtrado colaborativo, debido a la posible variación entre optativas y electivas elegidas por los estudiantes. Para este filtrado, se va a implementar técnicas de procesamiento natural de lenguaje (NLP) en las descripciones y títulos de los cursos. Primero, se aplicará un pre procesamiento, el cual permitirá eliminar palabras irrelevantes que no otorguen información alguna. Seguido, se realizará un proceso de lematización, el cual convertirá las palabras a su base, evitando redundancia [22]. Tercero, se aplicará el proceso de frecuencia de término y frecuencia inversa del documento (TF-IDF), para identificar aquellas palabras importantes en cada curso. TF-IDF determina la frecuencia relativa de una palabra en comparación con el inverso proporcional de la frecuencia en el conjunto de documentos que se le presenta. Esto, para determinar la relevancia de la palabra [23]. Con esta práctica, se puede obtener las palabras claves que se presentarán al usuario.

Una vez que el usuario escoja sus palabras clave, se procede a comparar dichas palabras con el documento, implementando la similitud del coseno. La similitud funciona debido a que,

tanto los documentos como las palabras claves del usuario (conocido como query) se colocan en un espacio vectorial. Entonces, se utiliza el coseno para determinar el ángulo de separación entre el query con cada documento, y así identificar que el documento con menor separación al query, es el que presenta mayor similitud [23]. Por lo que, se puede crear un ranking de asignaturas, basadas en la similitud con las palabras clave.

Existen otros proyectos que han optado por usar únicamente filtrado de contenido. Por ejemplo, un sistema de evaluación de desarrollo de software [24], donde el usuario ingresa parámetros, y el sistema examina el contenido y recomienda un puntaje [24]. También se utilizó para sugerir universidades a los estudiantes, según su interés y su GPA [25]. Por último, se ha implementado un sistema de recomendación de libros; se analiza los libros que ha leído el usuario y se compara las características con las de otros libros, para dar un puntaje de recomendación [26].

Desarrollo del proyecto

Toda el código del proyecto se encuentra almacenado en un repositorio en Github.¹

Para la implementación de la base de datos y el diseño de la aplicación web, se usará el editor de código Visual Studio Code (VS Code), un editor de código *free source*, que permite programar en varios lenguajes debido a su amplia oferta de extensiones [26], en su versión 1.93.1. Para la aplicación, los distintos módulos van a funcionar según la arquitectura expuesta en la Fig. 1.

¹ https://github.com/Edu-Guerrero/Proyecto_Integrador

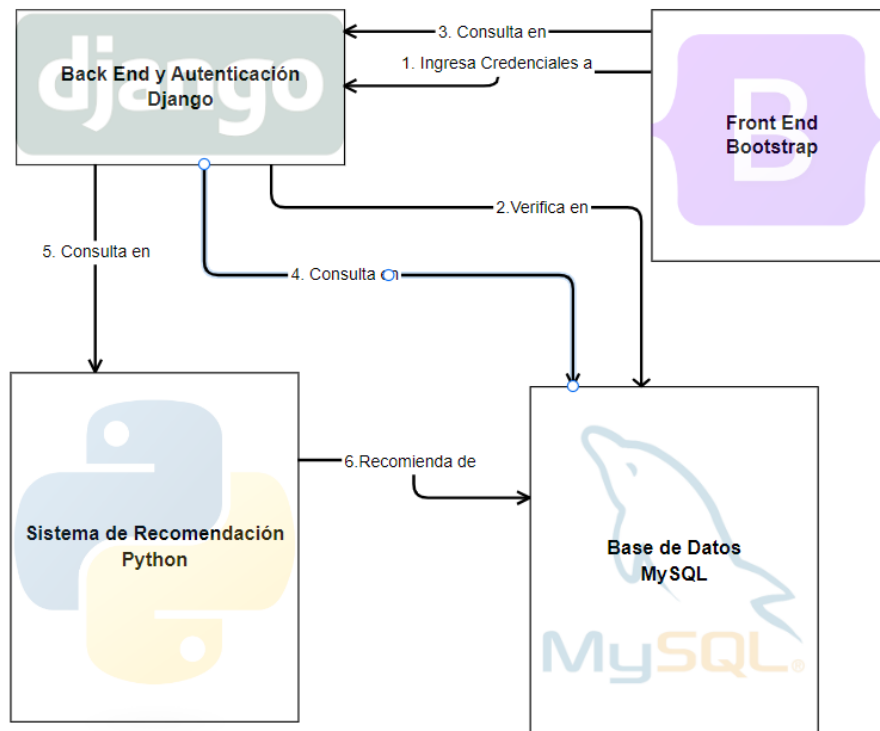


Fig. 1 Arquitectura de la aplicación

Base de Datos

En el desarrollo de la base de datos, se utiliza la extensión MySQL de Weijan Chen para la conexión con el servidor en el que se encuentra la base de datos, y MySQL Syntax de Jake Bathman para facilitar la escritura de código en el lenguaje sql.

Diseño del diagrama de entidades y modelo relacional

La estructura de la base de datos se define inicialmente mediante un diagrama de entidades y modelo relacional, donde se especificarán todas las entidades (tablas) que estarán presentes en la guía curricular, y la relación que comparten cada una de ellas. Este diagrama está hecho bajo el concepto de normalización, el cual tiene como objetivo la simplificación entre las relaciones de los campos de dos tablas [27], y las normas formales 1FN, 2FN y 3FN, las cuales permiten reducir errores, asegurar que las columnas que no son llaves sean dependientes de la llave primaria y evitar dependencias transitivas [27].

El diagrama de entidades muestra a los actores principales y su relación con cada uno.

Como se observa en la Fig. 2, existen 23 entidades propias, cada una con una relación específica a otra tabla, a excepción de Audit_log. Una vez definidas las relaciones, se procede a ingresar una tabla media entre aquellas entidades que compartan una relación muchos a muchos. Además, se definen las llaves primarias (PK por sus siglas en inglés), los identificadores únicos de cada entidad; las llaves foráneas (FK por sus siglas en inglés), aquellas que permitirán relacionar a la tabla actual con otras tablas; y el resto de atributos que pertenecen a la entidad.

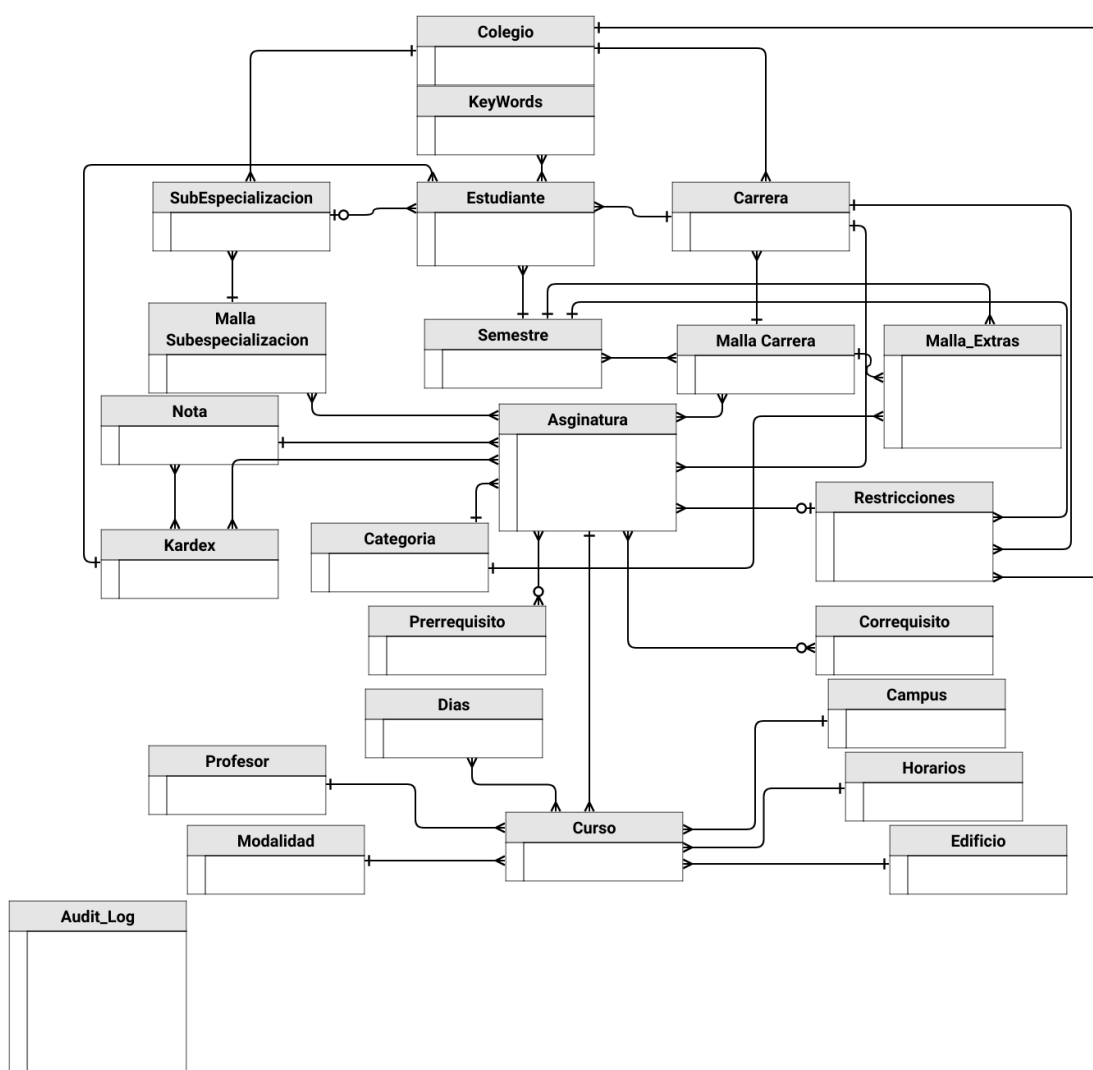


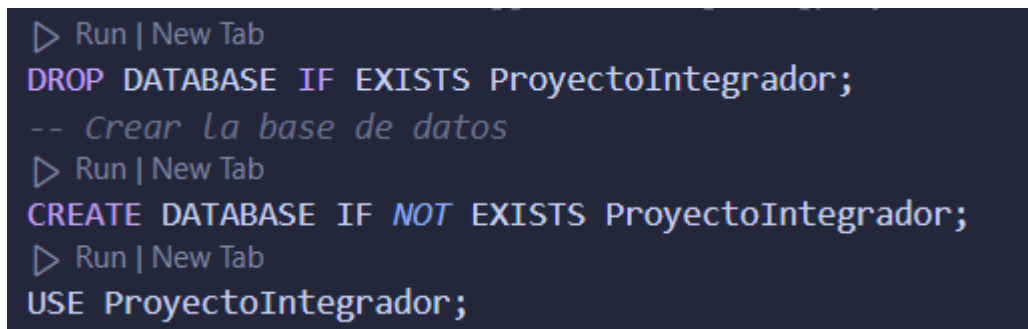
Fig. 2. Diagrama de entidades

Lo previamente mencionado se encuentra en el modelo relacional. Este modelo ahora cuenta con 29 entidades y define donde se encontrarán las relaciones entre cada entidad, para

visualizar este esquema, ver Anexo A: Modelo relacional de la base de datos. Una vez que se han realizado los cambios, se dice que la base de datos está normalizada. Sin embargo, por motivos de diseño y facilidad de uso, la tabla de prerequisites no ha sido normalizada y su relación muchos a muchos se mantiene.

Implementación de la base de datos

La base de datos se implementa como un servicio local, en la dirección 127.0.0.1 y el puerto 3306. Se utilizó el código DROP DATABASE IF EXISTS para eliminar la base de datos si ya existiera y luego CREATE DATABASE para crear la base de datos vacía, como se muestra en la Fig. 3.

A screenshot of a terminal window with a dark background. It shows three SQL commands being executed, each preceded by a 'Run | New Tab' prompt. The first command is 'DROP DATABASE IF EXISTS ProyectoIntegrador;'. The second command is a comment '-- Crear la base de datos'. The third command is 'CREATE DATABASE IF NOT EXISTS ProyectoIntegrador;'. Below these, there are two more 'Run | New Tab' prompts, followed by the command 'USE ProyectoIntegrador;'.

```
▷ Run | New Tab
DROP DATABASE IF EXISTS ProyectoIntegrador;
-- Crear la base de datos
▷ Run | New Tab
CREATE DATABASE IF NOT EXISTS ProyectoIntegrador;
▷ Run | New Tab
USE ProyectoIntegrador;
```

Fig. 3 Creación de la base de datos

Inserción de tablas

La inserción de todas las tablas se basó en el modelo relacional presentado en el Anexo A: Modelo relacional de la base de datos. De igual forma, se inicia con un DROP TABLE IF EXISTS para eliminar la tabla en caso de que haya existido previamente. Además, se agregan las relaciones de FK, algunas revisiones básicas a la hora de insertar variables y la sentencias ON UPDATE, ON DELETE, las cuales son acciones que protegen a las entidades en caso de que otra entidad relacionada se elimine o actualice.

```

-- Eliminar la tabla Estudiante si ya existe
> Run | New Tab
DROP TABLE IF EXISTS Estudiante;

-- Crear la tabla Estudiante
> Run | New Tab | Copy
CREATE TABLE Estudiante (
    Estudiante_id INT NOT NULL PRIMARY KEY, -- Identificador único del estudiante (código banner)
    Nombres VARCHAR(255) NOT NULL,         -- Nombres del estudiante
    Apellidos VARCHAR(255) NOT NULL,        -- Apellidos del estudiante
    Correo VARCHAR(255) NOT NULL,           -- Correo institucional
    Password VARCHAR(255) NOT NULL,         -- Contraseña del estudiante
    Carrera_id INT NOT NULL,
    Semestre_id INT NOT NULL,
    SubEsp_id INT NULL,
    CHECK (Correo LIKE '%@estud.usfq.edu.ec'), -- Verificar que el correo termine en "@estud.usfq.edu.ec"
    CHECK (Estudiante_id REGEXP '^[0-9]{6}$'), -- Verificar que el ID sea de 8 dígitos, comience con 00
    FOREIGN KEY (Carrera_id) REFERENCES Carrera(Carrera_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE, -- FK referencia a la tabla Carrera
    FOREIGN KEY (Semestre_id) REFERENCES Semestre(Semestre_id)
        ON DELETE RESTRICT
        ON UPDATE CASCADE, -- FK referencia a la tabla Semestre
    FOREIGN KEY (SubEsp_id) REFERENCES SubEspecializacion(SubEsp_id)
        ON DELETE SET NULL
        ON UPDATE CASCADE -- FK referencia a la tabla SubEspecializacion
);

```

Fig. 4 Inserción de la tabla Estudiantes

Como se puede observar en la Fig. 4, estudiante tiene dos verificaciones usando expresiones regulares que aseguran que el correo termine en “@estud.usfq.edu.ec” y que la variable Estudiante id, la cuál es la PK y simula el código del estudiante, contenga un total de seis números (no se consideran los dos primeros ceros que suelen estar presentes en el código). Además, las FK que asocian al estudiante con su carrera, semestre y subespecialización en caso de tenerla, contiene sentencias de restricciones. Primero, si se tratara de eliminar una carrera, pero esta está asociada a, al menos, un estudiante, no se permite la eliminación, lo mismo ocurre con un semestre. Segundo, si se modificara la PK de una carrera, semestre o subespecialización, todos los estudiantes que estén asociados a alguna de estas PK tendrán esos campos actualizados. Tercero, si se elimina una subespecialización, se eliminará ese campo en todos los estudiantes que la hayan estado tomando.

Para una vista más detallada de cada tabla, ver Anexo B: Inserción de tablas para la base de datos

Inserción de datos

Considerando que se pretende que el contenido de la base de datos represente fielmente

la realidad, se obtuvo información real de tres páginas web de la USFQ, las cuales contenían la información de los profesores, carreras y subespecializaciones. Para usar esta información, se inspeccionó el código HTML de cada una y mediante un script de Python se extrajo la información relevante implementando web scraping. El script utiliza BeautifulSoup, una librería que permite el análisis de HTML [28]. Gracias a esto, se insertó información verídica a las tablas de Profesor, Carrera y Subespecialización. Además, se inspeccionó el código HTML de cinco páginas que contienen asignaturas de las carreras de Ing. en Ciencias de la Computación, Gastronomía, Educación, Administración de Empresas y Artes Visuales; y cinco páginas con las asignaturas de las subespecializaciones de Videojuegos, Matemática, Danza, Ing. Industrial y Cine.

También, se implementó otro script de Python que genere datos al azar para crear cursos (por cada asignatura se obtienen cinco cursos), estudiantes, los días disponibles de un curso y entregar a los estudiantes palabras clave.

Las tablas de Días, Modalidad, Campus, KeyWords fueron insertadas manualmente con la información real de la universidad. La tabla de Prerrequisitos igual fue insertada manualmente para asignaturas de las carreras de Ing. en Ciencias de la Computación y las subespecializaciones.

Para un mayor detalle acerca de la cantidad y tipo de inserciones realizadas, ver Anexo C: Detalle de las inserciones en base de datos.

Creación de Índices

Como se mencionó previamente, los índices optimizan las búsquedas, por lo que se han insertado en la información que, se considera, es la más común en ser buscada. En la tabla Asignatura, se definió como índice al nombre y número de la asignatura. En Carrera, la abreviatura de la carrera y en Kardex_Asignatura, la variable que permite ver si una materia

fue aprobada o no, esta creación se observa en la Fig. 5.

```

> Run | New Tab | Active Connection
CREATE INDEX idx_kardex_asignatura_estudiante_asignatura ON Kardex_Asignatura (Aprobada);
> Run | New Tab
CREATE INDEX idx_asignatura ON Asignatura (Nombre);
> Run | New Tab
CREATE INDEX idx_carrera_abrv ON Colegio (Abreviatura);
> Run | New Tab
CREATE INDEX idx_asignatura_num ON Asignatura (Numero);

```

Fig. 5 Creación de Índices

Creación de triggers

Los disparadores, o triggers, son procedimientos que se realizan de forma automática antes o después de la inserción, actualización o eliminación de algún elemento. En este caso, los triggers están implementados para llenar la tabla audit_log tras cada modificación de alguna de las otras tablas. Esto con la intención de llevar un registro de los cambios. Se inserta en la tabla el usuario que realizó la modificación, la tabla en la que lo hizo y una pequeña descripción de la modificación. Un ejemplo de un trigger se puede observar en la Fig. 6

```

CREATE TRIGGER after_colegio_delete
AFTER DELETE ON Colegio
FOR EACH ROW
BEGIN
    INSERT INTO Audit_Log (Usuario, Tabla_Modificada, Descripcion)
    VALUES (CURRENT_USER(), 'Colegio', CONCAT('Se eliminó el colegio: ', OLD.Nombre, ' (ID: ', OLD.Colegio_id, ')'));
END;

```

Fig. 6 Creación de trigger para la eliminación de un elemento de la tabla Colegio

Creación de Stored Procedures

Como se mencionó en el estado del arte, los procedimientos almacenados o stored procedures se encargan de realizar un evento CRUD, dado uno o más atributos. En este caso, se crearon 14 stored procedures, todos realizan una lectura de los datos. Para más detalle, ver la Tabla I.

Tabla I Procedimientos almacenados

Nombre	Variables	Descripción
ObtenerAsignaturaRestantes	Estudiante_id	Devuelve las asignaturas restantes del estudiante según su malla curricular
ObtenerKardexEst	Estudiante_id	Devuelve el kardex del estudiante

GetMallaCarrera	Carrera_id	Devuelve las asignaturas de la malla de una carrera
ObtenerDetallesAsignatura	Asignatura_id	Presenta los prerrequisitos de una asignatura
VerificarPrerrequisitos	Estudiante_id, Asignatura_id	Verifica si un estudiante cumple con los prerrequisitos de una asignatura
ObtenerCorrequisitos	Asignatura_id	Presenta los correquisitos de una asignatura
ObtenerInformacionAsignatura	Asignatura_id	Presenta las restricciones de una asignatura
VerificarRestriccionesEstudiante	Estudiante_id, Asignatura_id	Verifica si un estudiante cumple con las restricciones de una asignatura
getOptativas	Estudiante_id	Devuelve una lista de asignaturas optativas que puede tomar el estudiante, considerando prerrequisitos y restricciones
getElectivas	Estudiante_id	Devuelve una lista de asignaturas electivas que puede tomar el estudiante, considerando prerrequisitos y restricciones
VerCursos	Asignatura_id	Devuelve los cursos ofrecidos por una asignatura
getSubEspecializacion	Subespecializacion_id	Devuelve la malla de una subespecialización
verSubEspRestante	Estudiante_id	Devuelve las asignaturas restantes para que un estudiante cumpla con una subespecialización
verSubEspTomadas	Estudiante_id	Devuelve las asignaturas que el estudiante ha tomado respecto a una subespecialización.

Back end con Django

Para realizar el back end de la página web, se instaló la versión 5.1.1 de Django.

Creación del proyecto

Para crear el proyecto es necesario especificarlo en la línea de comando (ver Fig. 7), una vez realizado, se crearán las carpetas y archivos necesarios para el desarrollo de forma automática. De los más importantes está manage.py, un programa de Python que facilitará la gestión del proyecto, permitiendo levantar el servidor, generar migraciones, crear nuevas aplicaciones, entre otras. De igual forma, es necesario crear una aplicación dentro del proyecto, para manejar de mejor forma la escalabilidad. Esto permitirá que, en el caso de ser necesario, otras aplicaciones puedan manejarse de forma independiente.

```
django-admin startproject proyecto
```

Fig. 7 Comando para crear proyecto en Django

En este caso, se crea una aplicación llamada cursos, la cual tendrá toda la información de la página web (ver Fig. 8). Aquí se realizará el proyecto.

```
python manage.py startapp cursos_
```

Fig. 8 Comando para crear aplicaciones en Django

Vinculación con la base de datos

Para poder vincular a Django con la base de datos creada previamente, se debe modificar el archivo settings.py, agregando la información necesaria para la conexión, para observar cómo se realiza la conexión, ver Fig. 9.

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'proyectointegrador',
        'USER': 'root',
        'PASSWORD': '123123',
        'HOST': 'localhost',
        'PORT': '3306',
    }
}
```

Fig. 9 Conexión a la base de datos con Django

Una vez declarada la conexión, es necesario crear una migración, esto con el comando migrate, en manage.py, para poder transformar los cambios hechos en Django a estructuras de la base de datos. En esta primera migración, se crean 10 tablas nuevas, que permitirán llevar un registro de los cambios realizados en el proyecto y la autenticación de usuarios (para observar las tablas ver Fig. 10)

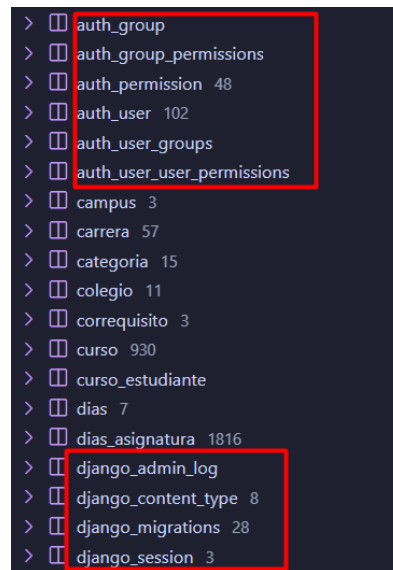


Fig. 10 Nuevas tablas creadas por Django

Debido a que, para la autenticación de usuarios, Django usa su propia tabla, es necesario pasar la información útil a la tabla `auth_user`, considerando que las contraseñas se encuentren cifradas. Si bien existen otras tablas de autenticación que permiten gestionar grupos y permisos, para este proyecto no es necesario modificarlas, ya que se considera que todos los usuarios son estudiantes con los mismos permisos.

* id	* password	last_login	* is_superuser	* username	* first_name	* last_name
int	varchar(128)	datetime(6)	tinyint(1)	varchar(150)	varchar(150)	varchar(150)
777152	pbkdf2_sha256\$870000\$IUI	2024-10-21 19:37:19.13653	0	rodrigo.ramos73@estud.usf	Rodrigo	Ramos
779297	pbkdf2_sha256\$870000\$Iza	(NULL)	0	juliana.pacheco80@estud.u	Juliana	Pacheco
819321	pbkdf2_sha256\$870000\$F6	(NULL)	0	valeria.diaz70@estud.usfq.e	Valeria	Diaz
822363	pbkdf2_sha256\$870000\$sB	(NULL)	0	bianca.fernandez90@estud.	Bianca	Fernandez

Fig. 11 Información de los usuarios en la tabla `auth_user`

Una vez realizado esto, se deben llamar a las tablas que se van a utilizar directamente en el proyecto. Estos son los modelos, la primera parte del “modelo vista template”. La declaración de los mismos se realizan en el archivo `models.py` de la aplicación y tiene el formato observado en la Fig. 12. En el caso del proyecto, se llama directamente a las siguientes tablas: Carrera, Semestre, Subespecialización, Estudiante, Keywords y PalabrasEstudiante. El resto de la información se la puede conseguir a través de los stored procedures mencionados previamente.

```

class Estudiante(models.Model):
    Estudiante_id = models.IntegerField(primary_key=True) # Identificador único del estudiante
    Nombres = models.CharField(max_length=255) # Nombres del estudiante
    Apellidos = models.CharField(max_length=255) # Apellidos del estudiante
    Correo = models.EmailField(max_length=255, unique=True) # Correo institucional
    Password = models.CharField(max_length=255) # Contraseña del estudiante (se manejará cifrada)
    Carrera = models.ForeignKey(Carrera, on_delete=models.CASCADE, default=1, db_column='Carrera_id') # FK hacia la carrera
    Semestre = models.ForeignKey(Semestre, on_delete=models.CASCADE, default=1, db_column='Semestre_id') # FK hacia el semestre
    Subespecializacion = models.ForeignKey(SubEspecializacion, on_delete=models.CASCADE, default=1, db_column='SubEsp_id')

    def __str__(self):
        return f'{self.Nombres} {self.Apellidos}'

class Meta:
    db_table = 'Estudiante'

```

Fig. 12 Modelo de la tabla Estudiante

Similar a lo realizado tras declarar la base de datos, ahora se debe crear una migración con el comando `makemigrations` y luego migrarla con el comando `migrate`, ambos en `manage.py`.

Módulo de autenticación del usuario

El primer paso de la aplicación web es iniciar sesión, por lo que se debe crear un módulo de autenticación, utilizando la información de la tabla `auth_user`. Para esto, se crea una vista, la segunda parte del “modelo vista template”. En la vista se define un método llamado `login_view`, el cual va a recibir como atributo un objeto `request`, este objeto tiene la información de la solicitud HTTP que recibe el servidor, ya sea un *GET* o *POST*.

Dentro del método, se verifica la recepción de un `request` de tipo *POST*, y de serlo, se verifica que exista un usuario y contraseña que se pueda obtener del html, del cual se hablará más adelante. Si se tiene ambos elementos de información, donde el usuario es el correo del estudiante, se procede a llamar a una función propia de Django llamada `authenticate`, que verifica que los datos sean correctos. En caso de serlo, se redirecciona a una nueva página. Si no lo son, se muestra un aviso y se mantiene en la misma página.

Además, el método direcciona al html de forma predeterminada al ser llamado por una solicitud HTTP. Para vincular el método, con la página web, se modifica el archivo `url.py`, de esta forma el path `/login` está vinculado con el método `login_view`, como se observa en la Fig.

```
def login_view(request):
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        print(username, password)
        user = authenticate(request, username=username, password=password)
        if user is not None:
            login(request, user)
            return redirect('table_user') # Redirige a la página principal después del inicio de sesión
        else:
            return render(request, 'login.html', {'error': 'Credenciales incorrectas'})
    return render(request, 'login.html')
```

Fig. 13 Vista del login

```
urlpatterns = [
    path('login/', login_view, name='login'),
```

Fig. 14 Vinculación del path en views.py

Por último, se define al `login.html`, que será renderizado cada vez que se ingrese al path previamente definido. Cualquier archivo html que se vaya a renderizar estará dentro de la carpeta *template*. Este es el tercer elemento del “modelo vista template”.

El html contiene un título para notificar al usuario que está en la página de iniciar sesión, dos campos de texto, uno para su correo y otro para su contraseña, y el botón para verificar su cuenta. Al presionar el botón, se envía el método *POST* que recibirá la vista para la verificación del usuario. Cualquier método *POST* deberá ser envuelto en la etiqueta *form* como se puede observar en la Fig. 15.

```
<body>
  <div class="container">
    <h1>USFQ - Buscador de Cursos</h1> <!-- Title -->
    <h2>Iniciar Sesión</h2>
    <form method="POST">
      {% csrf_token %}
      <label for="username">Nombre de Usuario:</label>
      <input type="text" name="username" required>

      <label for="password">Contraseña:</label>
      <input type="password" name="password" required>

      <button type="submit">Iniciar Sesión</button> <!-- Butt
    </form>
    {% if error %}
      <p style="color: red;">{{ error }}</p>
    {% endif %}
  </div>
</body>
```

Fig. 15 Template del login

Buscador de cursos

Para el buscador de cursos se va a utilizar los modelos previamente declarados en la vinculación de la base de datos. Todo esto se realizará en una nueva vista definida en `views.py`.

Primero, para asegurarse que solo aquellos usuarios que han sido verificados puedan ver la página, hay que especificar que el método requiere un login, con el comando “@login_required”. Tras esto, se vincula al usuario de la tabla de autenticación propia de Django, con la tabla de Estudiantes de la base de datos. Para esto, se obtiene el correo del usuario, y luego se guarda una variable de tipo Estudiante, que coincida con el correo (ver Fig. 16)

```
@login_required
def table_User(request):

    if request.method == 'POST' and 'logout' in request.POST:
        Logout(request)
        return redirect(reverse('login')) # Redirect to Login page after logout

    correo_usuario = request.user.email

    # Buscar al estudiante correspondiente en la tabla Estudiante
    try:
        estudiante = Estudiante.objects.get(Correo=correo_usuario)
    except Estudiante.DoesNotExist:
        estudiante = None
```

Fig. 16 Desarrollo de la vista table_user

Hecho esto, se puede llamar a la información específica del usuario que se encuentra en la base de datos. Primero, se guarda la información académica del usuario, la carrera, semestre y, si aplica, subespecialización que cursa el estudiante actualmente. Además, se obtiene una lista con las palabras claves elegidas y otra con todas las palabras clave, para poder elegir nuevas de ser necesario. Por último, se llama a los stored procedures para obtener las asignaturas que no han sido tomadas, tanto para su carrera como su subespecialización. Además, se obtienen sus prerrequisitos, correquisitos y restricciones, y se verifica si el estudiante cumple con los mismos o no.

```
# Call the stored procedure
with connection.cursor() as cursor:
    cursor.callproc('ObtenerAsignaturasRestantes', [estudiante.Estudiante_id])
    results = cursor.fetchall()

with connection.cursor() as cursor:
    cursor.callproc('verSubEspRestantes', [estudiante.Estudiante_id])
    subEsP = cursor.fetchall()
    cursor.close()
```

Fig. 17 Llamada a store procedures en la vista table_user

Toda esta información se pasa como contexto a la hora de renderizar la página, para poder utilizar los datos en el código html.

De forma similar a la vista de login, se crean verificaciones por si llega un método *POST*, pero, en este caso, se espera recibir varias llamadas de este tipo que deberán realizar diferentes acciones, por lo que se deberán identificar también con un nombre específico. Los métodos *POST*, sus nombres y su función se pueden observar en la Tabla II.

Tabla II Métodos *POST* de la vista *table_user*

Nombre	Función
logout	Finaliza la sesión del usuario y redirección a la página de inicio de sesión.
save_keywords	Guarda en la base de datos las nuevas palabras claves elegidas por el usuario
save_Sub	Guarda en la base de datos la nueva subespecialización elegida por el usuario
update_filter	Actualiza la tabla para ver los cursos elegidos por el estudiante.

El método *POST* con nombre *update_filter* tiene una particularidad y es que se trata de una solicitud *AJAX* (Asynchronous JavaScript and XML). Como indica su nombre, esta solicitud trabaja de forma asincrónica [29], cargando los datos nuevos de la tabla de cursos en un html diferente y exportando este html en la tabla. Esto lo realiza para no tener que cargar la página cada vez que se actualice la lista de cursos que se desea revisar y mejorar la experiencia del usuario.

La vista es asociada al path */table_user* y luego se procede a crear el código html que será renderizado. En el html, se despliega la información recolectada del estudiante (nombre, carrera, subespecialización, si aplica, y palabras claves seleccionadas). Además, se coloca un botón para observar todas las palabras clave y poder seleccionar hasta diez palabras. Y, otro botón para guardar dichas palabras. También existe un botón para seleccionar que materias se observan en la tabla de cursos y otro para cargar la información de nuevas asignaturas. Por último, hay un botón para observar las subespecializaciones disponibles y permite seleccionar una, y un botón para guardar la información.

Las asignaturas que se muestran automáticamente en la tabla son aquellas que, en orden según la malla curricular, el estudiante no ha tomado y la suma de todos sus créditos, da un máximo de 16. La tabla muestra 10 cursos a la vez, y se puede observar la información por medio de botones para pasar de página. Cada columna de la tabla tiene información útil para el usuario, y un botón para más información que permite ver los prerrequisitos, correquisitos y restricciones. En el caso que un estudiante no cumpla con los requisitos necesarios para tomar el curso, un mensaje en letras rojas se despliega en la tabla.

Por último, existe un botón de salir, que realiza el logout y un botón que abre una nueva pestaña para observar las asignaturas que el usuario ya ha tomado. Cada parte práctica de html tiene su propia función en las etiquetas de *script*. Para ver el detalle de cada una, se puede observar el Anexo D: Script de la vista `table_user`.

La nueva pestaña, para ver aquellos cursos tomados por el estudiante, llama al stored procedure que permite visualizar el kardex del estudiante, y despliega esta información en una tabla. En caso de haber tomado alguna materia, pero no aprobarla, se colocará su estado en letras rojas.

Por último, se realizó una descarga de un favicon, un archivo usado como ícono de las páginas web.

Front end con bootstrap.

Para el front end, se llama al archivo de estilos de bootstrap mediante una línea de código en el archivo html, como se observa en la Fig. 18. Con esta línea, se pueden aplicar los estilos ofrecidos por la librería, que no solo proporcionan una mejor interfaz gráfica, sino que también se encarga de que sus componentes sean adaptables y responsivos.

Todos los botones fueron normalizados para verse similares. Además, se incluye un banner de navegación fija, es decir, aparece siempre en la página web. Por último, la página

que permite ver el historial de asignaturas tomadas fue modificada para ser responsiva, permite ordenar las materias en orden alfabético, escoger la cantidad de elementos que visualizar y cambiar de página de ser necesario.

```
<link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">
```

Fig. 18 Llamada al URL de Bootstrap en el código html

Parte fundamental del front end fue mantener los colores base de la USFQ, los cuales se encuentran en el “manual de elementos para medios digitales, USFQ”.²

Sistema de recomendación

Para la implementación del sistema de recomendación primero se obtienen las palabras clave que el usuario puede elegir, y segundo, determinar un método por el cual se recomiendan cursos basados en las palabras escogidas.

Palabras clave

Las palabras clave se obtendrán tanto en español como en inglés, para contrastarlos, debido a que las librerías utilizadas en los scripts están pensadas principalmente para tratarse con textos en inglés.

Las palabras clave se obtienen del título y descripción de las asignaturas ofrecidas, por lo que, se debe extraer esta información y guardarla. Para esto se realiza un script de Python que genere una conexión con la base de datos y extraiga la información para guardarla en archivos txt, como se observa en la Fig. 19. Seguido de esto se procede a traducir los archivos con *deep_translator* de la librería de Python *GoogleTranslator*. Cada archivo traducido se guarda de igual forma con la extensión txt.

Una vez hecho esto, se preprocesa el texto, tanto en inglés como en español. Como primer paso, se carga la librería *spacy* que contiene modelos tanto en inglés como en español.

² <https://www.usfq.edu.ec/sites/default/files/2021-09/manual-de-marca-digital.pdf>

Estos modelos permiten eliminar *stop words* (palabras que no aportan información en el análisis de texto) y realizar un proceso de lematización. La lematización permite traer a las palabras a su raíz, esto permite que ciertas palabras y sus derivados se consideren una misma palabra. Además, la librería igual permite eliminar la puntuación, dejando solamente palabras. Para el preprocesamiento en español, es necesario también reemplazar los caracteres acentuados, como las letras con tilde o virgulilla. Ambos archivos preprocesados se guardan en un archivo csv para su análisis.

```
# Conectar a la base de datos
try:
    connection = pymysql.connect(**db_config)
    cursor = connection.cursor()

    # Consulta para obtener los datos de la tabla
    query = "SELECT * FROM asignatura"
    cursor.execute(query)
    rows = cursor.fetchall()

    # Directorio para guardar los archivos
    output_dir = 'output_files\\spanish'
    os.makedirs(output_dir, exist_ok=True)

    # Iterar sobre cada fila del resultado
    for row in rows:
        # Crear el nombre del archivo usando row[0] y row[1]
        filename_raw = f"{row[0]}_{row[1]}.txt" # Usar el formato row[0]_row[1]
        sanitized_filename = sanitize_filename(filename_raw)
        final_filename = replace_spaces_with_underscores(sanitized_filename)

        # Obtener el contenido de row[2]
        content = row[2] # Contenido de row[2]

        # Definir la ruta del archivo
        file_path = os.path.join(output_dir, final_filename)

        # Escribir el contenido en el archivo
        with open(file_path, 'w', encoding='utf-8') as file:
            file.write(str(content))

        print(f"Archivo creado: {final_filename} con contenido de row[2]")
```

Fig. 19 Script para guardar la descripción de las asignaturas

Con el texto preprocesado, se procede con la generación de las palabras clave. Para esto se probarán dos métodos. El primero utiliza *term frequency, inverse document frequency* (*tf_idf*) y el segundo es un modelo de aprendizaje profundo *Bidirectional Encoder Representations from Transformers* (*BERT*).

Para tf_idf , se probaron con variantes que se observan en la Tabla III. Para cada variante se obtiene un listado de 70 palabras junto con un puntaje calculado según la variante que se probó. Seguido de esto, se normalizaron los valores de cada variante y se sumaron, para obtener un listado con las palabras más significativas.

Tabla III Variantes de tf_idf

Term Frequency		Document Frequency	
Natural	$tf_{t,d}$	No	1
Logarítmico	$1 + \log(tf_{t,d})$	Idf	$\log \frac{N}{df_t}$
Aumentado	$0.5 + \frac{0.5 * tf_{t,d}}{\max(tf_{t,d})}$	Probabilístico	$\max\left\{0, \log \frac{N - df_t}{df_t}\right\}$
Boolean	$\begin{cases} 1 & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$		

Por último, se eliminó manualmente palabras que no podrían ser utilizadas como palabras clave. En este último paso se evidencia que las librerías están pensadas para ser usadas en inglés, dado que el listado en español contenía palabras de significado similar, como diseño y diseñar. Para ver las variantes usadas, los resultados finales y los resultados una vez realizada la eliminación manual, ver Anexo E: Resultados de tf_idf .

Para el segundo método de obtención de palabras clave, se utilizó KeyBERT, un componente que utiliza BERT para obtener las palabras clave de un documento. En este caso, se probó con un modelo en inglés y uno en español, y distintas variables, se probó con todas las descripciones como un solo documento, y con palabras claves individuales para cada documento. Los modelos y los parámetros se pueden visualizar en la Tabla IV.

Una vez obtenido los resultados, se eliminaron algunas redundancias y se procedió a unir los resultados de cada modelo. Se optó por utilizar como palabras clave las generadas por el modelo *distilbert-base-nli-mean-tokens*. Por lo que, se deberá traducir las palabras y generar un script que modifique la tabla *Keywords* de la base de datos. Por último, se verifica, en la aplicación web creada previamente, que las palabras se hayan actualizado de forma correcta,

como se observa en la Fig. 20.

Tabla IV Modelos usados para Keybert

Modelo	Parámetros
distilbert-base-nli-mean-tokens	N/A
distilbert-base-nli-mean-tokens	use_mmr = true diversity = 0.2
distilbert-base-nli-mean-tokens	use_mmr = true diversity = 0.7
sentence-transformers/distiluse-base-multilingual-cased-v1	N/A
sentence-transformers/distiluse-base-multilingual-cased-v1	use_mmr = true diversity = 0.2
sentence-transformers/distiluse-base-multilingual-cased-v1	use_mmr = true diversity = 0.7

social

software

taller

tecnica

teorico

Tu Información

Nombre:PRUEBA PRUEBA

Carrera:Ingenieria en Ciencias de la Computacion

Semestre:Quinto Semestre

SubEspecializacion:Danza - Artes Escenicas

Ver SubEspecializacion

Palabras Clave

Guardar Palabras Clave

☐ algebra
☐ analisis
☐ aplicacion
☐ arte
☐ biologia

Fig. 20 Actualización de las palabras clave

Creación de los vectores

Para crear los vectores, se utiliza un tf normal y un idf clásico, para crear un archivo csv con todos los documentos vectorizados.

Búsqueda de la similitud

Una vez creado los documentos, se obtiene el vector query tomando un conjunto de máximo diez palabras clave. Este vector debe ser creado en el mismo espacio en el que se creó

al resto de documentos, para que todos tengan la misma longitud. Una vez realizado esto, para encontrar la similitud existen varias opciones a probar. Estas opciones se pueden observar en la Tabla V.

Tabla V Métricas de distancia y similitud

Nombre	Descripción	Fórmula
Similitud del Coseno	Mide el coseno del ángulo entre dos vectores en un espacio multidimensional	$S.C = \frac{\vec{A} \cdot \vec{B}}{ \vec{A} \vec{B} }$
Distancia Euclidiana	Calcula la distancia euclidiana entre dos puntos en el espacio	$d(\vec{A}, \vec{B}) = \sqrt{\sum_{i=1}^n (A_i - B_i)^2}$
Similaridad de Jaccard	Mide el espacio sobrepuesto entre dos conjuntos. Usado principalmente en información binaria.	$J(A, B) = \frac{ A \cap B }{ A \cup B }$
Correlación de Pearson	Mide la correlación lineal entre dos vectores	$r = \frac{\sum_{i=1}^n (A_i - \bar{A})(B_i - \bar{B})}{\sqrt{\sum_{i=1}^n (A_i - \bar{A})^2 \sum_{i=1}^n (B_i - \bar{B})^2}}$
Distancia de Manhattan	Mide la suma de las diferencias absolutas entre elementos correspondientes de dos vectores	$d(\vec{A}, \vec{B}) = \sum_{i=1}^n A_i - B_i $
Divergencia KL	Mide como una distribución probabilística difiera de una distribución probabilística de una referencia	$D_{KL}(P Q) = \sum_{i=1}^n P_i \log\left(\frac{P_i}{Q_i}\right)$
Distancia de Hamming	Mide la proporción de las posiciones en las cuales los elementos correspondientes de dos vectores son distintos	$d(\vec{A}, \vec{B}) = \frac{1}{n} \sum_{i=1}^n A_i - B_i $
Distancia Bhattacharyya	Mide la similitud entre dos distribuciones probabilísticas	$D_B(P, Q) = -\log\left(\sum_{i=1}^n \sqrt{P_i Q_i}\right)$
Producto Punto	Mide la proyección de uno de los vectores en otro	$\vec{A} \cdot \vec{B} = \sum_{i=1}^n A_i B_i$

Tras probar las distintas medidas y observar resultados como el de la Fig. 21, se elige

la similitud del coseno.

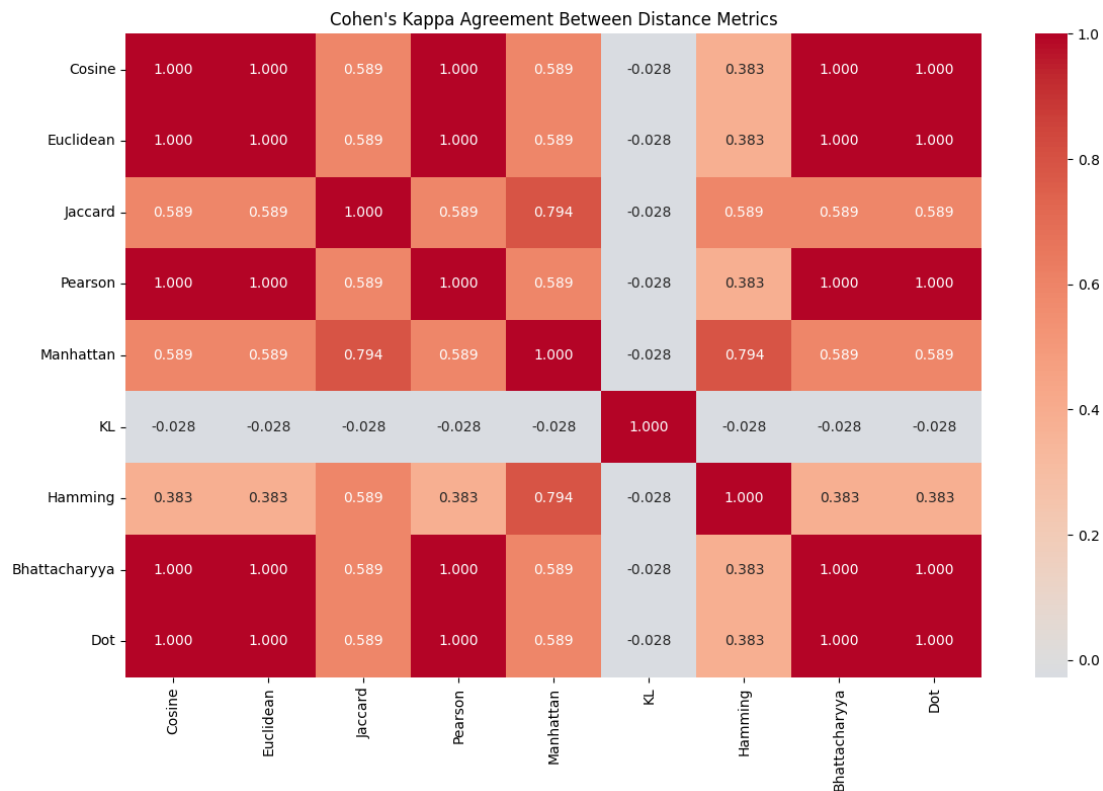


Fig. 21 Matriz de Acuerdo entre las métricas de distancia

Implementación en Django

Para implementar el sistema de recomendación en Django, se creó una clase llamada `recommendation.py`, una nueva vista con el mismo nombre y un nuevo template que sigue el mismo diseño que la página principal del buscador de cursos. Para obtener los cursos que aplican como optativas o electivas para el usuario, se llama a los stored procedures “getOptativa” y “getElectiva” y se realiza un filtro al archivo csv con los documentos vectorizados.

Por último, se realizaron pruebas con distintas cantidades de palabras claves, mientras se monitorea el resultado de la similitud del coseno. Con esto se concluye que, si bien más palabras clave no mejora necesariamente el puntaje, si presenta resultados variados con puntajes similares. Se determina un puntaje mínimo de 0.135 en la similitud del coseno, tras realizar distintas pruebas con una diferente cantidad de palabras clave. Los cursos con este

puntaje o un puntaje mayor mantienen relevancia con las palabras clave.

Pruebas de funcionalidad

Base de datos

Para las pruebas de la base de datos, se verificó que los stored procedures funcionen correctamente y entreguen la información correcta. Para esto, se agregó un estudiante de prueba con la información observada en la Fig. 22. Dado que el estudiante pertenece Ing. en Ciencias de la Computación, se harán las pruebas con esta carrera.

id int	nombre varchar	apellido varchar	correo varchar	contraseña varchar	carrera varchar	semestre varchar	minor varchar
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
999999	PRUEBA	PRUEBA	prueba@estud.usfq.edu.ec	prueba	Ingeniería en Ciencias de la	Quinto Semestre	(NULL)

Fig. 22 Datos del estudiante de prueba

DE LOS RESULTADOS MÁS IMPORTANTES DE LAS PRUEBAS ESTÁN, AL OBTENER LAS ASIGNATURAS RESTANTES DEL ESTUDIANTE, SI ELIMINA OPTATIVAS O ELECTIVAS QUE YA HAYA TOMADO Y TAMBIÉN AQUELLAS NO ESPECÍFICAS QUE IGUAL HAYA CURSADO; VERIFICA CORRECTAMENTE SI UN ESTUDIANTE CUMPLE O NO CON UN PRERREQUISITO O RESTRICCIÓN; PRESENTA CORRECTAMENTE UN LISTADO DE OPTATIVAS Y ELECTIVAS QUE EL ESTUDIANTE PUEDE TOMAR, TOMANDO EN CUENTA QUE CUMPLA CON LOS REQUISITOS Y RESTRICCIONES. PARA OBSERVAR EL RESULTADO DE TODAS LAS PRUEBAS, VER ANEXO E: RESULTADOS DE TF_IDF

Tabla XXXIX Variantes utilizadas de tf_idf

Term Frequency	Inverse Document Frequency
Natural	Idf
Natural	No
Logarítmico	Idf
Logarítmico	No
Logarítmico	Probabilístico
Aumentado	Idf
Aumentado	No
Aumentado	Probabilístico
Boolean	Idf
Boolean	No
Boolean	Probabilístico

Tabla XL Palabras Clave con tf_idf

Palabras Clave con tf_idf en español	
Keyword	NScore
tecnica	9.75609513
incluir	9.56397775
aprenderar	8.67371561
cubrir	7.99191647
concepto	7.90478924
proyecto	6.88556784
tema	6.73412794
explorar	6.60746242
desarrollar	6.48993546
aplicacion	6.05681665
habilidad	5.05418243
avanzado	4.73139341
enfocar	4.68934112
aplicar	4.63520369
analisis	4.4329143
practico	4.30818097
gestion	4.15165285
arte	4.05121639
teoria	3.6449584
problema	3.30664634
principio	3.25561757
introducir	3.09717744
abordar	2.98240665
fundamental	2.53697848
creacion	2.40841624
proporcionar	2.40046076
herramienta	2.10427964
practica	2.10020634
desarrollo	1.89666477
real	1.78623561
sistema	1.57743704
estudiantes	1.49763053
empresarial	1.45485036
artistico	1.4443009
estudio	1.43120654
estudiar	1.42308017
espacio	1.37922758
diseño	1.3398249
aprendizaje	1.26815328
tambien	1.20623417
laboratorio	1.20251717
dato	1.12008765

mas	0.96873664
enfoque	0.90524257
trabajar	0.89142857
analizar	0.86927412
planificacion	0.86779509
presentacion	0.77585179
uso	0.7390451
entorno	0.72136927
mejorar	0.71056182
campo	0.5916771
produccion	0.57544722
diverso	0.50918041
complejo	0.49560877
area	0.48171258
diferente	0.46484015
programacion	0.42516909
educativo	0.39165702
decision	0.36835389
practicar	0.35180634
aplicado	0.35014572
estructura	0.32139586
aprender	0.2427558
matematica	0.18854197
estrategia	0.08248506
caso	0.08057567
crear	0.06432187
basico	0.03051272
social	0.02112568
evaluacion	0.00822945
conocimiento	0.00275875
disenar	0.00257005
ingenieria	0.00225622
integral	0

Palabras clave con tf_idf en inglés

Keyword	NScore
learn	9.91938235
technique	9.26040112
cover	8.34567952
include	7.90738584
focus	7.70875653
develop	6.42938217
concept	6.4026158
apply	6.30984378
project	6.05900004
explore	5.66829069

art	5.07780167
topic	4.95943498
application	4.68646935
skill	4.66809989
study	4.56703682
analysis	4.24531688
advanced	4.00075197
management	3.93960613
practical	3.87116663
design	3.6527816
fundamental	3.62591903
provide	3.29911619
introduce	3.23775868
theory	3.22051201
business	3.21370918
principle	3.16270703
problem	2.80025377
work	2.18057203
basic	2.0869073
tool	1.85668984
artistic	1.64906882
real	1.5960847
create	1.49334664
field	1.32341783
space	1.30694173
use	1.24298852
system	1.12955258
process	0.99897777
creation	0.89853851
different	0.86740503
learning	0.63104061
complex	0.5939592
educational	0.57614077
area	0.56394007
production	0.55974959
structure	0.51243362
practice	0.5055745
development	0.49464487
solve	0.48754845
teach	0.48059504
address	0.48007167
decision	0.47816744
plan	0.3977533
planning	0.39595023
mathematic	0.37215069

environment	0.36511704
presentation	0.34808729
critical	0.27788994
programming	0.26341937
teaching	0.26124783
strategy	0.21112853
hand	0.20677753
improve	0.19184885
emphasis	0.17777637
allow	0.16172135
knowledge	0.1126256
case	0.10557134
social	0.06803295
current	0.01658876
contemporary	0.00883434
science	0.00830678
laboratory	0.00723836
prepare	0.00234074
engineering	0.00128396
control	0

Anexo F: Pruebas de la base de datos.

Proyecto Django

Para realizar las pruebas de front end y back end, se iniciará sesión con dos usuarios distintos, esto con el fin de observar el comportamiento del programa en diferentes situaciones. Primero, con un estudiante que no tiene una subespecialización elegida, y se encuentra en quinto semestre. Para esto, primero se inicia sesión y se verifica que los datos del estudiante sean correcto (ver Fig. 23). Una vez hecho esto, se escoge una subespecialización para el usuario y se verifica que se haya modificado en la base de datos. Para esto se hace click en el botón “Show Sub-Specializations” y tras escoger, el botón “Guardar Información”.

Tu Información
Nombre:PRUEBA PRUEBA
Carrera:Ingenieria en Ciencias de la Computacion
Semestre:Quinto Semestre
SubEspecializacion: Show Sub-Specializations Guardar Información

Fig. 23 Información del usuario de prueba

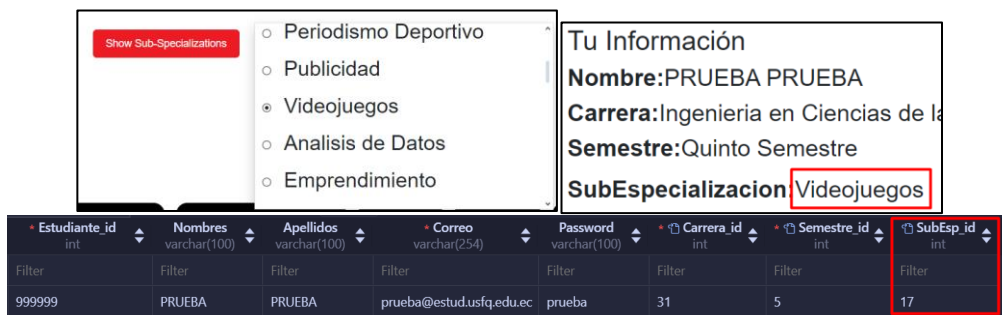


Fig. 24 Flujo para escoger una nueva subespecialización

Una vez observado que la subespecialización se guarda correctamente (ver Fig. 24), se prueba guardar las palabras claves elegidas por el usuario. Actualmente se observan las palabras vistas en Fig. 25. Luego, se despliegan todas las palabras claves de la base de datos, se verifica que solo se puedan escoger hasta diez palabras y se eligen nuevas opciones. Finalmente, se hace click en “Guardar Palabras Clave” y se verifica en la base de datos. El

flujo se observa en la Fig. 26.



Fig. 26 Flujo para cambiar las palabras clave

Seguido de esto, se verifican los cursos disponibles a elegir, verificando que sean aquellos que el usuario no ha tomado, constan en su carrera y en su subespecialización (ver Fig. 27).

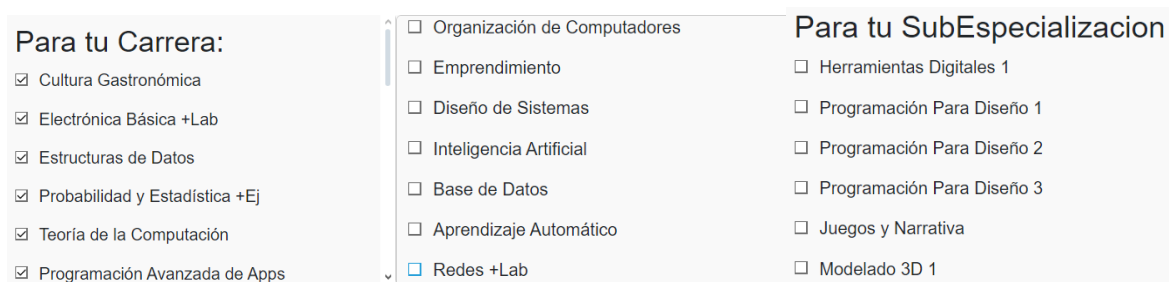


Fig. 27 Asignaturas disponibles para el usuario de prueba

Se verifica que se puedan escoger nuevos cursos y se pueda visualizar si no se cumple con una restricción o requisito.

Currently Selected:

Minería de Datos

Proyecto Integrador CMP

Herramientas Digitales 1

☐ Seguridad Informática

☒ Minería de Datos

☐ Aplicaciones Distribuidas

☒ Proyecto Integrador CMP

Para tu SubEspecialización

☒ Herramientas Digitales 1

☐ Programación Para Diseño 1

Minería de Datos

Profesor: Noel Perez Perez

Campus: Cumbaya

Día: L, I

Horario: 14:30 - 15:50

Edificio: S-116

Modalidad: Presencial

No se cumple con los prerrequisitos

Proyecto Integrador CMP

Profesor: Simeon Floyd

Campus: Galapagos

Día: L, I

Horario: 17:30 - 18:50

Edificio: MS-111

Modalidad: Virtual Sincrónico

No se cumple con las restricciones

Herramientas Digitales 1

Profesor: Daniel Bustillos Costales

Campus: Galapagos

Día: L, I

Horario: 11:30 - 12:50

Edificio: C-215

Modalidad: Presencial

Fig. 28 Flujo para elegir nuevas asignaturas que desplegar

Por último, se verifica que el historial de asignaturas funciona correctamente. Esto al hacer click en “Ver asignaturas previas” (ver Fig. 29).

Análisis Numérico	MAT-3001	B	Aprobada	Ciencias Exactas
Ser y Cosmos	ARL-2001	C	Aprobada	Colegio General
Inglés Nivel 5	ESL-0005	C	Aprobada	Lenguas

MAT-3001	Análisis Numérico	B	Aprobada
ARL-2001	Ser y Cosmos	C	Aprobada
ESL-0005	Inglés Nivel 5	C	Aprobada

Fig. 29 Asignaturas ya tomadas por el usuario

Finalmente, se hacen verificaciones similares con otro usuario, para asegurarse que aparezcan sus asignaturas propias, y en color rojo aquellas asignaturas que ya cursó, pero no ha aprobado. En la Fig. 30 se observa que la estudiante no aprobó la materia de Contabilidad

Empresarial, por lo que si se muestra en los cursos disponibles. Además, se muestra el curso restante de su subespecialización.

Código	Nombre	Nota	Estado
FIN-2002	Contabilidad Empresarial	D	Not Aprobada

Nombre:Paula Morales
Carrera:Administracion de Empresas
Semestre:Octavo Semestre
SubEspecializacion:Videojuegos Show

Ver tus asignaturas previas

Otros Cursos Actualizar

Currently S

Contabilidad Empre

Código

- ☒ Contabilidad Empresarial
- ☒ Cultura Gastronómica
- ☒ Innovacion y Sustentabilidad
- ☒ Tributacion y Entorno Legal
- ☒ Analisis Estrategico ADM

Para tu SubEspecializacion

- ☐ Modelado 3D 1

Fig. 30 Verificaciones con otros usuario

Para ver las pantallas de la aplicación en mayor detalle, ver Anexo G: Pantallas de la página web.

Sistema de Recomendación

La prueba se encarga de verificar el correcto funcionamiento del sistema de recomendación y la visualización de los cursos nuevos en una página nueva de la aplicación web. Los stored procedures utilizados para obtener las asignaturas optativas y electivas ya filtran aquellas que el usuario no podría tomar por motivos de requisitos y restricciones, por lo que la columna para obtener más información no es necesaria. Se prueba con dos conjuntos de palabras clave y se observa que en ambos casos se entregan resultados correctos.

arte
culinario
danza
improvisacion
panaderia

Palabras Clave
Guardar Palabras Clave

Ver Recomendaciones

Tu Información

DAN-1001	845	Improvisación Profesor: Victor Herrera Campus: Cumbaya Día: L, I Horario: 17:30 - 18:50 Edificio: EE-210 Modalidad: Presencial	20
CCO-1105	701	Temas en Comunicacion y Arte Profesor: Luis Rincon Campus: Galapagos Día: M, J Horario: 08:30 - 09:50 Edificio: LZ-302 Modalidad: Híbrido	45

Fig. 31 Prueba de funcionalidad sistema de recomendación

CONCLUSIONES

El proyecto ha permitido desarrollar una guía curricular, como una aplicación web, que permite visualizar los cursos obligatorios, electivos y optativos de un estudiante, mediante la información del proyecto académico. Además, proporciona un sistema de recomendaciones basado en inteligencia artificial para simplificar el proceso de selección de cursos.

Mediante una base de datos relacional se ha podido emular la estructura académica de la universidad. Esta base de datos permite generar las relaciones que diferentes entidades tienen entre ellas. Mediante procedimientos almacenados se facilita la obtención de esta información.

El sistema de autenticación asegura que la experiencia del usuario sea personalizada, de esta forma se obtienen solo las materias obligatorias para su carrera y subespecialización, y aquellas que puedan servir como optativas y electivas.

El sistema de recomendación elige las palabras clave implementando inteligencia artificial, para asegurarse que las palabras sean representativas y presenten diversidad a la hora de escogerlas. Además, la similitud del coseno permite encontrar aquellas que más se asemejan a la elección de palabras, de la lista de cursos accesibles para el usuario.

Por último, todo esto es visualizado gracias a la aplicación web realizada en Django, que sigue el esquema de modelo-vista-template. Este se conecta a la base de datos y permite realizar consultas en base a modelos, elegir la información a desplegar según su vista y crear una interfaz visual llamativa con sus templates y con la ayuda del framework de Bootstrap.

Trabajo futuro

El proyecto está pensado para ser implementado con la estructura universitaria, razón por la que la base de datos trató de ser lo más similar posible, por lo que sería necesario conectarse a la misma y realizar ciertas modificaciones a los modelos y vistas de Django de ser necesario.

Por otro lado, se recomienda obtener retroalimentación de los estudiantes al finalizar el proceso de selección de materias, acerca de fortalezas y debilidades del buscador de cursos, para continuar con su mejora y mantenimiento. Igualmente, se deberá registrar retroalimentación al final del semestre acerca de las materias optativas y electivas que cursaron. Esto con el fin de ampliar el sistema de recomendación para que igualmente sea colaborativo, y se recomienden materias por perfiles similares o por elecciones previas similares, parecido a como se lo hace en servicios de streaming actualmente.

Por último, se podría desarrollar una app para dispositivos móviles. Para esto sería necesario adaptar el framework de Django para ser usado en dispositivos móviles o utilizar un framework distinto. Sin embargo, las páginas de la universidad para inscripción y eliminación de cursos, información de retenciones, kárdex académico, entre otras, se encuentran disponibles solo en aplicaciones web. Por lo que sería recomendable desarrollar aplicaciones móviles para estas páginas, con el objetivo que toda la información académica pueda ser visualizada en conjunto.

REFERENCIAS BIBLIOGRÁFICAS

- [1] J. Adiego Rodriguez and D. Llanos Ferraris, Fundamentos de informática y programación en C. Ediciones Paraninfo, 2010.
- [2] M. T. González-Aparicio, M. Younas, J. Tuya, and R. Casado, “A new model for testing CRUD operations in a NoSQL database”, in 2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA), 2016, pp. 79–86.
- [3] C. J. Date, Introducción a los sistemas de bases de datos. Pearson Educación, 2001.
- [4] V. Valverde, N. Portalanza, P. Mora, and Others, “Análisis descriptivo de base de datos relacional y no relacional”, Revista Atlante: Cuadernos de Educación y Desarrollo, vol. 3, 2019.
- [5] N. Jatana, S. Puri, M. Ahuja, I. Kathuria, and D. Gosain, “A survey and comparison of relational and non-relational database”, International Journal of Engineering Research & Technology, vol. 1, no. 6, pp. 1–5, 2012.
- [6] L. A. C. Santillán, M. G. Ginestà, and Ó. P. Mora, ‘Bases de datos en MySQL’, Universitat oberta de Catalunya, 2014.
- [7] R. Muñoz et al., ‘Buenas prácticas en el diseño de estructuras de datos en bases de datos relacionales’, in XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste), 2018.
- [8] G. Harrison and S. Feuerstein, MySQL stored procedure programming. ‘ O’Reilly Media, Inc.’, 2006.
- [9] J. R. M. Ríos, N. M. L. Mora, M. P. Z. Ordóñez, and E. L. L. Sojos, “Evaluación de los Frameworks en el Desarrollo de Aplicaciones Web con Python”, Archivo de la revista Latinoamericana de Ingeniería de Software, vol. 4, no. 4, pp. 201–207, 2016.

- [10]N. Idris, C. F. M. Foozy, and P. Shamala, ‘A generic review of web technology: Django and flask’, *International Journal of Advanced Science Computing and Engineering*, vol. 2, no. 1, pp. 34–40, 2020.
- [11]C. G. R. R. Manuel Mirana Chinlli, «Repositorio Institucional de la Escuela Superior Politécnica de Chimborazo,» 2015.
- [12]I. H. Madurapperuma, M. S. Shafana, and M. J. A. Sabani, “State-of-art frameworks for Front-end and Back-end Web Development”, 2022.
- [13]J. A. Solórzano Ávila, ‘Desarrollo de una aplicación web multiplataforma usando el framework Django, para publicitar eventos sociales, aplicado en el municipio del cantón Morona’, *Escuela Superior Politécnica de Chimborazo*, 2018.
- [14]I. A. Bairagi, A. Sharma, B. K. Rana, and A. Singh, “UNO: A Web Application using Django”, in *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, 2021, pp. 1371–1374.
- [15]H. Silva, W. Ocampo, L. Ulloa, D. Salazar, R. Córdova, and M. Andrade, “Scrum y Django para el desarrollo rápido de un sistema web para el control de mantenimientos preventivos y correctivos de bienes del cuerpo de bomberos del GADM de Santo Domingo”, *La simulación en ingeniería, trascendiendo fronteras*, p. 29.
- [16]J. L. Navarro Rey and Others, “Desarrollo de una aplicación web con Django para evaluación automática de código Python”, 2017.
- [17]E. L. Medina Arias, “Desarrollo de aplicación web para automatizar procesos de venta para compañía transporte terrestre utilizando lenguaje de programación python, framework django”, 2017.
- [18]F. Ricci, L. Rokach, and B. Shapira, “Introduction to recommender systems handbook”, in *Recommender systems handbook*, Springer, 2010, pp. 1–35.

- [19]O. E. González & S. M. Jacques, “Estado del arte en los sistemas de recomendación”, Res. Comput. Sci., vol. 135, pp. 25–40, 2017.
- [20]P. Gómez et al., “Sistemas de Recomendación: un enfoque a las técnicas de filtrado”, Revista Ibérica de Sistemas de Tecnologías de Información, no. E18, pp. 286–293, 2019.
- [21]A. Collado Sanchez, “Sistema de recomendación de recursos basado en filtrado colaborativo para la plataforma edX”, Proyecto fin de carrera, Univ. Carlos III Madr., Leganés, 2014.
- [22]R. Gómez Díaz and A. Cultural, ‘La lematización en español: una aplicación para la recuperación de información’, Rev. Esp. Doc. Cient, vol. 29, no. 1, p. 175, 2006.
- [23]J. Ramos and Others, ‘Using tf-idf to determine word relevance in document queries’, in Proceedings of the first instructional conference on machine learning, 2003, vol. 242, pp. 29–48.
- [24]B. B. Fonseca, O. M. Cornelio, and I. P. Pupo, “Sistema de recomendaciones sobre la evaluación de proyectos de desarrollo de software”, Revista Cubana de Informática Médica, vol. 13, no. 2, 2021.
- [25]S. Baquero Arias, “Sistema de recomendación de intercambios para la dirección de internacionalización de la Universidad de los Andes”, 2022.
- [26]E. G. P. Cuadros, “Un modelo híbrido para la recomendación de libros utilizando reconocimiento facial, filtrado colaborativo y por contenido”, Revista General de Información y Documentación, vol. 34, no. 1, p. 45, 2024.
- [27]J. K. Rask, F. P. Madsen, N. Battle, H. D. Macedo, and P. G. Larsen, ‘Visual studio code vdm support’, in Proceedings of the 18th International Overture Workshop, 2021, pp. 35–49.
- [28]C. Normalización, ‘4.5 Normalización de Base de Datos’, Universidad Tecnológica de Puebla Tecnologías de la Información y Comunicación, p. 54, 2012.

- [28] A. C. Okumus and H. Nur, 'Performance Analysis for Web Scraping Tools: Case Studies on BeautifulSoup, Scrapy, Htmlunit and Jsoup'.
- [29] P. F. Rodríguez Jiménez, 'Ajax', Electiva IV: Arquitectura De Software Web, 2012.

ANEXO A: MODELO RELACIONAL DE LA BASE DE DATOS

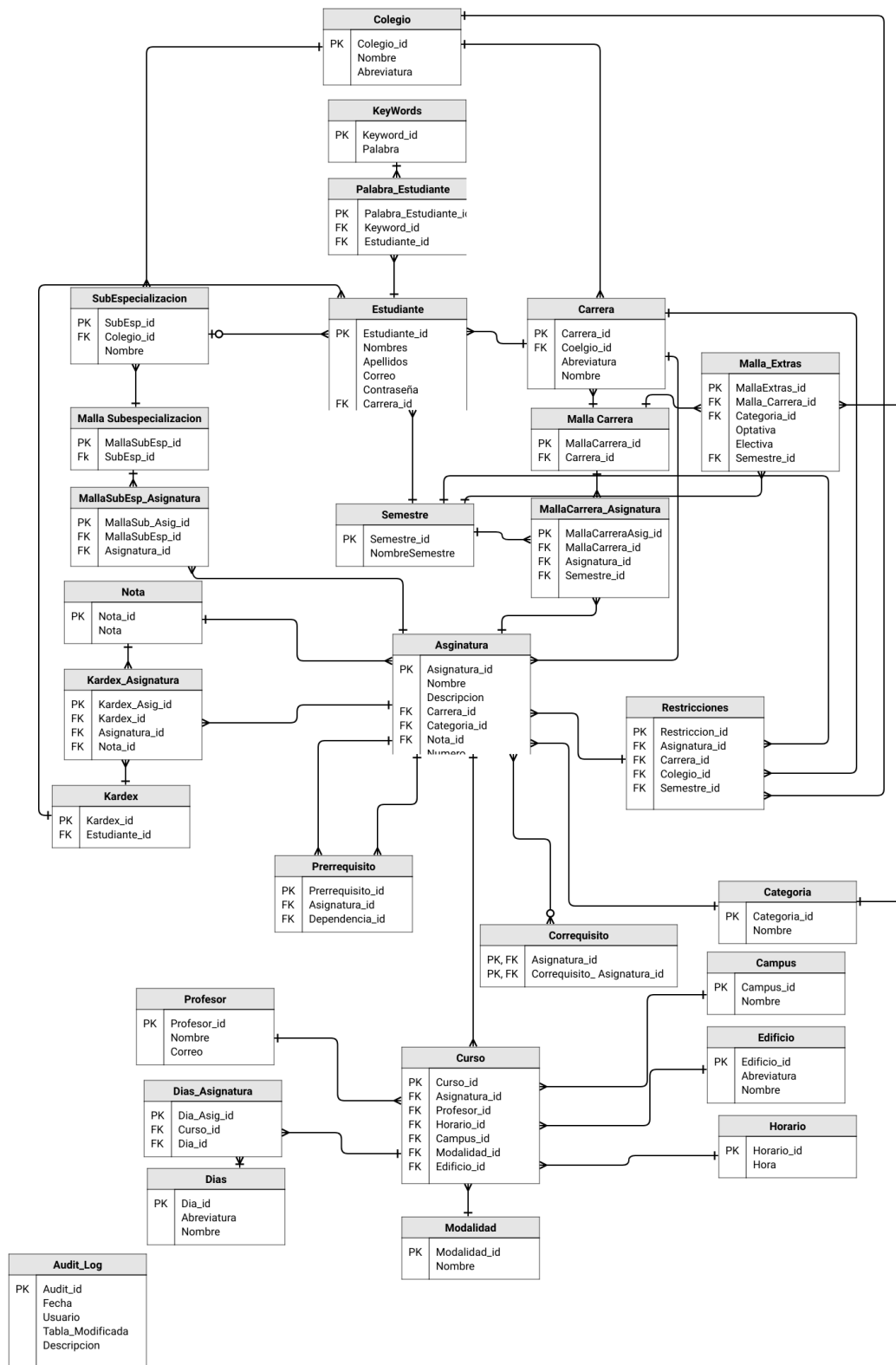


Fig. 32 Modelo Relacional

ANEXO B: INSERCIÓN DE TABLAS PARA LA BASE DE DATOS

Tabla VI Inserción de la tabla Colegio

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Colegio_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-
Abreviatura	VARCHAR(10)	NOT NULL	-	-	-

Tabla VII Inserción de la tabla KeyWords

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Keyword_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Palabra	VARCHAR(255)	NOT NULL	-	-	-

Tabla VIII Inserción de la tabla Semestre

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Semestre_id	INT	NOT NULL	PK	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla IX Inserción de la tabla Notas

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Nota_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nota	VARCHAR(10)	NOT NULL	-	-	-

Tabla X Inserción de la tabla Categoria

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Categoria_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla XI Inserción de la tabla Profesor

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Profesor_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-
Correo	VARCHAR(255)	NOT NULL	-	Correo LIKE '%@usfq.edu.ec'	-

Tabla XII Inserción de la tabla Dias

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Dia_id	INT	NOT NULL	PK	-	-
Abreviatura	CHAR(1)	NOT NULL	-	-	-
Nombre	VARCHAR(50)	NOT NULL	-	-	-

Tabla XIII Inserción de la tabla Horario

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Horario_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Hora	VARCHAR(20)	NOT NULL	-	-	-

Tabla XIV Inserción de la tabla Edificio

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Edificio_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Abreviatura	CHAR(2)	NOT NULL	-	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla XV Inserción de la tabla Modalidad

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Modalidad_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla XVI Inserción de la tabla Carrera

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Carrera_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Colegio_id	INT	NULL	FK (Colegio)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Abreviatura	VARCHAR(10)	NOT NULL	-	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla XVII Inserción de la tabla Subespecialización

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
SubEsp_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Colegio_id	INT	NULL	FK (Colegio)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Nombre	VARCHAR(255)	NOT NULL	-	-	-

Tabla XVIII Inserción de la tabla Estudiante

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Estudiante_id	INT	NOT NULL	PK	REGEXP '[0-9]{6}\$'	-
Nombres	VARCHAR(255)	NOT NULL	-	-	-
Apellidos	VARCHAR(255)	NOT NULL	-	-	-
Correo	VARCHAR(255)	NOT NULL	-	LIKE '%@estud.usfq.edu.ec'	-
Password	VARCHAR(255)	NOT NULL	-	-	-
Carrera_id	INT	NOT NULL	FK (Carrera)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Semestre_id	INT	NOT NULL	FK (Semestre)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
SubEsp_id	INT	NULL	FK (SubEspecialización)	-	ON DELETE SET NULL, ON UPDATE CASCADE

Tabla XIX Inserción de la tabla Palabra_Estudiante

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Palabra_Estudiante_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Keyword_id	INT	NOT NULL	FK (KeyWords)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Estudiante_id	INT	NOT NULL	FK (Estudiante)	-	ON DELETE CASCADE, ON

					UPDATE CASCADE
--	--	--	--	--	-------------------

Tabla XX Inserción de la tabla Asignatura

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Asignatura_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Nombre	VARCHAR(255)	NOT NULL	-	-	-
Descripcion	TEXT	NULL	-	-	-
Carrera_id	INT	NOT NULL	FK (Carrera)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Categoria_id	INT	NOT NULL	FK (Categoria)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Nota_id	INT	NOT NULL	FK (Notas)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Numero	VARCHAR(20)	NOT NULL	-	-	-
Creditos	INT	NOT NULL	-	-	-

Tabla XXI Inserción de la tabla Prerrequisito

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Prerrequisito_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Dependencia_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE

Tabla XXII Inserción de la tabla Subespecialización

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
MallaSubEsp_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
SubEsp_id	INT	NOT NULL	FK (SubEspecializacion)	-	ON DELETE CASCADE, ON UPDATE CASCADE

Tabla XXIII Inserción de la tabla MallaSubEsp_Asignatura

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
MallaSub_Asig_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
MallaSubEsp_id	INT	NOT NULL	FK (Malla_Subespecializacion)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE CASCADE, ON UPDATE CASCADE

Tabla XXIV Inserción de la tabla Malla_Carrera

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
MallaCarrera_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Carrera_id	INT	NOT NULL	FK (Carrera)	-	ON DELETE CASCADE, ON UPDATE CASCADE

Tabla XXV Inserción de la tabla MallaCarreraAsignatura

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
MallaCarreraAsignatura_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
MallaCarrera_id	INT	NOT NULL	FK (Malla_Carrera)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Semestre_id	INT	NOT NULL	FK (Semestre)	-	ON DELETE RESTRICT, ON UPDATE CASCADE

Tabla XXVI Inserción de la tabla Campus

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Campus_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-

Nombre	VARCHAR(255)	NOT NULL	-	-	-
--------	--------------	----------	---	---	---

Tabla XXVII Inserción de la tabla Kardex

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Kardex_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Estudiante_id	INT	NOT NULL	FK (Estudiante)	-	ON DELETE CASCADE, ON UPDATE CASCADE

Tabla XXVIII Inserción de la tabla Kardex_Asignatura

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Kardex_Asignatura_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Kardex_id	INT	NOT NULL	FK (Kardex)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Nota_id	INT	NOT NULL	FK (Notas)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Aprobada	BOOLEAN	NOT NULL	-	-	-

Tabla XXIX Inserción de la tabla Curso

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Curso_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Profesor_id	INT	NULL	FK (Profesor)	-	ON DELETE SET NULL, ON UPDATE CASCADE
Horario_id	INT	NOT NULL	FK (Horario)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Campus_id	INT	NOT NULL	FK (Campus)	-	ON DELETE RESTRICT, ON UPDATE CASCADE

Modalidad_id	INT	NOT NULL	FK (Modalidad)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Edificio_id	INT	NOT NULL	FK (Edificio)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Cupo	INT	NOT NULL	-	-	-
Aula	INT	NOT NULL	-	-	-

Tabla XXX Inserción de la tabla Dias_Asignatura

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Dias_Asignatura_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Curso_id	INT	NOT NULL	FK (Curso)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Dia_id	INT	NOT NULL	FK (Dias)	-	ON DELETE RESTRICT, ON UPDATE CASCADE

Tabla XXXI Inserción de la tabla Correquisito

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Correquisito_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE
Correquisito_Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE RESTRICT, ON UPDATE CASCADE

Tabla XXXII Inserción de la tabla Restricciones

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones
Restriccion_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Asignatura_id	INT	NOT NULL	FK (Asignatura)	-	ON DELETE CASCADE, ON UPDATE CASCADE

Carrera_id	INT	NULL	FK (Carrera)	-	ON DELETE SET NULL, ON UPDATE CASCADE
Colegio_id	INT	NULL	FK (Colegio)	-	ON DELETE SET NULL, ON UPDATE CASCADE
Semestre_id	INT				

Tabla XXXIII Inserción de la tabla Audit_Log

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones en Cascada
Audit_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
Fecha	TIMESTAMP	NULL	DEFAULT CURRENT_TIMESTAMP	-	-
Usuario	VARCHAR(255)	NOT NULL	-	-	-
Tabla_Modificada	VARCHAR(255)	NOT NULL	-	-	-
Descripcion	TEXT	NOT NULL	-	-	-

Tabla XXXIV Inserción de la tabla Malla_Extras

Nombre del Campo	Tipo de Dato	Nulo	Restricción	Checks	Acciones en Cascada
MallaExtras_id	INT	NOT NULL	PK, AUTO_INCREMENT	-	-
MallaCarrera_id	INT	NOT NULL	FK (Malla_Carrera)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Categoria_id	INT	NULL	FK (Categoria)	-	ON DELETE CASCADE, ON UPDATE CASCADE
Optativa	BOOLEAN	NULL	-	-	-
Electiva	BOOLEAN	NULL	-	-	-
Semestre_id	INT	NOT NULL	FK (Semestre)	-	ON DELETE CASCADE, ON UPDATE CASCADE

ANEXO C: DETALLE DE LAS INSERCIONES EN BASE DE DATOS

Tabla XXXV Cantidad de Inserción de datos por tabla

Tabla	Cantidad de Elementos
Asignatura	186
Campus	3 (Cumbaya, Galápagos, Tiputini)
Carrera	57
Categoría	15
Colegio	11
Correquisito	3
Curso	930 (5 por cada Asignatura)
Días	7
Días Asignatura	1816 (Cada asignatura tiene una probabilidad de tener 1 o 2 días)
Edificio	22
Estudiante	101 (999999 es de prueba)
Horario	10 (De 7:00 a 8:20 hasta 8:30 a 9:50 pm)
Kardex	5
Kardex Asignatura	125
Keywords	50 (Ninguna funcional)
Malla Carrera	5 (Compu, Gastronomía, Educación, Administración, Artes Visuales)
Malla Extras	48
Malla Subespecialización	5 (Videojuegos, Cine, Danza, Ing. Industrial, Matemáticas)
Modalidad	4 (Presencial, Híbrido, Virtual, En línea)
Notas	6 (A, B, C, D, F, P)
Palabra Estudiante	505 (5 por estudiante)
Prerrequisito	64
Profesor	212
Restricciones	49
Semestre	12
Subespecialización	74

ANEXO D: SCRIPT DE LA VISTA TABLE_USER

Tabla XXXVI Funciones de la vista table_user

Función	Descripción	Parámetros
loadTable(page)	Controla la paginación de la tabla mostrando solo un número específico de filas por página.	page: Número de página actual
changePage(direction)	Gestiona la navegación entre páginas de la tabla, actualizando la página actual.	direction: Número entero que indica dirección (+1 o -1)
toggleKeywordDropdown()	Alterna la visibilidad del menú desplegable de palabras clave.	ninguno
limitKeywords(checkbox)	Limita la selección de palabras clave a un máximo de 5, mostrando un mensaje de error si se excede.	checkbox: Elemento checkbox que fue clickeado
updateCourses()	Realiza una petición AJAX para actualizar la tabla de cursos basado en los filtros seleccionados.	ninguno
updatePagination()	Actualiza los controles de paginación y la información de páginas mostrada.	ninguno
toggleResults()	Alterna la visibilidad del menú desplegable de resultados.	ninguno
updateCheckedResults()	Actualiza las burbujas visuales que muestran los resultados seleccionados.	ninguno
openCustomDialog()	Abre un diálogo modal con información detallada del curso.	courseName, college, credits, prereq, correq, carr, coleg, semes, cumpleP, cumpleR
closeCustomDialog()	Cierra el diálogo modal.	ninguno
toggleSubEspecializaciones()	Alterna la visibilidad del menú desplegable de subespecializaciones.	ninguno
handleRadioClick(radioButton)	Maneja la lógica de selección/deselección de botones radio.	radioButton: Elemento radio button clickeado

Tabla XXXVII Variables globales de la vista table_user

Variable	Descripción	Valor Inicial
rowsPerPage	Número de filas a mostrar por página	10
currentPage	Página actual siendo mostrada	1
selectedCount	Contador de elementos seleccionados	0
previousRadio	Almacena el último radio button seleccionado	null

Tabla XXXVIII EventListeners de la vista table_user

Evento	Descripción
window.onclick	Cierra los dropdowns cuando se hace clic fuera de ellos
DOMContentLoaded	Inicializa la tabla y la paginación cuando la página carga

ANEXO E: RESULTADOS DE TF_IDF

Tabla XXXIX Variantes utilizadas de tf_idf

Term Frequency	Inverse Document Frequency
Natural	Idf
Natural	No
Logarítmico	Idf
Logarítmico	No
Logarítmico	Probabilístico
Aumentado	Idf
Aumentado	No
Aumentado	Probabilístico
Boolean	Idf
Boolean	No
Boolean	Probabilístico

*Tabla XL Palabras Clave con tf_idf***Palabras Clave con tf_idf en español**

Keyword	NScore
tecnica	9.75609513
incluir	9.56397775
aprenderar	8.67371561
cubrir	7.99191647
concepto	7.90478924
proyecto	6.88556784
tema	6.73412794
explorar	6.60746242
desarrollar	6.48993546
aplicacion	6.05681665
habilidad	5.05418243
avanzado	4.73139341
enfocar	4.68934112
aplicar	4.63520369
analisis	4.4329143
practico	4.30818097
gestion	4.15165285
arte	4.05121639
teoria	3.6449584
problema	3.30664634
principio	3.25561757
introducir	3.09717744
abordar	2.98240665
fundamental	2.53697848
creacion	2.40841624

proporcionar	2.40046076
herramienta	2.10427964
practica	2.10020634
desarrollo	1.89666477
real	1.78623561
sistema	1.57743704
estudiantes	1.49763053
empresarial	1.45485036
artistico	1.4443009
estudio	1.43120654
estudiar	1.42308017
espacio	1.37922758
diseño	1.3398249
aprendizaje	1.26815328
tambien	1.20623417
laboratorio	1.20251717
dato	1.12008765
mas	0.96873664
enfoque	0.90524257
trabajar	0.89142857
analizar	0.86927412
planificacion	0.86779509
presentacion	0.77585179
uso	0.7390451
entorno	0.72136927
mejorar	0.71056182
campo	0.5916771
produccion	0.57544722
diverso	0.50918041
complejo	0.49560877
area	0.48171258
diferente	0.46484015
programacion	0.42516909
educativo	0.39165702
decision	0.36835389
practicar	0.35180634
aplicado	0.35014572
estructura	0.32139586
aprender	0.2427558
matematica	0.18854197
estrategia	0.08248506
caso	0.08057567
crear	0.06432187
basico	0.03051272
social	0.02112568

evaluacion	0.00822945
conocimiento	0.00275875
disenar	0.00257005
ingenieria	0.00225622
integral	0
Palabras clave con tf_idf en inglés	
Keyword	NScore
learn	9.91938235
technique	9.26040112
cover	8.34567952
include	7.90738584
focus	7.70875653
develop	6.42938217
concept	6.4026158
apply	6.30984378
project	6.05900004
explore	5.66829069
art	5.07780167
topic	4.95943498
application	4.68646935
skill	4.66809989
study	4.56703682
analysis	4.24531688
advanced	4.00075197
management	3.93960613
practical	3.87116663
design	3.6527816
fundamental	3.62591903
provide	3.29911619
introduce	3.23775868
theory	3.22051201
business	3.21370918
principle	3.16270703
problem	2.80025377
work	2.18057203
basic	2.0869073
tool	1.85668984
artistic	1.64906882
real	1.5960847
create	1.49334664
field	1.32341783
space	1.30694173
use	1.24298852
system	1.12955258
process	0.99897777

creation	0.89853851
different	0.86740503
learning	0.63104061
complex	0.5939592
educational	0.57614077
area	0.56394007
production	0.55974959
structure	0.51243362
practice	0.5055745
development	0.49464487
solve	0.48754845
teach	0.48059504
address	0.48007167
decision	0.47816744
plan	0.3977533
planning	0.39595023
mathematic	0.37215069
environment	0.36511704
presentation	0.34808729
critical	0.27788994
programming	0.26341937
teaching	0.26124783
strategy	0.21112853
hand	0.20677753
improve	0.19184885
emphasis	0.17777637
allow	0.16172135
knowledge	0.1126256
case	0.10557134
social	0.06803295
current	0.01658876
contemporary	0.00883434
science	0.00830678
laboratory	0.00723836
prepare	0.00234074
engineering	0.00128396
control	0

ANEXO F: PRUEBAS DE LA BASE DE DATOS

Nombre Asignatura varchar	Codigo Asignatura varchar	Numero de Creditos bigint	Semestre varchar	Semestre_id int	Categoria varchar
Escritura Academica	CMP-1001	3	Primer Semestre	1	Colegio General
Taller de Ing. Cs. Computaci	CMP-1001	3	Primer Semestre	1	Ingenierías
Cálculo Diferencial + Ej	CMP-1201	3	Primer Semestre	1	Ciencias Exactas
Química General 1 +Lab/Ej	CMP-1003	3	Primer Semestre	1	Ingenierías
Cosmos	CMP-1002	3	Primer Semestre	1	Colegio General
Inglés Nivel 1	CMP-0001	0	Primer Semestre	1	Lenguas
Inglés Nivel 2	CMP-0002	0	Primer Semestre	1	Lenguas
Programación en C++ + Ej	CMP-1101	3	Segundo Semestre	2	Ingenierías
Cálculo Integral + Ej	CMP-1202	3	Segundo Semestre	2	Ciencias Exactas
Autoconocimiento	CMP-1001	3	Segundo Semestre	2	Colegio General
Ser y Cosmos	CMP-2001	3	Segundo Semestre	2	Colegio General
Inglés Nivel 3	CMP-0003	0	Segundo Semestre	2	Lenguas
Inglés Nivel 4	CMP-0004	0	Segundo Semestre	2	Lenguas
Humanidades	(NULL)	3	Segundo Semestre	2	Humanidades
Programación Avanzada en	CMP-2102	3	Tercer Semestre	3	Ingenierías

Fig. 33 GetMallaCarrera para Ing. en Ciencias de la Computacion

Subject Name varchar	Subject Code varchar	Grade varchar	Status varchar	Categoria varchar
Introducción a la Economía	ECN-1001	A	Aprobada	Ciencias Sociales
Programación de Apps	CMP-2103	A	Aprobada	Ingenierías
Álgebra Lineal 1 +Ej	MAT-1401	A	Aprobada	Ciencias Exactas
Física para Ingeniería 2 +Lab/Ej	FIS-2702	A	Aprobada	Ciencias Exactas
Aprendizaje y Servicio PASEC	PRC-2000	A	Aprobada	Colegio General
Fotografía 1	FOT-2101	A	Aprobada	Artes
Cálculo Diferencial + Ej	MAT-1201	B	Aprobada	Ciencias Exactas
Programación en C++ +Ej	CMP-1101	B	Aprobada	Ingenierías
Matemáticas Discretas	MAT-2004	B	Aprobada	Ciencias Exactas
Física para Ingeniería 1 +Lab/Ej	FIS-2701	B	Aprobada	Ciencias Exactas
Cálculo Vectorial	MAT-2203	B	Aprobada	Ciencias Exactas
Inglés Nivel 6	ESL-0006	B	Aprobada	Lenguas
Análisis Numérico	MAT-2001	B	Aprobada	Ciencias Exactas
Ser y Cosmos	ARL-2001	C	Aprobada	Colegio General
Inglés Nivel 5	ESL-0005	C	Aprobada	Lenguas

Fig. 34 GetKardexEst para el estudiante de prueba

Nombre Asignatura varchar	Codigo Asignatura varchar	Numero de Creditos bigint	Semestre varchar	Estado varchar
Proyectos: Gerencia y Análisis	CMP-4011	3	Octavo Semestre	No Tomada
Práctica Pre-Profesional PAS	CMP-4000	5	Octavo Semestre	No Tomada
Electiva	(NULL)	3	Octavo Semestre	No Tomada
Cultura Gastronómica	CMP-0010	1	Quinto Semestre	No Tomada
Electrónica Básica +Lab	CMP-2001	3	Quinto Semestre	No Tomada
Estructuras de Datos	CMP-3002	3	Quinto Semestre	No Tomada
Probabilidad y Estadística +	CMP-2008	3	Quinto Semestre	No Tomada
Teoría de la Computación	CMP-3005	3	Quinto Semestre	No Tomada
Ciencias Sociales	(NULL)	3	Quinto Semestre	No Tomada
Humanidades	(NULL)	3	Segundo Semestre	No Tomada
Inteligencia Artificial	CMP-4004	3	Séptimo Semestre	No Tomada
Base de Datos	CMP-4002	3	Séptimo Semestre	No Tomada
Optativa	(NULL)	3	Séptimo Semestre	No Tomada
Optativa	(NULL)	3	Séptimo Semestre	No Tomada

Fig. 35 ObtenerAsignaturasRestantes para el estudiante de prueba

Asignatura_id int	Nombre varchar	Codigo Asignatura varchar	Prerrequisitos long_blob
Filter	Filter	Filter	Filter
186	Sistemas Lean	IIN-4010	Ingeniería de la Calidad + Lab

Fig. 36 ObtenerDetallesdeAsignatura para ver prerrequisito de una asignatura

Estado varchar
Filter
No cumple con los prerrequisitos

Fig. 37 VerificarPrerrequisitos para el estudiando de prueba y la materia Sistemas Lean

Asignatura varchar	Correquisito varchar
Filter	Filter
Escritura Academica	Cosmos

Fig. 38 ObtenerCorrequisito

Asignatura varchar	Carrera varchar	Colegio varchar	Semestre varchar
Filter	Filter	Filter	Filter
Programación Avanzada de Apps	(NULL)	Colegio de Ciencias e Ingenierías	Quinto Semestre

Fig. 39 ObtenerInformacionAsignatura para ver las restricciones

Resultado varchar
Filter
> El estudiante cumple con todas las restricciones para tomar la asignatura.

Fig. 40 VerificarRestriccionesEstudiante para el estudiante de prueba y la asignatura de Programación Avanzada de Apps

Asignatura_id int	Numero varchar	Asignatura varchar	Carrera varchar	Colegio varchar
98	2009	Estadística Empresarial +Lab	Matematica	Colegio de Ciencias e Ingenierías
149	3501	Teoría de Grupos	Matematica	Colegio de Ciencias e Ingenierías
150	3002	Variable Compleja	Matematica	Colegio de Ciencias e Ingenierías
151	2002	Ecuaciones Diferenciales	Matematica	Colegio de Ciencias e Ingenierías
153	2003	Introducción a Probabilidades	Matematica	Colegio de Ciencias e Ingenierías
154	2006	Fundamentos de Geometría	Matematica	Colegio de Ciencias e Ingenierías
155	2007	Lógica y Teoría de Conjuntos	Matematica	Colegio de Ciencias e Ingenierías
156	2010	Inferencia Estadística	Matematica	Colegio de Ciencias e Ingenierías
158	3004	Teoría de Números	Matematica	Colegio de Ciencias e Ingenierías
159	3005	Combinatoria y Grafos	Matematica	Colegio de Ciencias e Ingenierías
160	3080	Ecuaciones Diferenciales Parciales	Matematica	Colegio de Ciencias e Ingenierías
161	3402	Álgebra Lineal 2	Matematica	Colegio de Ciencias e Ingenierías
162	3501	Álgebra Abstracta 1	Matematica	Colegio de Ciencias e Ingenierías

Fig. 41 GetOptativas para el estudiante de prueba

Asignatura_id int	Numero varchar	Asignatura varchar	Carrera varchar	Colegio varchar
Filter	Filter	Filter	Filter	Filter
47	1101	Conceptos y Tecnicas 1	Gastronomia	Colegio de Hospitalidad, Arte Culinario y Turismo
48	1001	Principios de Administracion	Administracion de Empresas	Colegio de Administración de Empresas
49	2002	Panaderia	Gastronomia	Colegio de Hospitalidad, Arte Culinario y Turismo
50	1001	Nutricion Humana +Lab	Nutricion y Dietetica	Colegio de Ciencias de la Salud
51	1001	Introduccion a la Hospitalidad	Hospitalidad y Hoteleria	Colegio de Hospitalidad, Arte Culinario y Turismo
52	2002	Contabilidad Empresarial	Finanzas	Colegio de Administración de Empresas
54	1200	Practica Culinaria 1	Gastronomia	Colegio de Hospitalidad, Arte Culinario y Turismo
55	2001	Administracion de A & B	Hospitalidad y Hoteleria	Colegio de Hospitalidad, Arte Culinario y Turismo
56	2003	Carniceria	Gastronomia	Colegio de Hospitalidad, Arte Culinario y Turismo
58	1101	Introducción a la Biología +Ej	Biologia	Colegio de Ciencias Biológicas y Ambientales
59	2003	Introduccion al Marketing HSP	Hospitalidad y Hoteleria	Colegio de Hospitalidad, Arte Culinario y Turismo
60	0001	Coloquios_Gastr	Hospitalidad y Hoteleria	Colegio de Hospitalidad, Arte Culinario y Turismo
61	3005	Alta Cocina Francesa	Gastronomia	Colegio de Hospitalidad, Arte Culinario y Turismo

Fig. 42 GetElectivas para el estudiante de prueba

Subject C varchar	Subject Name varchar	NRC int	Modalidad varchar	Day long	Schedule varchar	Professor varchar	Campus varchar	Location varchar	Credits int
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
CMP-5992	Proyecto Integrador CMP	221	Virtual Sincrónico	L, I	17:30 - 18:50	Simeon Floyd	Galapagos	Miguel de Santiago 111	5
CMP-5992	Proyecto Integrador CMP	222	Virtual Sincrónico	V	08:30 - 09:50	Rodrigo Xavier Jaramillo Almeida	Cumbaya	Da Vindi 202	5
CMP-5992	Proyecto Integrador CMP	223	Virtual Sincrónico	M, J	17:30 - 18:50	Alberto Cardenas Ruales	Cumbaya	Taller de metal 212	5
CMP-5992	Proyecto Integrador CMP	224	Virtual Sincrónico	M, J	20:30 - 21:50	Paul Rosero Contreras	Estacion Tiputini	La Casita Tomate 218	5
CMP-5992	Proyecto Integrador CMP	225	En Línea	M, J	10:00 - 11:20	Miguel Angel Mendez Silva	Cumbaya	Da Vindi 208	5

Fig. 43 VerCursos

Clave varchar	Asignatura varchar	Creditos int
Filter	Filter	Filter
CIN 1002	Lenguaje del Cine	3
CIN 2007	Lenguaje Visual y Montaje	3
CIN 2008	Cinematografía	3
CIN 2003	Sonido	3
CIN 2006	Storytelling	3
CIN 2012	Diseño de Producción	3

Fig. 44 getSubespecializacion para el minor de Cine

Clave varchar	Asignatura varchar	Creditos int
Filter	Filter	Filter
CIN 1002	Lenguaje del Cine	3
CIN 2007	Lenguaje Visual y Montaje	3
CIN 2008	Cinematografía	3

Fig. 45 verSubTomadas para un estudiante que toma el minor de Cine

Clave varchar	Asignatura varchar	Creditos int
Filter	Filter	Filter
CIN 2003	Sonido	3
CIN 2006	Storytelling	3
CIN 2012	Diseño de Producción	3

Fig. 46 verSubRestantes para un estudiante que toma el minor de Cine

ANEXO G: PANTALLAS DE LA PÁGINA WEB

The image shows a login form titled "USFQ - Buscador de Cursos" with a subtitle "Iniciar Sesión". It contains two input fields: "Correo institucional:" and "Contraseña:". Below the fields is a red button labeled "Iniciar Sesión". The form is centered on a light gray background.

USFQ - Buscador de Cursos

Iniciar Sesión

Correo institucional:

Contraseña:

Iniciar Sesión

Fig. 47 Inicio de Sesión

Fig. 48 Buscador de Cursos

Historial de asignaturas

Bienvenido

Cerrar

Show 10 entries

Search:

Código	Nombre	Nota	Estado
ARL-1001	Autoconocimiento	A	Aprobada
ARL-1002	Cosmos	A	Aprobada
ARL-2001	Ser y Cosmos	C	Aprobada
CMP-1001	Taller de Ing. Cs. Computación	A	Aprobada
CMP-1101	Programación en C++ +Ej	B	Aprobada
CMP-2102	Programación Avanzada en C++	A	Aprobada
CMP-2103	Programación de Apps	A	Aprobada
ECN-1001	Introducción a la Economía	A	Aprobada
ESL-0001	Inglés Nivel 1	A	Aprobada
ESL-0002	Inglés Nivel 2	A	Aprobada

Showing 1 to 10 of 26 entries

Previous 1 2 3 Next

Fig. 49 Historial de Asignaturas

Optativas y Electivas Recomendadas

Bienvenido

Cerrar

Código	NRC	Curso	Cupos
DIM-1001	711	Programación Para Diseño 1 Profesor: Pedro Javier Moncayo Herrera Campus: Galapagos Día: M, J Horario: 19:00 - 20:20 Edificio: D-215 Modalidad: Híbrido	45
DIM-1001	712	Programación Para Diseño 1 Profesor: Javier Roberto Arano Campus: Galapagos Día: M, J Horario: 13:00 - 14:20 Edificio: S-206 Modalidad: Híbrido	25
DIM-1001	713	Programación Para Diseño 1 Profesor: Jesus Enrique Estevez Monagas Campus: Estacion Tiputini Día: M, J Horario: 20:30 - 21:50	45

Fig. 50 Sistema de Recomendación