

Arquitecturas, lógica y despliegue de aplicaciones

Breve descripción:

El componente formativo está orientado a entregar los conocimientos sobre el manejo de las arquitecturas para la construcción de aplicaciones, lógica a aplicar para el funcionamiento de esta y el despliegue de la aplicación en un servidor de producción.

Mayo 2023

Tabla de contenido

Introducción.....	3
1. ¿Qué es un patrón de arquitectura?.....	6
2. Lenguaje de consulta.....	14
3. Integración de la capa lógica en las interfaces de usuario	23
4. Servicios web.....	32
5. Manejo y control de versiones	37
Síntesis	42
Material complementario	43
Glosario.....	44
Referencias bibliográficas.....	45
Créditos.....	46

Introducción

En los mercados actuales en la industria del software existen diferentes caminos y tecnologías a aplicar en la construcción de software, muchas de ellas se orientan a la funcionalidad de estos, pero otras se encuentran enmarcadas en el ámbito y la capacidad de poder adaptarse a las necesidades cambiantes de los usuarios de hoy en día.

A través del siguiente video se dará una introducción a esta interesante temática:

Video 1. Arquitecturas, lógica y despliegue de aplicaciones



[Enlace de reproducción del video](#)

Síntesis del video: arquitecturas, lógica y despliegue de aplicaciones

Se puede construir software con alta calidad de gráficos y demandas de interfaces, pero se debe resguardar el correcto funcionamiento de las aplicaciones. Por ello, nace un concepto que hasta los días modernos está en un constante crecimiento, y es el tema de las arquitecturas.

Si hace una exploración de todas las que existen actualmente en el mercado, encontrará que existe un patrón particular en su funcionamiento y su capacidad de brindar mejores servicios y despliegue de las aplicaciones. Una arquitectura no es más que la utilización de técnicas y software para garantizar que una aplicación no solo realice su proceso de manera correcta, sino que tenga las siguientes características: operatividad, fácil despliegue y escalabilidad.

Estas características significan que la aplicación opere de manera correcta, pero que también sea escalable. Esta palabra es fundamental para la mayoría del software actual, puesto que en los mercados actuales es necesario generar modificaciones y mejoras a los sistemas de información. Esto requiere que sean contruidos de tal manera que, al momento de agregar un cambio o mejora, se pueda hacer de la manera más rápida y fácil posible. Esto facilita mejorar los tiempos de entrega y producción de los sistemas de información.

Hoy en día, la industria del software se encuentra focalizada en el cliente, el cual cada vez más exige calidad y escalabilidad en las aplicaciones. Esto es un factor predominante en algunos sectores, como la construcción, las ventas, entre otros.

Estos sectores siempre buscan sacar el mayor beneficio al usar técnicas y procesos que ayuden al mejor desempeño de las actividades.

Podemos llegar a la conclusión de que el uso de patrones y modelos de producción o construcción de productos siempre generará el mejor resultado.

1. ¿Qué es un patrón de arquitectura?

Es un modelo que permite llegar a la solución de estructuración de un proyecto de software, este facilita que tanto el proceso de diseño como la construcción del producto se realice de manera correcta, usando técnicas que mantengan la escalabilidad, la seguridad y la operatividad del producto de software que se desea construir.

Con los patrones de arquitectura también se genera una ruta trazable donde se pueden visualizar cada uno de los componentes que conforman el producto y su aporte al funcionamiento del sistema en general, es decir, se puede desglosar cada parte del software y conocer su comportamiento y funcionamiento como parte integral del sistema.

En este caso al poder tener la posibilidad de conocer cada una de sus partes y la interacción que tienen con los demás componentes del software su modificación y construcción es mucho más sencilla. Por lo que se puede construir el software por componentes y luego agruparlos de manera completa en una sola solución, al tener dicha posibilidad si se requiere modificar una parte del software solo se concentrará en esa y no en todo el conjunto en general, por lo que hace que el escalamiento y la versatilidad de modificaciones sea mucho más fácil para quien está realizando dichas modificaciones.

La arquitectura de “software” es una disciplina muy relevante en el mundo del desarrollo web o App. Ahora, se va a profundizar en los diferentes tipos de arquitectura de “software” que existen, pero antes es necesario conocer qué significa esta ciencia.

En el mundo del desarrollo existen complejos problemas a tratar. Si bien es conocido que el software puede ser más o menos complejo, lo más probable es que vaya mejorando para cubrir nuevas necesidades o funcionalidades. Es por ello que la elección de una estructura o método en el que encajar todas las piezas o componentes es vital para obrar un sistema escalable que dé solución a los problemas del cliente.

En la actualidad existen muchos patrones de arquitectura de software, pero se centrará en 2 de ellos:

- MVC (Modelo Vista Controlador)
- MVT (Modelo Vista Template)

Estos patrones de arquitectura son los que actualmente rigen el proceso de construcción de las aplicaciones web en el mercado, recuerde que los patrones de arquitectura están diseñados para que un producto de software sea elevado a una calidad tal que pueda ser modificado y escalable sin necesidad de tantos esfuerzos por parte de los equipos de desarrollo.

Esto deriva en el uso de buenas prácticas al momento de construcción de un producto de software, eso quiere decir, que se puede garantizar no solo un buen proceso de construcción, sino también la utilización de un estándar de desarrollo a nivel internacional.

Gracias a este tipo de arquitecturas las aplicaciones de hoy en día pueden ser desplegadas y construidas con mayor facilidad que en otros tiempos. No es un secreto que muchas personas y hasta empresas no utilizan patrones de arquitectura para la construcción de sus productos de software, esto no quiere decir que estos productos no cumplan con los requerimientos que necesitan los clientes.

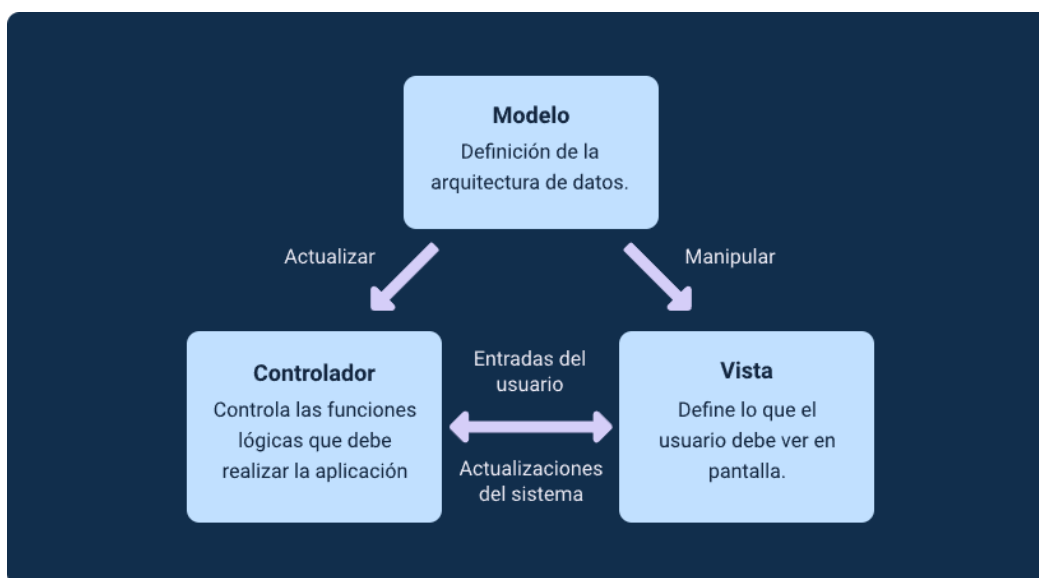
Estas pueden ser funcionales y de alguna manera solucionar problemas que requiera el sector empresarial; pero lo que sí no se puede dejar de lado es que al momento de realizar una modificación o agregar nuevas funcionalidades los tiempos de elaboración sí pueden variar de manera negativa, por eso siempre la invitación es a utilizar los estándares y los patrones de arquitectura para garantizar la calidad en los productos de software que se desarrollen.

Patrón Modelo Vista Controlador

Es un patrón en el diseño de “software” comúnmente utilizado para implementar las interfaces de usuario, los datos y la lógica de control. Enfatiza en una separación entre la lógica de negocios y su visualización.

Esta "separación de preocupaciones" proporciona una mejor división del trabajo y una mejora en el mantenimiento. Ver figura.

Figura 1. Diagrama del MVC



Las tres partes del patrón (modelo, vista y controlador) de diseño de software MVC se pueden describir de la siguiente manera:

Modelo

El modelo define qué datos debe contener la aplicación. Si el estado de estos datos cambia, el modelo generalmente notificará a la vista (para que la pantalla pueda cambiar según sea necesario) y, a veces al controlador (si se necesita una lógica diferente para controlar la vista actualizada). Volviendo a la aplicación de la lista de compras, el modelo especificará qué datos deben contener los artículos de la lista (artículo, precio, etc.) y qué artículos de la lista ya están presentes. Regularmente el modelo tiene estrecha comunicación con la base de datos o el modelo de almacenamiento que se esté utilizando para el aplicativo, este contiene todas las entidades, las relaciones y los datos que requiere el programa para operar y tiene la disposición de proporcionar a la aplicación esta información cuando sea requerido por el mismo.

Vista

La vista define cómo se deben mostrar los datos de la aplicación. En la aplicación de la lista de compras, la vista definiría cómo se presenta la lista al usuario y recibiría los datos para mostrar desde el modelo. Las vistas representan los objetos, los formularios o el despliegue de la información para el usuario que opera el sistema, es decir, es la cara visible al usuario con la que este interactúa y realiza el envío o solicitud de información, según sea lo requerido por el usuario del sistema, ahora bien, es importante tener en cuenta que los estados de la vista no solo se pueden ver No aplica afectados por el usuario de la aplicación, sino que pueden existir eventos y

procedimientos que de acuerdo con la información de entrada por otros sistemas o programadas puede realizar cambios en la vista.

Controlador

El controlador contiene una lógica que actualiza el modelo y/o vista en respuesta a las entradas de los usuarios de la aplicación, por ejemplo, la lista de compras podría tener formularios de entrada y botones que permitan agregar o eliminar artículos. Estas acciones requieren que se actualice el modelo, por lo que la entrada se envía al controlador, que luego manipula el modelo según corresponda, que luego envía los datos actualizados a la vista. Sin embargo, es posible que también se desee actualizar la vista para mostrar los datos en un formato diferente, por ejemplo, cambiar el orden de los artículos de menor a mayor precio o en orden alfabético. En este caso, el controlador podría manejar esto directamente sin necesidad de actualizar el modelo.

Implementar dicho patrón en sus proyectos puede beneficiarlo en los siguientes puntos:

- **Organización modular.** Podrá dividir la lógica del programa haciendo la aplicación más escalable.
- **Puede hacer abstracción de los datos para facilitar la realización de consultas a la base de datos.** Esto se da en el caso de los “framework” como Ruby on Rails, ASP NET, etc.
- **Código más organizado y legible.** Siendo ideal si trabaja en equipo o para continuar el trabajo de otro programador.
- **Este patrón se implementa hace muchos años y lo utilizan grandes empresas,** debido a su correcto diseño y extensibilidad.

Patrón Modelo Vista “Template”

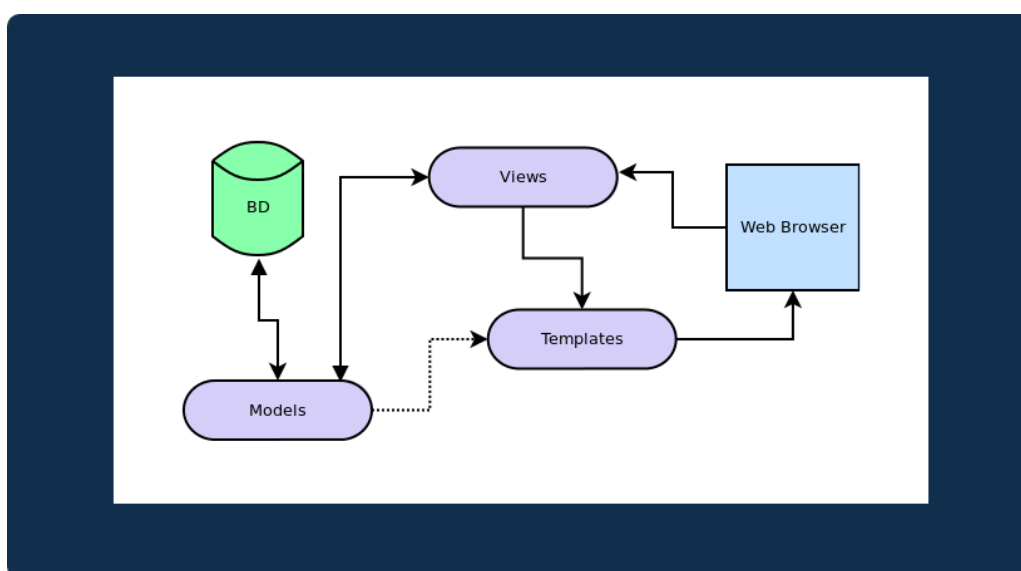
Es utilizado para mostrar inicialmente los datos al usuario con unas vistas ya definidas que se llaman “templates” o plantillas, las cuales tienen una estructura construida y solo se requiere que sean llamadas para enviar los datos a la misma y mostrarlos al usuario.

A diferencia del MVC en donde las vistas pueden variar de acuerdo con lo solicitado por el usuario, en este modelo se tiene una plantilla específica para mostrar la información.

Aunque esto puede parecer no ser una buena práctica, de hecho, facilita mucho la construcción de aplicaciones ya que las vistas están definidas y solo falta entregarle la información que se solicitará en otros aspectos, siendo mucho más rápida su ejecución y entendimiento.

A continuación, verá una figura que ilustra estos aspectos en general.

Figura 2. Modelo Vista “Template”



Nota. Adaptado de <https://docs.hektorprofe.net/django/web-personal/patron-mvt-modelo-vista-template/>

Como se puede apreciar en la figura este tipo de arquitectura realiza el paso de una plantilla para la visualización de los datos al cliente y como se comentó anteriormente esto permite que las vistas ya se encuentren definidas, por lo que facilita de alguna manera las consultas e información a visualizar.

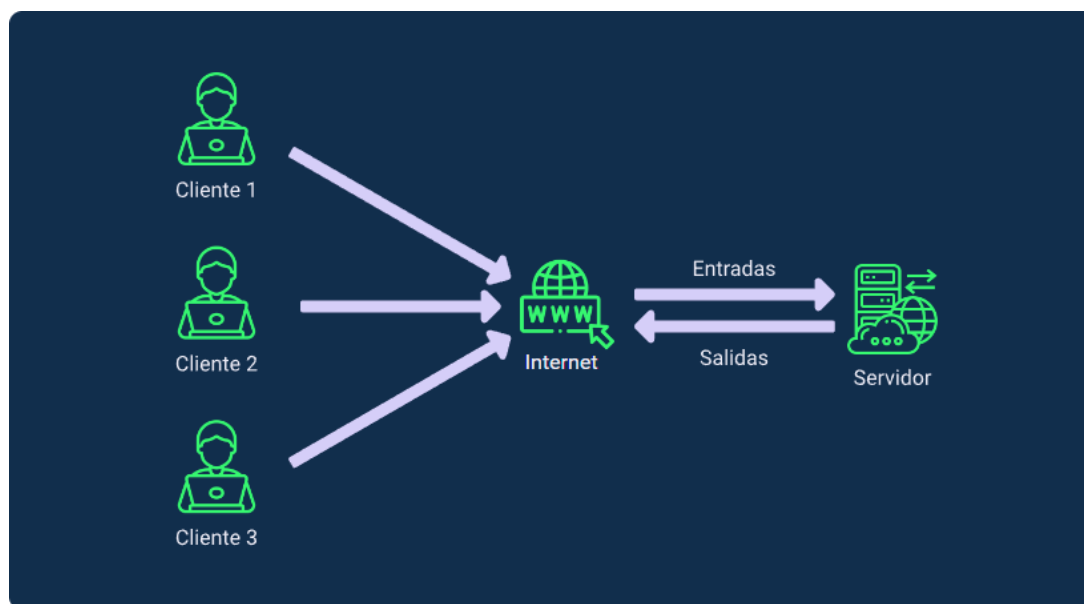
Si bien se centrará en los modelos MVC y MVT, también existe el Modelo Cliente-Servidor al cual se le abrirá un pequeño espacio.

Modelo Cliente - Servidor

Es un modelo de diseño de “software” en el que las tareas se reparten entre los proveedores de los recursos o los servicios, llamados los servidores y los demandantes, y los clientes. Un cliente realiza peticiones a otro programa y el servidor le da respuesta.

Este modelo es el utilizado por muchos años para el funcionamiento de los sitios web y servicios a nivel mundial, el servidor es una máquina dedicada a recibir peticiones de equipos que se encuentran repartidos por todo el mundo, contiene los aplicativos y los datos que requieren las aplicaciones y mediante un proceso de comunicación que incluye diferentes tipos de protocolos recibe la información, la procesa y la entrega a los equipos de clientes que la solicitan.

Figura 3. Modelo Cliente Servidor



Como se ve en la figura se tiene a tres clientes interactuando con el servidor a través del servicio de conectividad de Internet, lo que puede suceder es que cada uno de estos clientes se encuentra en diferentes partes del mundo y de esta manera pueden hacer uso de los recursos del aplicativo.

Cabe anotar que muchos de los servicios que se utilizan hoy en la vida cotidiana, tales como el correo electrónico, las redes sociales, entre otros, funcionan bajo este mismo esquema, por tanto, se espera que estos conceptos y conocimientos aporten a su vida profesional, personal y, que siga avanzando en su proceso de aprendizaje.

Tenga en cuenta que también podrá explorar otros patrones de arquitectura, ya que existen varios en el mercado, aunque para efectos prácticos se hará énfasis en los que se presentaron.

2. Lenguaje de consulta

También llamado SQL que significa “Lenguaje de consulta estructurado” es un estándar orientado a las bases de datos que permite recuperar información contenida en las mismas.

Este lenguaje utiliza las entidades y las relaciones que se encuentran en una base de datos, para que luego mediante una serie de comandos ejecutados de manera ordenada se logre la recuperación de la información contenida en la misma.

Es importante tener en cuenta que el lenguaje SQL es un estándar para todas las bases de datos, es decir, puede ser utilizado en todas. Ahora bien, existen algunas variaciones que se pueden dar dependiendo del fabricante del motor de base de datos, pero son mínimas ya que debe regirse por el estándar SQL implementado a nivel internacional.

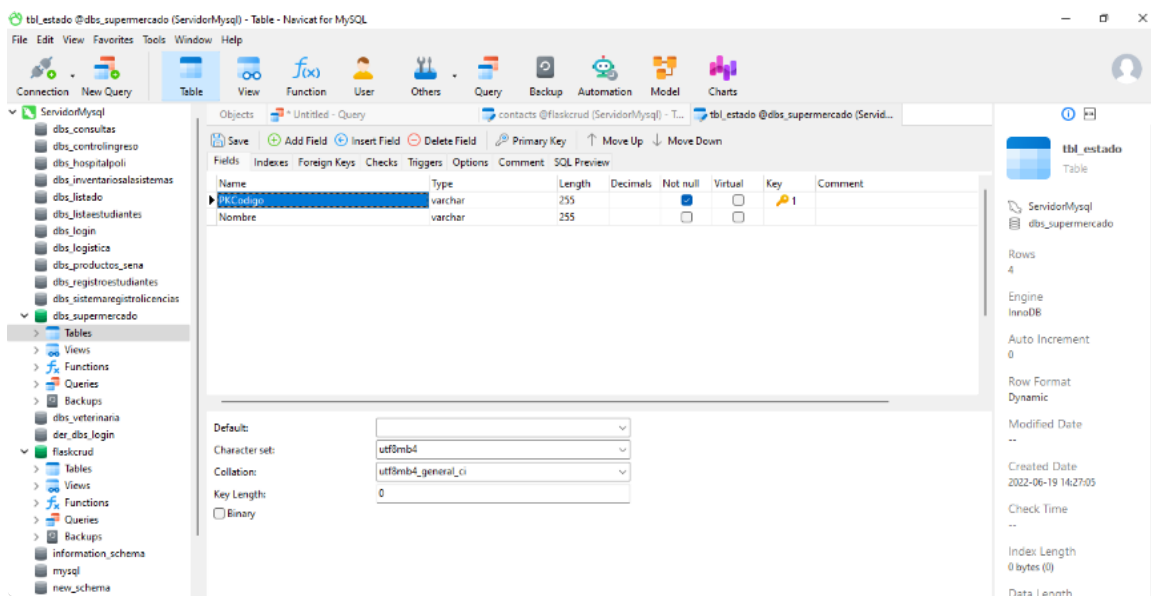
Hoy en día la información juega un papel fundamental para las organizaciones y las empresas a nivel mundial, siendo las bases de datos el sistema de almacenamiento principal de la información.

Requiere que no solo se puedan manipular los datos de manera correcta en cuanto a su inserción, modificación y eliminación sino que también se puedan recuperar estos con mecanismos seguros y que garanticen que dicha información sea entregada de una manera legible y entendible para el usuario, es allí donde el lenguaje SQL toma una gran importancia en los mercados actuales, ya que este permite personalizar la manera cómo se entregan los datos al usuario y así entregar información precisa y confiable. A continuación, verá varios ejemplos sobre cómo utilizar este lenguaje de consultas.

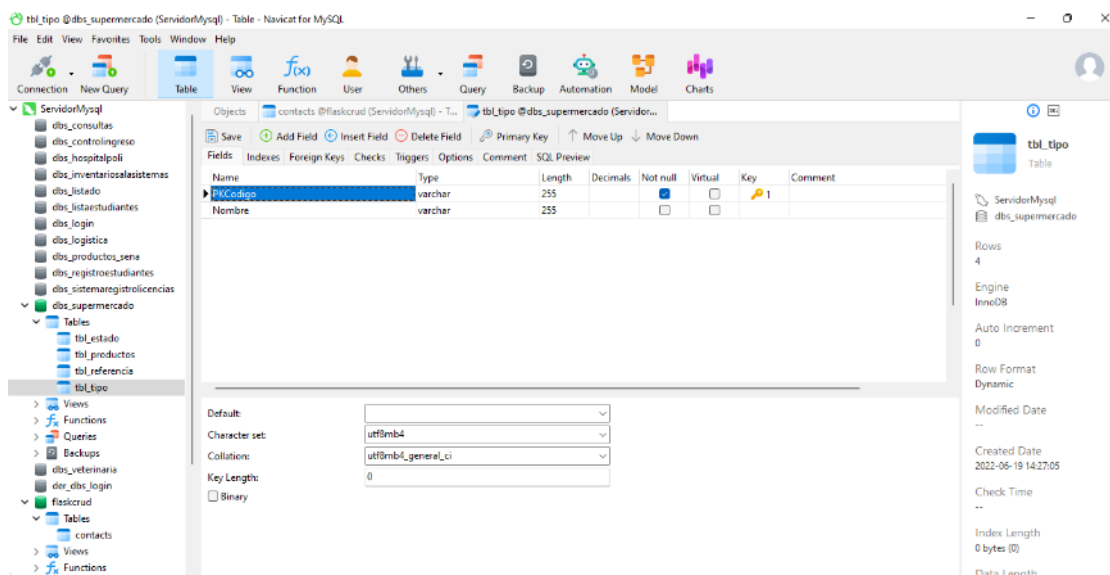
Lo primero que va a tener es una base de datos, que servirá como base para realizar las consultas que necesita.

Antes de iniciar con el proceso de consultas deberá crear la base de datos, para lo cual se entregan las tablas necesarias, con sus tipos de datos para que la cree con dicha información:

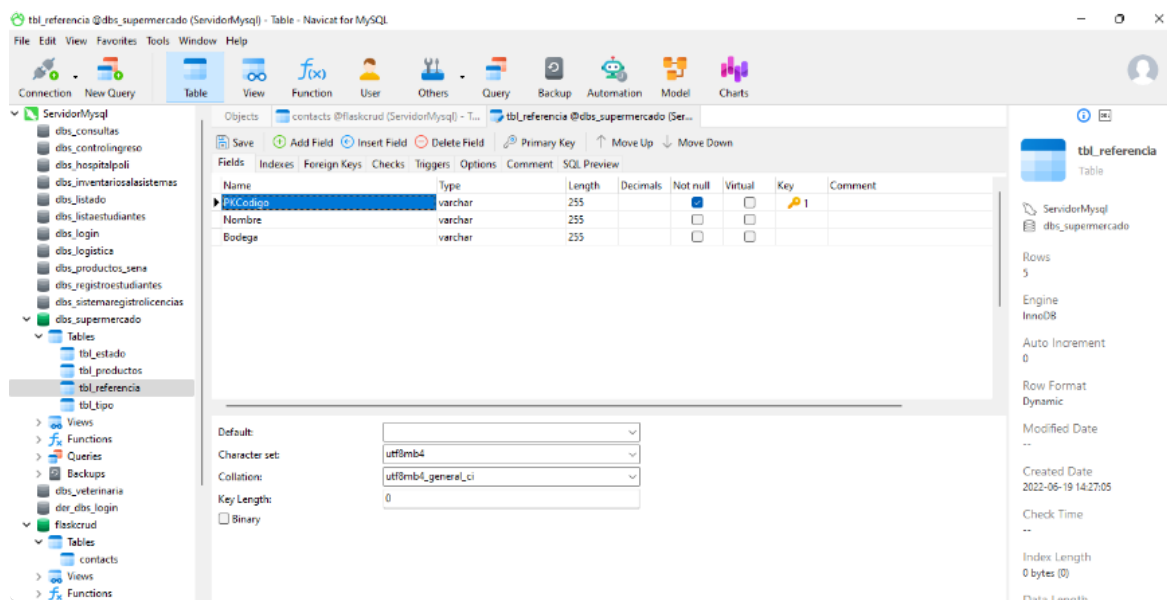
- a) Comience colocando el nombre de la base de datos que será **“dbs_supermercado”**.
- b) Y sigue con la configuración de las siguientes tablas, como se muestra a continuación:
 - Tabla tbl_estado
 - Tabla tbl_tipo
 - Tabla tbl_referencia
 - Tabla tbl_productos
- c) Tabla tbl_estado



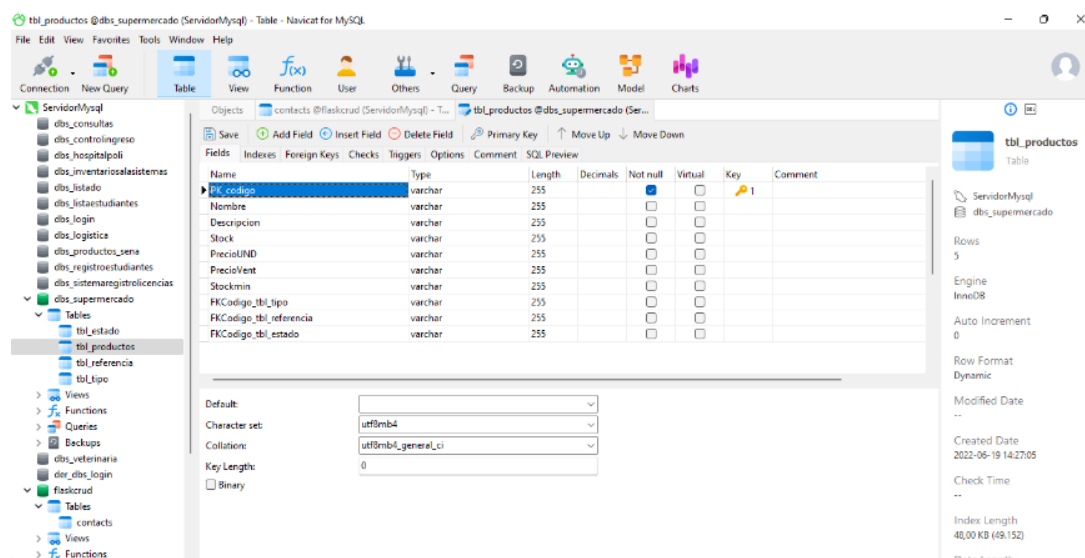
d) Tabla tbl_tipo



e) Tabla tbl_referencia



f) Tabla tbl_productos



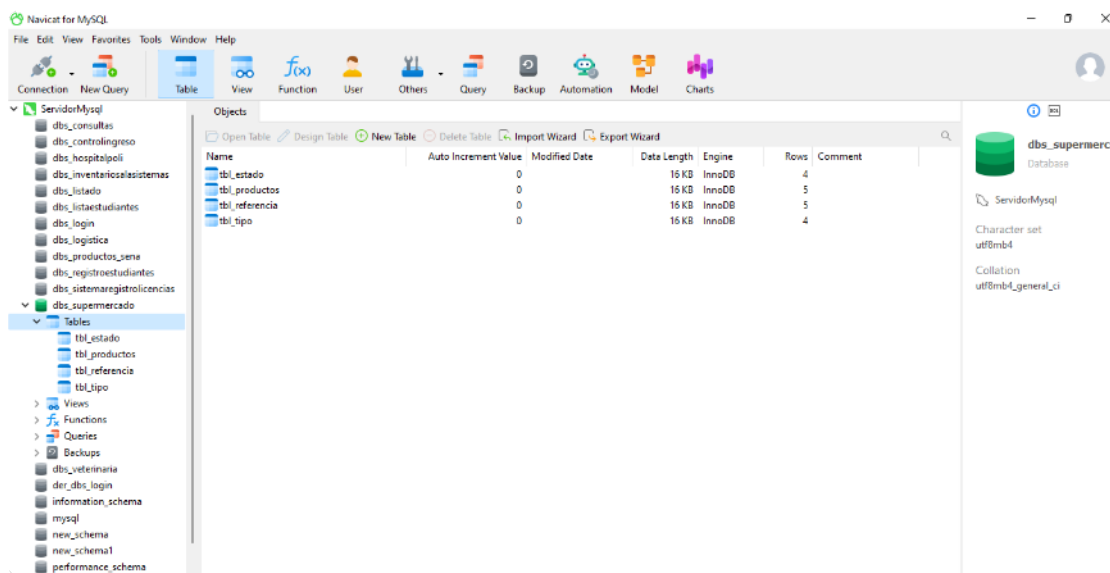
The screenshot shows the Navicat for MySQL interface with the 'tbl_productos' table selected. The table structure is as follows:

Name	Type	Length	Decimals	Not null	Virtual	Key	Comment
id_codigo	varchar	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	1	
Nombre	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
Descripcion	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
Stock	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
PrecioUND	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
PrecioVent	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
Stockmin	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
FKCodigo_tbl_tipo	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
FKCodigo_tbl_referencia	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		
FKCodigo_tbl_estado	varchar	255		<input type="checkbox"/>	<input type="checkbox"/>		

Additional table properties shown on the right:

- Rows: 5
- Engine: InnoDB
- Auto Increment: 0
- Row Format: Dynamic
- Modified Date: --
- Created Date: 2022-06-19 14:27:05
- Check Time: --
- Index Length: 48,00 KB (49,152)
- Data Length: --

g) Como se puede observar en la imagen se tiene el listado de las tablas o entidades que hacen parte de su base de datos, en la cual se encuentra la información que utilizará para realizar las consultas.



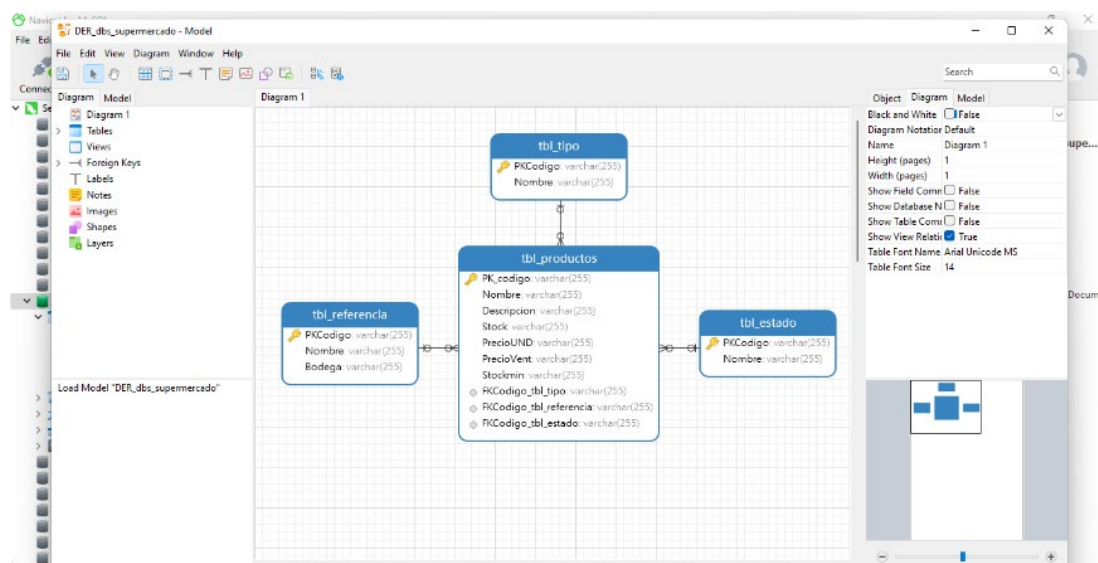
The screenshot shows the 'Objects' tab in Navicat for MySQL, displaying a list of tables in the 'db_supermercado' database. The table structure is as follows:

Name	Auto Increment Value	Modified Date	Data Length	Engine	Rows	Comment
tbl_estado	0		16 KB	InnoDB	4	
tbl_productos	0		16 KB	InnoDB	5	
tbl_referencia	0		16 KB	InnoDB	5	
tbl_tipo	0		16 KB	InnoDB	4	

Additional database properties shown on the right:

- Character set: utf8mb4
- Collation: utf8mb4_general_ci

h) Por último, se puede observar el modelo entidad relación de la base de datos donde se ilustra la distribución de los datos y la estructura de las tablas, adicional a ello se puede ver el tipo de datos que utiliza cada una y la extensión de estos.



Ahora, a través del siguiente video se aprecia las consultas para la recuperación de la información:

Video 2. Consultas



Enlace de reproducción del video

Síntesis del video: consultas

Esta es la consulta más básica que se puede realizar en un motor de bases de datos. Lo que realiza es la selección de todos los datos que se encuentran en la tabla "productos". El asterisco después de la instrucción "SELECT" permite que se puedan visualizar todos los datos que contiene la tabla en la base de datos. Visualizando en la parte inferior, se muestran todos los datos de la tabla.

En esta consulta, visualizará el sutil cambio que ha realizado al incluir la cláusula "WHERE" para colocar una condición. Solo devolverá el producto que cumple con la condición de que su código sea igual a 234. Esto hace que los demás

datos que se encuentran en la tabla no sean tenidos en cuenta, sino que se visualice únicamente la información requerida.

En esta consulta, se puede observar que ha reemplazado el asterisco por campos específicos que se requieren mostrar. Se debe tener en cuenta que todos los campos que se llamen después de la estructura "SELECT" son los que se mostrarán. De esta manera, se muestra al usuario no todos los campos que se encuentran en la tabla, sino aquellos que se requieran hasta ese momento.

En esta consulta, se evidencia que solo se muestran los datos indicados y que, adicional a ellos, se crea una condición con la cláusula "WHERE". Se estipula un rango para el stock o existencias del producto, que se encuentra entre 10 y 15. Por lo tanto, solo mostrará los datos que se encuentren en ese rango de valores determinados en la condición.

En este caso, la consulta se está realizando dentro de la condición "WHERE" con dos rangos de valores. Uno establecido por el stock actual de existencias del producto y el stock mínimo de ese producto. Como resultado, se ve que solo un producto cumple con dicha característica, que es el producto "casco" con un stock de 10 unidades y un stock mínimo de 3 unidades.

Hasta ahora, se ha utilizado una tabla para traer datos, pero ¿qué pasaría si se requiere utilizar más de una tabla? Debido a que los datos requieren ser usados desde otras entidades, a continuación, se dan algunos ejemplos sobre cómo realizar este proceso.

Observe la combinación que se ha realizado de las dos tablas en este caso: la tabla "productos" y la tabla "tipo". En la condición "WHERE", se hace un proceso de igualdad para verificar que el código del tipo del producto coincida con el tipo de producto que se encuentra en la tabla "productos". Esto es a lo que se llama comparación de llaves.

En este caso, la llave primaria se encuentra en la tabla "tipo Y" , por consiguiente, va primero y luego se iguala a la llave foránea que se encuentra en la tabla de "productos". Esta debe compartir el mismo código para poder visualizar dicha información. Esto se puede evidenciar en el diagrama entidad-relación donde ambas llaves se relacionan.

Como se puede observar en la imagen, se han traído varios datos y también se ha utilizado la cláusula o comando "AS", que permite renombrar la columna resultante de la instrucción SQL. Esto se utiliza para colocar un nombre mucho más entendible para el usuario final de la aplicación, ya que en muchas ocasiones el nombre de los campos de las tablas tiene nombres codificados que pueden llegar a confundir al usuario.

Adicionalmente en la cláusula "where" está incluyendo la comparación de los códigos, tanto del tipo de producto como del estado del producto y está especificando que solo traiga los productos que tienen como código 1, los cuales coinciden en la tabla de estado con el estado disponible, mostrando la información contenida en la tabla estado para corroborar dicha información.

Como se observa esta es la información que se encuentra almacenada y se corrobora que el código 1 coincide con el estado del producto disponible. Como se ha

visto hay muchas consultas que se pueden hacer siempre según el objetivo que se tenga previsto, ahora es momento de continuar con su desarrollo temático.

Al final del componente, en el material complementario, hay un enlace donde se podrá explorar mucho más sobre el tema visto.

3. Integración de la capa lógica en las interfaces de usuario

En esta sección se requiere de conocimientos sobre el lenguaje de programación a utilizar y sobre todo de la arquitectura y el diseño de software que se ha estado abordando.

En este paso se lleva a cabo la conexión entre la interfaz del usuario y los procesos que debe realizar el aplicativo, a través de la vista el usuario entrega a la lógica mediante el controlador los datos necesarios para ejecutar las operaciones requeridas en el sistema y este luego procede a hacerlas.

Si en alguno de los casos se requiere que se almacenen los datos en una base de datos entrarían en juego el modelo, que se encargaría de realizar el almacenamiento y posterior recuperación de la información solicitada por el usuario final de la aplicación.

Es importante que tenga en cuenta que este es un proceso de intercambio de estructuras, las cuales tienen como objetivo dividir las funcionalidades del sistema en espacios propios.

En pocas palabras, la vista se encarga de visualizar los formularios, el controlador se encarga de verificar dichos datos y procesarlos, y el modelo de almacenar y recuperar la información.

De acuerdo con la información y lo requerido por el usuario verá más adelante un ejemplo de cómo realizar este proceso paso a paso. Pero inicialmente, debe definir el “framework” que utilizará para la creación de su proyecto, “Flask”, vea ahora algunos conceptos básicos sobre este “framework”.

¿Qué es Flask?

Es un “framework” minimalista escrito en Python que permite crear aplicaciones web rápidamente y con un mínimo número de líneas de código.

Está basado en la especificación WSGI de Werkzeug, el motor de “templates” Jinja2 y tiene una licencia BSD. Dentro de este “framework” se puede encontrar un alto grado de integración con las librerías de Python para realizar el proceso de creación de sitios dinámicos que integren el uso de bases de datos, dentro de las cuales tiene compatibilidad con las principales que se encuentran en el mercado.

Flask es muy compacto, pero no débil; en ese sentido, se dice que es compacto por su practicidad en la construcción de una aplicación y es muy simple su codificación, lo cual lo hace uno de los “framework” más utilizados para la construcción de sitios web, integrando el lenguaje de programación Python.

Esto genera una gran compatibilidad con librerías externas para la generación de archivos de texto, entre otros. Este “framework” también tiene la ventaja de ocupar poco espacio y poder ser utilizado en equipos que no tienen mucha capacidad de procesamiento y ficha técnica, esto también se podría ver como una gran ventaja.

A continuación, se muestra un paso a paso de la creación de una página web en Python utilizando el “framework” de Flask y conectado con una base de datos MySQL, como se ha venido trabajando en este curso.

Nota: antes de comenzar descargue los archivos para la creación de las vistas en el Anexo. Archivos para creación de las vistas.

- a) Primero comience con la creación de las vistas, para lo cual previamente debió descargar los archivos indicados. En este primer paso ingresará a Visual Studio Code y creará una carpeta para implementar todos los documentos, que llamará PROYECTOFLASK.
- b) Ya con su carpeta de “templates” creada con las vistas que se le proyectarán al usuario, cree una carpeta llamada “static” donde se colocarán los arreglos de los archivos de CSS, adicional creará otra carpeta llamada JS donde colocará el archivo de arreglos para la visualización de la información de manera organizada.
- c) Luego de esto, podrá ver de esta manera la visualización de las carpetas del proyecto con los archivos main.css y main.js, estos deberán descargarlos del repositorio de documentos del proyecto dentro de la actividad.

Nota: el documento main.css contiene la configuración de fondo y las líneas que se visualizan en su programa.

Ya luego de estos pasos iniciales puede proceder a la creación de la base de datos y a las configuraciones que llevarán a la ejecución del programa:

Paso 1

Se procede a crear la base de datos para el almacenamiento de los datos que capturará desde sus vistas ya creadas.

Vea la secuencia de código **DDL**, que permite la creación de la base de datos para el almacenamiento de la información que suministra el usuario, tener en cuenta que el campo **pkid** es autonumérico y por lo tanto, no se solicitará al usuario, lo utilizará para

controlar la secuencia de registros y para poder realizar las operaciones de actualización y eliminación de elementos.

Paso 2

Se crea el archivo de código fuente llamado **app.py**, el cual utilizará para almacenar todos los códigos de la aplicación.

Paso 3

Lo primero que realizará es la instalación de **Flask** y luego la librería para el uso de la base de datos **Mysql**, comenzando con el comando **pip install flask**.

Con eso empezará la instalación desde el repositorio del framework de trabajo Flask. Para verificar que quedó bien instalado se verifica la versión con el comando que se visualiza a continuación: **flask --version**

Paso 4

Luego, realizará la instalación de las librerías de Mysql con el comando **pip install flask-mysqldb**.

Este comando permite realizar la instalación de la librería de Mysql para poder llevar a cabo la integración del “framework” Flask con la base de datos.

Paso 5

Llama las librerías necesarias para realizar la ejecución de cada uno de los componentes, se puede observar que se importan las librerías de Flask y las de **Mysqldb**, que serán utilizadas para la manipulación de las diferentes rutas de acceso del programa y al mismo tiempo la conexión con la base de datos.

Paso 6

Del paso anterior se tendrá la configuración de la conexión con el servidor, el tipo de conexión en este caso local (localhost), el nombre del usuario que en este caso es el usuario administrador (root), el password que en este caso no se tiene (en caso de colocarlo se ubicaría en ese espacio) y por último, el nombre de la base de datos (dbs_datos).

La línea de código llamada `app.secret_key` establece una clave secreta para la conexión y acceso a la información, se podrá configurar una palabra secreta (`mysecretkey`) como se ve en la imagen.

Paso 7

Se crea las rutas de acceso a los “templates” y la información que se visualizará en cada uno de ellos, la instrucción `@app.route` indica la ruta que se va a acceder y la función `def Index()`: indica que la información que se visualizará será mostrada en la ruta principal (en el archivo `index.html`), luego se procede a la creación de un cursor que realiza una consulta a la base de datos y busca en la tabla **`tbl_informacion`** y retorna los datos en el “template” con el nombre `index.html`, crea un arreglo llamado “contacts” y le asigna los datos traídos por medio de la consulta `sql`, para luego mostrar los datos en la vista.

Paso 8

Se codifica la ruta de acceso **add_contact**, que permite realizar el registro de un nuevo contacto y se crea una función llamada **def add_contact()**: la cual recibe a través de un método POST las variables enviadas desde la vista enviando los datos de nombre, contacto y correo y los almacena en estas variables, que son enviadas a través de parámetros a la instrucción “insert into”, que permite realizar un nuevo registro en la base de datos y envía un mensaje que el dato fue registrado y retorna a la ruta “Index” que realiza el despliegue de la información que se encuentra almacenada como se hizo en el punto anterior.

Paso 9

Ahora, se puede observar la ruta de acceso de la vista edit_contact.html, que permite enviar los datos que se actualizarán, en la ruta se observa lo siguiente **'/edit/<id>'** esto indica que se está llamando el botón de edición y se envía el id que contenga esa fila de datos para colocarlo en la instrucción “where” para que no actualice todos los registros, sino el registro que se presiona en el botón, luego se ejecuta la instrucción **select * from tbl_informacion where pkid = %s', (id)** que permite que se consulte solo ese dato en particular y se cargue a la vista edit_contact.html.

Paso 10

Observe ahora la ruta que permite realizar la actualización de los datos, se procede a recibir los datos en las variables de nombres, contacto y correo y, luego se ejecuta la instrucción “update”, la cual permite realizar el cambio del dato seleccionado, en ese momento se envía el mensaje correspondiente a la actualización de los datos y se procede a llamar la ruta “Index”, que utiliza la plantilla del archivo

index.html trayendo los datos después de realizar el cambio de los datos en la base de datos.

Paso 11

Por último, observe el proceso de eliminación de un dato, se crea la ruta que recibe el id de la persona que se desea sacar de la base de datos y luego se procede a ejecutar la instrucción “delete”, la cual permite eliminar un registro de la base de datos, pero en la cláusula “where” se le envía el id de la persona que se esté seleccionando para que no elimine todos los registros sino solo el que se está seleccionando. También se coloca el **app.run** donde se selecciona el puerto 3000 y se ejecuta la aplicación, la cual utilizará el navegador predeterminado para realizar el despliegue de las vistas y la integración del “backend” creado y se procede a ejecutar la aplicación para ver su operación.

Como se dijo con antelación, para la creación de una página web en Python utilizando el “framework” de Flask y conectado a una base de datos MySQL se tuvo que dar algunos pasos previos que llevan al momento de la ejecución del programa, como se verá en el siguiente video:

Video 3. Ejecución y prueba del programa



[Enlace de reproducción del video](#)

Síntesis del video: ejecución y prueba del programa

Como se puede apreciar en la imagen, al presionar el botón de ejecución se indica que se está ejecutando la aplicación. Proceda a abrir una pestaña nueva en su navegador y coloque el comando "localhost" indicado y el puerto de ejecución, que para este caso es 3000. La instrucción quedaría de la siguiente manera: "localhost:3000". Como puede visualizar, mostrará la aplicación que acaba de subir al servidor de Python.

Ahora proceda a ingresar los datos para verificar el funcionamiento de la aplicación. Ha ingresado información en las cajas de texto y ahora procede a dar clic

en el botón "registrar" para que estos datos se muestren en su base de datos. Puede observar cómo el sistema informa que se registraron los datos en la base de datos.

Ahora proceda a verificar si los datos se registraron correctamente en la base de datos. Como se puede visualizar, los datos fueron registrados en la base de datos de manera correcta.

Proceda a realizar una actualización de los datos dando clic en la opción "editar" de su vista. Al hacer eso, se desplegará la siguiente vista. Una vez ubicados en la vista de actualización, proceda a realizar un cambio, por ejemplo, cambiar el correo electrónico de la persona registrada, para verificar que se realiza esta función sin ningún problema. Puede ver que se han cambiado los datos de la persona registrada, en este caso, el correo electrónico. Al ir a la base de datos, verifique que se realizó el cambio de manera correcta.

Ahora proceda a realizar la eliminación de esta persona de los registros de nuestra base de datos. Este proceso se realiza presionando el botón "borrar", y esto elimina el dato desde la vista. Como se visualiza, al momento de realizar la consulta de la tabla "tbl información", se da cuenta de que se encuentra vacía y que el único dato que tenía registrado ha desaparecido. Por lo tanto, la función de eliminar registros también funciona.

Con esto, ha llegado al final de la ejecución de su programa web y su respectiva prueba de funcionalidad. Para garantizar que todo estuviese funcionando correctamente. Se espera que estas líneas de código y de conocimiento sean de mucha utilidad para su proceso de aprendizaje. Ahora las puede poner en práctica.

4. Servicios web

Son un conjunto de tecnologías que permiten el acceso de la información de diferentes fuentes y aplicaciones.

Su alimentación principal es la interacción con los diferentes subsistemas que se encuentran en la red, ahora bien, los servicios web proporcionan gran parte de la dinámica que se genera a través de Internet y que proporciona el flujo de información que se presenta todos los días, con ejemplos como:

- Los servicios de correo electrónico.
- Servicios de alojamiento de información.
- Servicios de redes sociales, entre muchos otros.

Todos estos se encuentran soportados por servicios web, se entiende que dependiendo de las necesidades que se tengan se pueden contratar según las capacidades técnicas requeridas, siendo su costo razonable de acuerdo con lo que ofrecen y que se convierte en la ventaja competitiva para las organizaciones.

Los servicios web varían desde el despliegue de las aplicaciones hasta complejas infraestructuras que permiten a grandes organizaciones soportar todos los servicios que estas requieren para funcionar.

Ahora bien, es importante tener en cuenta que los servicios web se utilizan para realizar el intercambio de información que puede requerir una aplicación. Para esto utiliza unos protocolos específicos para realizar estas operaciones teniendo como base el protocolo de transferencia http.

Se sabe que las comunicaciones son un punto importante hoy en día ya que todo está conectado con diversas fuentes de información y aplicaciones de propósito específico y general, es decir, la comunicación tal y como la ve hoy en día no se visualizaba de esa manera hace muchos años, la mayoría de las aplicaciones solucionaban problemas de manera local y solo algunos servicios tales como correo electrónico y consultas eran los que requerían la conectividad a Internet.

Hoy en día de acuerdo con la demanda de los procesos del mercado actual prácticamente una aplicación que no tenga la posibilidad de utilizar servicios de comunicación, compartir o intercambiar con otras aplicaciones está propensa a desaparecer. Esto ha generado toda una dinámica interesante en todos los sectores de la industria, ya que existe un acto llamado seguridad, que puede estar presente en cada uno de estos procesos.

¿Qué es un web service?

El concepto hace referencia al sistema de comunicación, los protocolos y los estándares entre dos dispositivos electrónicos conectados a la misma red para intercambiar datos entre sistemas o aplicaciones.

Es un software creado para la comunicación entre máquina y máquina y, la interoperabilidad entre ellas.

Uno de los elementos clave de este sistema es el XML (“Extensible Markup Language” o Lenguaje de Marcado Extensible en español). Es la manera en que están codificados o programados los datos para que se puedan procesar por diferentes sistemas, es decir, es un lenguaje estándar que permite que se dé la comunicación aun cuando las aplicaciones o sistemas usen lenguajes diferentes de programación.

Tipos y beneficios del “Web Service”

Existen dos tipos de “Web Services” que se diferencian por sus estándares (el SOAP y el RESTful) y varios beneficios del uso de estos servicios en diferentes sectores productivos, pero enfocarse en el sector industrial. Se profundiza en todo esto a continuación:

SOAP (Simple Object Access Protocol)

Se trata de un protocolo basado en el lenguaje XML, estándar de comunicación que permite enviar y recibir mensajes del servicio web entre varios sistemas o aplicaciones. Por lo tanto, se puede decir que SOAP define cómo es la comunicación y utiliza el protocolo HTTP para ello. El HTTP o protocolo de transferencia de hipertexto se refiere al protocolo que permite realizar una petición de datos y recursos. Es de los más utilizados en Internet por buscadores como Google y servidores.

RESTful

Son las aplicaciones de “Web Service” procedente de la arquitectura de software REST (“Representational State Transfer”) y, a diferencia del SOAP, este no se encuentra estructurado bajo estándares definidos y siendo este más ligero. Además, su flexibilidad permite que no funcione únicamente con lenguaje XML, sino también puede funcionar bajo JSON (“JavaScript Object Notation”) u otros lenguajes de programación.

Beneficios

Uno de los más relevantes en el sector industrial es poder acceder a programas o aplicaciones sin necesidad de tenerlos instalados en su ordenador, porque están

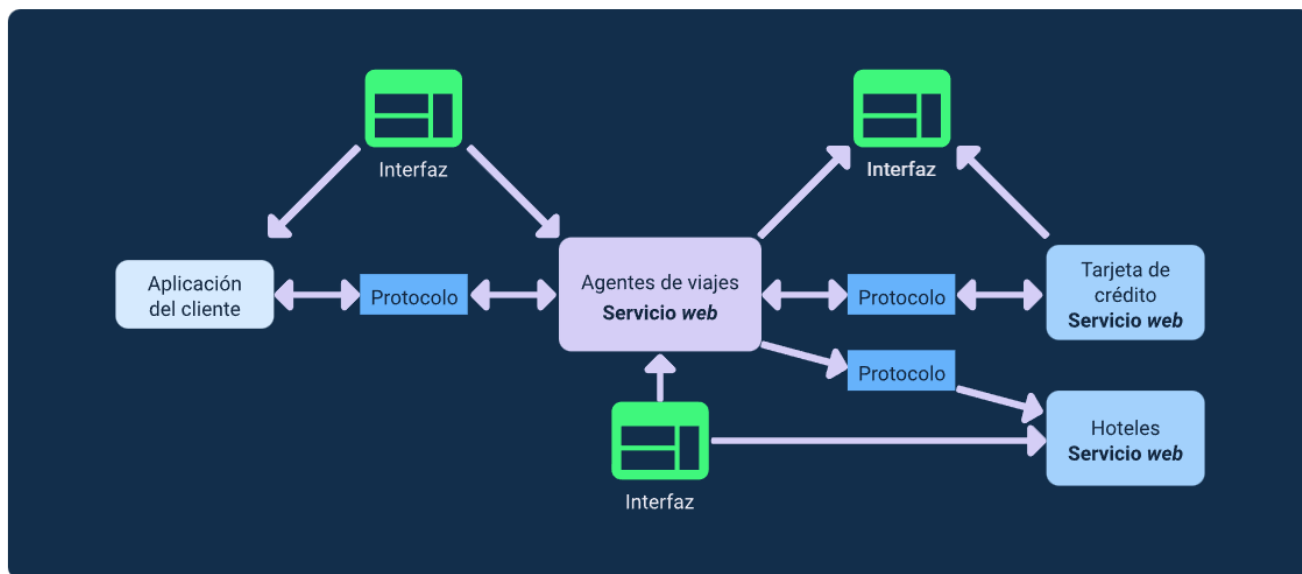
alojados en el “Web Service”. De esta manera, siempre accederá al mismo contenido sin que haya pérdidas de la información. No obstante, un “Web Service” sirve para poder acceder a cualquier tipo de archivos y contenidos que tenga alojados en un alojamiento web o hosting.

Es esencial contar con un “Web Service” de calidad si en su empresa tiene automatizados procesos en red con PLC’s, sistemas de supervisión y control de los procesos productivos o sistemas SCADA. Estos deben de contar con una buena conexión para asegurar un correcto funcionamiento.

La interacción que los servicios web ofrecen es la conexión mediante protocolos e interfaces, los diferentes servicios e información que se puede intercambiar entre estas mismas aplicaciones. Esto para obtener un beneficio propio, es decir, cómo se puede apreciar en la figura que ilustra el siguiente ejemplo:

Un cliente ingresa a través de una interfaz gráfica a la página de un sitio de un agente de viajes, tiene un servicio web instalado que proporciona información y datos al usuario sobre el proceso que este implica, pero a su vez está conectado con otra interfaz que permite, por ejemplo, si el cliente requiere utilizar los servicios del banco para pagar tener conexión con dichos procesos y, si el cliente requiere solicitar alojamiento en algún hotel a través de la página web lo podrá hacer. Ya que está le ofrecerá conexión con este servicio, que, de hecho, podrá utilizar el servicio de tarjeta de crédito para cancelar sus reservaciones.

Figura 4. Servicios web



Esto se puede observar en la páginas de “e-commerce”, en las cuales se ofrecen los servicios de pagos en línea, entre otros. Este es un claro ejemplo del uso de servicios web, ya que la aplicación de “e-commerce” está instalada en un servidor y está siendo administrada por una empresa o particular; pero en cierto modo, tiene la posibilidad de realizar el pago en línea y es allí donde el proceso toma fuerza, ya que al momento que se desee cancelar y utilizar medios electrónicos automáticamente se redireccionará al cliente a la página del banco para realizar el proceso.

Una vez este culmine retornará a la página del sitio de “e-commerce” para finalizar el proceso de compra.

5. Manejo y control de versiones

En este espacio se realizará el despliegue de la aplicación construida en Python, cabe aclarar que existen muchos hosting y sitios donde se puede llevar a cabo el despliegue de su sitio; pero va a utilizar un servicio gratuito llamado pythonanywhere.com el cual proporcionará todos los recursos para colocar en marcha su sitio sin ninguna dificultad.

A continuación, se mostrarán los pasos previos para lograr dicho objetivo:

Paso 1

Ingresa a [pythonanywhere](https://pythonanywhere.com) donde realizará el registro o login para poder utilizar los servicios. ¡Si es la primera vez dé clic en Sign up here! allí seleccione la opción Create a Beginner account y proceda a colocar sus datos para el registro. ¡Ya está listo para iniciar la creación de su sitio!

Paso 2

Le mostrará luego el Dashboard en [pythonanywhere](https://pythonanywhere.com) donde va a crear su espacio de sitio, ingresando en la barra superior a la opción Web. Puede observar que le aparece una opción llamada Add a new web app. Dé clic en Next y seleccione Flask, que es el framework seleccionado para la construcción de su sitio.

Paso 3

Luego, en la siguiente ventana va a seleccionar la versión de Python, en este caso utilizará la versión más actual, aunque en algunas ocasiones se debe realizar de acuerdo con la versión trabajada.

Paso 4

Por último, le mostrará el nombre del archivo principal, que va a dejar tal cual, ya que luego lo que hará será cambiar el código que este contiene y después dé clic en siguiente. Con esto verá que su sitio ha sido creado con éxito.

Luego de creado el sitio a través del servicio (pythonanywhere) siguiendo los pasos indicados, se continuará con su despliegue.

Ahora se mostrará el paso a paso en el siguiente video. Hay que tener en cuenta que si bien el ejercicio se hace con base en el ejemplo planteado servirá de guía para que lo pueda hacer después.

Video 4. Despliegue de aplicación en la web



Enlace de reproducción del video

Síntesis del video: despliegue de aplicación en la web

Con el sitio creado, comience con su despliegue. Para ello, vaya a la parte superior y seleccione la opción "Files". Esto lo llevará a esta pantalla donde va a subir los archivos de su programa.

En el menú de la parte inferior izquierda, seleccione la carpeta que dice "mysite" donde se encuentra el sitio creado. Cree las carpetas "templates", "css" y "javascript" como nuevos directorios, como se observa. Para esto, coloque el nombre del directorio y luego haga clic en "New directory" para crearlo. Luego, suba los archivos correspondientes a los templates, css y javascript dentro del directorio correspondiente a través del botón "Upload".

Ahora, concéntrese en su archivo "app.py", que es el archivo que contiene el código fuente o backend de su aplicación. Para subir los cambios del código que ha construido en su máquina local, antes de eso, llegará al archivo que ve actualmente, en el que puede sustituir la información que ha ingresado. En este caso, reemplácela por la que tiene su archivo "app.py". Copie y pegue el código y proceda a quitar el que se encuentra al final, ya que este no necesita ser ejecutado manualmente, ya que el servidor de PythonAnywhere se encarga de realizar este proceso.

Reemplace la información de conexión por la que está en su servidor y haga clic en la opción "Save" para guardar los cambios. Luego, haga clic en "Back" para ir a la opción "Database" y ver la información que requiere.

Coloque un nombre para su base de datos, que en este caso terminará en "dbs_datos". Con esto, se ha creado la base de datos. El "Host" es el que dice "Database Host address", el nombre de usuario es "username" y la contraseña la configura en la parte inferior. En esta parte, puede configurar la contraseña para su servidor de la base de datos y la usará en la configuración de conexión.

Una vez configurada, proceda a realizar cambios en su archivo de código para poder conectarse con su servidor y que los datos queden almacenados en el mismo servidor. Reemplace los datos de conexión por los que entrega el hosting de PythonAnywhere. En el caso del "password", debe colocar el que configuró. Luego, presione "Save" para que se guarden los cambios.

Puede observar una ventana de consola donde va a crear su tabla. Primero, utilice el comando "show tables" para verificar si existe una tabla creada. Si devuelve "Empty set", está indicando que no tiene ninguna tabla creada. Ahora, proceda a colocar el código de creación de su tabla. Para ello, copie en el portapapeles el código y péguelo en su ventana de consola. Al pegar la información desde su sistema de bases de datos local a su consola de PythonAnywhere, ejecute las líneas de comando. Indica que se ha creado y proceda a verificar la creación de la tabla, que aparecerá con el nombre de "tbl_informacion".

Retome a su "dashboard" y luego seleccione, dentro de la opción "New Console", la opción "Bash", ya que debe descargar el conector y la librería de conexión. Coloque los siguientes comandos: primero, la instalación de "flask-MySQLdb" y espere a que se instale con "pip install flask-MySQLdb".

El video expone, entonces, la puesta a punto del sitio a través de un servidor, logrando su visualización y la colocación de un dato para verificar que todo funcione de manera correcta.

Es importante tener presente que se deben seguir los pasos de manera continua para lograr el objetivo de forma idónea y que si bien, como se indicó, este es un ejemplo a través del servidor gratuito propuesto, se podrá seguir esta cadena lógica de pasos para poner a punto su sitio en este u otro servidor.

Se ha llegado al final del componente formativo, esperando que estos conocimientos le aporten a su vida personal y profesional.

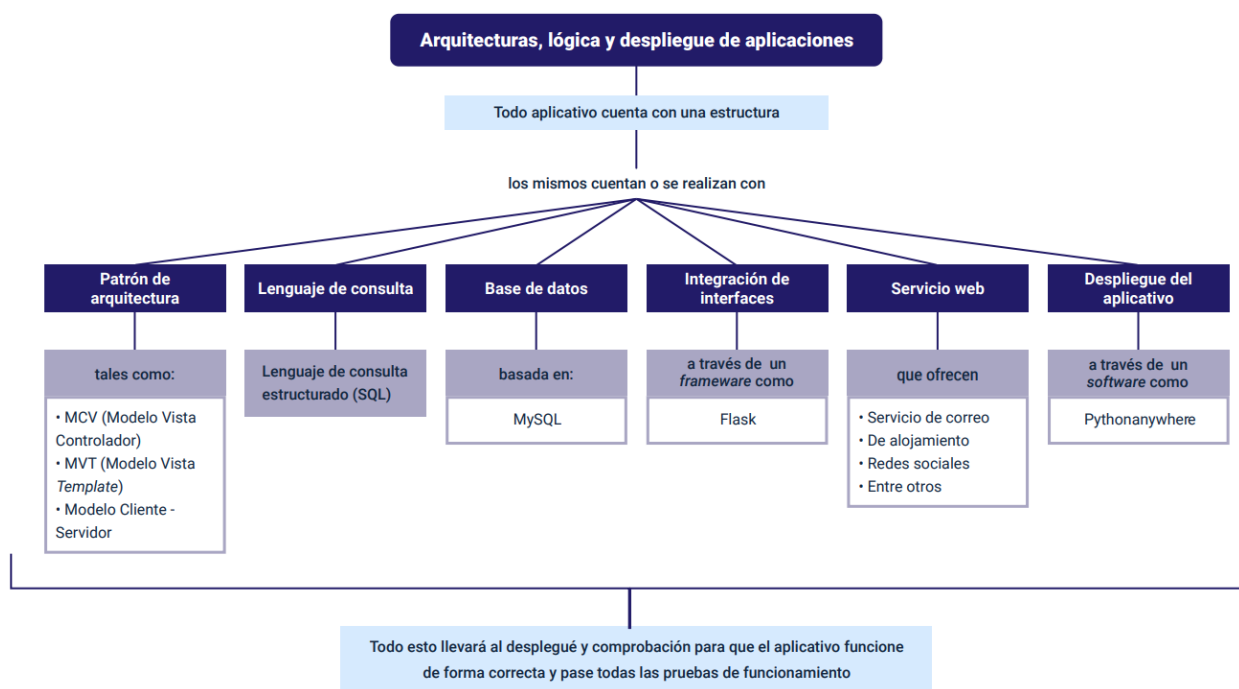
¡Muchos éxitos!

Síntesis

Como se ha visto hay diferentes caminos en la industria del software que ayudarán en su construcción, muchos orientados en la funcionalidad de este y otros enmarcados en el ámbito y la capacidad de poder adaptarse a las necesidades cambiantes de los usuarios de hoy en día.

Este material formativo ha llevado a desarrollar los conocimientos sobre el manejo de arquitecturas para la construcción de aplicaciones y lógica para el funcionamiento de estas, que resultan en la creación de algún aplicativo.

A continuación, podrá ver la síntesis de lo tratado en este componente:



Material complementario

Tema	Referencia APA	Tipo de material	Enlace del Recurso
Lenguaje de consulta	1Keydata.com. (2022). Tutorial de SQL: curso de SQL.	Tutorial web	https://www.1keydata.com/es/sql/
Servicios web	Rodríguez de Sepúlveda Maillo, D. (2015). Administración de servicios web. Ra-Ma.	Libro	https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_elibroELB106473
Servicios web	Reinosa, E., Maldonado, C., Muñoz, R., Damiano, L. y Abrutsky, M. (2012). Bases de datos. Alfaomega Grupo Editor	Libro	https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_aleph000061572

Glosario

Bases de datos: es una aplicación que permite el almacenamiento y recuperación de información que se encuentra debidamente organizada.

CSS: es un lenguaje que permite la creación de hojas de estilo para ubicar y mejorar el diseño de una página web.

Formulario: es un conjunto de controles que se agrupan para recolectar información a un posible usuario dentro de una página web.

“Front-end”: es el nombre técnico que recibe la parte del sistema que se encarga de la interacción con el usuario final.

HTML: lenguaje de etiquetado que permite estructurar una página web.

MVC Modelo Vista Controlador: es una estructura de trabajo que permite independizar cada una de las partes de un programa, en este caso un sitio web o aplicativo web.

Servidor web: es una máquina dedicada a colocar los archivos de ejecución de una página web y hacerla visible para muchos usuarios.

“Wireframe”: es un boceto previo que se utiliza para negociar con un posible cliente la ubicación y despliegue de componentes en una aplicación web o móvil.

Referencias bibliográficas

MDN Web Docs. (s.f.) Proyecto constrúyelo mejor. Mozilla recursos para desarrolladores.

Pavón, J. (2008). Estructura de las aplicaciones orientadas a objetos. El patrón Modelo-Vista-Controlador (MVC). Universidad Complutense de Madrid.
<https://www.fdi.ucm.es/profesor/jpavon/poo/2.14.mvc.pdf>

Reinosa, E., Maldonado, C., Muñoz, R., Damiano, L. y Abrutsky, M. (2012). Bases de datos. Alfaomega Grupo Editor. https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_aleph000061572

Rodríguez de Sepúlveda Maillo, D. (2015). Administración de servicios web. Rama. https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_elibroELB106473

Créditos

Nombre	Cargo	Regional y Centro de Formación
María Camila García Santamaria	Líder del equipo	Dirección General
Rafael Neftalí Lizcano Reyes	Asesor metodológico y pedagógico	Centro Industrial del Diseño y la Manufactura - Regional Santander
Dulfran Antonio Montaña Montaña	Experto temático	Centro De Diseño Y Metrología - Regional Distrito Capital
Zvi Daniel Grosman Landáez	Diseñadora instruccional	Centro de Gestión Industrial - Regional Distrito Capital
Carolina Coca Salaza	Asesora metodológica	Centro de Diseño y Metrología - Regional Distrito Capital
Julia Isabel Roberto	Corrector de estilo	Centro de Diseño y Metrología - Regional Distrito Capital
Juan Daniel Polanco Muñoz	Diseñador de Contenidos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Emilsen Alfonso Bautista	Desarrollador Full-Stack	Centro Industrial del Diseño y la Manufactura - Regional Santander
Zuleidy María Ruiz Torres	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Wilson Andrés Arenales Cáceres	Storyboard e ilustración	Centro Industrial del Diseño y la Manufactura - Regional Santander
Mary Jeans Palacio Camacho	Animador y Productor Multimedia	Centro Industrial del Diseño y la Manufactura - Regional Santander
Carlos Eduardo Garavito Parada	Animador y Productor Multimedia	Centro Industrial del Diseño y la Manufactura - Regional Santander

Luis Gabriel Urueta Álvarez	Validador de Recursos Educativos Digitales	Centro Industrial del Diseño y la Manufactura - Regional Santander
Daniel Ricardo Mutis Gómez	Evaluable para Contenidos Inclusivos y Accesibles	Centro Industrial del Diseño y la Manufactura - Regional Santander