**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

|  |  |
| --- | --- |
| PROGRAMA DE FORMACIÓN | Desarrollo de aplicaciones web con *Python* |

|  |  |  |  |
| --- | --- | --- | --- |
| COMPETENCIA | 220501123 - Construir sitios web según técnicas de interoperabilidad y protocolos técnicos. | RESULTADOS DE APRENDIZAJE | 220501123-04 Desarrollar la aplicación web con Python acorde con los requerimientos del cliente |

|  |  |
| --- | --- |
| NÚMERO DEL COMPONENTE FORMATIVO | 004 |
| NOMBRE DEL COMPONENTE FORMATIVO | 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, la lógica que se aplica para el funcionamiento de esta y el despliegue de la aplicación en un servidor de producción. |
| PALABRAS CLAVE | Arquitectura, lógica de negocio, despliegue, aplicaciones, producción. |

|  |  |
| --- | --- |
| ÁREA OCUPACIONAL | Ventas y Servicios |
| IDIOMA | Español |

**A. TABLA DE CONTENIDOS**

**Introducción**

**1. ¿Qué es un patrón de arquitectura?**

**2. Lenguaje de consulta**

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

**4. Servicios web**

**5. Despliegue de aplicaciones en la Web**

**Síntesis**

**B. INTRODUCCIÓN**

En el mercado actual de la industria del *software* existen diferentes caminos y tecnologías a aplicar en la construcción de *software*, muchas de ellas se orientan a la funcionalidad, 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:

**DI\_CF004\_Video\_Introducción**

**C. DESARROLLO DE CONTENIDOS**

**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; de allí que el escalamiento y la versatilidad de modificaciones sea mucho más fácil para quien está realizando dichos ajustes.



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 se encajen 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. Hay que recordar que los patrones 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*

Interfaz de usuario gráfica, Aplicación, PowerPoint

Descripción generada automáticamente

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

**DI\_CF004\_Acordeón\_1\_Partes del modelo MVC**

Al implementar dicho patrón en los proyectos puede verse beneficiado en los siguientes puntos:

* Organización modular. Se podrá dividir la lógica del programa haciendo la aplicación más escalable.
* 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 se 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.

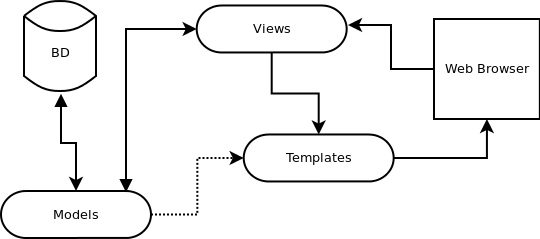
A diferencia del MVC en donde las vistas pueden variar de acuerdo con lo solicitado por el usuario, en este modelo se tiene una plantilla específica para mostrar la información.

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 se ilustran 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, facilitando 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.

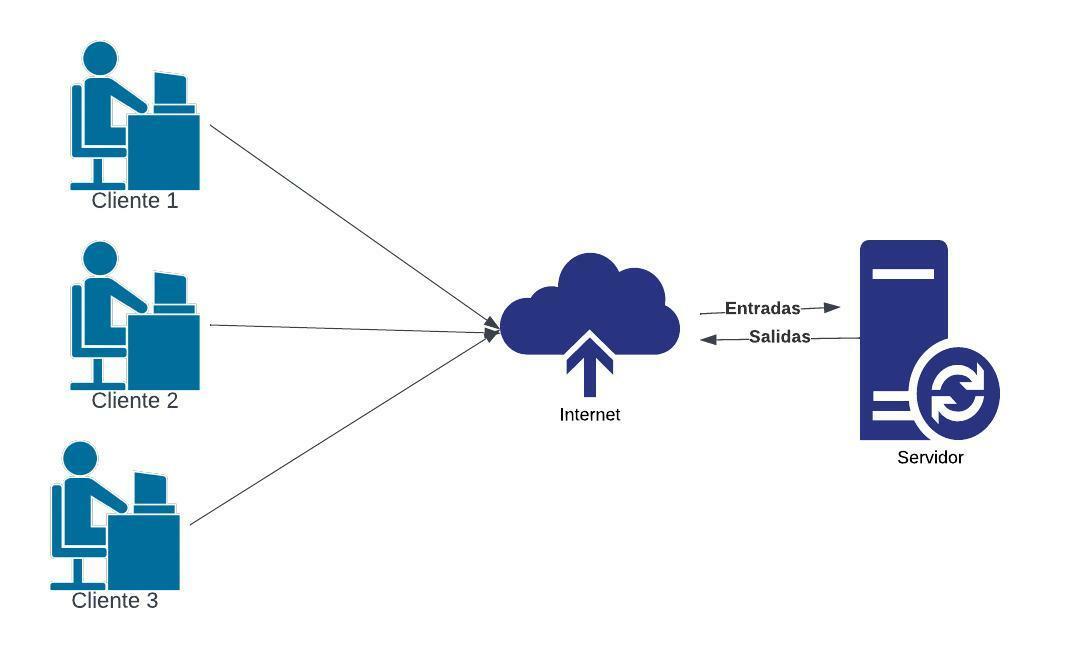
**Modelo Cliente - Servidor**

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. Ver figura.

**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 l1os 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 y 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. Existen algunas variaciones que se pueden dar dependiendo del fabricante del motor, 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, dando información precisa y confiable. A continuación se presentan varios ejemplos sobre cómo utilizar este lenguaje de consultas.

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

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

DI\_CF004\_Slider\_2\_Creando la base de datos

Luego de crear la base de datos, se invita a hacer todos estos pasos para que tenga una mejor comprensión.

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

DI\_CF004\_Video\_2\_Consultas

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 entraría en juego el modelo, que se encargaría de realizar el almacenamiento y la posterior recuperación de la información solicitada por el usuario final de la aplicación.

Es importante que tener 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 se apreciará más adelante un ejemplo de cómo realizar este proceso paso a paso. Pero inicialmente, se debe definir el *framework* que se utilizará para la creación del proyecto, Flask. Ahora se exponen 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 se requiere la descarga de los archivos para la creación de las vistas en el Anexo. Archivos para creación de las vistas.

Se debe comenzar con los pasos previos a la creación de la base de datos:

DI\_CF004\_Pasos simple\_3\_Pasos previos

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

DI\_CF004\_Pasos horizontales\_3\_Creando la base de datos

Como se dijo con antelación, para lacreación 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:

DI\_CF004\_Video\_3\_Ejecución y prueba del programa

**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 puede contratar según las capacidades técnicas requeridas, siendo su costo razonable de acuerdo con lo que ofrecen, convirtiéndose, además, 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 se 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, los cuales tienen varios beneficios de uso en diferentes sectores productivos. Se profundiza en todo esto a continuación:

DI\_CF004\_Pestañas\_4\_tipos y beneficios

La interacción que los servicios web ofrecen es la conexión mediante protocolos e interfaces y el intercambio de diferentes servicios e información que se puede dar entre estas mismas aplicaciones. Esto para obtener un beneficio propio, tal y como se puede apreciar en el siguiente ejemplo:

Un cliente ingresa a través de una interfaz gráfica a la página de un sitio de un agente de viajes. Este 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 solicitar alojamiento en algún hotel a través de la página web lo podrá hacer o si requiere utilizar los servicios del banco para pagar su reservación, también, ya que tiene la conexión con dichos procesos y servicios.

**Figura 4**

*Servicios web*

Interfaz de usuario gráfica, Diagrama, Aplicación

Descripción generada automáticamente

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 entra 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 culminar el proceso de compra.

**5. Despliegue de aplicaciones en la web**

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

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

DI\_CF004\_Pasos horizontales\_5\_Primeros pasos en el despliegue

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 se pueda hacer después.

DI\_CF004\_Video\_5\_Despliegue de aplicación en la web

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 estos conocimientos aporten a su vida personal y profesional.

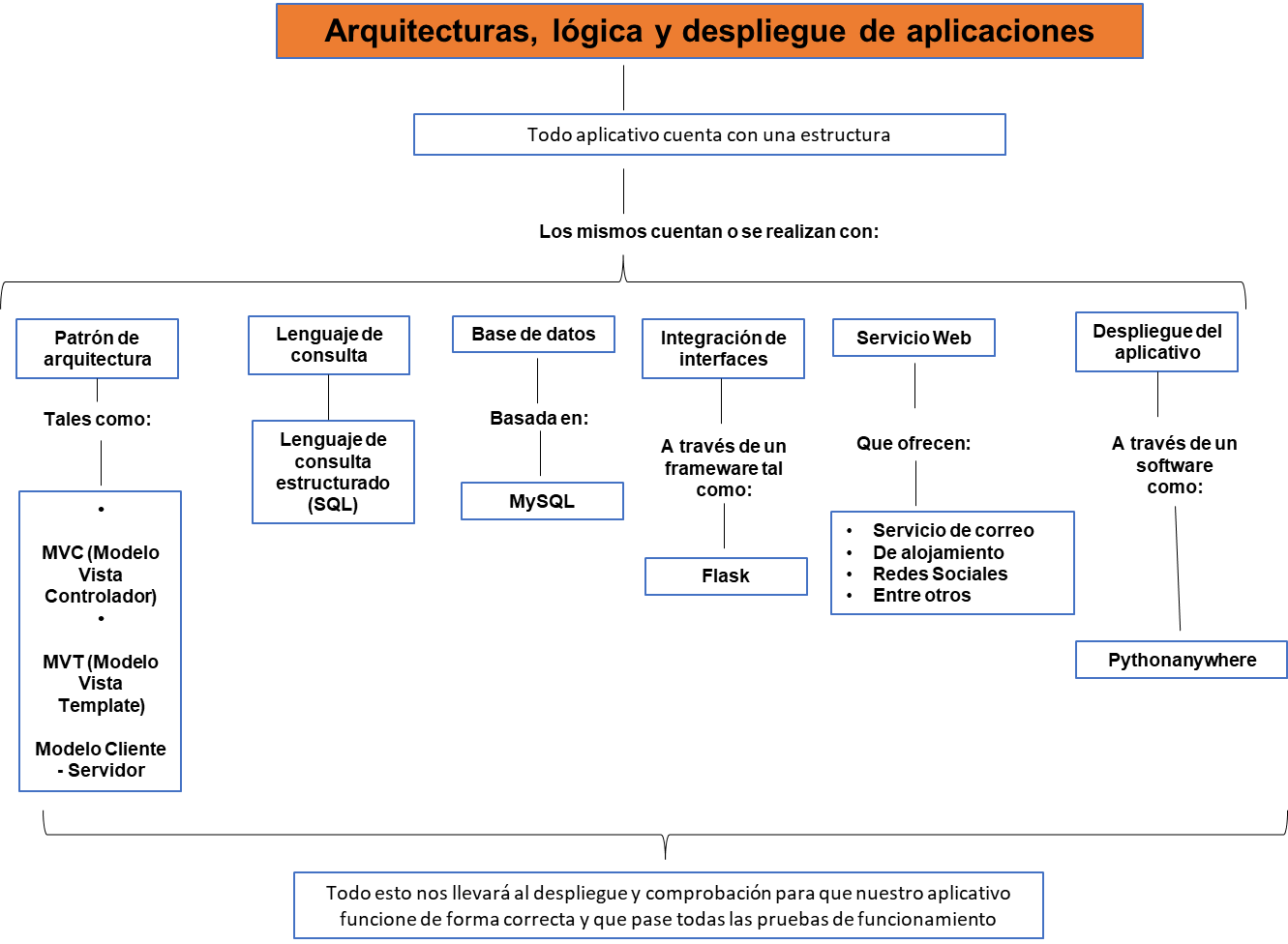
¡Muchos éxitos!

**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 se podrá ver la síntesis de lo tratado en este componente.



1. **ACTIVIDAD DIDÁCTICA**

|  |  |
| --- | --- |
| DESCRIPCIÓN DE LA ACTIVIDAD DIDÁCTICA | |
| Nombre de la actividad | Poniendo en marcha mi sitio |
| Objetivo de la actividad | Validar los conceptos aprendidos sobre cómo realizar la arquitectura, lógica y despliegue para poner en marcha un sitio web a través del lenguaje *Python.* |
| Tipo de actividad sugerida |  |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | CF004\_Actividad\_didactica\_cuestionario |

1. **MATERIAL COMPLEMENTARIO**

|  |  |  |  |
| --- | --- | --- | --- |
| Tema | Referencia APA del material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del recurso o  archivo del documento o material |
| Lenguaje de consulta | 1Keydata.com. (2022). *Tutorial de SQL: curso de SQL.* 1Keydata.com. <https://www.1keydata.com/es/sql/> | Tutorial web | <https://www.1keydata.com/es/sql/> |
| Servicios web | Rodríguez de Sepúlveda Maillo, D. (2015). *Administración de servicios web.* Ra-Ma*.* <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_elibroELB106473> | 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. <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_aleph000061572> | Libro | <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_aleph000061572> |

1. **GLOSARIO**

|  |  |
| --- | --- |
| TÉRMINO | SIGNIFICADO |
| 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. |

1. **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****.*** Ra-Ma***.*** <https://sena-primo.hosted.exlibrisgroup.com/permalink/f/1j5choe/sena_elibroELB106473>

1. **CONTROL DEL DOCUMENTO**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha |
| Autor (es) | Dulfrán Antonio Montaño Montaño | Experto temático | Regional Distrito Capital - Centro de Diseño y Metrología | Agosto de 2022 |
| Zvi Daniel Grosman Landáez | Diseñador instruccional | Regional Distrito Capital – Centro de Gestión Industrial | Agosto de 2022 |
| Carolina Coca Salazar | Asesora metodológica | Regional Distrito Capital - Centro de Diseño y Metrología | Agosto de 2022 |
| Rafael Neftalí Lizcano Reyes | Responsable Equipo desarrollo curricular | Regional Santander - Centro Industrial del Diseño y la Manufactura | Agosto de 2022 |
| Julia Isabel Roberto | Correctora de estilo | Regional Distrito Capital - Centro de Diseño y Metrología | Septiembre de 2022 |

1. **CONTROL DE CAMBIOS**

(Diligenciar únicamente si realiza ajustes a la Unidad Temática)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| Autor (es) |  |  |  |  |  |