



UTPL
La Universidad Católica de Loja

Modalidad Abierta y a Distancia

Itinerario 2: Desarrollo Basado en Plataformas Web

Guía didáctica

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Facultad de Ingenierías y Arquitectura

Departamento de Ciencias de la Computación y Electrónica

Itinerario 2: Desarrollo Basado en Plataformas Web

Guía didáctica

Carrera	PAO Nivel
▪ Tecnologías de la información	VI

Autor:

Elizalde Solano René Rolando



Asesoría virtual
www.utpl.edu.ec

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Universidad Técnica Particular de Loja

Itinerario 2: Desarrollo Basado en Plataformas Web

Guía didáctica

Elizalde Solano René Rolando

Diagramación y diseño digital:

Ediloja Cía. Ltda.

Telefax: 593-7-2611418.

San Cayetano Alto s/n.

www.ediloja.com.ec

edilojacialtda@ediloja.com.ec

Loja-Ecuador

ISBN digital - 978-9942-25-930-1



Reconocimiento-NoComercial-CompartirIgual
4.0 Internacional (CC BY-NC-SA 4.0)

Usted acepta y acuerda estar obligado por los términos y condiciones de esta Licencia, por lo que, si existe el incumplimiento de algunas de estas condiciones, no se autoriza el uso de ningún contenido.

Los contenidos de este trabajo están sujetos a una licencia internacional Creative Commons **Reconocimiento-NoComercial-CompartirIgual 4.0 (CC BY-NC-SA 4.0)**. Usted es libre de **Compartir** – copiar y redistribuir el material en cualquier medio o formato. **Adaptar** – remezclar, transformar y construir a partir del material citando la fuente, bajo los siguientes términos: **Reconocimiento**– debe dar crédito de manera adecuada, brindar un enlace a la licencia, e indicar si se han realizado cambios. Puede hacerlo en cualquier forma razonable, pero no de forma tal que sugiera que usted o su uso tienen el apoyo de la licenciatario. **No Comercial**-no puede hacer uso del material con propósitos comerciales. **Compartir igual**-Si remezcla, transforma o crea a partir del material, debe distribuir su contribución bajo la misma licencia del original. No puede aplicar términos legales ni medidas tecnológicas que restrinjan legalmente a otras a hacer cualquier uso permitido por la licencia. <https://creativecommons.org/licenses/by-nc-sa/4.0/>

Índice

1. Datos de información.....	8
1.1. Presentación de la asignatura	8
1.2. Competencias genéricas de la UTPL.....	8
1.3. Competencias específicas de la carrera.....	8
1.4. Problemática que aborda la asignatura	9
2. Metodología de aprendizaje.....	10
3. Orientaciones didácticas por resultados de aprendizaje	11
Primer bimestre.....	11
Resultado de aprendizaje 1	11
Contenidos, recursos y actividades de aprendizaje.....	11
Semana 1	11
Unidad 1. Conceptos generales de desarrollo de plataformas web	12
1.1. Características deseables de un sitio web	12
1.2. Tecnologías para creación de sitios web	14
Semana 2	20
1.3. Servicios en la nube (cloud)	20
1.4. Tipos de servicio en la nube	21
Actividad de aprendizaje recomendada.....	23
Autoevaluación 1.....	24
Resultado de aprendizaje 2	28
Contenidos, recursos y actividades de aprendizaje.....	28

Semana 3	28
Unidad 2. Programación del lado del cliente	28
2.1. Introducción	29
2.2. Ejemplos de HTML y CSS	30
2.3. Manejo de JavaScript	38
Semana 4	49
2.4. Base de Datos NoSql	49
Actividad de aprendizaje recomendada.....	71
Autoevaluación 2.....	72
Semana 5	79
Unidad 3. Acceso a base de datos relacionales mediante Object Relational Mapper (ORM)	79
3.1. Introducción	79
3.2. Manejo de datos con ORM SqlAlchemy.....	81
3.3. Consulta de datos con ORM SqlAlchemy	88
Semana 6	97
3.4. Eliminación de datos con ORM SqlAlchemy.....	97
3.5. Relaciones de entidades con ORM SqlAlchemy.....	102
Actividad de aprendizaje recomendada.....	115
Autoevaluación 3.....	117
Actividades finales del bimestre	129
Semana 7	129
Actividades de aprendizaje recomendadas.....	129

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Índice

Semana 8	130
Segundo bimestre	132
Resultado de aprendizaje 3	132
Contenidos, recursos y actividades de aprendizaje.....	132
 Semana 9	132
 Unidad 4. Programación en el servidor web	132
4.1. Desarrollo de aplicaciones con lenguaje PHP.....	133
 Semana 10	141
4.2. Acceso a base de datos relacional desde una aplicación en PHP	141
Actividad de aprendizaje recomendada.....	147
Autoevaluación 4.....	149
 Semana 11	166
 Unidad 5. Uso de frameworks de ambiente web	166
5.1. Modelo-vista-controlador base de los frameworks	166
5.2. Clasificación de frameworks de ambiente web según los lenguajes de programación.....	167
5.3. Desarrollo de aplicaciones mediante el framework Django.....	168
 Semana 12	179
Actividad de aprendizaje recomendada	187
Autoevaluación 5.....	188

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Índice

Semana 13	191
Semana 14	207
Actividad de aprendizaje recomendada.....	214
Autoevaluación 6.....	216
Actividades finales del bimestre	223
Semana 15	223
Actividades de aprendizaje recomendadas.....	223
Semana 16	225
4. Solucionario	226
5. Referencias bibliográficas	237

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



1. Datos de información

1.1. Presentación de la asignatura



1.2. Competencias genéricas de la UTPL

- Comportamiento Ético.

1.3. Competencias específicas de la carrera

- Implementar aplicaciones de baja, mediana y alta complejidad integrando diferentes herramientas y plataformas para dar solución a requerimientos de la organización.

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

1.4. Problemática que aborda la asignatura

- Las temáticas sobre desarrollo de aplicaciones web tradicionales desde el ámbito del cliente y el servidor; servicios en la nube; manejo de ORM para acceso a base de datos relacionales y acceso a bases de datos NoSql; brindarán un sustento para entender la importancia del uso tecnologías de ambiente web en el proyecto que involucra la creación de un marco para describir cómo una comunidad productiva puede organizarse con el apoyo de las tecnologías de la información para mejorar su producción y oportunidades de negocio.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



2. Metodología de aprendizaje

Para el desarrollo de los contenidos en la asignatura, se usa la metodología de aprendizaje denominada ***blended learning***. A través de estos procedimientos, el proceso de aprendizaje se divide en trabajo autónomo y actividades síncronas y asíncronas. Se enfoca en brindar al estudiante los espacios para organizar su tiempo para cumplir con las actividades propuestas en cada una de las unidades de estudio de la asignatura, destacando los aspectos teóricos y prácticos. Además, permite al docente acompañar al estudiante a través de actividades síncronas permanentes para dar solución y aclarar dudas. Finalmente, los profesionales en formación reciben una atención personalizada en la adquisición de los resultados de aprendizaje propuestos.

Para revisar la información relacionada a la metodología descrita usted puede ingresar al enlace que describe la [metodología blended learning](#).

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



3. Orientaciones didácticas por resultados de aprendizaje



Primer bimestre

Resultado de aprendizaje 1

Describe la diferencia entre SaaS y una aplicación Web tradicional.

Contenidos, recursos y actividades de aprendizaje

El resultado de aprendizaje planteado permitirá el estudio de las características y tecnologías en el desarrollo de aplicaciones web tradicionales, junto a los conceptos y ámbitos de aplicación de servicios en la nube que involucran infraestructura, plataforma y software.



Semana 1

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



Unidad 1. Conceptos generales de desarrollo de plataformas web

Estimado estudiante, en el estudio de la asignatura de Desarrollo Basado en Plataformas Web usted podrá revisar un conjunto de conceptos, metodologías y tecnologías en el desarrollo de aplicaciones web desde el punto de vista del servidor y del cliente.

1.1. Características deseables de un sitio web

*Estimado estudiante, es indispensable que se tome unos minutos para realizar una lectura comprensiva del texto básico, en lo relacionado al capítulo **¿Qué esperan los usuarios y las organizaciones de un sitio web?** sección **Algunas características funcionales de un sitio web.***

Luego que usted hizo una lectura de los apartados del texto básico, pudo apreciar que este realiza una descripción detallada de cada singularidad que debe tener un sitio web; el texto básico relaciona propiedades de calidad de software con el desarrollo de sitios web.

Las aplicaciones web deben poseer un conjunto de características indispensables para identificar su idoneidad en función de las necesidades para las que fueron formadas:

- Corrección y funcionalidad

El término corrección tiene concordancia con la generación de tareas o rutinas sin errores en función de las especificaciones dadas; a su vez la funcionalidad brinda el grupo de

posibilidades que se tiene en un sistema. El texto básico presenta un conjunto de interrogantes que deben plantearse antes de iniciar el proceso de desarrollo, ligados a:

- Delimitación en cuanto al alcance del proyecto
 - La claridad de los requerimientos
 - Diversos casos que se pueden presentar en función de los requerimientos
- Robustez

Como complemento de la corrección, esta característica permite identificar la capacidad que las aplicaciones poseen para poder trabajar o responder en escenarios fuera de lo común. Se trata de evitar que la aplicación deje de funcionar por algún caso inesperado. En el texto se realiza una reflexión sobre qué tan importante es la robustez para un sitio web, y se concluye que la relevancia aumenta mientras las aplicaciones tengan un nivel de criticidad alto.

- Facilidad de uso e imagen atractiva

Involucra la facilidad de uso de la aplicación por parte de personas que posean diversas formaciones y aptitudes. Además, se debe tomar en consideración el nivel de complejidad de la instalación y puesta en marcha.

- Portabilidad y compatibilidad

La portabilidad se refiere a la posibilidad de instalar en diversos entornos a nivel de hardware y software. La compatibilidad mide el nivel de interoperabilidad con elementos de otros programas o sistemas. Se resalta que estas características se deben fijar como objetivo fundamental desde el inicio del proyecto. COMENTARIO: Se refiere a la posibilidad de instalar (oración incompleta)

- Seguridad

Aquí se manifiesta la necesidad de generar procedimientos para impedir accesos que no estén autorizados, desde procesos muy básicos hasta alcanzar niveles complejos. Destaca que la integridad de un sistema web es un aspecto muy importante a para tomar en consideración para no sufrir inconvenientes en la implementación y uso de la aplicación. Un ejemplo de técnicas que debe tomar en consideración son las que prevengan la inyección SQL maliciosas.

- Facilidad de mantenimiento desde la visión del usuario

Durante la puesta en marcha de la aplicación, es posible que se deba incrementar alguna funcionalidad; el sistema tiene que estar construido de tal forma que la implementación de dicha funcionalidad sea lo más transparente posible para los desarrolladores y usuarios. Ejemplo, el cambio de la dirección física del servidor debe estar en un ambiente persistente que permita una actualización muy fácil para usuarios desarrolladores.

- Oportunidad y economía

Oportunidad se refiere a la característica de poder lanzar el sistema antes o en la fecha establecida en el cronograma. En cuanto a economía, se dice que el sistema pueda ser completado únicamente con el presupuesto asignado.

1.2. Tecnologías para creación de sitios web

Es momento de abordar el tema relacionado a lo descrito en el texto básico en el capítulo y sección bajo el nombre **Tecnologías para la creación de sitios web**. Luego de realizar la lectura ya se tiene una importante idea de conceptos fundamentales en el marco de la creación de sitios web.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

En el marco del entorno tecnológico de los sitios web, una empresa tiene la responsabilidad de estar en las mismas condiciones tecnológicas que sus competidores. A continuación, se exponen algunas ideas que refuerzan lo abordado en la temática propuesta en el texto básico:

Tabla 1. Conceptos básicos de tecnologías para la creación de sitios web

Conceptos básicos de tecnologías para la creación de sitios web	
W3C (Consorcio World Web) ¹	Se refiere a una comunidad internacional que trabaja en la generación de estándares para la web a nivel global; se enfoca de igual manera en garantizar el acceso a internet a cualquier persona, independientemente de su condición. La organización está representada por el inventor de la web, Tim Berners-Lee ²
Internet y la web (WWW)	Es importante mencionar que estos dos términos no son iguales. Internet se refiere al conjunto de: <ul style="list-style-type: none">▪ Servidores▪ Redes▪ Equipo de cómputo▪ Medios de comunicación La unión de lo antes mencionado genera la gran red de redes, con ámbito global. World Wide Web (www) o de forma abreviada web se manifiesta como un sistema de comunicación que tiene como base hipertexto o colecciones de: <ul style="list-style-type: none">▪ Documentos▪ Imágenes que se vinculan a través de hipervínculos que residen en servidores de la red de redes.

¹ <https://www.w3.org/>

² <https://www.w3.org/People/Berners-Lee/>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Conceptos básicos de tecnologías para la creación de sitios web

Página web	<p>Conjunto de documentos que poseen información en diferentes formatos:</p> <ul style="list-style-type: none">▪ Texto▪ Audio▪ Video-multimedias <p>Desde el punto de vista local, puede la página web estar almacenada en una computadora personal o ser almacenada en un servidor remoto.</p> <p>Importante, una página web puede estar compuesta por una o varias páginas y el lenguaje que se usa para la construcción es HyperText Markup Language (HTML).</p> <p>Existen dos tipos de páginas web básicamente:</p> <ul style="list-style-type: none">▪ Estáticas: caracterizadas por tener información que no varía en el tiempo. No tiene opción de interacción con los usuarios.▪ Dinámicas: se basa en la interacción con el usuario; puede acceder a bases de datos para almacenar de forma persistente la información.
------------	--

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Conceptos básicos de tecnologías para la creación de sitios web

HTML (HyperText Markup Language), XHTML (eXtensible HTML) y DHTML (Dynamic HTML)	<p>HTML es un lenguaje de etiquetado de documentos que permite la generación de páginas. A lo largo de los años las especificaciones HTML han evolucionado:</p> <ul style="list-style-type: none">• HTML 1 en el año 1991• HTML 3.0 en el año 1995• XML en el año 1996• HTML 4 en 1999• XHTML en 2002• HTML5 en 2008 <p>En el listado anterior se hace referencia al estándar XHTML que posee especificaciones de lenguaje estrictos. Está basado en XML y tiene como objetivo alinearse a la perspectiva de la web semántica³</p> <p>Otra variante al estándar HTML es DHTML, que añade funcionalidad en cuanto a la creación de estilos y etiquetas y permite la interoperabilidad con lenguajes como JavaScript.</p> <p>Se resalta un conjunto de lenguajes de programación que permiten el desarrollo de páginas web dinámicas como:</p> <ul style="list-style-type: none">• PHP• Python <p>En relación al estándar HTML5 maneja etiquetas que permiten manipular párrafos de texto, imágenes, videos, archivos multimedia y scripts.</p>
Proveedor de servicios o acceso a internet (ISP, Internet Service Provider)	Se hace referencia a las empresas o proveedores que ofrecen el servicio de acceso a internet. En el texto básico se describen las tecnologías que son empleadas por las empresas para brindar el servicio: <ul style="list-style-type: none">• DSL/ADSL• Internet por cable• Internet Inalámbrico (WISP – Wireless Service Provider)• Satelital

³ <https://www.w3.org/standards/semanticweb/>

Conceptos básicos de tecnologías para la creación de sitios web

Navegadores web (Browser)	Cuando se desea acceder a una página web se necesita tener instalado en las computadoras o dispositivos personales navegadores web. El objetivo de los visores web es visualizar el contenido que desee el usuario en cuanto a texto, imágenes, tablas, audio, vídeo, archivos XML, etc.
Protocolo HTTP	<p>Este protocolo permite administrar peticiones y respuestas que se puedan dar entre el cliente y el servidor. Usa el puerto 80. El cliente debe generar un proceso de comunicación. La petición será enviada y recibida por el servidor, quien generará una respuesta en formato HTML en la mayoría de los casos. A través del protocolo se accede a las direcciones únicas de internet (URL).</p> <p>Los navegadores pueden utilizar los siguientes protocolos adicionales:</p> <ul style="list-style-type: none"> • HTTPS (HyperText Transport Protocolo Secure) permite utilizar una conexión con servidores seguros de internet. El puerto usado por el protocolo es 443. • FTP, a través del puerto 21, permite la transferencia de archivos.
Trabajo cliente/servidor	Cuando se habla de ambientes distribuidos se hace referencia al trabajo cliente/servidor. El Cliente genera una petición a una entidad remota que toma el nombre de servidor. El servidor busca los mecanismos a través de programas o procedimientos establecidos, con el objetivo de satisfacer la petición del cliente.
Servidor	<p>Las características de un equipo servidor son muy singulares en cuanto al almacenamiento y procesamiento de información.</p> <p>De acuerdo a diversas necesidades, los servidores se clasifican en:</p> <ul style="list-style-type: none"> • Servidores de impresión • Servidores web • Servidores de archivos • Servidores de base de datos • Servidor proxy

Conceptos básicos de tecnologías para la creación de sitios web	
Sitio web (web site)	En un sitio web se agrupa un conjunto de archivos o páginas web en diversos formatos y se los organiza en forma de jerarquía. Es muy importante señalar que el sitio web parte de una página que tiene el nombre de índice (index)
Dominio	El nombre compuesto por caracteres alfabéticos y numéricos que necesariamente debe estar asociada a una dirección física de un computador o dispositivo que se vincula a internet. Seguramente usted ha escuchado el término hosting (hospedaje); cuando se ubica los archivos de una página web en un servidor, toma el nombre de hosting.
Direcciones IP (Internet Protocol)	Se conoce como dirección IP a la dirección que se le da a los servidores de internet. La IP está formada por cuatro números de tres dígitos separados por un punto; tienen un rango de 0 a 255. Existen diferentes tipos de direcciones IP. <ul style="list-style-type: none">• Públicas• Privadas
Servidor de nombres de dominio (DNS)	Su tarea fundamental es relacionar la dirección IP con el nombre de dominio. Ejemplo: cuando abrimos un navegador y se escribe en la barra de direcciones una URL (Uniform Resource Locator): https://www.utpl.edu.ec ; la información será atendida por el servidor DNS que realiza el proceso de identificación de la página web.
Localizador uniforme de recursos (URL, Uniform Resource Locator)	Permite nombrar recursos en internet; asigna una dirección a cada recurso que está disponible en internet. Está formada por: <ul style="list-style-type: none">• Protocolo: HTTP o HTTPS• Dominio• Ruta Ejemplo: https://www.utpl.edu.ec/internacional Donde: https:// : protocolo www.utpl.edu.ec : dominio /internacional : ruta



Semana 2

1.3. Servicios en la nube (cloud)

Estimado estudiante, continuamos con la planificación de la asignatura para abordar el tema relacionado a Servicios en la nube. Usted debe dar lectura en el texto básico al capítulo 2: **Tecnologías para la creación de sitios web sección cómputo o servicios en la nube (cloud).**

Con base en lo mencionado por el autor en el texto básico, cuando se habla de servicios en la nube se considerará como afirmativas las siguientes ideas:

- Agrupamiento de servidores que se encuentran en centros de información en todo el mundo.
- Modelo que permite acceder a diversos servicios como:
 - Almacenamiento en servidores
 - Acceso a redes de información
- Busca que los servicios estén a disposición de usuarios de manera transparente.
- Los servicios que se brinda a través de cloud computing son:
 - Infraestructura como Servicio (IaaS, Infrastructure as a Service)
 - Plataforma como Servicio (PaaS, Platform as a Service)
 - Software como Servicio (SaaS, Software as a Service)

1.4. Tipos de servicio en la nube

Estimado estudiante, para abordar el presente ítem de la asignatura se pide la revisión literaria del recurso educativo abierto denominado: **Del cloud computing al big data**, sección **Cloud Computing** subsección **Definiendo el cloud computing y La flexibilidad del cloud computing**.

Luego de la lectura y la relación que usted puede realizar con los conceptos del texto básico en la sección **Cómputo o servicios en la nube (cloud)**, ya se tiene una idea concreta de los ámbitos de aplicación de cada uno de los tipos de servicio descritos.

En los siguientes apartados se presentan los escenarios donde se puede aplicar los tipos de servicios en la nube en el ámbito empresarial, desde el punto de vista de la empresa Microsoft⁴

1.4.1. Infraestructura como servicio

A nivel de infraestructura como servicio, Microsoft ofrece los siguientes escenarios para empresas⁵:

- Desarrollo de pruebas
- Hospedaje de sitios web
- Almacenamiento, copias de seguridad y recuperación
- Aplicaciones web
- Informática de alto rendimiento

1.4.2. Plataforma como servicio

A nivel de plataforma como servicio Microsoft ofrece los siguientes escenarios para empresas⁶ que los soliciten:

⁴ <https://www.microsoft.com/>

⁵ <https://azure.microsoft.com/es-es/overview/what-is-iaas/>

⁶ <https://azure.microsoft.com/es-es/overview/what-is-paas/>

- Marco de desarrollo para personalizar aplicaciones que residen en la nube
- Análisis o inteligencia empresarial para generar análisis y minería de datos
- Servicios de mejora que involucran flujos de trabajo, seguridad y programación

1.4.3. Software como Servicio

En el recurso sugerido para el presente apartado se explica que el uso de software como servicio puede estar vinculado a áreas de la empresa como: gestión de recursos humanos, gestión financiera y uso de herramientas, que antes solo estaban disponibles en los computadores personales de forma local.

Microsoft ofrece los siguientes servicios⁷:

- A través de Outlook acceder a correos electrónicos desde cualquier dispositivo.
- Adquirir aplicaciones para productividad de las empresas como: administrador de las relaciones con el cliente – CRM, planeamiento de recursos empresariales – ERP. Se resalta que el pago por el consumo tiene relación directa con el nivel de uso.

⁷ <https://azure.microsoft.com/es-es/overview/what-is-saas/>



Actividad de aprendizaje recomendada

Actividad 1

- **Actividad de aprendizaje:** Realizar la autoevaluación 1, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 1: Conceptos generales de desarrollo de plataformas web. En la parte introductoria de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección **solucionario de la guía didáctica**.



Autoevaluación 1

Es momento de medir el entendimiento de algunos conceptos estudiados en la Unidad 1. Conceptos generales de desarrollo de plataformas web. Se solicita revisar:

- La unidad 1 de la presente guía.
- Capítulo: ¿Qué esperan los usuarios y las organizaciones de un sitio web?, sección: Algunas características funcionales de un sitio web; del texto básico
- Capítulo y sección: Tecnologías para la creación de sitios web del texto básico
- Capítulo: Tecnologías para la creación de sitios web, sección: Cómputo o servicios en la nube (cloud) del texto básico
- Sección: Cloud computing subsección definiendo el cloud computing y la flexibilidad del cloud computing del recurso abierto del cloud computing al big data.

La autoevaluación le permitirá reafirmar los conceptos estudiados. Los enunciados tienen una sola opción correcta. Lea de forma atenta cada respuesta y seleccione la opción que usted considere correcta.

1. **Identifique una propiedad de calidad de software en el desarrollo de sitios web, de las siguientes expresiones propuestas:**

- a. Framework.
- b. Corrección y funcionalidad.
- c. HTML.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- 2. Prevenir la inyección de SQL maliciosas ¿a qué propiedad de software en el desarrollo de sitios web se refiere?**
 - a. Facilidad de mantenimiento.
 - b. Portabilidad.
 - c. Seguridad.
- 3. ¿A qué literal se refiere la siguiente idea? Una comunidad internacional que trabaja en la generación de estándares para la web a nivel global.**
 - a. HTML.
 - b. XHTML.
 - c. W3C.
- 4. Identifique cuál de los siguientes protocolos es usado por los navegadores:**
 - a. SSH.
 - b. HTTPS.
 - c. UDP.
- 5. ¿Cuáles son las partes de una URL?**
 - a. Protocolo, recurso, ruta.
 - b. Protocolo, dominio, ruta.
 - c. Protocolo, dominio, HTTP.

- 6. ¿Cuál de las siguientes ideas es válida para describir los servicios en la nube?**
- a. Busca que los servicios estén a disposición de usuarios solo en procesos críticos.
 - b. Modelo que permite acceder a entornos locales como: almacenamiento en servidores y acceso a redes de información.
 - c. Modelo que permite acceder a diversos servicios como: almacenamiento en servidores y acceso a redes de información.
- 7. Identifique el concepto correcto para HyperText Markup Language.**
- a. Es un lenguaje de programación funcional que permite la generación de páginas.
 - b. Es un lenguaje de orientación a objetos que permite la generación de páginas.
 - c. Es un lenguaje de etiquetado de documentos que permite la generación de páginas.
- 8. Identifique dos complementos del lenguaje XML.**
- a. XSL, XPath.
 - b. HTML5, Ajax.
 - c. JavaScript, DHTML.
- 9. Alojar una página web en un servidor. ¿Con qué nombre se lo conoce?**
- a. Dirección IP.
 - b. Hosting.
 - c. Dominio.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

10. ¿A qué concepto hace referencia lo siguiente?: un equipo con características particulares en lo relacionado a almacenamiento y procesamiento de información.

- a. Cliente.
- b. Hosting.
- c. Servidor.

¡Excelente! Usted ha finalizado el estudio de la primera unidad. Es importante recordarle que si necesita afianzar los conceptos que pudieran resultarle un poco más difíciles puede revisar los apartados correspondientes.

[Ir al solucionario](#)

Avancemos a la siguiente unidad.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Resultado de aprendizaje 2

Discute el impacto de la implementación de estándares y/o atributos de calidad en aplicaciones web.

Contenidos, recursos y actividades de aprendizaje

Las temáticas abordadas a través del recurso de aprendizaje descrito son relacionadas al entendimiento conceptual y práctico de programación de lado del cliente usando HTML, CSS y JavaScript y el estudio de conceptos de ORM para la interacción con base de datos relacionales.



Semana 3



Unidad 2. Programación del lado del cliente

Las herramientas como HTML, CSS y JavaScript facilitan la programación de lado del cliente, es decir sus procesos se ejecutan en el navegador del usuario. De la aplicación backend se hablará en las próximas unidades.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

En la unidad número 2 de la presente guía se estudiará conceptos y ejemplos sobre la programación del lado del cliente; se solicita leer de manera analítica el capítulo 4 del texto básico denominado Hacia la interfaz con el usuario.

2.1. Introducción

Es momento de realizar una lectura de la sección Un primer acercamiento a la programación del lado del cliente, subsecciones: ¿Qué es HTML, CSS y JavaScript?; La coherencia entre el diseño visual y la labor del programador; y Funcionamiento obvio y adaptable, el modelo actual de los sitios web; el autor describe conceptos importantes para entender el proceso de desarrollo de aplicaciones del lado del cliente.

Es necesario analizar la reflexión del texto básico: el porqué de las recomendaciones de usar HTML, CSS y JavaScript; en el transcurso de los próximos años los estándares evolucionarán. La sintaxis puede variar, pero la responsabilidad como profesionales de las Tecnologías de la Información es saber utilizar las herramientas en diferentes contextos o problemáticas.

Se resalta la importancia de dejar de lado el diseño de plano de pantallas y siempre asociar el diseño a las funcionalidades que se desea construir; en el marco de desarrollo de aplicaciones web.

En cuanto al funcionamiento obvio y adaptable, se indica la necesidad obligatoria de crear aplicaciones web intuitivas, evitando distracciones innecesarias por parte de los usuarios al momento de usarlas; sobresalen las siguientes recomendaciones:

- Generar pruebas con usuarios reales.
- Los sitios deben ser muy fáciles de usar.
- Estructuras de las páginas fáciles de identificar.
- Bajar el ruido visual.

Para la generación de elementos HTML en el ámbito de diseño web adaptativo, en el texto básico se sugiere dos características:

- Los elementos HTML se deben expresar en porcentajes, no pixeles.
- Cuando sea indispensable se debe generar dimensiones mínimas en los elementos.

2.2. Ejemplos de HTML y CSS

Para entender este punto de la guía didáctica se solicita a usted, estimado estudiante, realizar una lectura del capítulo 4 del texto básico: **Hacia la interfaz con el usuario; secciones: Los primeros elementos de HTML y CSS; y Panorama general de HTML y CSS3.**

Ejemplo 1

Para el primer ejemplo básico de HTML se usará el siguiente código:

Ejemplo 1

```
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
2 <html>
3 <head>
4   <meta http-equiv="content-type" content="text/html; charset=utf-8" />
5   <title>El País Ecuador</title>
6 </head>
7 <body>
8   <h1>Ecuador</h1>
9   <h2>
10 Oficialmente República del Ecuador, es un país soberano ubicado en la región noroccidental de América del Sur, compuesto por veinticuatro provincias. Limita al norte con Colombia, al sur y al este con Perú y al oeste con el océano Pacífico, el cual lo separa de las islas Galápagos por mil kilómetros entre la península de Santa Elena y la isla San Cristóbal.
11 </h2>
12
13 <h3>Tomado de <a href="https://es.wikipedia.org/wiki/Ecuador">wikipedia</a></h3>
14 </body>
15 </html>
```

Usted debe realizar los siguientes pasos para visualizar el HTML propuesto en un navegador en su computador:

- Copiar el código en un editor de texto: bloc de notas, SublimeText, etc.
- Guardar el archivo con la siguiente estructura:
 - [nombre del archivo].html
- Para visualizar el contenido:
 - Opción 1: hacer doble click en el archivo HTML y se abrirá el navegador que tenga configurado por defecto.
 - Opción 2: abrir el navegador que usted desee (Chrome, Mozilla Firefox), ir al menú *archivo*, opción *abrir archivo* y buscar el archivo en el directorio donde está el archivo HTML.

Luego de realizar los pasos descritos se visualiza la Figura 1.

Ecuador

Oficialmente República del Ecuador es un país soberano ubicado en la región noroccidental de América del Sur, compuesto por veinticuatro provincias. Limita al norte con Colombia, al sur y al este con Perú y al oeste con el océano Pacífico, el cual lo separa de las islas Galápagos por mil kilómetros entre la península de Santa Elena y la isla San Cristóbal.

Tomado de [wikipedia](#)

Figura 1. Ejemplo de HTML básico

Explicación del ejemplo:

- La línea 1 indica en qué versión de HTML serán interpretadas las líneas expuestas en el archivo.
- Línea 2 da inicio al documento a través de la etiqueta **html**.
- Línea 3 se inicia el encabezado de la página.
- La línea 4 permite especificar el estándar UTF-8 para representar caracteres en cualquier navegador.
- Línea 7. desde esta etiqueta se ubican las etiquetas que deseamos incorporar a la página web.
- Tomar en consideración que cada etiqueta usada en la página ha sido cerrada.

En el texto básico se recomienda revisar que los contenidos generados en la aplicación web estén bajo las recomendaciones de la W3C; para ello se debe usar la página web <https://validator.w3.org/>; para el ejemplo expuesto se tiene la siguiente salida luego de realizar la comprobación, ver Figura 2. Según los resultados, manifestar algunas consideraciones:

- La primera salida indica una advertencia; se sugiere según el estándar de la W3C, agregar el atributo lang a la etiqueta html.

- La segunda salida es un error. La versión de HTML está obsoleta; es correcto pues el estándar sugiere el uso de <!DOCTYPE html>. En el texto básico se manifiesta que a pesar de que el estándar del ejemplo está en desuso es muy importante que los estudiantes en formación se familiaricen con encabezados que se siguen usando en algunas aplicaciones en el mercado.

Nu Html Checker

This tool is an ongoing experiment in better HTML checking, and its behavior remains subject to change

Showing results for ejemplo1.html

Checker Input

Show source outline image report

Check by No se eligió archivo

Uploaded files with .xhtml or .xht extensions are parsed using the XML parser.

Use the Message Filtering button below to hide/show particular messages, and to see total counts of errors and warnings.

1. **Warning** Consider adding a `[lang]` attribute to the `[html]` start tag to declare the language of this document.
From line 1, column 51; to line 2, column 6
`4.01//EN">><html><head`
For further guidance, consult [Declaring the overall language of a page](#) and [Choosing language tags](#).
If the HTML checker has misidentified the language of this document, please [file an issue report](#) or [send e-mail to report the problem](#).

2. **Error** Obsolete doctype. Expected `<!DOCTYPE html>`.
From line 1, column 1; to line 1, column 50
`<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"><html>`

Figura 2. Uso de validador de contenido HTML bajo las consideraciones de la W3C

En los siguientes ejemplos se hace uso del estándar HTML5 y CSS3; como lo indica el texto básico, las principales diferencias en el uso del estándar HTML5 se listan a continuación.

- El encabezado tiene la siguiente estructura <!DOCTYPE html> .
- Se usa como recomendación el grupo de caracteres UTF-8.
- Los estilos se los administra desde las hojas de estilo CSS.

En el texto básico se reflexiona sobre las hojas de estilo y su importancia en el desarrollo de aplicaciones web; se refiere a la modularidad y mantenimiento de código, los cambios a futuro deben ser fáciles de realizar por los desarrolladores. Las hojas de estilo permiten tener una uniformidad en cuanto a formato y presentación en una aplicación web.

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - [Desarrollo del ejemplo](#)

Ejemplo 2

En el ejemplo se usa dos archivos:

- html; que tiene el siguiente código

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo 2

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <link rel="stylesheet" href="estilos.css" type="text/css" />
7   <title>Ejemplo HTML5 y CSS3</title>
8 </head>
9 <body>
10  <h2>Provincias del Ecuador</h2>
11  <table>
12    <tr>
13      <td class="titulodecampo">Nombre</td>
14      <td class="titulodecampo">Capital</td>
15    </tr>
16    <tr>
17      <td>Loja</td>
18      <td>Loja</td>
19    </tr>
20    <tr>
21      <td>Pichincha</td>
22      <td>Quito</td>
23    </tr>
24
25  </table>
26 </body>
27 </html>
```

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

- Estilo css, con las siguientes sentencias:

Estilo.css

```
1 /* manejo de las características de la etiqueta body*/
2
3 body {
4   background: #FEFEFD;
5   padding: 20px;
6
7 }
8
9 /* manejo de las características de la etiqueta table*/
10 th, td {
11   padding: 15px;
12   border-bottom: 1px solid black;
13 }
14
15 tr:hover {
16   background-color: #258EAD;
17 }
18
19 /* manejo de las características de las clases*/
20
21 titulodecampo {
22   background: #9999FF;
23   color: #000000;
24   font-family: Arial, Helvetica, sans-serif;
25   font-size: 14px;
26   font-weight: bold;
27
28 }
```

Si usted ejecuta en su computador personal los archivos anteriores, se tiene la siguiente salida de la Figura 3.

Provincias del Ecuador

Nombre	Capital
Loja	Loja
Pichincha	Quito

Figura 3. Visualización del ejemplo 2, usando el estándar HTML5 y CCS3
Elaborado por: Elizalde, R. (2020)

Los ejemplos anteriores permiten recordar conceptos ya estudiados en asignaturas anteriores. En el desarrollo de las temáticas se muestran algunas recomendaciones en lo relacionado a librerías, extensiones y páginas en general que apoyan el desarrollo de aplicaciones web.

- En los navegadores Mozilla, Firefox y Chrome se puede utilizar dentro de las herramientas de Desarrollo Web las opciones de inspector, consola, editor de estilos, etc con el objetivo de poder realizar acciones como: realizar cambios en nuestra página sin necesidad de acudir a los archivos fuentes, los cambios no serán persistentes. Revise Figura 4

The screenshot shows the Mozilla Firefox Developer Tools interface with the "Inspector" tab selected. On the left, the DOM tree displays the HTML structure of a table with two rows. The right side shows the CSS panel where styles are being applied to specific elements. A style for ".titlelocaledcampo" is defined with a background color of #00FFFF and a font-family of Arial, Helvetica, sans-serif. Another style for "th, td" is defined with padding of 15px and a black border-bottom. The "Disposición" (Layout) tab is active, showing a visual representation of a table cell with dimensions of 79.367x16 pixels and margins/padding values. Below the layout panel, the "Propiedades del modelo de caja" (Box Model Properties) are listed, including border-sizing, display, float, content-box, position, z-index, line-height, and position.

Figura 4. Herramientas de Desarrollo Web de Mozilla Firefox

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - [Desarrollo del ejemplo](#)

2.3. Manejo de JavaScript

Para abordar este ítem, se solicita leer la sección **JavaScript** del capítulo **Hacia la interfaz con el usuario** del texto básico.

Luego de revisar la sección indicada, usted debe tener en cuenta la importancia del uso de JavaScript, el manejo de variables, condicionales, ciclos repetitivos, entrada de datos.

A continuación, se presentan ejemplos con el lenguaje de programación interpretado JavaScript; el texto básico manifiesta que en este punto del aprendizaje se tengan claros conceptos de Programación Estructurada y Programación Orientada a Objetos.

Ejemplo 1

- Las líneas de código propuestas permiten ingresar por teclado dos valores; se procede a realizar el cálculo de la suma y diferencia de los mismos; finalmente se presenta en pantalla los valores resultantes.

Ejemplo 1

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>Demo JavaScript</title>
7   </head>
8   <body>
9     <h1>Suma de dos números</h1>
10    <script type="text/javascript" charset="utf-8">
11      var entrada1 = window.prompt("Escriba el primer valor: ", "0");
12      var entrada2 = window.prompt("Escriba el segundo valor: ", "0");
13      ...
14      var a = parseFloat(entrada1);
15      var b = parseFloat(entrada2);
16      var c = a + b;
17      var d = a - b;
18      alert("La suma de los valores es: " + c);
19      alert("La diferencia de los valores es: " + d);
20    </script>
21  </body>
22 </html>
```

Luego de ejecutar en un navegador, usted puede confirmar que el ejemplo cumple con el sencillo objetivo propuesto.

Explicación

- En la línea 10 la etiqueta **script**⁸ permite indicar que se va a insertar un script ejecutable dentro de la página HTML.
- La etiqueta script tiene una propiedad **type**, cuyo valor determina el lenguaje usado. Para el ejemplo se denota **text/javascript**, que determina el lenguaje JavaScript.

⁸ <https://developer.mozilla.org/es/docs/Web/HTML/Elemento/script>

- En la línea 11 y 12 se crean variables que van a almacenar el valor que se ingrese por pantalla, a través del dialogo mostrado por el método **window.prompt**⁹; dicha propiedad acepta una cadena que será el mensaje que se presenta al usuario, además se determina el valor por defecto, para el ejemplo es cero (0). Ver Figura 5

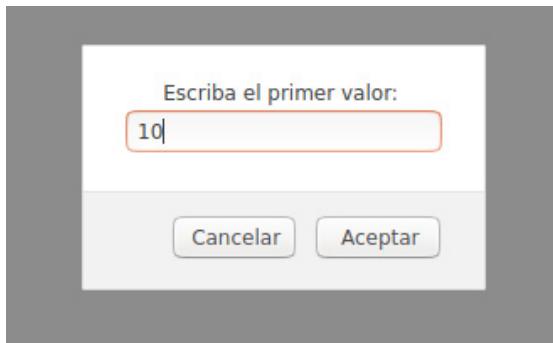


Figura 5. Diálogo mostrado por el método window.prompt

Elaborado por: Elizalde, R. (2020)

- En las líneas 14 y 15 se transforma el valor ingresado por teclado; es una cadena, en un valor numérico de tipo float, a través de la palabra reservada **parseFloat**.
- En las líneas 16 y 17 se realizan las operaciones indicadas en la problemática.
- En las sentencias de las líneas 18 y 19 se usa el método **alert** para mostrar un diálogo para presentar un texto y los valores de las operaciones del punto anterior, ver Figura 6.

⁹ <https://developer.mozilla.org/es/docs/Web/API/Window/prompt>

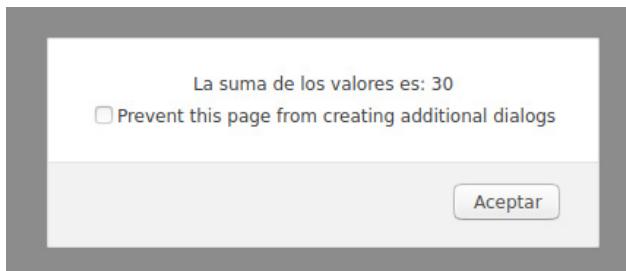


Figura 6. Usa el método alert para mostrar un dialogo para presentar un texto.

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - [Desarrollo del ejemplo](#)

Ejemplo 2

- El siguiente ejemplo, sigue la misma dinámica del ejemplo anterior; la diferencia única es la de presentar los valores resultantes de las operaciones en la misma página. Se deja de usar el método **alert**.

Ejemplo 2

```

1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>Demo JavaScript</title>
7   </head>
8   <body>
9     <h1>Suma de dos números</h1>
10    <p id="resultadoSuma"></p>
11    <p id="resultadoResta"></p>
12  __
13    <script type="text/javascript" charset="utf-8">
14      var entrada1 = window.prompt("Escriba el primer valor: ", "0");
15      var entrada2 = window.prompt("Escriba el segundo valor: ", "0");
16  __
17      var a = parseFloat(entrada1);
18      var b = parseFloat(entrada2);
19      var c = a + b;
20      var d = a - b;
21      // alert("La suma de los valores es: " + c);
22      // alert("La diferencia de los valores es: " + d);
23      document.getElementById("resultadoSuma").innerHTML = "<b>La suma de
valores<br>es: " + c + "</b>";
24      document.getElementById("resultadoResta").innerText = "<b>La diferencia de
valores<br>es: " + d + "</b>";
25    </script>
26  </body>
27 </html>

```

Explicación

- En las líneas 23 y 25 se expresan las diferencias en relación al ejemplo anterior. La palabra reservada **document**¹⁰ permite acceder a cualquier componente de la página web; y **getElementById** retorna el elemento o etiqueta que tenga el id

¹⁰ <https://developer.mozilla.org/es/docs/Web/API/Document>

pasado como parámetro, para el ejemplo es resultadoSuma y resultadoResta, respectivamente.

- Con el elemento obtenido en la línea 23 se usa la propiedad innerHTML para agregar un HTML al elemento; para el ejemplo se establece una etiqueta **** con formato de letra negrita o bold.
- Con el elemento obtenido en la línea 25 se usa la propiedad innerText para agregar un texto al elemento; a pesar que se está agregando una expresión que se podría visualizar como un HTML, solo será visible como una cadena, en base a la propiedad innerText.

El resultado final del ejemplo anterior se puede observar en la Figura 7.

Suma de dos números

La suma de valores es: 30

La diferencia de valores es: -10

Figura 7. Resultado final del ejemplo 2 de lenguaje JavaScript.

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

2.3.1. Librería jQuery

En el presente apartado se estudiará la librería de JavaScript denominada jQuery¹¹; que posee características descritas por Pardo et al., (2011) como:

- Mejorar el manejo del DOM¹²
- Manejo de eventos
- Manejo de animación
- Mejora y simplicidad en el uso de la técnica de desarrollo de ambiente web Asynchronous JavaScript And XML (AJAX¹³) para crear aplicaciones interactivas.

Se explicará su funcionamiento a través de algunos ejemplos:

Ejemplo 1

```
Ejemplo 1
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>Demo Jquery</title>
7     <script src="https://code.jquery.com/jquery-3.5.0.js"></script>
8   </head>
9   <body>
10    <h1>Ejemplo de Uso de Jquery</h1>
11    <h3 id="nombreCarrera">Tecnologías de la Información</h3>
12    <h3 id="nombreUniversidad">Universidad Técnica Particular de Loja</h3>
13    <button type="submit">Generar Mensaje</button>
14    <p id="mensajeFinal">
15    </p>
16  </body>
17  <script>
18    $(document).ready(function() {
```

¹¹ <https://jquery.com/>

¹² <https://developer.mozilla.org/es/docs/DOM>

¹³ https://developer.mozilla.org/es/docs/Web/Guide/AJAX_

Ejemplo 1

```

19 $( "button" ).click(function() {
20     var texto1 = $('#nombreCarrera').text();
21     var texto2 = $('#nombreUniversidad').text();
22     $('#mensajeFinal').text("Su carrera es: " + texto1 + ". Su universidad es: "
23         + texto2);
24     console.log(texto1);
25 });
26 });
27 </script>
28 </html>

```

Explicación

- En el ejemplo se toma el texto de las etiquetas nombreCarrera y nombreUniversidad y se lo asigna a la etiqueta mensajeFinal. Figura 8.

Ejemplo de Uso de Jquery**Tecnologías de la Información****Universidad Técnica Particular de Loja****Generar Mensaje**

Su carrera es: Tecnologías de la Información. Su universidad es: Universidad Técnica Particular de Loja

Figura 8. Salida del ejemplo de uso de jQuery.

- Para poder usar las características de jQuery en la página web se debe agregar su referencia. Existen dos formas:
 - En la dirección web <https://jquery.com/download/> se busca el archivo js de la librería y se guarda en la carpeta en la que se está trabajando el proyecto web.
 - Usar la versión de las librerías en línea, a la que se accede a través de la Red de Distribución de Contenido (Content Delivery Network – CDN). En el ejemplo se usa esta opción. En la línea 7 se especifica lo indicado.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- Para acceder a las propiedades de la librería jQuery se usa el carácter \$; además los procedimientos deben estar entre las etiquetas <script>. En la línea 19 se indica que cuando se haga un clic en cualquier etiqueta **button** de la página web, se realizará una acción. En la línea 20 se está asignando un valor en la variable texto2, el texto que tenga la etiqueta que tiene como atributo id el valor nombreCarrera, a través de la propiedad text().
- En la línea 22 se accede a la etiqueta que tiene el **id** mensajeFinal y a través del método text se le asigna una cadena de texto.

- Estimado estudiante puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

La librería jQuery tiene algunos proyectos relacionados:

- jQuery Mobile¹⁴
- jQuery User Interface¹⁵
- QUnit - JavaScript Unit Testing framework¹⁶

En relación a jQuery User Interface permite agregar plugins, interacciones y widgets de manera sencilla, permitiendo generar páginas interactivas en un alto nivel (<http://learn.jquery.com/jquery-ui/gettingstarted/>, 2020)

¹⁴ <https://jquerymobile.com/>

¹⁵ <https://jqueryui.com/>

¹⁶ <https://qunitjs.com/>

Ejemplo 2

- Se realiza el mismo proceso del Ejemplo 1. Ver Figura 9, pero se hace uso de uno de los componentes de jQueryUI denominado Accordion¹⁷, que permite ver el contenido en paneles en función de las necesidades que el usuario lo requiera.

Ejemplo 2

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>Demo Jquery y JqueryUI</title>
7     <link rel="stylesheet" href="https://code.jquery.com/ui/1.12.1/themes/base/
jquery-ui.css">
8     <script src="https://code.jquery.com/jquery-3.5.0.js"></script>
9     <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
10   </head>
11   <body>
12     <h2>Ejemplo de Uso de jQuery y jQueryUI</h2>
13     <div id="accordion">
14       <h3>Sección 1</h3>
15       <div>
16         <p id="nombreCarrera">Tecnologías de la Información</p>
17         <p id="nombreUniversidad">Universidad Técnica Particular de Loja</p>
18         <button type="submit">Generar Mensaje</button>
19       </div>
20       <h3>Sección 2</h3>
21       <div>
22         <p id="mensajeFinal"></p>
23       </div>
24     </div>
25   </body>
```

¹⁷ <https://jqueryui.com/accordion/>

Ejemplo 2

```

26 <script>
27   $(document).ready(function() {
28     $("#accordion").accordion();
29     $("button").click(function() {
30       var texto1 = $('#nombreCarrera').text();
31       var texto2 = $('#nombreUniversidad').text();
32       $('#mensajeFinal').text("Su carrera es: " + texto1 + ". Su universidad es "
33         + texto2);
34       console.log(texto1);
35     });
36   });
37 </script>
38 </html>

```

Ejemplo de Uso de jQuery y jQueryUI

▼ Sección 1

Tecnologías de la Información

Universidad Técnica Particular de Loja

Generar Mensaje

▶ Sección 2

*Figura 9. Ejemplo de uso de jQuery y jQueryUI***Explicación**

- Además de hacer uso de las librerías jQuery a través de su versión en línea vía CDN, se debe agregar dos referencias más para usar las características descritas para jQueryUI. La línea 7 hace referencia a las hojas de estilo y la línea 9 permite referenciar al archivo de JavaScript de jQueryUI.
- Existe una estructura sugerida desde la línea 13 hasta la línea 24.

- En la línea 13 se da inicio a una etiqueta **div** con atributo **id accordion**.
- La estructura tiene la siguiente secuencia:
 - Una etiqueta **h3** y una etiqueta **div**
 - La secuencia se repite dos veces; en el ejemplo hay dos paneles.
 - Importante, dentro de cada div en la secuencia se puede ubicar las etiquetas que se requieran sin importar el orden.
- Para poder visualizar los paneles, bajo las características de jQuery, en la etiqueta **script** se debe acceder al componente de la línea 13 y agregarle la funcionalidad **accordion()**; lo mencionado se realiza en la línea 28.

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - [Desarrollo del ejemplo](#)



Semana 4

2.4. Base de Datos NoSql

El desarrollo y evolución de las tecnologías surgen con la necesidad que se presenta en las diversas problemáticas; en el texto básico en capítulo 2 y sección **Tecnologías para la creación de sitios web, subsección Datos a gran escala (Big Data)** se hace un resumen

sobre la generación de datos e información atribuido a las redes sociales, sin dejar de lado los elevados volúmenes de datos que a diario se originan en bancos, compañías telefónicas, sistemas GPS, etc. El tipo de información generada es tipo estructurada, no estructurada y semiestructurada.

En la obra de Herrera et al. (2016), los autores manifiestan ideas sobre el surgimiento de NoSql (no only SQL), concepto que nace a raíz de los inconvenientes que tiene con la cantidad de información y el no acoplamiento de la misma a esquemas estrictos relacionales.

De acuerdo con Robles et al. (2015), NoSql es una arquitectura en el ámbito de base de datos que no necesitan esquemas a nivel de entidades o tablas fijas. Algunas características de este tipo de base de datos son: escalado horizontal, baja de costos de operación y mantenimiento. Algunos modelos de datos NoSql más comunes se lista a continuación:

- Modelo llave-valor, donde a cada llave se le asigna un dato, genera procesamiento de información de manera ágil.
- Modelo orientado a columnas, se usa el concepto de tabla, pero no se usa relaciones; cada dato es almacenado en columnas.
- Modelo de documentos, se caracteriza por usar formatos como: XML, JSON¹⁸ para el almacenamiento de la información.
- Modelo orientado a grafos, realiza una administración de datos provenientes de redes sociales principalmente. La información es dividida en fracciones más pequeñas o básicas y guardando relación entre los datos. (Herrera et al., 2016)

Robles et al. (2015) presenta una clasificación de sistemas NoSql.

¹⁸ <https://www.json.org/json-en.html>

Tabla 2. Clasificación de sistemas NoSql

Modelo NoSql	Ejemplos de sistemas NoSql
Llave – valor	<ul style="list-style-type: none"> ▪ Redis¹⁹ ▪ Azure Table Storage²⁰ ▪ Couchbase²¹ ▪ Amazon Dynamo²²
Orientado a columnas	<ul style="list-style-type: none"> ▪ Hadoop ▪ Hbase ▪ Cassandra
Basado en documentos	<ul style="list-style-type: none"> ▪ CouchDB ▪ MongoDB
Basado en grafos	<ul style="list-style-type: none"> ▪ OrientDB²³ ▪ Neo4J²⁴

Luego de revisar la tabla anterior usted puede identificar los principales modelos de sistemas NoSql y ejemplificaciones de cada uno de ellos.

2.4.1. Manipulación de datos con CouchDB

Para la exemplificación en el uso de base de datos NoSql se usará un base de datos de modelo basado en documentos llamada CouchDB; una de las principales ventajas para proponer su uso es la interfaz gráfica a nivel web que posee para la administración de información.

En el documento de Robles et al. (2015) se indican algunas consideraciones de CouchDB:

- Para el almacenamiento de datos se usa el formato JSON.
- Se puede realizar consultas a través del uso de Map/Reduce y el API del base datos a través del protocolo HTTP.

¹⁹ <https://aws.amazon.com/es/redis/>

²⁰ <https://azure.microsoft.com/es-es/services/storage/tables/>

²¹ <https://www.couchbase.com/press-releases/membase-general-availability>

²² <https://aws.amazon.com/es/dynamodb/>

²³ <https://orientdb.com/>

²⁴ <https://neo4j.com/>

Las ventajas de usar CouchDB radica en:

- Adaptabilidad en aplicaciones comunes.
- Un motor de búsqueda muy robusto.
- Puede ser escalable.

Instalación de CouchDB

La base de datos CouchDB puede ser instalada en sistemas operativos Windows, GNU/Linux y MacOS.

El proceso de instalación para sistema operativo Windows se lo puede realizar a través del siguiente enlace web [[instalación de couchDB en Windows](#)]; en el mismo se destacan algunos puntos:

- Se requiere tener instalado .NET Framework v3.5
- Muy importante, la ruta de instalación de CouchDB debe ser sin espacios. Ejemplo: C:\CouchDB
- En los ejercicios de la presente guía se usará la instalación en modo “single node”.

Interfaz Fauxton

- Luego de realizar los pasos de instalación, se puede acceder a la interfaz web de la base de datos; a través de la aplicación Fauxton²⁵, desde donde se puede crear, actualizar, borrar documentos. La dirección de acceso para usar dicha interfaz es:

http://127.0.0.1:5984/_utils/

Se pedirá el ingreso del usuario de administrador y la clave. Figura 10.

²⁵ <https://couchdb.apache.org/fauxton-visual-guide/>

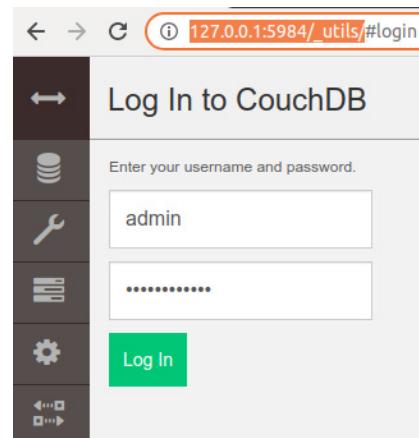
Índice

Primer bimestre

Segundo bimestre

Solucionario

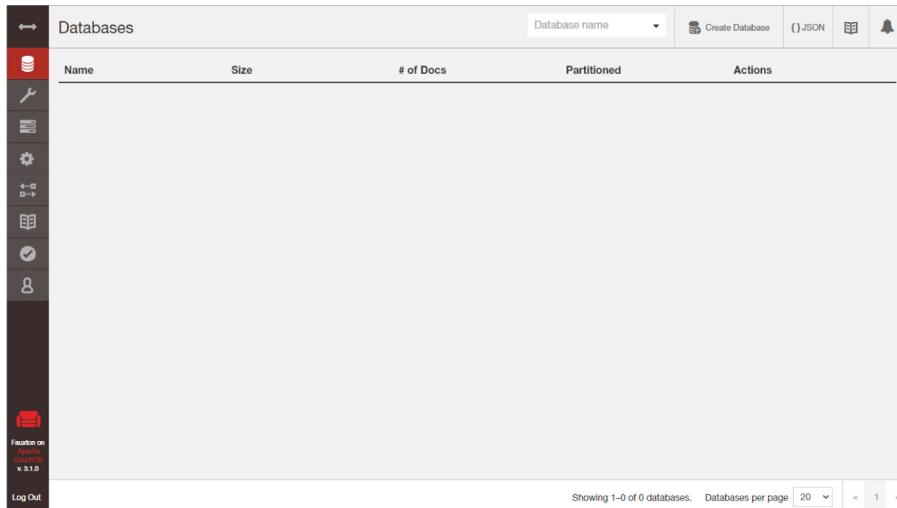
Referencias bibliográficas



The image shows a web browser window with the URL `127.0.0.1:5984/_utils/#login`. The title bar says "Log In to CouchDB". On the left is a sidebar with icons for database management, replication, and system settings. The main area has a form for entering a username and password. The username field contains "admin" and the password field contains a series of dots. A green "Log In" button is at the bottom.

Figura 10. Ingreso de credenciales de acceso de administrado de CouchDB

Luego de ingresar las credenciales de acceso, la interfaz tendrá la siguiente apariencia. **Figura 11.**



The image shows the main administration interface for CouchDB. The top navigation bar includes "Databases", "Database name", "Create Database", and "JSON" buttons. The main content area is titled "Databases" and lists one database entry: "Name" (empty), "Size" (empty), "# of Docs" (empty), "Partitioned" (empty), and "Actions" (empty). On the left is a sidebar with various icons for database management, replication, and system settings. At the bottom, it shows "Showing 1-0 of 0 databases.", "Databases per page" set to 20, and pagination controls. A footer note says "Forked on GitHub v 3.1.0" and a "Log Out" link.

Figura 11. Interfaz principal para la administración de CouchDB

- En la interfaz se pueden realizar gran parte de la administración en CouchDB, **Figura 12:**

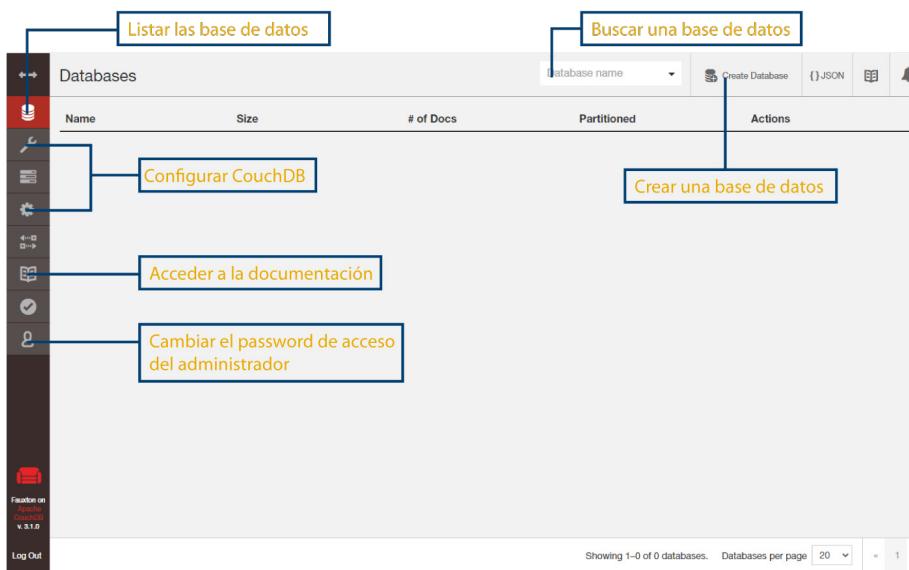


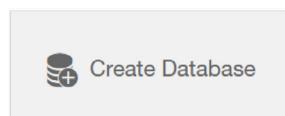
Figura 12. Opciones de administración de CouchDB a través de Fauxton.

- Listar las bases de datos.
- Configurar CouchDB.
- Acceder a la documentación.
- Cambiar el password de acceso del administrador.
- Crear una base de datos.
- Buscar una base de datos.

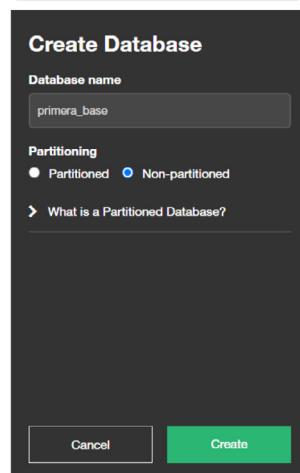
Creación de una base de datos

- Para la creación de una base de datos hacemos clic en la opción “Create Database” de la interfaz Fauxton.
- Luego, en la pantalla que se despliega se digita el nombre de la base de datos; en el particionamiento se selecciona **Non-partitioned**; finalmente se pulsa el botón **Create**. Figura 13.

Paso 1



Paso 2



Paso 3



Figura 13. Creación de una base de datos en CouchDB a través de interfaz Fauxton

- Se despliega la pantalla de acceso a la base de datos donde se puede:
 - Visualizar el nombre de la base de datos
 - Crear documentos en la base datos
 - Permisos de acceso a la base de datos
 - Diseño de documentos

Creación de documentos en la base de datos

Se recuerda que en couchDB se guardan documentos, bajo el formato JSON. El proceso para crear un documento es:

- Se pulsa el botón **Create Document**

- Se presenta una pantalla donde se podrá ingresar los documentos en formato JSON. CouchDB genera el JSON de inicio con una llave denominada “_id” y asignado un valor tipo cadena que contiene número y letras. Este valor “_id” es importante; sirve de referencia para buscar un documento y debe ser un valor único por cada documento.
- Se ingresa los valores al documento
 - A partir de JSON creado de manera automática.

```
{  
    "_id": "aac322d3ae479d4c3434ae6587001944",  
    "ciudad": "Catamayo",  
    "Provincia": "Loja"  
}
```

- Se agrega los datos, bajo el considerando; llave: valor.

```
{  
    "_id": "aac322d3ae479d4c3434ae6587001944",  
    "ciudad": "Catamayo",  
    "Provincia": "Loja"  
}
```

En el JSON se agregó dos llaves con dos valores: la llave ciudad se le asignó el valor de Catamayo; y la llave Provincia se le agregó la llave Loja.

- Se pulsa el botón **Create Document**

El proceso se puede repetir de acuerdo a las necesidades de la problemática; además es importante manifestar que en cada documento puede existir un esquema (conjunto de llaves) diferente; es una de potencialidades de CouchDB. Ver Figura 14



Figura 14. Crear un documento en CouchDB

Consulta de datos

- Cuando se agrega un documento, en la interfaz, dentro de la base de datos se muestra un listado de los documentos; se puede visualizar en formato Tabla, Metadata y JSON.
- Formato Tabla Figura 15

<input type="checkbox"/>	<input checked="" type="checkbox"/> Table	Metadata	{ } JSON	<input type="button" value="Create Document"/>
	<input type="checkbox"/>	<input checked="" type="checkbox"/> ciudad	<input type="checkbox"/>	<input type="checkbox"/> Provincia
	<input checked="" type="checkbox"/>	6f1bdff4641f6985aea235fb030...	Catamayo	Loja
	<input checked="" type="checkbox"/>	6f1bdff4641f6985aea235fb030...	Catacocha	Loja
	<input checked="" type="checkbox"/>	6f1bdff4641f6985aea235fb030...	Macará	Loja

Figura 15. Visualización de datos en formato tabla de un documento en CouchDB.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- Formato Metadata Figura 16

The screenshot shows a web-based interface for managing CouchDB documents. At the top, there are tabs for 'Table' (disabled), 'Metadata' (selected), 'JSON' (disabled), and a search bar. To the right is a 'Create Document' button. Below the header is a table with three columns: 'id', 'key', and 'value'. There are three rows of data, each with a small icon next to the id. The 'value' column contains JSON objects.

id	key	value
6f1bdff4641f6985aea235fb030...	6f1bdff4641f6985aea235fb030...	{ "rev": "1-c0c6bee7c89cc2e65...
6f1bdff4641f6985aea235fb030...	6f1bdff4641f6985aea235fb030...	{ "rev": "1-a4dc6f10790dfc5a2b...
6f1bdff4641f6985aea235fb030...	6f1bdff4641f6985aea235fb030...	{ "rev": "1-2b6ea18bdb959af4b...

Figura 16. Visualización de datos en formato metadata de un documento en CouchDB

- Formato JSON Figura 17.

The screenshot shows a web-based interface for managing CouchDB documents. The URL in the address bar is 'http://127.0.0.1:5984/_temp_view结果?group=true&group_level=1'. The page displays a single document in JSON format. The 'id' is highlighted in red. The document structure includes an '_id', '_rev', 'ciudad', and 'Provincia' field.

```
id "6f1bdff4641f6985aea235fb03001edf"
{
  "_id": "6f1bdff4641f6985aea235fb03001edf",
  "_key": "6f1bdff4641f6985aea235fb03001edf",
  "value": {
    "rev": "1-c0c6bee7c89cc2e651872955f3e30249"
  },
  "doc": {
    "_id": "6f1bdff4641f6985aea235fb03001edf",
    "_rev": "1-c0c6bee7c89cc2e651872955f3e30249",
    "ciudad": "Catamayo",
    "Provincia": "Loja"
  }
}
```

Figura 17. Visualización de datos en formato JSON de un documento en CouchDB

Modificar documentos de base de datos

- En el listado de documentos, dentro de la base de datos se puede pulsar sobre un documento y se procede a cambiar los datos que se necesiten.

Eliminación de documentos de base de datos

- En el listado de documentos, dentro de la base de datos se puede seleccionar los documentos que se requieran borrar y se pulsa el botón borrar. Ver Figura 18.

1	Provincia	_id	ciudad
<input checked="" type="checkbox"/>	Loja	6f1bdff4641f6985aea235fb030...	Catamayo
<input type="checkbox"/>	Loja	6f1bdff4641f6985aea235fb030...	Catacocha
<input type="checkbox"/>	Loja	6f1bdff4641f6985aea235fb030...	Macará

Figura 18. Eliminación de documentos de base de datos CouchDB

Generación de vistas en la base de datos

Para realizar consultas a los datos de las bases desarrolladas en CouchDB se hace uso del concepto de vistas. En este punto se debe mencionar que las consultas difieren de cómo se las realiza en modelos tipo relacional.

Una vista es un proceso que se ejecutará por cada documento. Para la creación de vistas en CouchDB a través de Fauxton se realizan los siguientes pasos Figura 19:

- En la base de datos que se requiera, pulsamos la opción **Design Documents** y luego la subsección **New View**.
- En la siguiente pantalla se escribe el nombre del **documento**; para el ejemplo se usa primero.
 - El nombre de vista en el campo **index name**; para el ejemplo se usa **new-view**.
 - Se modifica el campo **Map function**.
 - Se pulsa el botón **Create Document and the Build Index**.

The screenshot shows the Fauxton interface for managing a database named 'base_uno'. On the left, there's a sidebar with various icons and a list of database documents. The main area shows a 'New View' dialog.

Paso 1: The sidebar shows 'Design Documents' with a red '+' button. A tooltip 'Add New' is visible over it.

Paso 2: The 'New View' dialog is open. It has fields for 'Index name' (set to 'new-view'), 'Map function' (containing the code '1 - function (doc) { 2 | emit(doc._id, 1); 3 }'), and 'Reduce (optional)' (set to 'NONE'). At the bottom, there are buttons for 'Create Document and then Build Index' and 'Cancel'.

Figura 19. Creación de vistas en CouchDB a través de Fauxton

Se debe indicar algunas consideraciones en cuanto al funcionamiento de las vistas. En los puntos anteriores se nombró la expresión **creación de documento**; eso quiere decir que para crear una vista se debe crear un documento, que será tomado como un documento dentro de la propia base de datos.

El punto principal para la creación de la vista es el relacionado con Map function:

- En este campo se debe agregar código en lenguaje JavaScript.
- La estructura por defecto del JavaScript al momento de generar la vista es la siguiente.

```
function (doc) {
  emit(doc._id, 1);
}
```

Donde **doc** hace referencia a un documento de la base de datos; se debe recordar qué vista será ejecutada para cada uno de los documentos que forman la base; **emit** es la respuesta. Es una lista clave-valor de los documentos que coincidan con la lógica propuesta. En el ejemplo la llave será los `_id` de cada documento

y el valor será 1. La salida al momento de ejecutar la vista es la siguiente. Figura 20

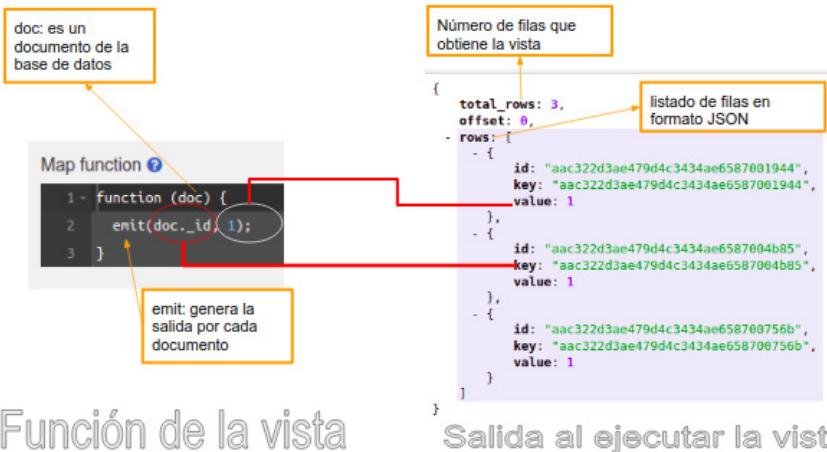


Figura 20. Salida generada por una vista en CouchDB

En líneas anteriores se manifestó que se puede consumir los datos de una base de datos de CouchDB a través del API REST que posee; se puede consumir desde un terminal de cualquier sistema operativo a través del comando **curl**²⁶ por ejemplo, desde una aplicación web, etc. Veamos un ejemplo:

- Si se quiere obtener el total de documentos de la base de datos se lo hace a través de la siguiente dirección:
http://127.0.0.1:5984/base-ejemplo-1/_all_docs

donde:

- <http://127.0.0.1:5984> es la dirección y puerto de nuestro servidor de CouchDB.
- `base-ejemplo-1` es el nombre de la base de datos.
- `_all_docs` es la función propia de CouchDB que retorna los documentos de la base de datos; la función

²⁶ <https://github.com/curl/curl>

```
{
  "total_rows": 4,
  "offset": 0,
  "rows": [
    {
      "id": "_design/primero",
      "key": "_design/primero",
      "value": {
        "rev": "3-bdf00b34820c8b704c8c6310627edf05"
      }
    },
    {
      "id": "aac322d3ae479d4c3434ae6587001944",
      "key": "aac322d3ae479d4c3434ae6587001944",
      "value": {
        "rev": "2-115c7b725cf3479a7176707ecf5764e4"
      }
    },
    {
      "id": "aac322d3ae479d4c3434ae6587004b85",
      "key": "aac322d3ae479d4c3434ae6587004b85",
      "value": {
        "rev": "2-6c1df38d25d3159894274fab5f63e7bc"
      }
    },
    {
      "id": "aac322d3ae479d4c3434ae658700756b",
      "key": "aac322d3ae479d4c3434ae658700756b",
      "value": {
        "rev": "2-360519e7940fcac778c3d864fb1b0f87"
      }
    }
  ]
}
```

Figura 21. Uso de la función _all_docs para una base de datos en CouchDB

Ejemplo 1

Generar una base de datos en couchDB que almacena la siguiente información:

Ejemplo 1

```
{
  "ciudad": "Catamayo",
  "provincia": "Loja"
}
{
  "ciudad": "Machala",
  "provincia": "El Oro"
}
{
  "ciudad": "Loja",
  "provincia": "Loja"
}
{
  "ciudad": "Cuenca",
}
```

Generar una vista que devuelva todos los documentos que tenga una llave provincia.

The screenshot shows a CouchDB interface. On the left, a modal window displays the following JSON document:

```

1 > [
2   "_id": "aac322d3ae479d4c3434ee6597001944",
3   "_rev": "2-115c7b725cf3479a7176707ecf5764e4",
4   "ciudad": "Catamayo",
5   "provincia": "Loja"
6 ]

```

On the right, a table lists four documents from a view:

<input type="checkbox"/>	aac322d3ae479d4c3...	Catamayo	Loja
<input type="checkbox"/>	aac322d3ae479d4c3...	Machala	El Oro
<input type="checkbox"/>	aac322d3ae479d4c3...	Loja	Loja
<input type="checkbox"/>	aac322d3ae479d4c3...	Cuenca	

Figura 22. Datos ingresados a la base de datos CouchDB

- Se crea la base de datos y se agrega los elementos dados en la problemática. Figura 22.
- Se genera el documento y la vista solicitada. Figura 23.



Figura 23. Vista generada en base a la problemática del Ejemplo 1

- Salida, a través del uso del comando curl, desde un terminal.
Figura 24.

- En terminal de su sistema operativo digitamos lo siguiente

curl http://127.0.0.1:5984/base-ejemplo-1/_design/primer/_view/vistados

donde:

- http://127.0.0.1:5984 es la dirección y puerto de nuestro servidor de CouchDB.
- base-ejemplo-1 es el nombre de la base de datos.
- _design, expresión que va en la URL que se quiere consumir una vista.
- primero, nombre del documento generado
- _view, expresión que va en la URL que se quiere consumir una vista.
- Vistados, nombre de la vista.

```
~$ curl http://127.0.0.1:5984/base-ejemplo-1/_design/primeros/_view/vistados
{"total_rows":3,"offset":0,"rows":[
{"id":"aac322d3ae479d4c3434ae6587004b85","key":"El Oro","value":{"_id":"aac322d3ae479d4c3434ae6587004b85","_rev":"2-6c1df38d25d3159894274fab5f63e7bc","ciudad":"Machala","provincia":"El Oro"}},
 {"id":"aac322d3ae479d4c3434ae6587001944","key":"Loja","value":{"_id":"aac322d3ae479d4c3434ae6587001944","_rev":"2-115c7b725cf3479a7176707ecf5764e4","ciudad":"Catamayo","provincia":"Loja"}},
 {"id":"aac322d3ae479d4c3434ae658700756b","key":"Loja","value":{"_id":"aac322d3ae479d4c3434ae658700756b","_rev":"2-360519e7940fcac778c3d864fb1b0f87","ciudad":"Loja","provincia":"Loja"}}
]}
```

Figura 24. Salida generada en base a la vista para solucionar la problemática

- Estimado estudiante, puede descargar la información para que pueda agregar información a su base de datos en NoSql.
 - Desarrollo del ejemplo

2.4.2. Consumo de datos de CouchDB desde JavaScript

Para el presente ítem de estudio se plantea usar la librería jQuery. Se presenta el siguiente ejemplo que demuestra la potencialidad de consumo que posee la base de datos CouchDB, a través de su servicio REST-API

Ejemplo 1

- Dada una base de datos en CouchDB que tiene los siguientes datos

Ejemplo 1

```
{  
    "ciudad": "Catamayo",  
    "provincia": "Loja"  
}  
{  
    "ciudad": "Machala",  
    "provincia": "El Oro"  
}  
{  
    "ciudad": "Loja",  
    "provincia": "Loja"  
}  
{  
    "ciudad": "Cuenca",  
}
```

- Crear una página web que haga uso de la librería jQuery y la técnica de programación AJAX para acceder a los documentos de la base de datos que tengan la llave provincia y los despliegue en la página creada.

Para solucionar lo anterior en primer lugar se genera la base de datos en couchDB, luego se agrega los datos y se crea la vista correspondiente. Se usará la base de datos y vista generados en el Ejemplo 1 del punto 2.4.2 de la presente guía.

El diseño de la página web se lo realiza a través de las siguientes líneas de código.

Ejemplo 1

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width" />
6   <title>Demo jQuery - jQueryUI - CouchDB</title>
7   <link rel="stylesheet" href="https://code.jquery.com/ui/1.12.1/themes/base/
jquery-ui.css">
8   <script src="https://code.jquery.com/jquery-3.5.0.js"></script>
9   <script src="https://code.jquery.com/ui/1.12.1/jquery-ui.js"></script>
10 </head>
11 <body>
12 ....
13   <h3>Ejemplo de Uso de Jquery + CouchDB</h3>
14   <div id="pestanias">
15     <ul>
16       <li><a href="#">Principal</a></li>
17     </ul>
18     <div id="t1">
19       <p id="mensajeFinal"></p>
20       <button type="submit">Obtener información de CouchDB</button>
21     </div>
22   </div>
23 </body>
24 <script>
25 $(document).ready(function() {
26   $("#pestanias").tabs();
27   $("button").click(function() {
28     $.ajax({
29       dataType: 'json',
30       url: "http://127.0.0.1:5984/base-ejemplo-1/_design/primero/_view/
vistados"
31     }).done(function(data) {
32       for(var i=0; i<data.rows.length; i++){
33         var c = data.rows[i].value.ciudad;
34         var prov = data.rows[i].value.provincia;
35         $('#mensajeFinal').append("<p> <b>Ciudad:</b> " + c + " <b>Provincia:</b> " +
36           prov + "</p>");
37     }
38     }).fail(function(msg, texto, textoerror) {
39       console.log( "error" );
40       console.log( msg );
41     }
42   }
43 )
44 
```

Ejemplo 1

```

41     console.log( textoerror );
42   });
43 });
44 });
45 </script>
46 </html>
```

Para la ejecución del código se solicita alojar el archivo en un servidor web; además su servidor de base de datos CouchDB debe estar en el mismo computador.

En el código se hace uso de la técnica de desarrollo de ambiente web AJAX para acceder al REST API de CouchDB, particularmente a la vista generada. Se explica a continuación el proceso realizado desde la línea 24 del código presentado.

- En la línea 26 se hace uso del contenedor de paneles tabs²⁷ de jQueryUI.
- La línea 27 indica que cuando se haga un clic en el componente button, se generan algunos procesos.
- En la línea 28 empieza una petición AJAX²⁸ a través de la librería jQuery.
- Con dataType = 'json' en la línea 29 se especifica el tipo de dato esperado del servidor.
- El atributo **url** permite indicar la URL a la cual se hace la petición; en este caso se hace uso de la URL que hace la petición servicio REST API que devuelve el resultado de la vista generada en CouchDB.
- Cuando la respuesta del servicio generado es correcta se hace uso del controlador **done** línea 31; y se realiza todo lo

²⁷ <https://jqueryui.com/tabs/>

²⁸ <https://api.jquery.com/jQuery.ajax/>

REST API de CouchDB.

```
{"total_rows":3,"offset":0,
"rows":[
{"id":"aac322d3ae479d4c3434ae6587004b85","key":"El Oro","value":{"_id":"aac322d3ae479d4c3434ae6587004b85","_rev":"2-6c1df38d25d3159894274fab5f63e7bc","ciudad":"Machala","provincia":"El Oro"}},
 {"id":"aac322d3ae479d4c3434ae6587001944","key":"Loja","value":{"_id":"aac322d3ae479d4c3434ae6587001944","_rev":"2-115c7b725cf3479a7176707ecf5764e4","ciudad":"Catamayo","provincia":"Loja"}},
 {"id":"aac322d3ae479d4c3434ae658700756b","key":"Loja","value":{"_id":"aac322d3ae479d4c3434ae658700756b","_rev":"2-360519e7940fcae778c3d864fb1b0f87","ciudad":"Loja","provincia":"Loja"}}
]
```

- La variable **data** está en formato JSON y posee las siguientes llaves:
 - "total_rows", "offset", "rows"
- De las llaves, la información que se necesita está en la llave "rows"; la misma es una lista,. Por lo tanto, se debe hacer un proceso repetitivo para acceder a cada elemento de la lista. Los elementos tienen las siguientes llaves:
 - "id", "key", "value"
- Interesa la llave "value" de cada elemento, para acceder a los valores requeridos. Cada "value" tiene las siguientes llaves
 - "_id", "_rev", "ciudad", "provincia"

- De las llaves anteriores se usa "ciudad" y "provincia" y sus valores correspondientes en cada iteración para agregar una etiqueta tipo párrafo `<p>` al elemento con `id = "mensajeFinal"`.
 - Si existe algún inconveniente con la petición:
 - No está bien formada la URL
 - No se tiene los permisos de acceso
 - Los datos devueltos están mal formados
- Se usa el controlador `fail` para manejar el proceso, línea 38 del ejemplo.

El resultado final de la aplicación se muestra en las siguientes figuras:

- Cuando se lanza la aplicación, Figura 25.



Figura 25. Salida inicial de la aplicación web del Ejemplo 1

- Cuando se pulsa el botón “**Obtener información de CouchDB**”. Figura 26.



Figura 26. Salida final del Ejemplo 1 de la aplicación web luego de ejecutar la acción del botón “Obtener información de CouchDB”



Actividad de aprendizaje recomendada

Actividad 1

Actividad de aprendizaje: realizar la autoevaluación 2, donde se exponen un conjunto de preguntas en función de las temáticas descritas en la unidad 2 Programación del lado del cliente. En la parte introductoria de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.



Autoevaluación 2

Es momento de medir el entendimiento de algunos conceptos estudiados en la Unidad 2. Programación del lado del cliente. Se solicita revisar:

- La unidad 2 de la presente guía.
- Capítulo 4 Hacia la interfaz con el usuario del texto básico
 - Sección: Un primer acercamiento a la programación del lado del cliente.
 - Subsecciones: ¿Qué es HTML, CSS y JavaScript?; La coherencia entre el diseño visual y la labor del programador; y Funcionamiento obvio y adaptable, el modelo actual de los sitios web
 - Capítulo 4 Hacia la interfaz con el usuario del texto básico.
 - Sección: Los primeros elementos de HTML y CSS
 - Sección: Panorama general de HTML y CSS3.
- Capítulo 4 Haca la interfaz con el usuario del texto básico.
 - Sección: JavaScript
- Capítulo 2: Tecnologías para la creación de sitios web del texto básico
 - Sección: Tecnologías para la creación de sitios web.
 - Subsección: Datos a gran escala (Big Data)

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

La autoevaluación le permitirá reforzar los conceptos estudiados. Los enunciados tienen una sola opción correcta. Lea de forma atenta cada respuesta y seleccione la opción que usted considere correcta.

1. Para la creación de aplicaciones web intuitivas existen algunas recomendaciones; identifique una de ellas del siguiente listado:

- a. Generar pruebas con usuarios desarrolladores.
- b. Generar pruebas con usuario reales.
- c. Generar pruebas con el gerente de la empresa solicitante.

2. En un archivo HTML existe la siguiente línea de código:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

Identifique lo que significa dicha línea:

- a. La versión de HTML que será interpretada.
- b. La versión de JavaScript que será usada.
- c. La versión de CSS que será asociada.

3. ¿Cuál es la razón principal de usar la dirección <https://validator.w3.org/>?

- a. Verificar que la aplicación web esté desarrollada bajo los estándares de Mozilla Firefox, Chrome.
- b. Verificar que la aplicación web esté desarrollada bajo los estándares expuestos por los clientes.
- c. Verificar que la aplicación web esté desarrollada bajo los estándares de la World Wide Web Consortium.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

4. Dado el siguiente código HTML, identifique la sentencia que hace falta agregar en la línea 4 para establecer como conjunto de caracteres por interpretar: UTF-8

Código HTML

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4
5     <meta name="viewport" content="width=device-width" />
6   <title></title>
7 </head>
8 <body>
9   <p> Nombre</p>
10  <p>Apellido</p>
11  <p>Cédula</p>
12 </body>
13 </html>
```

- a. <script charset="utf-8" />
- b. <link charset="utf-8" />
- c. <meta charset="utf-8" />

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Código HTML

```
index.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width" />
6   <link rel="stylesheet" href="estilos.css" type="text/css" />
7   <title>Pregunta Repaso</title>
8 </head>
9 <body>
10  <p>Nombre</p>
11  <p>Apellido</p>
12  <p>Cédula</p>
13 </body>
14 </html>
```

```
estilos.css
```

```
1
2 .tres{
3   background: blue;
4   color: white;
5 }
```

- a. <p id="tres">Cédula</p>
- b. <p class="tres">Cédula</p>
- c. <p class[id]="tres">Cédula</p>

6. Dado el siguiente código, ¿cuál es el resultado que se visualizará en el navegador, si se ingresa el valor de 10 para la variable entrada1?

Código HTML

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>Pregunta Repaso</title>
7   </head>
8   <body>
9     <p id="uno"></p>
10    <p id="dos"></p>
11  _____
12    <script type="text/javascript" charset="utf-8">
13      var entrada1 = window.prompt("Escriba el primer valor: ", "0");
14      var a = parseFloat(entrada1);
15      for (var i = 0; i < a; i++) {
16        document.getElementById("uno").innerText = i
17      }
18      document.getElementById("dos").innerText = "<b>" + i + "</b>";
19    </script>
20  </body>
21 </html>
```

Respuestas

- a. 9
10
- b. 9
10
- c. 9
10

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

7. **Para acceder a la librería jQuery a través de la Red de Distribución de Contenido y usar sus propiedades en una página web. ¿Cuál de las siguientes líneas de código se debe agregar?**
- <script src="<https://code.jquery.com/jquery-3.5.0.js>"></script>
 - <a href="<https://code.jquery.com/jquery-3.5.0.js>"></script>
 - <script href="<https://code.jquery.com/jquery-3.5.0.js>"></script>
8. **¿Cuál es el modelo NoSql usado en CouchDB y Neo4J?**
- CouchDb: basado en documentos; Neo4J: basado en gráficos
 - CouchDb: orientado a columnas; Neo4J: basado en gráficos
 - CouchDb: basado en documentos; Neo4J: Llave-valor
9. **¿Cómo se llama la interfaz administrativa de CouchDB y cuál es la dirección de acceso local por defecto?**
- Interfaz: Fauxton; dirección de acceso local:
[http://127.0.0.1 :5984/](http://127.0.0.1:5984/)
 - Interfaz: CouchDBWeb; dirección de acceso local:
[http://127.0.0.1 :5984/_utils/](http://127.0.0.1:5984/_utils/)
 - Interfaz: Fauxton; dirección de acceso local:
[http://127.0.0.1 :5984/_utils/](http://127.0.0.1:5984/_utils/)

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

10. ¿Cuál de los siguientes documentos son válidos para guardarlos en una base de datos CouchDB?

- a. {
 "_id": "aac322d3ae479d4c3434ae6587001944,
 "ciudad": "Quito,
 "Provincia": "Pichincha
 }"}
- b. {
 _id: "aac322d3ae479d4c3434ae6587001944",
 ciudad: "Quito",
 Provincia: "Pichincha"
 }}
- c. {
 "_id": "aac322d3ae479d4c3434ae6587001944",
 "ciudad": "Quito",
 "Provincia": "Pichincha"
 }}

Hemos finalizado con éxito nuestra segunda unidad de estudios.

¡Avancemos con ánimo!

[Ir al solucionario](#)



Semana 5



Unidad 3. Acceso a base de datos relacionales mediante Object Relational Mapper (ORM)

Estimado estudiante, durante su formación académica ha tenido la oportunidad de interactuar con base de datos relacionales y base de datos no relacionales. En este apartado se retoman conceptos de base de datos relacionales y se aborda una nueva temática que tiene relación con el manejo de datos a través de Object Relational Mapper o conocido por sus iniciales ORM.

3.1. Introducción

En referencia a Object Relational Mapper (ORM), Cabedo et al. (2010) indica que es una técnica que permite a desarrolladores pasar los datos de lenguajes de programación bajo el paradigma de Orientación a Objetos a datos persistentes para su posterior almacenamiento en bases de datos relacionales.

En el documento de Riera, "OBJECT/RELATIONAL MAPPING", se presentan algunas ventajas y limitaciones del uso de ORM en el desarrollo de una solución.

Ventajas del uso de ORM

- Proceso de desarrollo más rápido
- Separación de las bases de datos; cuando se utiliza un ORM se puede cambiar el motor de base de datos en cualquier momento. Dado que los ORM transforman automáticamente las consultas, también deben tener la capacidad de adaptarse a diversos gestores de base datos como: MySql²⁹, Postgres³⁰, Oracle³¹, Sqlite³², etc.
- Seguridad, en los ORM existen procedimientos que impiden ataques como SQL injections³³.

Limitaciones del uso de ORM

- Curva de aprendizaje: es difícil explotar en forma completa un ORM por su extensión; por tal razón, se debe tomar un tiempo considerable para dominar y aplicarlo en una solución.
- Aplicaciones ralentizadas; la aplicación de ORM genera que los procesos sean más lentos en un porcentaje bajo; se debe considerar y sopesar con la velocidad del desarrollo.

Librerías ORM según los lenguajes de programación

- Lenguaje de programación PHP
 - Doctrine³⁴
 - Eloquent³⁵

²⁹ <https://www.mysql.com/>

³⁰ <https://www.postgresql.org/>

³¹ <https://www.oracle.com/es/database/>

³² <https://www.sqlite.org/index.html>

³³ https://www.w3schools.com/sql/sql_injection.asp

³⁴ <https://www.doctrine-project.org/>

³⁵ <https://laravel.com/docs/7.x/eloquent>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- Active Record Class³⁶, asociado al framework Codeigniter.
- Zend-db³⁷
- Lenguaje de programación Java
 - Hibernate³⁸
 - Ebean³⁹
- Lenguaje de programación Ruby
 - Active Record⁴⁰
 - Ruby Object Mapper⁴¹
 - Sequel⁴²
- Lenguaje de programación Python
 - SQLAlchemy⁴³
 - Django-ORM⁴⁴
 - Pony-ORM⁴⁵
 - Peewee⁴⁶

3.2. Manejo de datos con ORM SQLAlchemy

En el presente apartado se va a ejemplificar el uso de la librería ORM denominada SQLAlchemy que está desarrollada en lenguaje Python.

³⁶ https://www.codeigniter.com/userguide2/database/active_record.html

³⁷ <https://docs.zendframework.com/zend-db/>

³⁸ <https://in.relation.to/2020/04/30/hibernate-orm-5415-final-out/>

³⁹ <https://ebean.io/docs/intro/database/>

⁴⁰ https://guides.rubyonrails.org/active_record_basics.html

⁴¹ <https://rom-rb.org/>

⁴² <https://github.com/jeremyevans/sequel>

⁴³ <https://www.sqlalchemy.org/>

⁴⁴ <https://docs.djangoproject.com/en/3.0/topics/db/queries/>

⁴⁵ <https://github.com/ponyorm/pony>

⁴⁶ <https://github.com/coleifer/peewee>

El recurso educativo abierto denominado **Learning SQLAlchemy** será usado como base para el aprendizaje de la librería.

Estimado estudiante en su computador necesita crear el ambiente de desarrollo óptimo para realizar los ejemplos propuestos.

Se necesita instalar:

- Lenguaje de programación Python (puede revisar el enlace que permite la [descarga e instalación en diversos sistemas operativos](#))
- Instalación de la librería SQLAlchemy ([revisar proceso de instalación](#)); se recomienda usar el proceso vía el gestor de librerías de Python denominado pip⁴⁷

SqlAlchemy está dividido en dos grandes funcionalidades y áreas como lo indica el recurso Learning sqlalchemy; la relacionada con SqlAlchemy Core y SqlAlchemy ORM. Se revisará la segunda área en la presente guía.

SqlAlchemy ORM permite manejar un patrón que permite pasar las clases desarrolladas en Python a una base de datos de forma simple y elegante. Manifestar que los ejemplos siguientes pueden ser desarrollados o probados de dos formas:

- Agregando el código a través de un editor de texto en un archivo con extensión py y luego ejecutar desde un terminal o consola con el patrón: python [nombre_archivo].py
- Usando la consola por defecto de python o la librería ipython⁴⁸

Para hacer uso de la librería existen algunas consideraciones:

- Conectarnos a la base de datos a través del siguiente código

⁴⁷ <https://pypi.org/project/pip/>

⁴⁸ <https://ipython.org/>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

código

```
1 from sqlalchemy import create_engine  
2  
3 # se genera enlace al gestor de base de  
4 # datos  
5 # para el ejemplo se usa la base de datos  
6 # sqlite  
7 engine = create_engine('sqlite:///demobase.db')
```

En el código anterior se importa el módulo `create_engine` (línea 1) que permite el enlace hacia un motor de base de datos; en la línea 7 se crea una variable llamada `engine` que hace uso de `create_engine`, a la cual se envía como parámetro una cadena de texto que indica el motor a usar. Para el ejemplo se usa un enlace para una base de datos SQLite, el nombre asignado para la base de datos es **demobase.db**.

Estimado estudiante, se solicita instalar en su computador la [base de datos](#) y el [visor de base de datos](#) SQLite.

Ejemplos de conexiones para algunos gestores de base de datos⁴⁹

Tabla 3. Ejemplos de conexiones para algunos gestores de base de datos

Gestor de base datos	Conexión
Postgres	# por defecto <code>create_engine('postgresql://usuario:clave@localhost:5432/basedatos')</code> # usando la librería psycopg2 ⁵⁰ <code>create_engine('postgresql+psycopg2://usuario:clave@localhost:5432/basedatos')</code>

⁴⁹ <https://docs.sqlalchemy.org/en/13/core/engines.html>

⁵⁰ <https://pypi.org/project/psycopg2/>

Gestor de base datos	Conexión
Mysql	# por defecto create_engine ('mysql://usuario:clave@localhost/base-de-datos') # usando la librería de Python mysqlclient ⁵¹ create_engine ('mysql+mysql://usuario:clave@localhost/base-de-datos')
Oracle	# por defecto create_engine ('oracle://usuario:clave@127.0.0.1:1521/esquema') # usando la librería de python ⁵² create_engine ('oracle+cx_oracle://usuario:clave@esquema')

- Declaración de las clases que en lo posterior se transforman en tablas de la base de datos.

Base de datos

```

8
9 from sqlalchemy.ext.declarative import declarative_base
10 Base = declarative_base()
11

```

Para realizar el proceso se necesita importar la función **declarative_base** como se observa en la línea 9.

En la línea 10 se crea una clase llamada Base para definir las clases en lenguaje Python.

La clase Base se usa como superclase para todas las clases que se necesitan inicializar.

⁵¹ <https://pypi.org/project/mysqlclient/>

⁵² https://oracle.github.io/python-cx_Oracle/

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

```
Código  
13 from sqlalchemy import Column, Integer, String  
14  
15 class Saludo(Base):  
16     __tablename__ = 'saludo'  
17     id = Column(Integer, primary_key=True)  
18     mensaje = Column(String)  
19  
20 Base.metadata.create_all(engine)
```

En la línea 13 se importa las clases Column, Integer y String, donde Column representa una columna en la base de datos. Integer y String son clases que permiten asignar un tipo de dato a la columna.

En la línea 15 se crea una clase llamada Saludo que hereda de la clase Base. A través del atributo de la clase __tablename__ se identifica el nombre de la tabla en la base de datos.

En la línea 17 se crea un atributo de clase llamado id que será una columna que tiene como características que acepta datos de tipo entero y es la clave primaria de la entidad.

En la línea 18 se crea un atributo llamado mensaje que será una columna de tipo cadena.

Con la sentencia de la línea 20 se crean las tablas que no estén creadas aún en la base de datos, ver Figura 27.

The screenshot shows the DB browser for SQLite interface. At the top, there's a menu bar with File, Edit, View, Help. Below the menu are buttons for New Database, Open Database, Write Changes, and Revert Changes. The main area has tabs for Database Structure, Browse Data, Edit Pragmas, and Execute SQL. Under Database Structure, there are buttons for Create Table, Create Index, Modify Table, and Delete Table. A table structure is displayed with columns for Name, Type, and Schema. The 'Tables (1)' section contains a single entry 'saludo'. The 'saludo' table has two columns: 'id' (INTEGER) and 'mensaje' (VARCHAR). The 'Schema' column shows the CREATE TABLE statement: `CREATE TABLE saludo (id INTEGER NOT NULL, mensaje VARCHAR, PRIMARY KEY (id))`. Below the table, there are sections for Indices (0), Views (0), and Triggers (0).

Figura 27. Tabla generada desde SQLAlchemy, visualizada desde DB
Elaborado por: Elizalde, R. (2020).

Ahora, se revisará algunos puntos para guardar información en la base de datos y entidad recién creada.

Base de datos

```
22 from sqlalchemy.orm import sessionmaker  
23  
24 Session = sessionmaker(bind=engine)  
25 session = Session()
```

Algunas explicaciones; sessionmaker es una clase generadora de clases de tipo sesión; se usa como configuración los parámetros enviados a través del constructor. Para el ejemplo, se envía el enlace creado para la base de datos, **engine**.

En la línea 21 se crea una clase de Python llamada Sesión, desde el generador sessionmaker. Comentario: No está claro si se intenta utilizar la palabra "sesión" o "session"

En la línea 22 se crea un objeto session de tipo Session, el que va a permitir guardar, eliminar, actualizar, generar consultas en la base de datos respecto a las entidades creadas. Comentario: no se entiende bien la diferencia entre "session" y "Session".

Ahora, se deja unas líneas de código que permiten crear un objeto de tipo Saludo y guardar dicho objeto como registro en la base de datos.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Código

```
27 # se crea un objeto de tipo
28 # Saludo
29
30 miSaludo = Saludo()
31 miSaludo.mensaje = "Hola mundo desde SQLAlchemy y SQLite"
32
33 # se agrega el objeto miSaludo
34 # a la entidad Saludo a la sesión
35 # a la espera de un commit
36 # para agregar un registro a la base de-
37 # datos demobase.db
38 session.add(miSaludo)
39
40 # se confirma las transacciones
41 session.commit()
```

Como se puede visualizar en Figura 28 la base de datos demobase.db ya tiene registros.

The screenshot shows the DB browser for SQLite interface. At the top, there's a menu bar with File, Edit, View, Help, and several database-related buttons: New Database, Open Database, Write Changes, and Revert Changes. Below the menu is a toolbar with Database Structure, Browse Data, Edit Pragmas, and Execute SQL buttons. A dropdown menu labeled 'Table:' is open, showing 'saludo' as the selected table. To the right of the dropdown are 'New Record' and 'D' buttons. The main area displays a table with two columns: 'id' and 'mensaje'. A single row is visible, with 'id' having a value of 1 and 'mensaje' having a value of 'Hola mundo desde SQLAlchemy y SQLite'. There are also 'Filter' buttons for both columns.

id	mensaje
1	Hola mundo desde SQLAlchemy y SQLite

Figura 28. Tabla generada desde SQLAlchemy con registros, visualizada desde DB browser for SQLite

Elaborado por: Elizalde, R. (2020)

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

3.3. Consulta de datos con ORM SQLAlchemy

Para realizar la consulta de información a una base de datos a través del ORM de SQLAlchemy se deben considerar algunos puntos:

- Usar la variable **session**
- Se usa el método **query()**; se accede desde *session*
- Al método query se le puede enviar como argumentos clases o características de una clase
- Al método query se le agrega algunas opciones como: *all*, *order_by*, *filter*, *filter_by*

Ejemplo 1:

- Se genera un archivo de Python para crear la base datos; la entidad Saludo tiene tres atributos:
 - Id, que la clave primaria; no es necesario agregarla en cada instancia; se agrega de forma autoincremental por defecto
 - Mensaje, de tipo cadena
 - Tipo, con tipo de dato cadena

Ejemplo 1

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.ext.declarative import declarative_base
3 from sqlalchemy.orm import sessionmaker
4 from sqlalchemy import Column, Integer, String
5
6 # se genera enlace al gestor de base de
7 # datos
8 # para el ejemplo se usa la base de datos
9 # sqlite
10 engine = create_engine('sqlite:///demobase2.db')
11
12 Base = declarative_base()
13
14 class Saludo(Base):
15     __tablename__ = 'saludo'
16     id = Column(Integer, primary_key=True)
17     mensaje = Column(String)
18     tipo = Column(String)
19
20 Base.metadata.create_all(engine)
```

- Se agrega registros a la base datos, a través de otro archivo de Python.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Base de datos

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3
4 # se importa la clase(s) del-
5 # archivo genera_tablas
6 from genera_tablas import Saludo
7
8 # se genera enlace al gestor de base de
9 # datos
10 # para el ejemplo se usa la base de datos
11 # sqlite
12 engine = create_engine('sqlite:///demobase2.db')
13
14 Session = sessionmaker(bind=engine)
15 session = Session()
16
17 # se crea un objetos de tipo Saludo
18
19 saludo1 = Saludo()
20 saludo1.mensaje = "Hola que tal"
21 saludo1.tipo = "informal"
22
23 saludo2 = Saludo()
24 saludo2.mensaje = "Buenos días"
25 saludo2.tipo = "formal"
26
27 saludo3 = Saludo()
28 saludo3.mensaje = "Que hay"
29 saludo3.tipo= "informal"
30
31 saludo4 = Saludo()
32 saludo4.mensaje = "Buenas noches"
33 saludo4.tipo = "formal"
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Base de datos

```
34  
35 # se agrega los objetos de tipo Saludo  
36 # a la sesión  
37 # a la espera de un commit  
38 # para agregar un registro a la base de-  
39 # datos demobase2.db  
40 session.add(saludo1)  
41 session.add(saludo2)  
42 session.add(saludo3)  
43 session.add(saludo4)  
44  
45 # se confirma las transacciones  
46 session.commit()
```

- Se generan varios archivos de Python, que permita realizar diversos tipos de consultas.
 - Selección de todos los registros de la clase Saludo.

Archivos de Python

```
1 from sqlalchemy import create_engine  
2 from sqlalchemy.orm import sessionmaker  
3  
4 # se importa la clase(s) del-  
5 # archivo genera_tablas  
6 from genera_tablas import Saludo  
7  
8 # se genera enlace al gestor de base de  
9 # datos  
10 # para el ejemplo se usa la base de datos  
11 # sqlite  
12 engine = create_engine('sqlite:///demobase2.db')  
13  
14 Session = sessionmaker(bind=engine)  
15 session = Session()
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Archivos de Python

```
16
17 # Obtener todos los registros de-
18 # la tabla Saludo
19 losSaludos = session.query(Saludo).all()
20 # la consulta con .all(), devuelve
21 # una lista de objetos de tipo Saludo
22 # que se le asigna como valor a la variable
23 # losSaludos
24 # Se recorre la lista a través de un ciclo
25 # repetitivo for en python
26
27 for s in losSaludos:
28     print(s.mensaje)
29     print(s.tipo)
30     print("_____")
Hola que tal
informal
```

Buenos días
formal

Que hay
informal

Buenas noches
formal

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- Selección de todos los registros de la clase Saludo, ordenados en función del atributo mensaje.

Registros de la clase Saludo

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3
4 from genera_tablas import Saludo
5
6 # datos
7 # para el ejemplo se usa la base de datos
8 # sqlite
9 engine = create_engine('sqlite:///demobase2.db')
10
11 Session = sessionmaker(bind=engine)
12 session = Session()
13
14 # Obtener todos los registros de-
15 # la tabla saludo ordenados por el atribut
16 # mensaje
17 losSaludos = session.query(Saludo).order_by(Saludo.mensaje)
18 # la consulta con .order_by, ordena los resultados en función del
19 # atributo de la clase Saludo, mensaje;
20 # Devuelve una lista de objetos de tipo Saludo
21 # y se le asigna como valor a la variable
22 # losSaludos
23
24 # Se recorre la lista a través de un ciclo
25 # repetitivo for en python
26
27 for s in losSaludos:
28     print("Mensaje: %s" % (s.mensaje))
29     print("Tipo: %s" % (s.tipo))
30     print("id: %s" % (s.id))
31     print("_____")
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la clase Saludo

Mensaje: Buenas noches

Tipo: formal

id: 4

Mensaje: Buenos días

Tipo: formal

id: 2

Mensaje: Hola que tal

Tipo: informal

id: 1

Mensaje: Que hay

Tipo: informal

id: 3

- Obtener todos los registros de la tabla saludo que tengan como valor en el atributo **tipo** la expresión "formal".

Registros de la tabla saludo

```
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from genera_tablas import Saludo

# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase2.db')
Session = sessionmaker(bind=engine)
session = Session()

# Obtener todos los registros de
# la tabla saludo que tengan como valor en
# el atributo tipo la expresión "formal"
losSaludos = session.query(Saludo).filter(Saludo.tipo=="formal")
# la consulta con .filter con argumento
# Saludo.tipo=="formal"
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la tabla saludo

```
# Devuelve una lista de objetos de tipo Saludo  
# y se le asigna como valor a la variable  
# losSaludos  
# Se recorre la lista a través de un ciclo  
# repetitivo for en python  
for s in losSaludos:  
    print("Mensaje: %s" % (s.mensaje))  
    print("Tipo: %s" % (s.tipo))  
    print("id: %s" % (s.id))  
    print("_____")  
    Mensaje: Buenos días  
    Tipo: formal  
    id: 2
```

Mensaje: Buenas noches

Tipo: formal

id: 4

- Obtener todos los registros de la tabla saludo que tengan la vocal "o" en el atributo mensaje y sean de tipo "informal".

Registros de la tabla saludo

```
1 from sqlalchemy import create_engine  
2 from sqlalchemy.orm import sessionmaker  
3 from sqlalchemy import and_ # se importa el operador and  
4 from genera_tablas import Saludo  
5  
6 # datos  
7 # para el ejemplo se usa la base de datos  
8 # sqlite  
9 engine = create_engine('sqlite:///demobase2.db')  
10  
11 Session = sessionmaker(bind=engine)  
12 session = Session()  
13  
14 # Obtener todos los registros de-  
15 # la tabla saludo que tengan la vocal "o" en el atributo mensaje
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la tabla saludo

```
16 # y sean de tipo "informal"
17 losSaludos = session.query(Saludo).filter(and_(Saludo.mensaje.like("%o%"), \
18 Saludo.tipo=="informal"))
19 # a la consulta con .filter:
20 # se le agrega una expresión lógica AND
21 # la primera parte de la expresión usa el·
22 # operador like para determinar todos los registros
23 # que tengan la vocal "o" en el atributo mensaje;
24 # la segunda parte de la expresión, filtra todos·
25 # los registros que tienen como valor en el atributo tipo
26 # "informal"
27 # Devuelve una lista de objetos de tipo Saludo
28 # y se le asigna como valor a la variable
29 # losSaludos
30
31 # Se recorre la lista a través de un ciclo
32 # repetitivo for en python
33
34 for s in losSaludos:
35     print("Mensaje: %s" % (s.mensaje))
36     print("Tipo: %s" % (s.tipo))
37     print("id: %s" % (s.id))
38     print("_____")
Mensaje: Hola que tal
Tipo: informal
id: 1
_____
```

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

Se puede generar muchos ejemplos, revisar la referencia de manejo de Querying⁵³ en la página oficial de SQLAlchemy.

⁵³ <https://docs.sqlalchemy.org/en/13/orm/tutorial.html>



Semana 6

3.4. Eliminación de datos con ORM SQLAlchemy

La eliminación de registros a través del ORM del SqlAlchemy se lo realiza mediante el uso de la expresión **delete** asociada a la sesión creada en el proceso.

Ejemplo

- Crear una estructura que permita manipular docentes con las características como: nombre, apellido, ciudad.

Ejemplo

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.ext.declarative import declarative_base
3 from sqlalchemy.orm import sessionmaker, relationship
4 from sqlalchemy import Column, Integer, String, ForeignKey
5
6 # se importa información del archivo configuración
7 from configuracion import cadena_base_datos
8
9 # se genera enlace al gestor de base de
10 # datos
11 # para el ejemplo se usa la base de datos
12 # sqlite
13 engine = create_engine(cadena_base_datos)
14
15 Base = declarative_base()
16
17 class Docente(Base):
18     __tablename__ = 'docentes'
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo

```
19 id = Column(Integer, primary_key=True)
20 nombre = Column(String)
21 apellido = Column(String)
22 ciudad = Column(String, nullable=False)
23 ....
24 def __repr__(self):
25     return "Docente: nombre=%s apellido=%s ciudad:%s" %
26             self.nombre,
27             self.apellido,
28             self.ciudad)
29
30 Base.metadata.create_all(engine)
```

- Ingresar información.

Ejemplo

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3
4 # se importa la clase(s) del-
5 # archivo genera_tablas
6 from genera_tablas import Docente
7
8 # se importa información del archivo configuracion
9 from configuracion import cadena_base_datos
10 # se genera enlace al gestor de base de
11 # datos
12 # para el ejemplo se usa la base de datos
13 # sqlite
14 engine = create_engine(cadena_base_datos)
15
16 Session = sessionmaker(bind=engine)
17 session = Session()
18
19 # se crea un objetos de tipo Docente-
20 docente1 = Docente(nombre="Tony", apellido="García", \
21                     ciudad="Loja")
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo

```
22  
23 docente2 = Docente(nombre="Luis", apellido="Borrero", \  
24     ciudad="Loja")  
25  
26 docente3 = Docente(nombre="Ana", apellido="Salcedo", \  
27     ciudad="Zamora")  
28  
29 docente4 = Docente(nombre="Monica", apellido="Valenzuela", \  
30     ciudad="Zamora")  
31  
32 # se agrega los objetos  
33 # a la sesión  
34 # a la espera de un commit  
35 # para agregar un registro a la base de-  
36 # datos  
37 session.add(docente1)  
38 session.add(docente2)  
39 session.add(docente3)  
40 session.add(docente4)  
41  
42 # se confirma las transacciones  
43 session.commit()  
Presentación de Docentes  
Docente: nombre=Tony apellido=García ciudad:Loja  
  
_____  
Docente: nombre=Luis apellido=Borrero ciudad:Loja  
  
_____  
Docente: nombre=Ana apellido=Salcedo ciudad:Zamora  
  
_____  
Docente: nombre=Monica apellido=Valenzuela ciudad:Zamora  
_____
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

- Eliminar los docentes que tengan el apellido igual a "Valenzuela".

Ejemplo

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3 from sqlalchemy import and_ # se importa el operador and
4
5 # se importa la clase(s) del-
6 # archivo genera_tablas
7 from genera_tablas import Docente
8
9 # se importa información del archivo configuracion
10 from configuracion import cadena_base_datos
11 # se genera enlace al gestor de base de
12 # datos
13 # para el ejemplo se usa la base de datos
14 # sqlite
15 engine = create_engine(cadena_base_datos)
16
17
18 Session = sessionmaker(bind=engine)
19 session = Session()
20
21 # Obtener todos los registros de-
22 # la entidad docentes
23 docentes = session.query(Docente).all()
24
25 # Se recorre la lista a través de un ciclo
26 # repetitivo for en python
27 print("Presentación de Docentes")
28 for s in docentes:
29     print("%s" % (s))
30     print("_____")
31
32 # Eliminar datos o registros de los docentes
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo

```
33 # que tiene como apellido "Valenzuela"
34 docentes_eliminar = session.query(Docente).filter(Docente.apellido=="Valenzuela")\
35 .all()
36
37 for d in docentes_eliminar:
38 # se agrega a la sesión los elementos que
39 # se quiere eliminar, a través de
40 # session.delete()
41 session.delete(d)
42
43 # se confirma las transacciones
44 session.commit()
45
46 # Obtener todos los registros de-
47 # la entidad docentes
48 docentes = session.query(Docente).all()
49
50 # Se recorre la lista a través de un ciclo
51 # repetitivo for en python
52 print("Presentación de Docentes luego de eliminar")
53 for s in docentes:
54 print("%s" % (s))
55 print("_____")
56
```

Presentación de Docentes

Docente: nombre=Tony apellido=García ciudad:Loja

Docente: nombre=Luis apellido=Borrero ciudad:Loja

Docente: nombre=Ana apellido=Salcedo ciudad:Zamora

Docente: nombre=Monica apellido=Valenzuela ciudad:Zamora

Presentación de Docentes luego de eliminar

Docente: nombre=Tony apellido=García ciudad:Loja

Docente: nombre=Luis apellido=Borrero ciudad:Loja

Docente: nombre=Ana apellido=Salcedo ciudad:Zamora

En la **Figura 29**, se puede visualizar los cambios en la base de datos.

	id	nombre	apellido	ciudad
1	1	Tony	Garcia	Loja
2	2	Luis	Borrero	Loja
3	3	Ana	Salcedo	Zamora
4	4	Monica	Valenzuela	Zamora

	id	nombre	apellido	ciudad
1	1	Tony	Garcia	Loja
2	2	Luis	Borrero	Loja
3	3	Ana	Salcedo	Zamora

Figura 29. Visualización de registros de la tabla docentes

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

Usted puede revisar información del proceso de eliminación a través de la expresión `delete` en SQLAlchemy en [Query.delete](#)⁵⁴

3.5. Relaciones de entidades con ORM SQLAlchemy

A través del ORM de SQLAlchemy se puede crear las relaciones entre clases, de la misma forma que se hace cuando se diseña e implementa una base de datos con lenguaje SQL:

⁵⁴ <https://docs.sqlalchemy.org/en/13/orm/query.html?highlight=delete#sqlalchemy.orm.query.Query.delete>

- Uno a uno.
- Uno a muchos.
- Muchos a muchos.

Ejemplo 1

Dadas las entidades Estudiante y Número Telefónico, que poseen las siguientes características:

- Estudiante:
 - nombre.
 - apellido.
 - cedula.
- NumeroTeléfono.
 - numero_telefónico.
 - tipo.
 - estudiante.

Entre las dos entidades existe una **relación de uno a muchos**; un estudiante puede tener muchos números telefónicos; para el ejemplo propuesto se necesita agregar una **llave foránea** en la entidad **NúmeroTeléfono**.

La generación de las entidades en Python usando el ORM de SQLAlchemy es de la siguiente forma:

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Ejemplo 1

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.ext.declarative import declarative_base
3 from sqlalchemy.orm import sessionmaker, relationship
4 from sqlalchemy import Column, Integer, String, ForeignKey
5
6 # se importa información del archivo configuracion
7 from configuracion import cadena_base_datos
8
9 # se genera enlace al gestor de base de
10 # datos
11 # para el ejemplo se usa la base de datos
12 # sqlite
13 engine = create_engine(cadena_base_datos)
14
15 Base = declarative_base()
16
17 class Estudiante(Base):
18     __tablename__ = 'estudiantes'
19     id = Column(Integer, primary_key=True)
20     nombre = Column(String)
21     apellido = Column(String)
22     cedula = Column(String, nullable=False)
23
24     def __repr__(self):
25         return "Estudiante: nombre=%s apellido=%s cedula=%s" % (
26             self.nombre,
27             self.apellido,
28             self.cedula)
29
30 class NumeroTelefonico(Base):
```

Ejemplo 1

```

31 __tablename__ = 'numerotelefonicos'
32 id = Column(Integer, primary_key=True)
33 numero_telefonico = Column(String, nullable=False)
34 tipo = Column(String)
35 estudiante_id = Column(Integer, ForeignKey('estudiantes.id'))
36 estudiante = relationship("Estudiante", back_populates="numerostelefonicos")
37 ....
38 def __repr__(self):
39     return "Número Telefónico %s" % (self.numero_telefonico)
40
41 Estudiante.numerostelefonicos = relationship("NumeroTelefonico", \
42     back_populates="estudiante")
43
44 Base.metadata.create_all(engine)

```

Se explica las líneas más importantes:

- En la línea 35 se crea un atributo **estudiante_id**, que será una columna de tipo entero; se le agrega la directiva **ForeignKey** que indica que los valores en dicha columna deben estar presentes en la columna de la entidad que se pasa como parámetro, para el ejemplo '**estudiantes.id**' (**entidad estudiantes, columna id**). Se crea una relación entre numerotelefonicos. **estudiante_id** y **estudiantes.id**
- Se usa la directiva **relationship** en la línea 36 para indicarle al ORM que la clase **NumeroTelefonico** está relacionada con **Estudiante** a través de **NumeroTelefonico.estudiante**.
- En la línea 41 se agrega una directiva **relationship** que permite representar la relación desde la clase **Estudiantes** con sus números telefónicos.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Name	Type	Schema
Tables (2)		
estudiantes		CREATE TABLE estudiantes (id INTEGER NOT NULL, nombre VARCHAR, apellido VARCHAR, cedula VARCHAR NOT NULL)
id	INTEGER	'id' INTEGER NOT NULL
nombre	VARCHAR	'nombre' VARCHAR
apellido	VARCHAR	'apellido' VARCHAR
cedula	VARCHAR	'cedula' VARCHAR NOT NULL
numerotelefonicos		CREATE TABLE numerotelefonicos (id INTEGER NOT NULL, numero_telefono VARCHAR NOT NULL, tipo VARCHAR)
id	INTEGER	'id' INTEGER NOT NULL
numero_telefo...	VARCHAR	'numero_telefono' VARCHAR NOT NULL
tipo	VARCHAR	'tipo' VARCHAR
estudiante_id	INTEGER	'estudiante_id' INTEGER
Indices (0)		
Views (0)		
Triggers (0)		

Figura 30. Tablas generadas a través de la relación expuesta entre las clases

El ingreso de información o registros a las tablas se lo describe en las siguientes líneas:

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registro a las tablas

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3
4 # se importa la clase(s) del-
5 # archivo genera_tablas
6 from genera_tablas import Estudiante, NumeroTelefonico
7
8 # se importa información del archivo configuracion
9 from configuracion import cadena_base_datos
10 # se genera enlace al gestor de base de
11 # datos
12 # para el ejemplo se usa la base de datos
13 # sqlite
14 engine = create_engine(cadena_base_datos)
15
16 Session = sessionmaker(bind=engine)
17 session = Session()
18
19 # se crea un objetos de tipo Estudiante-
20 estudiante1 = Estudiante(nombre="Tony", apellido="García", \
21     cedula="123456789")
22 estudiante2 = Estudiante(nombre="Annette", apellido="García", \
23     cedula="223456789")
24 estudiante3 = Estudiante(nombre="David", apellido="Phillips", \
25     cedula="323456789")
26
27 # Se crean objeto de tipo NumeroTelefonico
28 # con sus propiedades
29 # numero_telefonico
30 # tipo
31 # adicional a cada objeto se le agrega
32 # un objeto de tipo Estudiante para el-
33 # atributo estudiante
34 # que representa la relación
35 #
36 telefono1 = NumeroTelefonico(numero_telefonico="0912345678", \
37     tipo="personal", \
38 estudiante=estudiante1)
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registro a las tablas

```
39 telefono2 = NumeroTelefonico(numero_telefonico="0212345678", \
40     tipo="domicilio";
41     estudiante=estudiante1)
42 telefono3 = NumeroTelefonico(numero_telefonico="0942345678", \
43     tipo="personal";
44     estudiante=estudiante2)
45 telefono4 = NumeroTelefonico(numero_telefonico="0342345678", \
46     tipo="domicilio";
47     estudiante=estudiante2)
48 telefono5 = NumeroTelefonico(numero_telefonico="0952345678", \
49     tipo="personal";
50     estudiante=estudiante3)
51
52
53
54 # se agrega los objetos
55 # a la sesión
56 # a la espera de un commit
57 # para agregar un registro a la base de
58 # datos
59 session.add(estudiante1)
60 session.add(estudiante2)
61 session.add(estudiante3)
62 session.add(telefono1)
63 session.add(telefono2)
64
65 # se confirma las transacciones
66 session.commit()
```

En la base de datos ya se puede verificar la información ingresada y debidamente relacionada. Figura 31.

The figure shows two database tables in a SQLite interface. The first table, 'estudiantes', has columns 'id', 'nombre', 'apellido', and 'cedula'. It contains three rows with data: (1, Tony, Garcia, 123456789), (2, Annette, Garcia, 223456789), and (3, David, Phillips, 323456789). The second table, 'numerotelefonicos', has columns 'id', 'numero_telefonico', 'tipo', and 'estudiante_id'. It also contains three rows with data: (1, 0912345678, personal, 1), (2, 0212345678, domicilio, 1), and (3, 0942345678, personal, 2). Red circles highlight the values 1, 2, and 3 in both tables, corresponding to the student IDs.

Figura 31. Registros en la base de datos, luego de ejecuta los procesos de inserción respectivos para la tabla estudiantes y numerostelefonicos

La consulta de información se describe a continuación

- Obtener un listado de todos los registros de las tablas estudiantes y numerotelefonicos. Comentario: no está muy claro si se ha usado el infinitivo “obtener” como parte de una instrucción. Por esta razón, se ha decidido mantenerlo y no cambiarlo a la segunda persona del singular.

Registros de las tablas estudiantes y numerotelefonicos

```

1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3 from sqlalchemy import and_ # se importa el operador and
4
5 # se importa la clase(s) del-
6 # archivo genera_tablas
7 from genera_tablas import Estudiante, NumeroTelefonico
8
9 # se importa información del archivo configuracion
10 from configuracion import cadena_base_datos
11 # se genera enlace al gestor de base de
12 # datos
13 # para el ejemplo se usa la base de datos
14 # sqlite
15 engine = create_engine(cadena_base_datos)
16
17
18 Session = sessionmaker(bind=engine)

```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de las tablas estudiantes y numerotelefonicos

```
19 session = Session()  
20  
21 # Obtener todos los registros de-  
22 # la entidad estudiantes (clase Estudiante)-  
23 estudiantes = session.query(Estudiante).all()  
24  
25 # Se recorre la lista a través de un ciclo  
26 # repetitivo for en python  
27 print("Presentación de Estudiantes")  
28 for s in estudiantes:  
29     print("%s" % (s))  
30     print("_____")  
31  
32 # Obtener todos los registros de-  
33 # la entidad numerotelefonicos (clase NumeroTelefonico)-  
34 numeros_telefonicos = session.query(NumeroTelefonico).all()  
35  
36 # Se recorre la lista a través de un ciclo  
37 # repetitivo for en python  
38  
39 print("Presentación de Números Telefónicos")  
40 for s in numeros_telefonicos:  
41     print("%s" % (s))  
42     print("_____")  
43
```

Presentación de Estudiantes

Estudiante: nombre=Tony apellido=García cedula=123456789

Estudiante: nombre=Annette apellido=García cedula=223456789

Estudiante: nombre=David apellido=Phillips cedula=323456789

Presentación de Números Telefónicos

Número Telefónico 0912345678

Número Telefónico 0212345678

Número Telefónico 0942345678

Número Telefónico 0342345678

Número Telefónico 0952345678



- Obtener un listado de todos los registros de la tabla estudiantes; por cada registro (objeto tipo Estudiante), presentar sus números telefónicos.

Registros de la tabla estudiantes

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3 from sqlalchemy import and_ # se importa el operador and
4
5 # se importa la clase(s) del-
6 # archivo genera_tablas
7 from genera_tablas import Estudiante, NumeroTelefonico
8
9 # se importa información del archivo configuracion
10 from configuracion import cadena_base_datos
11 # se genera enlace al gestor de base de
12 # datos
13 # para el ejemplo se usa la base de datos
14 # sqlite
15 engine = create_engine(cadena_base_datos)
16
17
18 Session = sessionmaker(bind=engine)
19 session = Session()
20
21 # Obtener todos los registros de-
22 # la entidad estudiantes (clase Estudiante)-
23 estudiantes = session.query(Estudiante).all()
24
25 # Se recorre la lista a través de un ciclo
26 # repetitivo for en python
27 print("Presentación de Estudiantes")
28 for s in estudiantes:
29     print("%s" % (s))
30 # desde cada objeto de la lista
31 # estudiantes
32 # se puede acceder a sus número telefónicos
33 # haciendo uso de la relación
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la tabla estudiantes

34 # Importante

35 # s.numerostelefonicos es un lista de

36 # número telefónicos

37 # Se hace uso de un ciclo repetitivo para

38 # la presentación

39 for t in s.numerostelefonicos:

40 print("\t%s" % (t))

41 print("_____")

Presentación de Estudiantes

Estudiante: nombre=Tony apellido=García cedula=123456789

Número Telefónico 0912345678

Número Telefónico 0212345678

Estudiante: nombre=Annette apellido=García cedula=223456789

Número Telefónico 0942345678

Número Telefónico 0342345678

Estudiante: nombre=David apellido=Phillips cedula=323456789

Número Telefónico 0952345678

- Generar

- Obtener un listado de todos los registros de la tabla estudiantes, que tengan al menos un número telefónico con la cadena "091";
- Obtener un listado de todos los registros de la tabla numerostelefonicos, que tengan en su atributo numero_telefonico la cadena "091" y
- Obtener un listado de todos los registros de la tabla estudiantes, que tengan al menos un número telefónico con la cadena "091" (opción 2)

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la tabla estudiantes

```
1 from sqlalchemy import create_engine
2 from sqlalchemy.orm import sessionmaker
3 from sqlalchemy import and_ # se importa el operador and
4
5 # se importa la clase(s) del-
6 # archivo genera_tablas
7 from genera_tablas import Estudiante, NumeroTelefonico
8
9 # se importa información del archivo configuracion
10 from configuracion import cadena_base_datos
11 # se genera enlace al gestor de base de
12 # datos
13 # para el ejemplo se usa la base de datos
14 # sqlite
15 engine = create_engine(cadena_base_datos)
16
17
18 Session = sessionmaker(bind=engine)
19 session = Session()
20
21 # Obtener un listado de todos los registros-
22 # de la tabla estudiantes, que tengan al menos-
23 # un número telefónico con la cadena "091"
24
25 # para la solución se hace uso del método-
26 # join aplicado a query
27
28 estudiantes = session.query(Estudiante).join(NumeroTelefonico).
filter(NumeroTelefonico.numero_telefonico.like("091%")).all()
29
30 print("Consulta 1 ")
31 for e in estudiantes:
32     print(e)
33     for t in e.numerostelefonicos:
34         print(t)
35
36 # obtener un listado de todos los registros-
37 # de la tabla numerostelefonicos, que tengan-
38 # en su atributo numero_telefonico la-
39 # cadena "091".
```



Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Registros de la tabla estudiantes

```
40  
41 numeros_telefonicos = session.query(NumerosTelefonico).  
filter(NumerosTelefonico.numero_telefonico.like("091%")).all()  
42  
43 print("Consulta 2 ")  
44 for t in numeros_telefonicos:  
45     print("%s - %s" % (t, t.estudiante))  
46  
47 # Obtener un listado de todos los registros:  
48 # de la tabla estudiantes y numerostelefonicos, que tengan al menos:  
49 # un número telefónico con la cadena "091"  
50  
51 # para la solución se hace uso del método:  
52 # join aplicado a query  
53 # en el query se ubican las dos entidades involucradas  
54 #  
55  
56 estudiantes = session.query(Estudiante, NumeroTelefonico).  
join(NumeroTelefonico).filter(NumeroTelefonico.numero_telefonico.like("091%")).  
all()  
57  
58 print("Consulta 3 ")  
59  
60 for registro in estudiantes:  
61     # el registro contiene:  
62     # dos valores en un tupla  
63     # posición 0 el estudiante  
64     # posición 1 el número telefónico  
65     # que cumplen con la condición  
66     print(registro[0])  
67     print(registro[1])
```

Registros de la tabla estudiantes

Consulta 1

Estudiante: nombre=Tony apellido=García cedula=123456789

Número Telefónico 0912345678

Número Telefónico 0212345678

Consulta 2

Número Telefónico 0912345678 - Estudiante: nombre=Tony apellido=García cedula=123456789

Consulta 3

Estudiante: nombre=Tony apellido=García cedula=123456789

Número Telefónico 0912345678

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

Finalmente, usted puede revisar los siguientes enlaces para analizar ejemplos del manejo de información a través de:

- QueryAPI⁵⁵ de SQLAlchemy
- Querying with Joins⁵⁶ en SQLAlchemy



Actividad de aprendizaje recomendada

Actividad 1

Actividad de aprendizaje: Realice la autoevaluación 3, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 3 Acceso a base de datos relacionales

⁵⁵ <https://docs.sqlalchemy.org/en/13/orm/query.html#sqlalchemy.orm.query.Query.filter>

⁵⁶ <https://docs.sqlalchemy.org/en/13/orm/tutorial.html>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

mediante Object Relational Mapper (ORM). En la parte inicial de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.



Autoevaluación 3

Es momento de medir el entendimiento de algunos conceptos estudiados en la Unidad 3. Acceso a base de datos relacionales mediante Object Relational Mapper (ORM). Se solicita revisar:

- La unidad 3 de la presente guía
- Recurso educativo abierto: Learning sqlalchemy

La autoevaluación le permitirá reforzar los conceptos estudiados. Los enunciados tienen una sola opción correcta. Lea atentamente cada respuesta y seleccione la opción que usted considere correcta.

1. De acuerdo al concepto de un ORM, ¿cuál de las siguientes afirmaciones es correcta?

- a. Pasar objetos de clases realizadas en lenguajes de programación a registros en base de datos relacionales.
- b. Pasar objetos de clases realizadas en lenguajes de programación a registros en archivos planos como txt y csv.
- c. Pasar objetos de clases abstractas realizadas en lenguajes de programación a registros en base de datos relacionales.

2. ¿Cuál de las siguientes afirmaciones es verdadera, respecto a los ORM?

- a. Procesos de desarrollo lentos.
- b. Con el uso de ORM se puede cambiar de motor de base datos en cualquier momento.
- c. ORM no impide la ejecución de inyecciones SQL.

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

- 3. Identifique un ORM usado bajo el lenguaje de programación PHP**
 - a. Ebean.
 - b. Doctrine.
 - c. SqlAlchemy.

- 4. ¿Cuál de las siguientes opciones es correcta para la instalación de la librería SqlAlchemy en un entorno local?**
 - a. Pip install SqlAlchemy.
 - b. Pip SqlAlchemy.
 - c. Python install SqlAlchemy.

- 5. ¿Cuál de las siguientes sentencias permiten conectarse a una base de datos SQLite haciendo uso de la librería SqlAlchemy?**
 - a. Engine = create_engine('sqlite:///demobase.db').
 - b. Engine = createEngine('sqlite:///demobase.db').
 - c. Engine = create('sqlite:///demobase.db').

6. Identique la sentencia que permite crear las tablas en la base de datos en función de las clases de Python previamente creadas, haciendo uso de la librería SQLAlchemy.

- a.

```
from sqlalchemy.ext.declarative import declarative_base
engine = create_engine(<<cadena_base_datos>>)
Base = declarative_base()
Base.metadata.create_all(engine)
```
- b.

```
from sqlalchemy.ext.declarative import declarative_base
engine = create_engine(<<cadena_base_datos>>)
Base = declarative_base()
Base.metadata.create_all()
```
- c.

```
from sqlalchemy.ext.declarative import declarative_base
engine = create_engine(<<cadena_base_datos>>)
Base = declarative_base()
Base.metadata.create_all()
```

7. Revise las siguientes líneas de código, identifique la salida que se generaría al ejecutar el archivo `consulta_datos.py`.

Líneas de código

```
# genera_tablas.py
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
from sqlalchemy import Column, Integer, String, ForeignKey
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Base = declarative_base()
class Hospital(Base):
    __tablename__ = 'estudiantes'
    id = Column(Integer, primary_key=True)
    nombre = Column(String)
    numero_camas = Column(Integer)
    numero pisos = Column(Integer)
    def __repr__(self):
        return "Hospital: %s - camas: %d - pisos: %d" % (
            self.nombre,
            self.numero_camas,
            self.numero pisos)
Base.metadata.create_all(engine)

# genera_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
h1 = Hospital(nombre="Isidro Ayora", numero_camas=100, \
    numero pisos=10)
h2 = Hospital(nombre="Andrade Marín", numero_camas=200, \
```

Líneas de código

```
numero pisos=20)
h3 = Hospital(nombre="Machala", numero_camas=80, \
    numero_pisos=8)
h4 = Hospital(nombre="Luis Vernaza", numero_camas=80, \
    numero_pisos=10)
# se agrega los objetos
# a la sesión
# a la espera de un commit
# para agregar un registro a la base de
# datos
session.add(h1)
session.add(h2)
session.add(h3)
session.add(h4)
# se confirma las transacciones
session.commit()

consulta_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy import and_ # se importa el operador and
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
# Obtener todos los registros de
# la entidad estudiantes (clase Estudiante)
datos = session.query(Hospital).filter(Hospital.numero_camas==80).all()
# Se recorre la lista a través de un ciclo
# repetitivo for en python
print("Presentación de Hospitales")
for s in datos:
    print("%s" % (s))
    print("_____")
```

Respuestas

- a. Presentación de Hospitales

Hospital: Machala - camas: 80 - pisos: 8

Hospital: Luis Vernaza - camas: 80 - pisos: 8

- b. Presentación de Hospitales

Hospital: Isidro Ayora - camas: 80 - pisos: 8

Hospital: Luis Vernaza - camas: 80 - pisos: 10

- c. Presentación de Hospitales

Hospital: Machala - camas: 80 - pisos: 8

Hospital: Luis Vernaza - camas: 80 - pisos: 10

8. Revise las siguientes líneas de código, identifique la salida que se generaría al ejecutar el archivo consulta_datos.py

Líneas de código

```
# genera_tablas.py
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
from sqlalchemy import Column, Integer, String, ForeignKey
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Base = declarative_base()
class Hospital(Base):
    __tablename__ = 'estudiantes'
    id = Column(Integer, primary_key=True)
    nombre = Column(String)
    numero_camas = Column(Integer)
    numero_pisos = Column(Integer)
def __repr__(self):
    return "Hospital: %s - camas: %d - pisos: %d" % (
        self.nombre,
        self.numero_camas,
        self.numero_pisos)
Base.metadata.create_all(engine)

# genera_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
h1 = Hospital(nombre="Isidro Ayora", numero_camas=100, \
    numero pisos=10)
h2 = Hospital(nombre="Andrade Marín", numero_camas=200, \
    numero pisos=20)
h3 = Hospital(nombre="Machala", numero_camas=80, \
    numero pisos=8)
h4 = Hospital(nombre="Luis Vernaza", numero_camas=80, \
    numero pisos=10)
# se agrega los objetos
# a la sesión
# a la espera de un commit
# para agregar un registro a la base de
# datos
session.add(h1)
session.add(h2)
session.add(h3)
session.add(h4)
# se confirma las transacciones
session.commit()

consulta_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy import and_ # se importa el operador and
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
# Obtener todos los registros de
datos = session.query(Hospital).order_by(Hospital.numero_pisos).all()
# Se recorre la lista a través de un ciclo
# repetitivo for en python
print("Presentación de Hospitales")
for s in datos:
    print("%s" % (s))
    print("_____")
```



Respuestas

a. Presentación de Hospitales

Hospital: Andrade Marín - camas: 200 - pisos: 20

Hospital: Isidro Ayora - camas: 100 - pisos: 10

Hospital: Luis Vernaza - camas: 80 - pisos: 10

Hospital: Machala - camas: 80 - pisos: 8

b. Presentación de Hospitales

Hospital: Isidro Ayora - camas: 100 - pisos: 10

Hospital: Andrade Marín - camas: 200 - pisos: 20

Hospital: Machala - camas: 80 - pisos: 8

Hospital: Luis Vernaza - camas: 80 - pisos: 10

c. Presentación de Hospitales

Hospital: Machala - camas: 80 - pisos: 8

Hospital: Isidro Ayora - camas: 100 - pisos: 10

Hospital: Luis Vernaza - camas: 80 - pisos: 10

Hospital: Andrade Marín - camas: 200 - pisos: 20

9. Revise las siguientes líneas de código; identifique la salida que se generaría al ejecutar el archivo consulta_datos.py

Líneas de código

```
# genera_tablas.py
from sqlalchemy import create_engine
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
from sqlalchemy import Column, Integer, String, ForeignKey
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Base = declarative_base()
class Hospital(Base):
    __tablename__ = 'estudiantes'
    id = Column(Integer, primary_key=True)
    nombre = Column(String)
    numero_camas = Column(Integer)
    numero_pisos = Column(Integer)
def __repr__(self):
    return "Hospital: %s - camas: %d - pisos: %d" % (
        self.nombre,
        self.numero_camas,
        self.numero_pisos)
Base.metadata.create_all(engine)

# genera_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
h1 = Hospital(nombre="Isidro Ayora", numero_camas=100, \
    numero_pisos=10)
h2 = Hospital(nombre="Andrade Marín", numero_camas=200, \
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
numero pisos=20)
h3 = Hospital(nombre="Machala", numero_camas=80, \
    numero_pisos=8)
h4 = Hospital(nombre="Pedro Ayora", numero_camas=80, \
    numero_pisos=10)
# se agrega los objetos
# a la sesión
# a la espera de un commit
# para agregar un registro a la base de
# datos
session.add(h1)
session.add(h2)
session.add(h3)
session.add(h4)
# se confirma las transacciones
session.commit()

consulta_datos.py
from sqlalchemy import create_engine
from sqlalchemy.orm import sessionmaker
from sqlalchemy import and_ # se importa el operador and
# se importa la clase(s) del
# archivo genera_tablas
from genera_tablas import Hospital
# se genera enlace al gestor de base de
# datos
# para el ejemplo se usa la base de datos
# sqlite
engine = create_engine('sqlite:///demobase3.db')
Session = sessionmaker(bind=engine)
session = Session()
# Obtener todos los registros de
datos = session.query(Hospital).filter(Hospital.nombre.like("%Ay%")).all()
# Se recorre la lista a través de un ciclo
# repetitivo for en python
print("Presentación de Hospitales")
for s in datos:
    print("%s" % (s))
    print("_____")
```

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Respuestas

- a. Presentación de Hospitales

Hospital: Andrade Marín - camas: 200 - pisos: 20

Hospital: Isidro Ayora - camas: 100 - pisos: 10

- b. Presentación de Hospitales

Hospital: Isidro Ayora - camas: 100 - pisos: 10

Hospital: Pedro Ayora - camas: 80 - pisos: 10

- c. Presentación de Hospitales

Hospital: Machala - camas: 80 - pisos: 8

Hospital: Isidro Ayora - camas: 100 - pisos: 10

10. ¿Cuál de las sentencias siguientes permitiría identificar a un atributo de una clase como clave foránea?

- a. persona_id = Column(Integer, ForeignKey('persona.id'))
- b. persona_id = ForeignKe(Integer, Column('persona.id'))
- c. persona_id = Column(Integer, ForeignKey())

[Ir al solucionario](#)



Actividades finales del bimestre



Semana 7



Actividades de aprendizaje recomendadas

En la semana 7 de estudio se recomienda volver a revisar, analizar y contestar las preguntas de las autoevaluaciones del primer bimestre.

Actividad 1

- Actividad de aprendizaje: realice la autoevaluación 1 donde existen preguntas relacionadas a las temáticas descritas en la unidad 1 Conceptos generales de desarrollo de plataformas web. Se indican los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá, estimado estudiante, identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, se solicita la formulación de interrogantes o ejercicios similares para reafirmar lo analizado y estudiado.

Se recuerda que se puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Actividad 2

- Actividad de aprendizaje: considere realizar la autoevaluación 2, donde se exponen un conjunto de preguntas de la unidad 2 Programación del lado del cliente. La autoevaluación es fundamental para medir el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Luego de realizar la autoevaluación, la guía didáctica le permite revisar las respuestas correctas de las preguntas.

Actividad 3

- Actividad de aprendizaje: al momento de revisar la autoevaluación 3, usted está realizando un repaso importante de las temáticas descritas en la unidad 3 (Acceso a base de datos relacionales mediante Object Relational Mapper - ORM). La autoevaluación tiene como objetivo identificar el nivel de aprendizaje alcanzado, así como encontrar los ítems que se debe volver a revisar para lograr. Finalmente, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, se pide revisar las respuestas correctas de las preguntas en la sección solucionario.



Semana 8

La semana 8 es un período de tiempo importante destinado para que usted, estimado estudiante, pueda reafirmar temáticas del primer bimestre de estudio que aún están pendientes de comprender. Esto se realiza con el objetivo de prepararse para rendir la evaluación presencial del bimestre.

Por ello se sugiere el repaso de las siguientes unidades mediante la generación de resúmenes, cuadros sinópticos, mapas conceptuales, entre otros.

- Unidad 1. Conceptos Generales de desarrollo de plataformas web.
- Unidad 2. Programación del lado del cliente.
- Unidad 3: Acceso a base de datos relacionales mediante Object Relational Mapper (ORM).

Además, es importante realizar las siguientes acciones:

- Revisar los recursos de aprendizaje recomendados en el plan docente de la asignatura.
- Plantearse ejercicios y problemáticas similares a las expuestas en los contenidos de la guía.
- Revisar las preguntas que conforman los cuestionarios de repaso del bimestre planteados en el entorno virtual.

¿Estamos listos?

Avancemos...

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



Segundo bimestre

Resultado de aprendizaje 3

Aplica estándares web en el desarrollo de aplicaciones.

Contenidos, recursos y actividades de aprendizaje

El resultado de aprendizaje ayuda a comprender el desarrollo de aplicaciones web desde el punto de vista de lado del servidor mediante dos lenguajes de programación: PHP y Python. Además se estudiará el uso del patrón modelo-vista-controlador con el desarrollo de aplicaciones a través del framework de ambiente web Django.



Semana 9



Unidad 4. Programación en el servidor web

Estimados estudiantes, para el inicio del segundo bimestre vamos a revisar la información que brinda el texto básico a través del capítulo 5: Programación en el servidor web.



4.1. Desarrollo de aplicaciones con lenguaje PHP

Para el presente ítem de estudio se solicita la revisión del capítulo 5 del texto básico, sección *Preparación del entorno para el desarrollo web con PHP y sus subsecciones: Lenguajes del lado del servidor e Instalación del servidor WampServer*. Además, se toma como referencia la literatura de la sección *Introducción a la programación de aplicaciones web con lenguaje PHP*; la subsección *Aspectos básicos de la programación en PHP* que permiten analizar conceptos y ejemplos relacionados a tipos de datos, estructuras de control, manejo de arreglos y funciones en lenguaje PHP.

El texto básico recomienda el uso de la herramienta de código abierto denominado WampServer¹, misma que permite la instalación del lenguaje de programación PHP, gestor de base de datos MariaDB² y el servidor web Apache. Usted puede tomar otras opciones:

Opción 1

Instalar la herramienta XAMPP³ que permite descargas para sistemas operativos como: Windows⁴, GNU/Linux⁵, MacOS⁶.

Opción 2

Instalar las herramientas por separado en cada computador personal. En primer lugar instalar [PHP](#) ; luego el gestor de base de datos, ejemplo [MariaDB](#); y finalmente en relación a los servidores puede elegir entre [Apache](#) y [Nginx](#)

¹ <https://www.wampserver.com/en/>

² <https://mariadb.org/>

³ <https://www.apachefriends.org/es/index.html>

⁴ <https://www.microsoft.com/es-es/windows>

⁵ <https://www.gnu.org/gnu/linux-and-gnu.es.html>

⁶ <https://www.apple.com/la/macos/what-is/>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Se detalla a continuación algunos ejemplos en función de los descrito en el texto básico, en relación al desarrollo de aplicaciones con lenguaje PHP.

Ejemplo 1

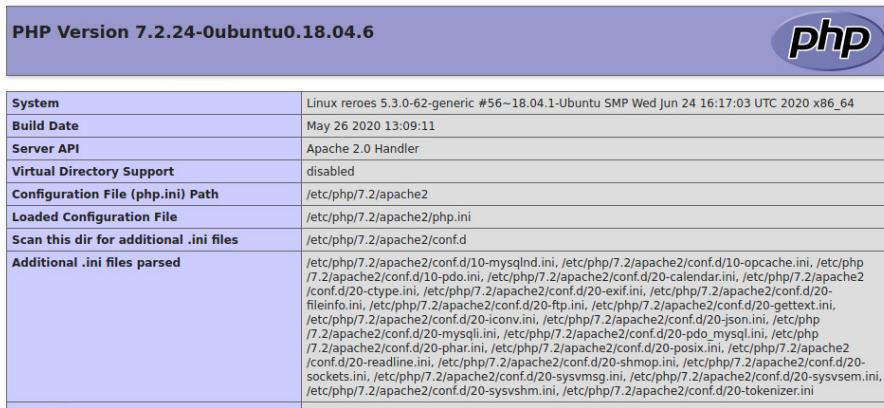
Verificación del funcionamiento del servidor con lenguaje PHP. Se recuerda que todo el código de PHP debe ir entre las etiquetas:

- <?php ?>

Ejemplo 1

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <meta name="viewport" content="width=device-width" />
6 <title>Verificación funcionamiento PHP</title>
7 </head>
8 <body>
9 <?php
10 ....
11 phpinfo();
12 ....
13 ?>
14 </body>
15 </html>
```

La salida debe ser como se muestra a continuación Figura 32:



```
PHP Version 7.2.24-0ubuntu0.18.04.6
php
```

System	Linux reroes 5.3.0-62-generic #56~18.04.1-Ubuntu SMP Wed Jun 24 16:17:03 UTC 2020 x86_64
Build Date	May 26 2020 13:09:11
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc/php/7.2/apache2
Loaded Configuration File	/etc/php/7.2/apache2/php.ini
Scan this dir for additional .ini files	/etc/php/7.2/apache2/conf.d
Additional .ini files parsed	/etc/php/7.2/apache2/conf.d/10-mysqlind.ini, /etc/php/7.2/apache2/conf.d/10-opcache.ini, /etc/php/7.2/apache2/conf.d/10-pdo.ini, /etc/php/7.2/apache2/conf.d/20-calendar.ini, /etc/php/7.2/apache2/conf.d/20-crypt.ini, /etc/php/7.2/apache2/conf.d/20-exif.ini, /etc/php/7.2/apache2/conf.d/20-finfo.ini, /etc/php/7.2/apache2/conf.d/20-ftp.ini, /etc/php/7.2/apache2/conf.d/20-gettext.ini, /etc/php/7.2/apache2/conf.d/20-iconv.ini, /etc/php/7.2/apache2/conf.d/20-json.ini, /etc/php/7.2/apache2/conf.d/20-mysqli.ini, /etc/php/7.2/apache2/conf.d/20-pdo_mysql.ini, /etc/php/7.2/apache2/conf.d/20-phar.ini, /etc/php/7.2/apache2/conf.d/20-posix.ini, /etc/php/7.2/apache2/conf.d/20-readline.ini, /etc/php/7.2/apache2/conf.d/20-shmop.ini, /etc/php/7.2/apache2/conf.d/20-sysvsem.ini, /etc/php/7.2/apache2/conf.d/20-sysvshm.ini, /etc/php/7.2/apache2/conf.d/20-tokenizer.ini

Figura 32. Verificación del funcionamiento del servidor con lenguaje PHP

- Estimado estudiante puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

Ejemplo 2

Generar un ejemplo que haga uso de:

- Concatenación de cadenas a través de sprintf⁷
- Uso de funciones
- Ciclos repetitivos y contadores
- Hojas de estilo

Se presenta el siguiente código

⁷ <https://www.php.net/manual/es/function.sprintf.php>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo 2

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <link rel="stylesheet" href="css/estilos.css" type="text/css" />
7     <title>Uso básico de PHP</title>
8   </head>
9   <body>
10
11   <table>
12     <tr>
13       <td>Multiplicando</td>
14       <td>Símbolo</td>
15       <td>Multiplicador</td>
16       <td>Resultado</td>
17     </tr>
18     <?php
19       // función en PHP
20       function operacionMultiplicacion($num1, $num2){
21         return $num1 * $num2;
22       }
23       $tabla = 5;
24       $inicio = 1;
25       echo '<h3>Tabla del '. $tabla, "</h3>";
26       // uso de ciclo repetitivo
27       while($inicio <= 12){
28         $formato = "<tr>
29           <td>%d</td><td>%s</td><td>%d</td><td class='colorcelda'>%d</td>
30         </tr>";
31         // concatenar cadenas a través de sprintf
32         echo sprintf($formato, $tabla, "*", $inicio,
33           operacionMultiplicacion($tabla, $inicio));
34         // contador
35         $inicio++;
36       }
37 ....
38     ?>
39   </table>
40 </body>
41 </html>
```

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

El resultado es, ver Figura 33.

Tabla del 5

Multiplicando	Símbolo	Multiplicador	Resultado
5	*	1	5
5	*	2	10
5	*	3	15
5	*	4	20
5	*	5	25
5	*	6	30
5	*	7	35
5	*	8	40
5	*	9	45
5	*	10	50
5	*	11	55
5	*	12	60

Figura 33. Uso básico de lenguaje PHP

En la subsección Formularios de la sección **Introducción a la programación de aplicaciones web** con lenguaje PHP del capítulo 5 del texto básico, se indica el uso de formularios mediante lenguaje PHP con la meta de lograr interacción entre el servidor y el usuario.

Ejemplo 3

En el siguiente ejemplo se evidencia el envío de parámetros entre dos páginas en lenguaje PHP, haciendo uso de un formulario mediante el método POST. Se solicita a través de un formulario nombre, apellido, la tabla de multiplicar y el límite de la tabla; con base en los valores ingresados, presentar el nombre, apellido y la tabla generada.

Se utiliza el siguiente código

Ejemplo 3

```
uno.php
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <link rel="stylesheet" href="css/estilos.css" type="text/css" />
7     <title>Uso básico de PHP con formularios</title>
8   </head>
9   <body>
10    <h2>Ingrese datos para generación</h2>
11    <div class="miformulario">
12      <form action="dos.php" method="post">
13        <label for="nombre">Ingrese su nombre</label>
14        <input type="text" name="nombre" id="nombre" required/>
15        <br/>
16        <br/>
17        <label for="apellido">Ingrese su apellido</label>
18        <input type="text" name="apellido" id="apellido" required/>
19        <br/>
20        <br/>
21        <label for="tabla">Ingrese la tabla a generar</label>
22        <input type="number" name="tabla" id="tabla" required/>
23        <br/>
24        <br/>
25        <label for="limite">Ingrese el límite</label>
26        <input type="number" name="limite" id="limite" required/>
```

Ejemplo 3

```
27      <br/>
28      <br/>
29      <input type="submit" name="enviar" id="enviar" value="Enviar" />
30      </form>
31  </div>
32 </body>
33 </html>
```

dos.php

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width" />
6   <link rel="stylesheet" href="css/estilos.css" type="text/css" />
7   <title>Uso básico de PHP</title>
8 </head>
9 <body>
10 <h3>Datos Ingresados</h3>
11 <?php
12 $nombre = $_REQUEST["nombre"];
13 $apellido = $_REQUEST["apellido"];
14 $datos = "<b>Nombre:</b>%s<br/><b>Apellido:</b>%s<br/>";
15 echo sprintf($datos, $nombre, $apellido) ;
16
17 ?>
18 <table>
19 <tr>
20   <td>Multiplicando</td>
21   <td>Símbolo</td>
22   <td>Multiplicador</td>
23   <td>Resultado</td>
24 </tr>
25 <?php
26 // función en PHP
27 function operacionMultiplicacion($num1, $num2){
28   return $num1 * $num2;
29 }
30 $tabla = intval($_REQUEST['tabla']);
31 $inicio = 1;
32 $limiteTabla = intval($_REQUEST['limite']);
```

Ejemplo 3

```

33 echo '<h3>Tabla del '. $tabla, "</h3>";
34 // uso de ciclo repetitivo
35 while($inicio <= $limiteTabla){
36     $formato = "<tr>
37         <td>%d</td><td>%s</td><td>%d</td><td class='colorcelda'>%d</td>
38     </tr>";
39     // concatenar cadenas a través de sprintf
40     echo sprintf($formato, $tabla, "*", $inicio,
41     operacionMultiplicacion($tabla, $inicio));
42     // contador
43     $inicio++;
44 }
45 ....
46 ?>
47 </table>
48 <a href="uno.php">Regresar</a>
49 </body>
50 </html>

```

La salida al ejecutar es la siguiente. Ver Figura 34:

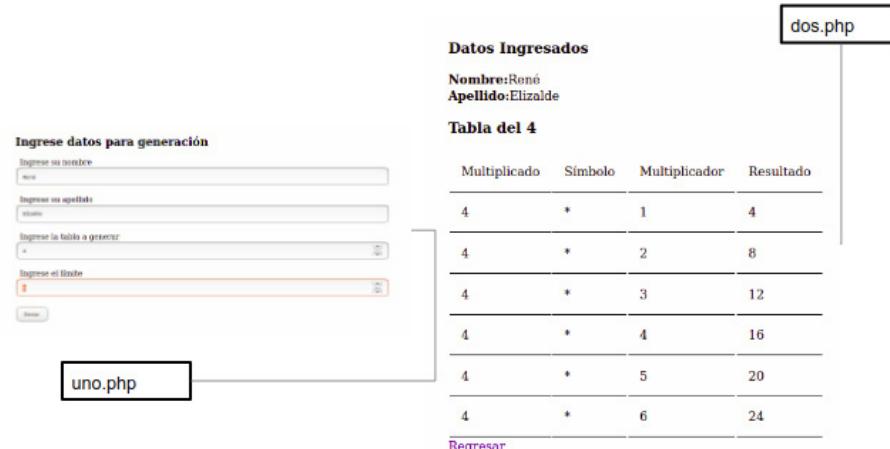


Figura 34. Interacción entre páginas con lenguaje PHP

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - [Desarrollo del ejemplo](#)



Semana 10

4.2. Acceso a base de datos relacional desde una aplicación en PHP

Señor estudiante, para el entendimiento del presente apartado es necesario realizar una lectura comprensiva del capítulo 5 del texto básico, Programación en el servidor web, sección Uso de bases de datos en programación web.

En el apartado mencionado se realiza una explicación que permite recordar conceptos, características y el proceso de acceso a bases de datos relacionales; además se explica el uso del lenguaje SQL para la generación de tablas, selección de información en función de un base de datos.

A continuación, se realiza un ejemplo que permita reforzar los conceptos abordados en los ítems del texto básico.

Ejemplo 1

Generar una entidad en base de datos para describir vehículos con las siguientes características: marca, placa, año fabricación, tipo (particular o público) y propietario. Se usa el gestor de base de datos MariaDB.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo 1

```
# Creación de la base de datos
create database proyectoejemplo;

# Creación de la tabla vehículos
create table `vehiculos` (
    `id` int(10) NOT NULL auto_increment primary key,
    `marca` varchar(100) not null,
    `placa` varchar(100) not null unique,
    `anio_fabricacion` int(100) not null,
    `tipo` varchar(100) not null,
    `propietario` varchar(100) not null
);

# Ingreso de datos a la tabla vehículos
insert into `vehiculos`(
    `marca`,
    `placa`,
    `anio_fabricacion`,
    `tipo`,
    `propietario`
) values (
    "chevrolet",
    "LCH0101",
    1990,
    "particular",
    "Patricio Rojas"
);
```

A través de una aplicación web generada a través del lenguaje PHP.

- Acceder a la base de datos y listar los vehículos ingresados.
- Agregar nuevos vehículos.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de Código

```
basedatos.php
1 <?php
2 // datos para enlace la base de datos
3 $servidor = "localhost";
4 $usuario = "susuario";
5 $password = "suclave";
6 $base_datos = "proyectoejemplo";
7 $conectaBD = new mysqli($servidor, $usuario, $password, $base_datos);
8 ?>

index.php
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="utf-8" />
5 <meta name="viewport" content="width=device-width" />
6 <link rel="stylesheet" href="css/estilos.css" type="text/css" />
7 <title>Uso básico de PHP</title>
8 </head>
9 <body>
10 <h3>Listado de Vehículos</h3>
11 <table>
12 <tr>
13 <td>Propietario</td>
14 <td>Placa</td>
15 <td>Marca</td>
16 <td>Año Fabricación</td>
17 <td>Tipo</td>
18 </tr>
19 <?php
20 include("basedatos.php");
21 function convertirMayuscula($dato){
22     // función que permite-
23     // pasar una cadena a mayúscula-
24     return strtoupper($dato);
25 }
26 // se realizar la consulta a la base de datos
27 $consultaBD = $conectaBD -> query("Select * from vehiculos");
28 while($registro = $consultaBD -> fetch_array(MYSQLI_ASSOC)){
29     $formato = "<tr>
30         <td>%s</td><td>%s</td><td>%s</td><td>%d</td><td>%s</td>
```

Líneas de Código

```

31      </tr>";
32 // se agrega una fila a la tabla HTML
33 echo sprintf($formato, convertirMayuscula($registro['propietario']),
34           convertirMayuscula($registro['placa']),
35           convertirMayuscula($registro['marca']),
36           convertirMayuscula($registro['anio_fabricacion']),
37           convertirMayuscula($registro['tipo']));
38 }
39 ?>
40 </table>
41 <a href="nuevo.php">Nuevo vehículo</a>
42 </body>
43 </html>
```

nuevo.php

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="utf-8" />
5   <meta name="viewport" content="width=device-width" />
6   <link rel="stylesheet" href="css/estilos.css" type="text/css" />
7   <title>Uso básico de PHP con formularios</title>
8 </head>
9 <body>
10  <?php
11  if($_GET['error']){
12    echo "<h5>". $_GET['error'] . "</h5>";
13  }
14 ?>
15  <h2>Ingrese datos vehículo</h2>
16  <div class="miformulario">
17    <form action="nuevo_agrega.php" method="post">
18      <label for="marca">Ingrese marca de vehículo</label>
19      <input type="text" name="marca" id="marca" required/>
20      <br/>
21      <br/>
22      <label for="placa">Ingrese placa del vehículo</label>
23      <input type="text" name="placa" id="placa" required/>
24      <br/>
25      <br/>
26      <label for="anio_fabricacion">Ingrese año de fabricación de vehículo</
label>
```

Líneas de Código

```

27      <input type="number" name="anio_fabricacion" id="anio_fabricacion"
required/>
28      <br/>
29      <br/>
30      <label for="tipo">Ingrese tipo de vehículo</label>
31      <input type="text" name="tipo" id="tipo" required/>
32      <br/>
33      <br/>
34      <label for="proprietario">Ingrese nombres de propietario</label>
35      <input type="text" name="proprietario" id="proprietario" required/>
36      <br/>
37      <br/>
38
39      <input type="submit" name="enviar" id="enviar" value="Enviar" />
40  </form>
41  </div>
42 </body>
43 </html>
```

nuevo_agrega.php

```

1 <?php
2 // proceso para guardar un nuevo registro a la base de datos
3 include("basedatos.php");
4 $marca = $_REQUEST['marca'];
5 $placa = $_REQUEST['placa'];
6 $tipo = $_REQUEST['tipo'];
7 $proprietario = $_REQUEST['proprietario'];
8 $anio = intval($_REQUEST['anio_fabricacion']);
9 $cadena = "insert into `vehiculos`(`marca`, `placa`, `anio_fabricacion`,
10 `tipo`, `proprietario`) values ('%s', '%s', %d, '%s', '%s');";
11 $cadena = sprintf($cadena, $marca, $placa, $anio, $tipo, $proprietario);
12 echo $cadena;
13 $consultaBD = $conectaBD -> query($cadena);
14 .....
15 if($consultaBD){
16 // si hay existo con la inserción,
17 // se realiza un redirect a index.php
18 header("Location: index.php");
19 }else{
20 // si existe un error
21 // se captura el error
```

Líneas de Código

```

22 // se hace un redirect a nuevo.php
23 // además se envía el mensaje de error
24 // como parámetro
25 $mensaje = $conectaBD -> error;
26 header("Location: nuevo.php?error=". $mensaje);
27 }
28 ?>
```

Luego de ejecutar los archivos anteriores, se realizaron tareas como:

- Listar la información de la tabla vehículos a través del archivo index.php; ver Figura 35.

Listado de Vehículos

Propietario	Placa	Marca	Año Fabricación	Tipo
ANDRES RIVAS	LCH0102	MAZDA	2000	PUBLICO
LUIS VALENCIA	LLF0022	SAN REMO	2020	ALQUILER
RENé ELIZALDE	LLF0023	TOYOTA	2019	ALQUILER
RENé ELIZALDE	LLF0024	SAN REMO	2018	PARTICULAR
ROLANDO VERA	LCG0011	CHEVROLET	2016	ALQUILER
PATRICIO ROJAS	LCH0101	CHEVROLET	1990	PARTICULAR
Nuevo vehículo				

Figura 35. Listado de registros de la tabla vehículos

- Generar nuevos registros a través de los archivos: nuevo.php y nuevo_agrega.php; ver Figura 36.

Ingrese datos vehículo

Ingrese marca de vehículo

Ingrese placa del vehículo

Ingrese año de fabricación de vehículo

Ingrese tipo de vehículo

Ingrese nombres de propietario

Enviar

Figura 36. Formulario de ingreso de datos para un nuevo vehículo

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo



Actividad de aprendizaje recomendada

Actividad 1

Actividad de aprendizaje: Realice la autoevaluación 4 en donde se presenta un conjunto de preguntas en función de las temáticas descritas en la unidad 4, **Programación en el servidor web**. En la parte inicial de la actividad se sugieren los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



Autoevaluación 4

Es momento de medir el entendimiento de algunos conceptos estudiados en la Unidad 4 Programación en el servidor web. Usted puede revisar los siguientes temas:

- La unidad 4 de la presente guía
- Capítulo 5: Programación en el servidor web del texto básico
 - Sección: Preparación del entorno para el desarrollo web con PHP
 - Subsección: Lenguajes del lado del servidor
 - Subsección: Instalación del servidor WampServer
 - Capítulo 5: Programación en el servidor web del texto básico
 - Sección Introducción a la programación de aplicaciones web con lenguaje PHP
 - Subsección: Aspectos básicos de la programación en PHP
 - Subsección: Formularios
 - Capítulo 5 Programación en el servidor web del texto básico
 - Sección Uso de bases de datos en programación web

La autoevaluación le permitirá reforzar los conceptos estudiados. Los enunciados tienen una sola opción correcta, lea de forma atenta cada respuesta y seleccione la opción que usted considere correcta.

1. Identifique el nombre de dos servidores web.

- a. WampServer, XAMPP.
- b. Apache, WampServer.
- c. Nginx, Apache.

2. Indique el código que, al ejecutarse en un servidor web, presente en el navegador la cadena: verificando conocimientos

- a.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<title>Verificación funcionamiento PHP</title>
</head>
<body>
    echo "Verificando conocimientos" ;
</body>
</html>
```
- b.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<title>Verificación funcionamiento PHP</title>
</head>
<body>
<%
    echo "Verificando conocimientos" ;
%
</body>
</html>
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

c. <!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<title>Verificación funcionamiento PHP</title>
</head>
<body>
<?php
 echo "Verificando conocimientos" ;
?>
</body>
</html>

3. Dados los siguientes archivos, ¿cuál es la salida que se puede visualizar a través de la ejecución en un navegador?

Líneas de Código

```
index.php
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<table>
<tr>
    <td>Nro 1</td>
    <td>Operador</td>
    <td>Nro 2</td>
    <td>Resultado</td>
    while($inicio <= 5){
        $formato = "<tr>
        <td>%d</td><td>%s</td><td>%d</td><td class='celda'>%d</td>
        </tr>";
        echo sprintf($formato, $inicio, "+", $tabla, operacion($tabla, $inicio));
        $inicio++;
    }
}
```

[Índice](#)[Primer bimestre](#)[Segundo bimestre](#)[Solucionario](#)[Referencias bibliográficas](#)

Líneas de Código

```
}
```

```
?>
```

```
</table>
```

```
</body>
```

```
</html>
```

```
css/estilos.css
```

```
th, td {
```

```
    padding: 15px;
```

```
    border-bottom: 1px solid black;
```

```
}
```

```
tr:hover {
```

```
    background-color: #258EAD;
```

```
}
```

```
/* manejo de las características de las clases*/
```

```
.celda{
```

```
    background: #9999FF;
```

```
    color: #000000;
```

```
    font-family: Arial, Helvetica, sans-serif;
```

```
    font-size: 14px;
```

```
    font-weight: bold;
```

```
}
```

a.

Tabla del 3

Nro 1	Operador	Nro 2	Resultado
3	+	1	4
3	+	2	5
3	+	3	6
3	+	4	7
3	+	5	8



[Índice](#)[Primer bimestre](#)[Segundo bimestre](#)[Solucionario](#)[Referencias bibliográficas](#)

b. **Tabla del 3**

Nro 1	Operador	Nro 2	Resultado
3	+	1	6
3	+	2	6
3	+	3	6
3	+	4	6
3	+	5	6

c. **Tabla del 3**

Nro 1	Operador	Nro 2	Resultado
1	+	3	4
2	+	3	5
3	+	3	6
4	+	3	7
5	+	3	8

4. Dado el siguiente archivo, ¿cuál es la salida que se puede visualizar a través de la ejecución en un navegador?

Líneas de Código

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<?php
    function operacion($num1){
        return $num1 * 10;
    }
    echo "<hr/>";
    for ($i = 0; $i <= 6; $i++) {
        if ($i>=3) {
            $cadena = "Valor generado: <b>%s</b><br/>";
        }else{
            $cadena = "Valor generado: %s<br/>";
        }
        echo sprintf($cadena, operacion($i));
    }
    echo "<hr/>";
?>
</table>
</body>
</html>
```

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

a.

Valor generado: **0**
Valor generado: **10**
Valor generado: **20**
Valor generado: 30
Valor generado: 40
Valor generado: 50
Valor generado: 60

b.

Valor generado: 0
Valor generado: 10
Valor generado: 20
Valor generado: **30**
Valor generado: **40**
Valor generado: **50**
Valor generado: **60**

c.

Valor generado: **0**
Valor generado: **10**
Valor generado: **20**
Valor generado: **30**
Valor generado: **40**
Valor generado: **50**
Valor generado: **60**

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

5. Dados los siguientes archivos, ¿cuál es la salida que se puede visualizar a través de la ejecución en un navegador?

Líneas de Código

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="css/estilos.css" type="text/css" />
    <title>Uso básico de PHP</title>
  </head>
  <body>
    <?php
      function operacion($num1){
        return $num1 * 2;
      }
      function operacion2($num1){
        return $num1 - 1;
      }
      echo "<hr/>";
      for ($i = 0; $i <= 6; $i++) {
        $cadena = "<span class='presentacion'>Valor generado: " . $i . "</span><br/>";
        echo sprintf($cadena, operacion2(operacion($i)));
      }
      echo "<hr/>";
    ?>
    </table>
  </body>
</html>
```



Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

a.

VALOR GENERADO: -1
VALOR GENERADO: 1
VALOR GENERADO: 3
VALOR GENERADO: 5
VALOR GENERADO: 7
VALOR GENERADO: 9
VALOR GENERADO: 11

b.

Valor generado: -1
Valor generado: 1
Valor generado: 3
Valor generado: 5
Valor generado: 7
Valor generado: 9
Valor generado: 11

c.

VALOR GENERADO: 1
VALOR GENERADO: 3
VALOR GENERADO: 5
VALOR GENERADO: 7
VALOR GENERADO: 9
VALOR GENERADO: 11
VALOR GENERADO: 13

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

6. Dado el siguiente archivo denominado uno.php, identifique el archivo dos.php correcto.

Líneas de Código

```
uno.php
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <link rel="stylesheet" href="css/estilos.css" type="text/css" />
    <title>Uso básico de PHP con formularios</title>
  </head>
  <body>
    <h2>Ingrese datos para generación</h2>
    <div class="miformulario">
      <form action="dos.php" method="post">
        <label for="ciudad">Ingrese su ciudad</label>
        <input type="text" name="ciudad" id="ciudad" required/>
        <br/>
        <br/>
        <label for="provincia">Ingrese su provincia</label>
        <input type="text" name="provincia" id="provincia" required/>
        <br/>
        <br/>
        <input type="submit" name="enviar" id="enviar" value="Enviar" />
      </form>
    </div>
  </body>
</html>
```

Respuestas

a. dos.php

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<h3>Datos Ingresados</h3>
<?php
$ciudad = ["ciudad"];
$provincia = ["provincia"];
$datos = "<b>Provincia:</b>%s<br/><b>Ciudad:</b>%s<br/>";
echo sprintf($datos, $ciudad, $provincia) ;
?>
</body>
</html>
```

b. dos.php

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<h3>Datos Ingresados</h3>
<?php
$ciudad = $_REQUEST[ciudad];
$provincia = $_REQUEST[provincia];
$datos = "<b>Provincia:</b>%s<br/><b>Ciudad:</b>%s<br/>";
echo sprintf($datos, $ciudad, $provincia) ;
?>
</body>
</html>
```

c. dos.php

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<h3>Datos Ingresados</h3>
<?php
$ciudad = $_REQUEST["ciudad"];
$provincia = $_REQUEST["provincia"];
$datos = "<b>Provincia:</b>%s<br/><b>Ciudad:</b>%s<br/>";
echo sprintf($datos, $ciudad, $provincia) ;
?>
</body>
</html>
```

7. ¿Cuál de las siguientes sentencias permite crear una conexión a una base de datos MariaDB o MySQL?

- a. \$conectaBD = new MySqlDB(\$servidor, \$usuario, \$password, \$base_datos);
- b. \$conectaBD = new mysqli(\$servidor, \$usuario, \$password, \$base_datos);
- c. \$conectaBD = mysqli(\$servidor, \$usuario, \$password, \$base_datos);

8. ¿Cuál de las siguientes sentencias permite realizar una redirect a una página llamada destino.php a la cual se envía un parámetro llamado texto?

- a. header("Location_destino.php?texto="mensaje");
- b. header("destino.php?texto="mensaje");
- c. header("Location: destino.php?texto="mensaje");

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

9. ¿Cuál de los siguientes archivos genera una salida como la que se muestra en la figura?

Seleccione una ciudad

- Loja
- Quito
- Cuenca
- Machala

Enviar

a. <!DOCTYPE html>
<html>
 <head>
 <meta charset="utf-8" />
 <meta name="viewport" content="width=device-width" />
 <title>Uso básico de PHP con formularios</title>
 </head>
 <body>
 <h2>Seleccione una ciudad</h2>
 <div class="miformulario">
 <form action="dos.php" method="post">
 <?php
 function informacion(\$v1){
 \$variable = '<input type="radio" id="%s" name="ciudades" value="%s">'
 <label for="%s">%s</label>

';
 \$variable = sprintf(\$variable, \$v1, \$v1, \$v1, \$v1);
 return \$variable;
 }
 \$ciudades = array('Loja', 'Quito', 'Cuenca', 'Machala');
 for (\$i = 0; \$i < 4; \$i++) {
 echo informacion(\$ciudades[\$i]);
 }
 ?>

 <input type="submit" name="enviar" id="enviar" value="Enviar" />
 </form>
 </div>
 </body>
</html>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

b.

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<title>Uso básico de PHP con formularios</title>
</head>
<body>
<h2>Seleccione una ciudad</h2>
<div class="miformulario">
<form action="dos.php" method="post">
<?php
function informacion($v1){
$variable = '<input type="radio" id="%s" name="ciudades" value="%s">
<label for="%s">%s</label>
<br/>';
$variable = sprintf($variable, $v1);
return $variable;
}
$ciudades = array('Loja', 'Quito', 'Cuenca', 'Machala');
for ($i = 0; $i < 4; $i++) {
echo informacion($ciudades[$i]);
}
?>
<br/>
<input type="submit" name="enviar" id="enviar" value="Enviar" />
</form>
</div>
</body>
</html>
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

c. <!DOCTYPE html>
<html>
 <head>
 <meta charset="utf-8" />
 <meta name="viewport" content="width=device-width" />
 <title>Uso básico de PHP con formularios</title>
 </head>
 <body>
 <h2>Seleccione una ciudad</h2>
 <div class="miformulario">
 <form action="dos.php" method="post">
 <?php
 function informacion(\$v1){
 \$variable = '<input type="radio" id="%s" name="ciudades" value="%s">
 <label for="%s">%s</label>

';
 \$variable = sprintf(\$variable, \$v1, \$v1, \$v1, \$v1);
 }
 \$ciudades = array('Loja', 'Quito', 'Cuenca', 'Machala');
 for (\$i = 0; \$i < 4; \$i++) {
 echo informacion(\$ciudades[\$i]);
 }
 ?

 <input type="submit" name="enviar" id="enviar" value="Enviar" />
 </form>
 </div>
 </body>
</html>

10. Dados los siguientes archivos, identifique la salida correcta.

Líneas de Código

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<title>Uso básico de PHP con formularios</title>
</head>
<body>
<h2>Seleccione una ciudad</h2>
<div class="miformulario">
  form action="dos.php" method="post">
    <label for="fecha">Fecha</label>
    <input type="date" id="fecha" name="fecha" max="2020-12-31"><br><br>
    <input type="submit" name="enviar" id="enviar" value="Enviar" />
  </form>
</div>
</body>
</html>
```

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<meta name="viewport" content="width=device-width" />
<link rel="stylesheet" href="css/estilos.css" type="text/css" />
<title>Uso básico de PHP</title>
</head>
<body>
<h1>Presentar Datos</h1>
<?php
  function mifecha($v){
    return date('Ymd',strtotime($v));
  }
  $fecha = $_REQUEST["fecha"];
  $fecha2 = mifecha($fecha);
  $datos = "<h2>Fecha</h2><h3>%s</h3>";
  echo sprintf($datos, $fecha2) ;
?>
</body>
</html>
```

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

a.

Presentar Datos

Fecha

07.25.20

b.

Presentar Datos

Fecha

25,7,2020

c.

Presentar Datos

Fecha

20200725

Ir al solucionario



Semana 11



Unidad 5. Uso de frameworks de ambiente web

5.1. Modelo-vista-controlador base de los frameworks

Estimado estudiante, en este apartado de la guía se hace uso de la sección **Uso de frameworks** del capítulo 6 *Introducción a los frameworks* del texto básico. En este apartado sugerimos realizar una lectura de la subsección del texto básico denominada **Modelo-vista-controlador, la base de todos los frameworks**.

Algunos aspectos importantes de la lectura recomendada:

- Cuando se habla de un *framework* se abarca temas como: lenguaje de programación, librerías, bibliotecas, herramientas y metodologías de desarrollo.
- Se resalta el uso del patrón Modelo-Vista-Controlador que permite reutilización de código, facilitar los procesos de desarrollo y mejorar el mantenimiento a futuro de las aplicaciones.

- La rapidez en los tiempos de desarrollo a través del uso de propiedades de cada *framework*.
- Se facilita la corrección de posibles errores en la construcción de la solución.
- Generación de test de desarrollo para probar las funcionalidades.
- La seguridad implícita en los *frameworks* permite generar calidad en los productos entregados.

5.2. Clasificación de *frameworks* de ambiente web según los lenguajes de programación

En el presente ítem de estudio se describe los más importantes *frameworks* de acuerdo a los lenguajes de programación para la comprensión de un determinado *framework* existen dos puntos claves:

- Manejo de conceptos de: paradigma de programación de orientación a objetos, HTML, CSS y JavaScript.
- Entender el modelo-vista-controlador explicado en el apartado anterior de la guía didáctica.

Resaltar que la lista de *frameworks* es enorme en función de los lenguajes de programación.

A continuación, se listan los *frameworks* más importantes con documentación actualizada.

Lenguaje PHP

Framework	Repositorio de versiones	Sistema de plantillas
Symfony	https://github.com/symfony/symfony	▪ Twig
Laravel	https://github.com/laravel/laravel	▪ Blade

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Framework	Repositorio de versiones	Sistema de plantillas
Codeigniter	https://github.com/bcit-ci/CodeIgniter	<ul style="list-style-type: none">▪ Template Parser Class

Lenguaje Java

Framework	Repositorio de versiones	Sistema de plantillas
Spring	https://github.com/spring-projects/spring-framework	<ul style="list-style-type: none">▪ Thymeleaf
Ninja	https://github.com/ninjaframework/ninja	<ul style="list-style-type: none">▪ Apache FreeMarker
Apache Struts	https://github.com/apache/struts	<ul style="list-style-type: none">▪ Apache FreeMarker

Lenguaje Ruby

Framework	Repositorio de versiones	Sistema de plantillas
Ruby on Rails	https://github.com/rails/rails	Ruby Templating Engines

Lenguaje Python

Framework	Repositorio de versiones	Sistema de plantillas
Django	https://github.com/django/django	<ul style="list-style-type: none">▪ Django template language (DTL)▪ Jinja
Flask	https://github.com/pallets/flask	<ul style="list-style-type: none">▪ Ninja
Pylons	https://github.com/Pylons/pyramid	<ul style="list-style-type: none">▪ Mako▪ Chameleon
TurboGears	https://github.com/TurboGears/tg2	<ul style="list-style-type: none">▪ Kajiki

5.3. Desarrollo de aplicaciones mediante el framework Django

Estimado estudiante para el estudio del presente apartado relacionado con el *framework* Django se hará uso del recurso abierto

denominado ***La guía definitiva de django – Desarrolla aplicaciones Web de forma rápida y sencilla.***

5.3.1. Introducción

El capítulo Introducción a Django del recurso abierto destaca algunas características importantes del *framework* Django en el marco de la generación de aplicación web; se menciona algunas de ellas:

- Ahorra tiempo en el desarrollo de aplicaciones.
- Genera atajos para algunas tareas frecuentes.
- Maneja el patrón Modelo-Vista-Controlador mediante su filosofía Modelo-Vista-Template.
- Existe un acoplamiento débil en el desarrollo de componentes.

La [documentación oficial de Django](#) siempre será un recurso importante para profundizar en algunas temáticas.

5.3.2. Creación de proyecto y modelado

Usted debe revisar el capítulo 2 del recurso recomendado, cuya denominación es Empezando, donde se detalla información respecto a los procesos de instalación de lenguaje Python y del *framework* Django. Para la instalación de Django se recomienda hacer uso del gestor de paquetes PIP de Python.

La versión de Django usada en la guía es Django-3.0.8, instalada a través del comando:

- `pip install Django`

Ejemplo 1:

A continuación, se realiza la explicación de un ejemplo para la creación de un proyecto en Django. Se inicia el proyecto usando el siguiente comando:

- django-admin startproject proyectoUno

Los archivos generados son Figura 37:

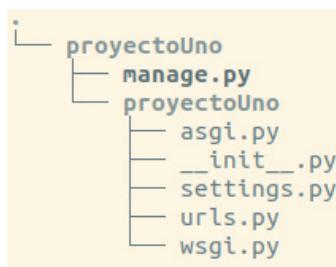


Figura 37. Archivos generados por el comando que crea un proyecto en Django

Donde:

- **proyectoUno** se convierte en la carpeta que contiene los archivos del proyecto.
- **manage.py** es usado para interactuar a través de la línea de comandos con el proyecto; se puede crear aplicaciones, acceder a una consola interactiva, actualizar la base de datos, crear usuarios administradores para el proyecto.
- **__init__.py** es un archivo que permite indicar a Python que el directorio sea considerado como paquete.
- **settings.py**, en este archivo se tienen las configuraciones del proyecto de Django.
- **urls.py**, se ubica el conjunto de direcciones internas del proyecto de Django.

- wsgi.py y asgi.py son archivos usados cuando se requiera ubicar el proyecto en un servidor como Apache o Nginx.

Indicar que en el desarrollo de este ejemplo demostrativo se usará la base de datos Sqlite. En el archivo setting.py se busca la variable DATABASES, misma que indica el gestor de base de datos para usar.

Ejemplo

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': os.path.join(BASE_DIR, 'db.sqlite3'),
    }
}
```

Se crean las tablas por defecto del proyecto a través del comando:

- python manage.py migrate

Las tablas creadas en la base de datos son las descritas en la Figura 38:

Database Structure		
	Type	Schema
Create Table	Create Index	Modify Table
Tables (11)		
auth_group		CREATE TABLE "auth_group" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name" varchar(50))
auth_group_permissions		CREATE TABLE "auth_group_permissions" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "group_id" integer NOT NULL, "permission_id" integer NOT NULL)
auth_permission		CREATE TABLE "auth_permission" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "content_type_id" integer NOT NULL, "codename" varchar(100), "name" varchar(50), "model" varchar(100))
auth_user		CREATE TABLE "auth_user" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "date_joined" datetime, "email" varchar(254), "first_name" varchar(30), "groups_id" integer NOT NULL, "is_active" integer NOT NULL, "is_staff" integer NOT NULL, "last_login" datetime, "last_name" varchar(30), "password" varchar(128), "username" varchar(150))
auth_user_groups		CREATE TABLE "auth_user_groups" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "group_id" integer NOT NULL, "user_id" integer NOT NULL)
auth_user_user_permissions		CREATE TABLE "auth_user_user_permissions" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "permission_id" integer NOT NULL, "user_id" integer NOT NULL)
django_admin_log		CREATE TABLE "django_admin_log" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "action_flag" integer NOT NULL, "change_message" text, "content_type_id" integer NOT NULL, "object_id" varchar(255), "user_id" integer NOT NULL)
django_content_type		CREATE TABLE "django_content_type" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "app_label" varchar(100), "model" varchar(100))
django_migrations		CREATE TABLE "django_migrations" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "app" varchar(100), "migration" varchar(255), "timestamp" timestamp)
django_session		CREATE TABLE "django_session" ("session_key" varchar(40) NOT NULL PRIMARY KEY, "session_data" text, "expire_date" datetime)
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)

Figura 38. Tablas generadas por con el comando python manage.py migrate

Se hace uso del servidor ligero para revisar el funcionamiento de la aplicación; el comando es python manage.py runserver, y se muestra su ejecución en la Figura 39:

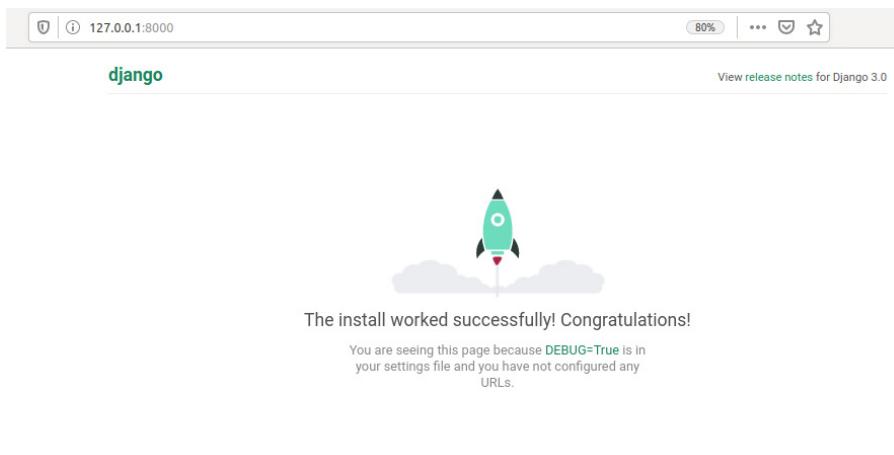


Figura 39. Levantar la aplicación a través del servidor propio de Django, usando el comando python manage.py runserver

Se procede a crear una **aplicación** para el proyecto a través del comando:

- `python manage.py startapp administrativo`

Lo anterior agrega una nueva estructura al proyecto (Figura 40).

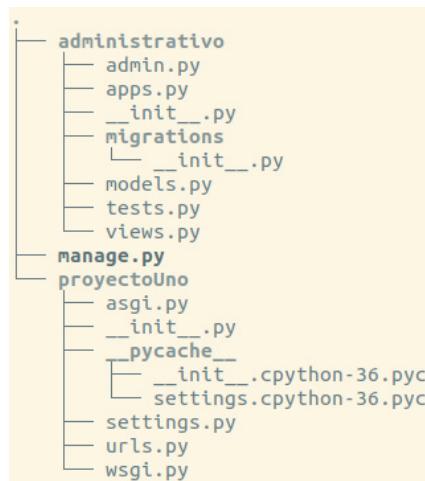


Figura 40. Estructura inicial de un proyecto y una aplicación en Django

- **administrativo** es la carpeta que contiene los archivos de la aplicación.
- **admin.py** permite agregar las clases o entidades que serán puestas en el proceso de administración del proyecto
- **__init__.py**, es el archivo que permite indicar a Python que el directorio sea considerado com paquete.
- **models.py**, se ubican las clases en Python que luego se convierten tablas de la base de datos.
- **views.py** se gestiona principalmente a través de funciones.

Luego de tener generada la aplicación, se procede a ingresar el modelo de la misma; se hace uso de archivo models.py; se recuerda que el modelado se lo hace mediante clases estructuradas en lenguaje Python, de manera similar al proceso realizado en SQLAlchemy.

Clases estructuradas en lenguaje Python

```
1 from django.db import models
2
3 # Create your models here.
4
5 class Estudiante(models.Model):
6     nombre = models.CharField(max_length=30)
7     apellido = models.CharField(max_length=30)
8     cedula = models.CharField(max_length=30, unique=True)
9
10    def __str__(self):
11        return "%s %s %s" % (self.nombre,
12                             self.apellido,
13                             self.cedula)
14
15 class NumeroTelefonico(models.Model):
16     telefono = models.CharField(max_length=100)
17     tipo = models.CharField(max_length=100)
18     estudiante = models.ForeignKey(Estudiante, on_delete=models.CASCADE)
19
20    def __str__(self):
21        return "%s %s" % (self.telefono, self.tipo)
```

En el modelo generado se ha creado dos entidades Estudiante y NumeroTelefonico.

- La entidad Estudiante hereda de la clase Model y tiene los siguientes atributos:
 - nombre; de tipo cadena representado en Django por campo CharField⁸, se le asigna un máximo de 30 caracteres.
 - apellido; de tipo cadena representado en Django por campo CharField, se le asigna un máximo de 30 caracteres.
 - cedula; de tipo cadena representado en Django por campo CharField, se le asigna un máximo de 30 caracteres y a través del atributo unique se indica que los valores deben ser únicos.
- La entidad NumeroTelefonico hereda de la clase Model y tiene los siguientes atributos:
 - telefono; de tipo cadena representado en Django por campo CharField se le asigna un máximo de 100 caracteres.
 - tipo; de tipo cadena representado en Django por campo CharField, se le asigna un máximo de 100 caracteres.
 - estudiante, atributo de tipo clave foránea representado a través de ForeignKey⁹. Se resalta que al atributo se le establece la entidad con cual se genera la relación (Estudiante). Cuando se genere un objeto de tipo

⁸ <https://docs.djangoproject.com/en/3.0/ref/models/fields/#django.db.models.CharField>

⁹ <https://docs.djangoproject.com/en/3.0/ref/models/fields/#foreignkey>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

NumeroTelefonico, al atributo estudiante se le debe asignar como valor un objeto de tipo Estudiante.

En las clases expresadas en el modelo, no se ha especificado un atributo como clave primaria - primary_key¹⁰; en estos casos, Django crea automáticamente un campo AutoField¹¹ de tipo entero, no es necesario agregar este atributo cuando se generan objetos.

En la documentación se detalla información referente a [Modelo Field](#).

Es necesario agregar la aplicación en el arreglo representado por la variable INSTALLED_APPS en el archivo settings.py. Se procede a ejecutar los comandos para almacenar en archivos de control propios de Django las migraciones:

- `python manage.py makemigrations administrativo`

¹⁰ <https://docs.djangoproject.com/en/3.0/ref/models/fields/#primary-key>

¹¹ <https://docs.djangoproject.com/en/3.0/ref/models/fields/#django.db.models.AutoField>

Líneas de código

Opcional: se usa el comando `python manage.py sqlmigrate administrativo 0001` para revisar en formato SQL el makemigration ejecutado

```
BEGIN;
```

```
—  
__ Create model Estudiante
```

```
—  
CREATE TABLE "administrativo_estudiante" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "nombre" varchar(30) NOT NULL, "apellido" varchar(30) NOT NULL, "cedula" varchar(30) NOT NULL UNIQUE);
```

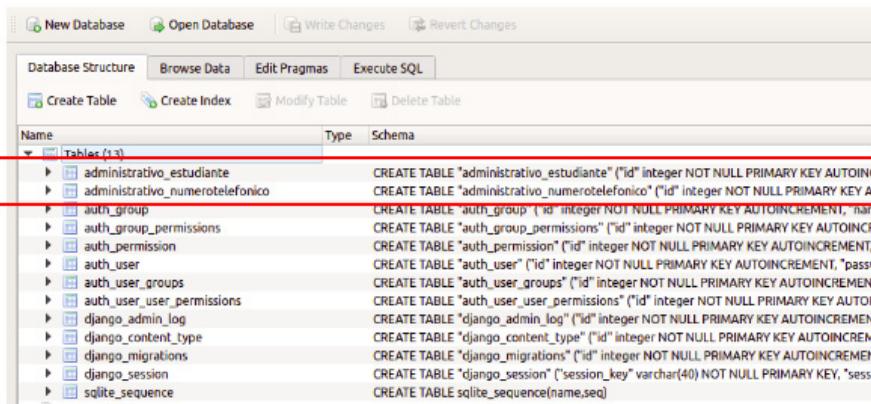
```
—  
__ Create model NumeroTelefonico
```

```
—  
CREATE TABLE "administrativo_numerotelefonico" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "telefono" varchar(100) NOT NULL, "tipo" varchar(100) NOT NULL, "estudiante_id" integer NOT NULL REFERENCES "administrativo_estudiante" ("id") DEFERRABLE INITIALLY DEFERRED);  
CREATE INDEX "administrativo_numerotelefonico_estudiante_id_a580aa13" ON "administrativo_numerotelefonico" ("estudiante_id");  
COMMIT;
```

Luego, se necesita realizar los cambios en la base de datos a través del comando:

- `python manage.py migrate`

La base de datos tiene la siguiente apariencia, ver Figura 41:



Name	Type	Schema
Tablas (13)		
administrativo_estudiante		CREATE TABLE "administrativo_estudiante" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT);
administrativo_numerotelefonico		CREATE TABLE "administrativo_numerotelefonico" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT);
auth_group		CREATE TABLE "auth_group" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name" varchar(50));
auth_group_permissions		CREATE TABLE "auth_group_permissions" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "group_id" integer NOT NULL, "permission_id" integer NOT NULL);
auth_permission		CREATE TABLE "auth_permission" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "name" varchar(50), "content_type_id" integer NOT NULL, "codename" varchar(100));
auth_user		CREATE TABLE "auth_user" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "password" varchar(128), "last_login" datetime, "is_superuser" integer, "username" varchar(150), "first_name" varchar(30), "last_name" varchar(30), "email" varchar(254), "is_staff" integer);
auth_user_groups		CREATE TABLE "auth_user_groups" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "user_id" integer NOT NULL, "group_id" integer NOT NULL);
auth_user_user_permissions		CREATE TABLE "auth_user_user_permissions" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "user_id" integer NOT NULL, "permission_id" integer NOT NULL);
django_admin_log		CREATE TABLE "django_admin_log" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "action_time" datetime, "object_id" varchar(255), "object_repr" varchar(255), "change_message" longtext, "content_type_id" integer NOT NULL, "user_id" integer NOT NULL);
django_content_type		CREATE TABLE "django_content_type" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "model" varchar(100), "name" varchar(100));
django_migrations		CREATE TABLE "django_migrations" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "app" varchar(100), "name" varchar(50), "applied" datetime);
django_session		CREATE TABLE "django_session" ("session_key" varchar(40) NOT NULL PRIMARY KEY, "session_data" longtext, "expire_date" datetime);
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq);

Figura 41. Modelo de la aplicación migrado a la base de datos

Es momento de usar el shell de Django para interactuar con la base de datos a través del uso del ORM de Django mediante las clases del modelo. Para acceder se usa el comando:

- `python manage.py shell`

Líneas de código

```
# La sentencias mostradas a continuación se usando el shell de Django
# Primero, se importa las clases del modelo
from administrativo.models import Estudiante, NumeroTelefonico
# Inserción de datos
# se crea un objeto de tipo Estudiante
# se hace referencia a los atributos de la clase
e1 = Estudiante(nombre="Luisa", apellido="Tene", cedula="1100991133")
e2 = Estudiante(nombre="Rolando", apellido="Sarango", cedula="1100991122")
# se guarda el objeto en la base de datos, mediante el método save()
e1.save()
e2.save()
# se crea un objeto de tipo NumeroTelefonico
# para el atributo estudiante del objeto t, se usa el objeto de tipo Estudiante e
t = NumeroTelefonico(telefono="09988776655", tipo="particular", estudiante=e1)
# Consulta de datos
# Seleccionar todos los datos del modelo Estudiante
estudiantes = Estudiante.objects.all()
for e in estudiantes:
    print(e)
#salida:
#####
# Rolando Sarango 1100991122
# Luisa Tene 1100991133
#####
# Filtrar todos los estudiantes que tiene como valor en nombre: "Rolando"
estudiantes = Estudiante.objects.filter(nombre="Rolando").all()
# Filtrar todos los estudiantes que tengan un número de teléfono con la secuencia
"8888"
# numerotelefonico es la clase asociada, se recuerda la relación
Estudiante.objects.filter(numerotelefonico__telefono__contains="8888")
# Filtrar todos los números telefónicos que pertenecen a estudiantes que tengan
en su apellido la secuencia "Te"
telefonos = NumeroTelefonico.objects.filter(estudiante_apellido_contains="Te")
for t in telefonos:
    print("%s - %s" % (t, t.estudiante))
#salida
##
# 09988776655 particular - Luisa Tene 1100991133
##
```

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

Se sugiere la revisión de la documentación oficial del framework Django en relación a manejo de relaciones entre entidades:

- Relación muchos a uno
- Relación uno a uno
- Relación muchos a muchos
- Manejo de ORM de Django



Semana 12

5.3.3. Uso de interfaz administrativa

En el capítulo 6 *El sitio de administración*, del recurso abierto *La guía definitiva de django – Desarrolla aplicaciones web de forma rápida y sencilla*, se detalla los procesos para manejar la interfaz de administración de proyectos en Django. Se solicita una lectura de dicho apartado.

Se hace uso de los metadatos de los modelos generados en las aplicaciones para crear una interfaz de usuario con prestaciones listas para ser usadas por el usuario que disponga de los permisos respectivos.

Para la activación se sugiere los siguientes pasos:

- Verificar en el archivo `setting.py`, en la variable `INSTALLED_APPS` tenga en listado de elementos: `django.contrib.admin`.

- Mediante el comando: `python manage.py createsuperuser`, se crea un súper usuario del proyecto.
- Editar el archivo `admin.py` de la aplicación para indicar los modelos que serán accesibles para administración desde la interfaz.

Líneas de código

```
1 from django.contrib import admin
2
3 # Importar las clases del modelo
4 from administrativo.models import Estudiante, NumeroTelefonico
5
6 # Agregar la clase Estudiante para administrar desde
7 # interfaz de administración
8 admin.site.register(Estudiante)
9
10 # Agregar la clase NumeroTelefonico para administrar desde
11 # interfaz de administración
12 admin.site.register(NumeroTelefonico)
```

Se digita el comando que permite levantar el servidor propio de Django (`python manage.py runserver`) y se ingresa a la siguiente dirección en un navegador web (`http://127.0.0.1:8000/admin/`). Se solicitará el ingreso del **Username** y **Password**, recordar los datos ingresados en el comando **createsuperuser**. Se visualiza una pantalla que lista las clases o entidades disponibles en la interfaz de administrador para manipular; ver Figura 42



Figura 42. Interfaz de administración de Django

Se puede seleccionar cualquiera de las entidades listadas para acceder las opciones dadas por la interfaz. Si se selecciona la entidad **Estudiantes**, se visualiza una pantalla con algunas opciones (ver Figura 43).



Figura 43. Interfaz de administración de una clase específica

Para agregar un registro a la clase o entidad **Estudiante** se tiene la siguiente pantalla ver Figura 44.

Llenar los campos disponibles, en base a las propiedades de la clase.

Guardar la información en base de datos y generar una nueva pantalla para ingreso de información,

Guardar la información en base de datos y continuar en la misma pantalla para actualizar alguna información pendiente

Guardar y añadir otro Guardar y continuar editando GRABAR

Figura 44. Ingreso de un registro a la clase o entidad Estudiante

A continuación, se representa el proceso de ingreso de un registro de la clase NumeroTelefonico, para demostrar el potencial del componente administrativo de Django. Se recuerda que la clase NumeroTelefonico requiere como datos o propiedades teléfono, tipo y estudiante; este último es de tipo Estudiante; ver Figura 45.

El atributo estudiante para un objeto de tipo NumeroTelefonico se lo obtiene de un listado de estudiantes. El formulario maneja la relación entre las clases.

Figura 45. Manejo de relaciones en interfaz administrativa de Django

A continuación, se muestra algunas opciones de personalización para la interfaz administrativa; se trabaja sobre el archivo admin.py.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
1 from django.contrib import admin
2
3 # Importar las clases del modelo
4 from administrativo.models import Estudiante, NumeroTelefonico
5
6 # Agregar la clase Estudiante para administrar desde
7 # interfaz de administración
8 # admin.site.register(Estudiante)
9
10 # Se crea una clase que hereda
11 # de ModelAdmin para el modelo
12 # Estudiante
13 class EstudianteAdmin(admin.ModelAdmin):
14 # listado de atributos que se mostrará
15 # por cada registro
16 # se deja de usar la representación (str)
17 # de la clase
18 list_display = ('nombre', 'apellido', 'cedula')
19
20 # admin.site.register se lo altera
21 # el primer argumento es el modelo (Estudiante)
22 # el segundo argumento la clase EstudianteAdmin
23 admin.site.register(Estudiante, EstudianteAdmin)
24
25 # Agregar la clase NumeroTelefonico para administrar desde
26 # interfaz de administración
27 admin.site.register(NumeroTelefonico)
```

En la Figura 46 se indica como se visualizan los registros con los cambios realizados.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Inicio > Administrativo > Estudiantes

Escoja estudiante a modificar

Acción: Ir seleccionados 0 de 3

<input type="checkbox"/>	NOMBRE	APELLIDO	CEDULA
<input type="checkbox"/>	Luisa	Tene	1100991133
<input type="checkbox"/>	Rolando	Sarango	1100991122
<input type="checkbox"/>	Pedro	Pérez	1134565678

3 estudiantes

Figura 46. Listar los registros de la clase Estudiante mediante columnas

- Agregar la opción de búsqueda en base a los atributos de la clase. Se agrega el atributo `search_fields`, al cual se da como valor una tupla de atributos que serán los campos de búsqueda para cada registro (Figura 47).
 - Para el modelo estudiante se tiene: `search_fields = ("nombre", "cedula")`

Escoja estudiante a modificar

Buscar

Acción: Ir seleccionados 0 de 3

<input type="checkbox"/>	NOMBRE	APELLIDO	CEDULA
<input type="checkbox"/>	Luisa	Tene	1100991133
<input type="checkbox"/>	Rolando	Sarango	1100991122
<input type="checkbox"/>	Pedro	Pérez	1134565678

3 estudiantes

Figura 47. Agregar la opción de búsqueda al listado de un modelo en la interfaz administrativa

- Cuando se crea un registro de tipo `NumeroTelefonico`, al momento de asignar el valor del atributo `estudiante` (clave foránea), se debe seleccionar de un listado de estudiantes a través de un `select`. Se puede considerar otra forma de seleccionar el campo `estudiante`. En una clase que hereda de `ModelAdmin` se agrega el atributo `raw_id_fields` que permite acceder a una interfaz para buscar los estudiantes y seleccionar el que se desee (ver Figura 48).

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
1 from django.contrib import admin
2
3 # Importar las clases del modelo
4 from administrativo.models import Estudiante, NumeroTelefonico
5
6 # Agregar la clase Estudiante para administrar desde
7 # interfaz de administración
8 # admin.site.register(Estudiante)
9
10 # Se crea una clase que hereda
11 # de ModelAdmin para el modelo
12 # Estudiante
13 class EstudianteAdmin(admin.ModelAdmin):
14     # listado de atributos que se mostrará
15     # por cada registro
16     # se deja de usar la representación (str)
17     # de la clase
18     list_display = ('nombre', 'apellido', 'cedula')
19     search_fields = ('nombre', 'cedula')
20
21 # admin.site.register se lo altera
22 # el primer argumento es el modelo (Estudiante)
23 # el segundo argumento la clase EstudianteAdmin
24 admin.site.register(Estudiante, EstudianteAdmin)
25
26 # Agregar la clase NumeroTelefonico para administrar desde
27 # interfaz de administración
28 # admin.site.register(NumeroTelefonico)
29
30 # Se crea una clase que hereda
31 # de ModelAdmin para el modelo
32 # NumeroTelefonico
33 class NumeroTelefonicoAdmin(admin.ModelAdmin):
34     # listado de atributos que se mostrará
35     # por cada registro
36     # se deja de usar la representación (str)
37     # de la clase
38     list_display = ('telefono', 'tipo', 'estudiante')
39     # se agrega el atributo
```



Líneas de código

```

40 # raw_id_fields que permite acceder a una interfaz·
41 # para buscar los estudiantes y seleccionar el que·
42 # se dese·
43 raw_id_fields = ('estudiante',)
44
45 admin.site.register(NúmeroTelefónico, NúmeroTelefónicoAdmin)

```

The screenshot shows three parts of a Django admin interface:

- Añadir numero telefonico**: A form with fields for 'Telefono' (07712341233), 'Tipo' (público), and 'Estudiante' (with a dropdown menu). A purple arrow points from this form to the 'Modificar numero telefonico' form.
- Escoja estudiante**: A search interface for selecting a student. It shows a list of students with columns for 'NOMBRE', 'APELITOS', 'CÉDULA', and 'TELÉFONO'. One student, Luisa Tene, is selected and highlighted with an orange border. Another student, Pedro Pérez, is also shown. A blue arrow points from the 'Estudiante' field in the first form to this search interface.
- Modificar numero telefonico**: A form with the same fields as the first one, but with the 'Estudiante' field populated with the ID '2' and the name 'Luisa Tene 1100991133'. A red box highlights this entire form.

Below the forms, a message states: "el registro NúmeroTelefónico con datos guardados".

Figura 48. Uso del atributo raw_id_fields en un atributo llave foránea

- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo



Actividad de aprendizaje recomendada

Actividad 1

Actividad de aprendizaje: realice la autoevaluación 5, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 5, Uso de *frameworks de ambiente web*, subsecciones 5.1, 5.2, 5.3. En la parte inicial de la actividad se indican los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.



Autoevaluación 5

Es momento de medir el entendimiento de algunos conceptos estudiados en la primera parte de la Unidad 5 Uso de frameworks de ambiente web. Usted puede revisar los siguientes temas:

- Ítems 5.1, 5.2, 5.3 de la presente guía.
- Capítulo 6: Introducción a los *frameworks* del texto básico
 - Sección Uso de *frameworks*
 - Subsección: Modelo-vista-controlador, la base de todos los *frameworks*
- Recurso abierto denominado: La guía definitiva de django – Desarrolla aplicaciones web de forma rápida y sencilla
 - Capítulo 1: Introducción a Django
 - Capítulo 2: Empezando
 - Capítulo 6: El sitio de administración

La autoevaluación le permitirá reforzar los conceptos estudiados. Los enunciados tienen una sola opción correcta, lea de forma atenta cada respuesta y seleccione la opción que usted considere correcta.

1. **Identifique tres temáticas que se abarca cuando se estudia y aplica un *framework*.**
 - a. Lenguaje de programación, librerías y bibliotecas
 - b. Lenguaje de programación, sistema operativo y bibliotecas
 - c. Lenguaje de programación, librerías y IDE de programación

2. Identifique dos sistemas de plantillas usados por los frameworks con lenguaje de programación PHP.
 - a. Twing, Thymeleaf.
 - b. Twing, Blade.
 - c. Apache FreeMarker, Jinja.
3. ¿Cuál es el comando para crear un proyecto en el framework Django?
 - a. Django-admin startproject nombre-proyecto.
 - b. Django-admin start_project nombre-proyecto.
 - c. Django-admin project nombre-proyecto.
4. Identifique el archivo generado en los proyectos de Django que permite: interactuar a través de la línea de comandos con el proyecto para crear aplicaciones, actualizar la base de datos, crear usuarios administradores para el proyecto.
 - a. Settings.py.
 - b. Manage.py.
 - c. Wsgi.py.
5. Identifique el comando que permite crear y actualizar las tablas en la base de datos en un proyecto de Django.
 - a. Python manage.py engine.
 - b. Python manage.py sql.
 - c. Python manage.py migrate.
6. ¿Cuál es el comando usado en Django para crear una aplicación?
 - a. Python manage.py startapp nombre-app.
 - b. Python manage.py start administrativo.
 - c. Python manage.py start_app administrativo.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

7. Dado un modelo con denominación Universidad, identifique el comando que permite obtener todos los registros de la base de datos mediante el ORM de Django.
- Universidad.session().objects.all()
 - Universidad.objects.all()
 - Universidad.objects.filter()
8. La entidad Universidad posee un atributo llamado dirección de tipo cadena. ¿Cuál es el comando que permita buscar todos los registros que contengan la cadena “Loja” en alguna parte del valor del atributo?
- Universidad.objects.filter(direccion__contains="Loja").all()
 - Universidad.objects.filter(direccion.contains="Loja").all()
 - Universidad.objects.filter(direccion__contains("Loja")).all()
9. ¿Cuál es el atributo que se debe agregar en una clase que hereda de admin.ModelAdmin, que permita mostrar atributos seleccionados de un modelo?
- Losta_display.
 - List_display.
 - Listdisplay.
10. ¿Cuál es el comando que permite crear un super-usuario administrador en un proyecto en Django?
- Python manage.py createsuperuser.
 - Python manage.py create_superuser.
 - Python manage.py create_supe_ruser.

Ir al solucionario



Semana 13

Es momento de realizar una revisión detenida del capítulo 7 **Procesamiento de formularios** del recurso abierto *La guía definitiva de django – Desarrolla aplicaciones web de forma rápida y sencilla*.

5.3.4. Manejo de vistas, templates

El manejo de vistas en los proyectos en Django se realizan editando el archivo `views.py` de las aplicaciones el proyecto. Se recuerda que en este módulo se genera la lógica de la aplicación que permite acceder a la base de datos y construir una respuesta para ser visualizada en una plantilla. Las funciones que se codifican en el archivo `views.py` tiene como primer argumento un objeto de tipo `HttpRequest`. Se considera un estándar usar la palabra `request`; en dicho objeto se agrupa un conjunto de información acerca de la petición generada.

En la Figura 49 se describe el flujo de información al momento que se realiza una petición a un proyecto de Django.

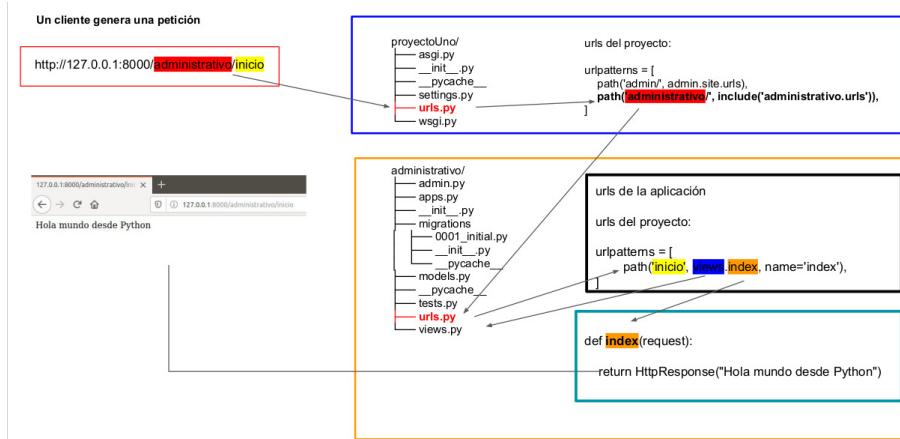


Figura 49. Flujo de información mediante una petición a Django

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
55 TEMPLATES = [
56   {
57     'BACKEND': 'django.template.backends.django.DjangoTemplates',
58
59     'DIRS': [os.path.join(BASE_DIR, 'templates')],
60     'APP_DIRS': True,
61     'OPTIONS': {
62       'context_processors': [
63         'django.template.context_processors.debug',
64         'django.template.context_processors.request',
65         'django.contrib.auth.context_processors.auth',
66         'django.contrib.messages.context_processors.messages',
67       ],
68     },
69   },
70 ]
```

La estructura de la aplicación **administrativo** queda de la siguiente forma. Se define el nombre del directorio como **templates**:

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

Líneas de código

```
administrativo/  
admin.py  
apps.py  
__init__.py  
migrations  
0001_initial.py  
__init__.py  
__pycache__  
models.py  
__pycache__  
templates  
listadoEstudiantes.html  
tests.py  
urls.py  
views.py
```

Ejemplo 1

A continuación se define un proceso que permite **listar los registros del modelo Estudiante** obtenidos de la base de datos.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo 1

```
views.py
1 from django.shortcuts import render
2 from django.http import HttpResponseRedirect
3 from django.template import RequestContext
4 from django.shortcuts import render
5
6 # importar las clases de models.py
7 from administrativo.models import *
8
9 # Create your views here.
10
11 def index(request):
12     # return HttpResponseRedirect("Hola mundo desde Python")
13     return HttpResponseRedirect("Hola mundo desde Python<br/>%s" % (request.path))
14
15 def listadoEstudiantes(request):
16     """
17     Listar los registros del modelo Estudiante,
18     obtenidos de la base de datos.
19     """
20     # a través del ORM de django se obtiene
21     # los registros de la entidad; el listado obtenido
22     # se lo almacena en una variable llamada
23     # estudiantes
24     estudiantes = Estudiante.objects.all()
25     informacion_template = {'estudiantes': estudiantes, 'numero_estudiantes':
len(estudiantes)}
26     return render(request, 'listadoEstudiantes.html', informacion_template)
27 
```

```
templates/listadEstudiantes.html
1 Número de estudiantes {{ numero_estudiantes }}
2 <br/><br/>
3
4 {% for e in estudiantes %}
5
6 {{e}}<br/>
7
8 {% endfor %}
```

Ejemplo 1

urls.py de la aplicación

```

1 """
2 Manejo de urls para la aplicación
3 administrativo
4 """
5 from django.urls import path
6 # se importa las vistas de la aplicación
7 from . import views
8
9
10 urlpatterns = [
11     path('inicio', views.index, name='index'),
12     path('listado-estudiantes', views.listadoEstudiantes,
13          name='listadoEstudiantes'),
14 ]

```

Al momento de ejecutar el servidor con los cambios realizados se obtiene una imagen como de la Figura 50.



Figura 50. Pantalla final del proceso de generación de información en views.py y utilización de templates

Algunas observaciones de la función **listadoEstudiantes** del archivo **views.py**:

- **estudiantes = Estudiante.objects.all()** ; a través del ORM de django se obtiene los registros de la entidad; el listado obtenido se lo almacena en una variable llamada estudiantes; se entiende que la variable estudiantes es un listado.

- **informacion_template**; es la variable tipo diccionario donde se agregará la información que estará disponible en el *template* para presentar.
- **{'estudiantes': estudiantes, 'numero_estudiantes': len(estudiantes)}**; es el valor asignado a la variable **informacion_template**; se generan dos llaves, la primera llamada '**estudiantes**' cuyo valor específico es la variable **estudiantes**, la segunda es '**numero_estudiantes**' que tiene asignado el número de elementos que contienen la lista llamada **estudiantes**.
- En la parte final de la función esta expresado una sentencia de retorno:
 - return **render**¹²; la función render recibe como argumentos: el objeto request, la plantilla o template donde se desplegará la lógica desarrollada en la función y un diccionario con datos que necesitamos pasar a la plantilla.
 - **request**¹³, el objeto que tiene los datos de la petición.
 - 'listadoEstudiantes.html', se detalla la ruta y la plantilla asociada a la función.
 - **informacion_template**, es un diccionario con todo el contexto que estará disponible en el template.

Ahora se detalla lo realizado en el archivo `listadoEstudiantes.html`, se recuerda que dicho archivo está ubicado en la carpeta **templates** dentro la aplicación **administrativo**.

¹² <https://docs.djangoproject.com/es/3.0/topics/http/shortcuts/>

¹³ <https://docs.djangoproject.com/en/3.0/ref/request-response/>

En los templates¹⁴ o plantillas de Django se puede usar HTML, JAVASCRIPT, CSS, junto a:

- **Variables.**- son representadas de la siguiente forma `{{ nombre-de-la-variable }}`. Dentro del template evalúa si una variable está bien estructurada y presenta en valor que tenga en ese momento.
- **Filtros.**- una característica que permite modificar la presentación de una variable. Se usa el carácter pipe (|) después del nombre de la variable; por ejemplo `{{ nombre-de-variable | lower}}`, presenta el valor de variable en minúscula. Existen filtros establecidos en el **framework** y existe la posibilidad de crear filtros personalizados¹⁵.
- **Tags.**- la estructura que sigue un tag es `{% nombre-tag %}`. Según la funcionalidad, para algunos tags se necesita establecer el inicio y fin:

Tags
<code>{% nombre-tag %}</code>
<code>{% end-nombre-tag %}</code>

A través de los tags se puede implementar ciclos repetitivos mediante for, condicionales if, elif y else.

En el archivo listadoEstudiantes.html se expresa lo siguiente:

- En la línea 1, se usa la representación de una variable mediante `{{ numero_estudiantes }}`
- En la línea 5, se inicia un ciclo repetitivo mediante el tag `{% for e in estudiantes %}`, que permite recorrer una secuencia de información (listado de **estudiantes**), que llega a la plantilla desde la función generada en views.py a través del diccionario.

¹⁴ <https://docs.djangoproject.com/en/3.0/ref/templates/language/>

¹⁵ <https://docs.djangoproject.com/en/3.0/howto/custom-template-tags/>

- La expresión {{ e }} de la línea 5, presenta el valor de la variable que itera en el **ciclo for**.
- Se finaliza el tag del ciclo for en la línea 9, mediante {% endfor %}.

En el archivo urls.py se agrega un nuevo PATH, que asocia 'listado-estudiantes' a la vista **listadoEstudiantes**.

Ejemplo 2

A continuación se define un proceso que permite **listar los registros del modelo Estudiante** y los números telefónicos asociados a cada estudiante (ver Figura 51).

Ejemplo 2

```
views.py
32 def listadoEstudiantesDos(request):
33 """
34 Listar los registros del modelo Estudiante,
35 obtenidos de la base de datos.
36 """
37 estudiantes = Estudiante.objects.all()
38 # en la variable tipo diccionario llamada informacion_template
39 # se agregará la información que estará disponible
40 # en el template
41 informacion_template = {'estudiantes': estudiantes, 'numero_estudiantes':
len(estudiantes)}
42 return render(request, 'listadoEstudiantesDos.html', informacion_template)
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Ejemplo 2

```
listadoEstudiantesDos.html
1 <h2>Listado de Estudiantes</h3>
2 <h3>Número de estudiantes {{ numero_estudiantes }}</h3>
3
4 {% for e in estudiantes %}
5 ....
6 {{e}}<br/>
7 <ul>
8 {% for numero in e.numerotelefonico_set.all%}
9   <li>
10    Número telefónico {{numero}}<br/>
11   </li>
12 {% endfor %}
13 </ul>
14 <br/>
15 {% endfor %}
```

```
urls.py
1 """
2 Manejo de urls para la aplicación
3 administrativo
4 """
5 from django.urls import path
6 # se importa las vistas de la aplicación
7 from . import views
8
9
10 urlpatterns = [
11     path('inicio', views.index, name='index'),
12     path('listado-estudiantes', views.listadoEstudiantes,
13          name='listadoEstudiantes'),
14     path('listado-estudiantes-dos', views.listadoEstudiantesDos,
15          name='listadoEstudiantesDos'),
16 ]
```

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

The screenshot shows a web browser window with the URL `127.0.0.1:8000/administrativo/listado-estudiantes-dos`. The page title is **Listado de Estudiantes**. Below it, a section header says **Número de estudiantes 3**. Three student entries are listed:

- Rolando Sarango 1100991122
 - Número telefónico 08888776655 alquiler
 - Número telefónico 07712341234 alquiler
- Luisa Tene 1100991133
 - Número telefónico 09988776655 particular
 - Número telefónico 07712341233 público
- Pedro Pérez 1134565678
 - Número telefónico 0550551234 alquiler

Figura 51. Listar los registros del modelo Estudiante y los números telefónicos asociados a cada estudiante

- Estimado estudiante puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

5.3.5. Manejo de formularios

En Django existen varias formas para crear formularios. Asociadas a la filosofía del *framework* se recomienda usar la **clase Form¹⁶** y la clase **ModelForm¹⁷**. La segunda se la revisará en este ítem de estudio. A partir de los modelos expresados en el archivo `models.py` se puede generar formularios.

Ejemplo 1

Generar un proceso que permita crear un registro de tipo Estudiante haciendo uso de la clase **ModelForm**.

¹⁶ <https://docs.djangoproject.com/en/3.0/topics/forms/>

¹⁷ <https://docs.djangoproject.com/en/3.0/topics/forms/modelforms/>

- Se crea un archivo bajo la denominación **forms.py**, en el cual se genera una clase llamada **EstudianteForm** que hereda de **ModelForm**; dentro de ella se adiciona una clase interna **Meta**, que permite indicar en el atributo **model** la entidad del archivo **models.py** para generar el formulario. Además, mediante el atributo **fields** se establece el listado de atributos de la clase del modelo que se necesita expresar en el modelo.

Ejemplo 1

```

1 from django.forms import ModelForm
2 from administrativo.models import Estudiante
3
4
5 class EstudianteForm(ModelForm):
6     class Meta:
7         model = Estudiante
8         fields = ['nombre', 'apellido', 'cedula']
9

```

- En el archivo **views.py** se agrega una nueva función denominada **crearEstudiante**.
 - En la línea 51 se realiza un condicional que verifica que la petición realizada sea un 'POST' (`request.method`); si esto es verdadero, se crea un objeto de tipo **EstudianteForm** llamado **formulario**, al cual se le envía como argumentos los parámetros que tenga el `request`. **POST** (los valores ingresados en el formulario, asociados a los atributos de la clase). Si el formulario es válido (`formulario.is_valid`), a través de la función `save()` se guarda la información en el base de datos. Finalmente se realiza un `redirect` al función `listadoEstudiantesDos`.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
48 def crearEstudiante(request):
49     """
50     """
51     if request.method=='POST':
52         formulario = EstudianteForm(request.POST)
53         print(formulario.errors)
54         if formulario.is_valid():
55             formulario.save()
56             return redirect(listadoEstudiantesDos)
57     else:
58         formulario = EstudianteForm()
59     diccionario = {'formulario': formulario}
60
61     return render(request, 'crearEstudiante.html', diccionario)
62
```

- En el archivo crearEstudiante.html se crea un formulario en HTML, se especifica en el atributo method el valor "POST".
 - En el interior de formulario se hace referencia a la variable formulario, que es recibida desde la función del views.py, mediante {{formulario.as_p}}. La expresión {{formulario.as_p}} genera el siguiente HTML

Líneas de código

```
'<p><label for="id_nombre">Nombre:</label> <input type="text" name="nombre" maxlength="30" required id="id_nombre"></p>\n<p><label for="id_apellido">Apellido:</label> <input type="text" name="apellido" maxlength="30" required id="id_apellido"></p>\n<p><label for="id_cedula">Cedula:</label> <input type="text" name="cedula" maxlength="30" required id="id_cedula"></p>'
```

- Cuando se usa formularios en Django se debe usar el tag `{% csrf_token %}`¹⁸; Django usa esta directiva para prevenir ataques de sitios que generan solicitudes falsas.

Líneas de código

```
1 <h2>Crear Estudiante</h3>
2
3 <form action="" method="post">
4   {% csrf_token %}
5   {{formulario.as_p}}
6   <p><input type='submit' value='Agregar'/></p>
7 </form>
~
```

- En el archivo urls.py se agrega la nueva ruta llamada **crear-estudiante** que se asocia a la función crearEstudiante agregada en views.py.

En la Figura 52, se indica el flujo de información expresada en las líneas anteriores al momento de ingresar la dirección en un navegador web <http://127.0.0.1:8000/administrativo/crear-estudiante>.

¹⁸ <https://docs.djangoproject.com/en/3.0/ref/csrf/>

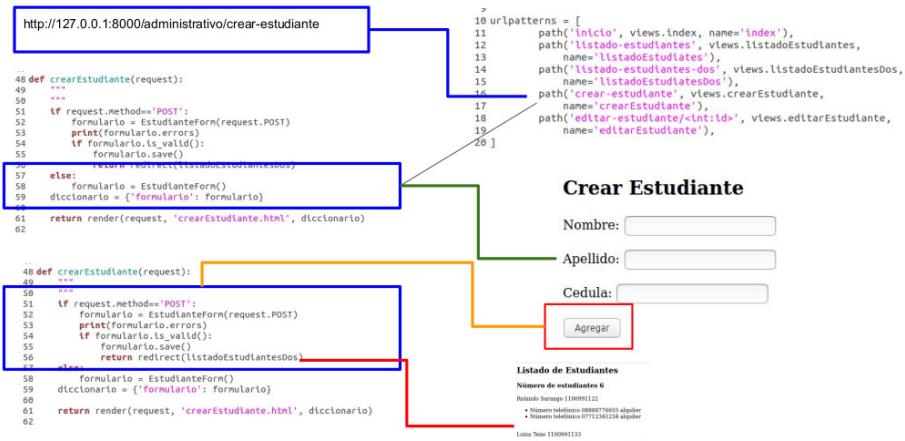


Figura 52. Proceso que permita crear un registro de tipo Estudiante, haciendo uso de la clase ModelForm

Fuente: Elizalde, R. (2020).

Ejemplo 2

Generar un proceso que permita editar un registro de tipo Estudiante, haciendo uso de la clase **ModelForm**.

- Se usa la clase llamada `EstudianteForm` del archivo `forms.py`
- En el archivo `views.py` se agrega una nueva función denominada **editarEstudiante**.
 - La función tiene dos argumentos;
 - El primero el objeto tipo `request`.
 - El segundo argumento permite recibir en la función una variable que permita buscar el objeto o registro de tipo Estudiante mediante el ORM.
 - Para buscar un estudiante a través del ORM en base al modelo Estudiante se usa la sentencia de la línea 67 (ver Figura 53).

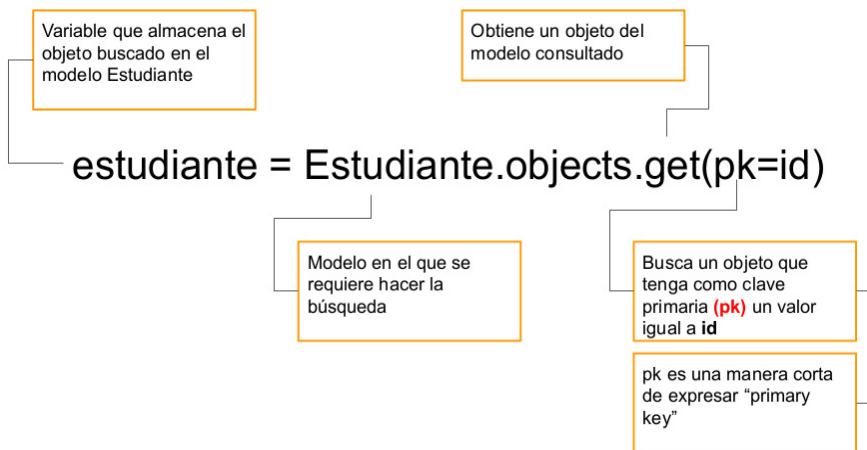


Figura 53. Consulta de un objeto de un modelo mediante la clave primaria
Fuente: Elizalde, R. (2020)

- En la línea 68 se realiza un condicional que verifica que la petición realizada sea un 'POST' (`request.method`); si esto es verdadero, se crea un objeto de tipo `EstudianteForm` llamado `formulario`, al cual se le envía como argumentos los parámetros que tenga el `request`. `POST` (los valores ingresados en el formulario, asociados a los atributos de la clase) y en el atributo `instance` se envía la instancia del objeto buscado. Si el formulario es válido (`formulario.is_valid`), a través de la función `save()` se guarda la información en la base de datos. Finalmente se realiza un redirecciónamiento (`redirect`) a la función `listadoEstudiantesDos`.
- Si el método recibido a través del objeto `request` es diferente de 'POST'; se crea un objeto llamado `formulario` de tipo `EstudianteForm`, enviando como argumento la instancia del objeto buscado (`instance=estudiante`); con el objetivo de cargar el formulario con los datos del objeto y proceder a realizar la edición.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
64 def editarEstudiante(request, id):
65     """
66     """
67     estudiante = Estudiante.objects.get(pk=id)
68     if request.method=='POST':
69         formulario = EstudianteForm(request.POST, instance=estudiante)
70         print(formulario.errors)
71         if formulario.is_valid():
72             formulario.save()
73             return redirect(listadoEstudiantesDos)
74     else:
75         formulario = EstudianteForm(instance=estudiante)
76     diccionario = {'formulario': formulario}
77
78 return render(request, 'editarEstudiante.html', diccionario)
```

- En el archivo editarEstudiante.html se crea un formulario en HTML y se renderiza el formulario enviado desde la vista.

Líneas de código

```
1 <h2>Editar Estudiante</h3>
2
3 <form action="" method="post">
4     {% csrf_token %}
5     {{formulario.as_p}}
6     <p><input type='submit' value='Actualizar' /></p>
7 </form>
```

- En el archivo urls.py se agrega la nueva ruta llamada **editar-estudiante que permite un parámetro tipo entero** que se asocia a la función editarEstudiante agregada en views.py (Figura 54).



Figura 54. Generar un path en el archivo urls.py con un argumento tipo entero

- Estimado estudiante puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo



Semana 14

5.3.6. Herencia de plantillas

Para el estudio del este apartado se recomienda seguir lo descrito en el capítulo 4 denominado ***El sistema de plantillas*** del recurso abierto *La guía definitiva de django – Desarrolla aplicaciones Web* de forma rápida y sencilla, de manera particular los apartados relacionados con ***Herencia de plantillas***.

Como se menciona en el recurso, la herencia de plantillas es una característica que permite crear un esqueleto – plantilla - template base -, donde se agregan partes que se pudieran usar en todas las plantillas de la aplicación; dejando expresado el conjunto de “bloques”, secciones que se pueden personalizar en las plantillas secundarias que heredan del template base.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Líneas de código

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6   <title>
7   {% block title %}
8   {% endblock %}
9   </title>
10  </head>
11  <body>
12    <h1>Aplicación Administración </h1>
13    <hr>
14    {% block content %}
15    {% endblock %}
16
17    <hr>
18    <footer>
19      <p>Loja-Ecuador</p>
20      <p>{% now "jS F Y H:i" %}</p>
21    </footer>
22  </body>
23 </html>
```

En las líneas descritas en el archivo master.html, se aprecia la estructura total de una página web; pero existen los denominados bloques, expresados con **{% block inicio %} ... {% endblock%}**. Estos bloques pueden ser particularizados en cada plantilla que herede de master.html. Todo lo que no está expresado entre un tag **block** será renderizado en la plantilla hijas.

La plantilla de nombre listadoEstudiantesDos.html queda de la siguiente manera:

Líneas de código

```
1 {% extends "master.html" %}  
2  
3 {% block title %}  
4 Listado de Estudiantes  
5 {% endblock %}  
6  
7 {% block content %}  
8  
9 <h2>Listado de Estudiantes</h3>  
10 <h3>Número de estudiantes {{ numero_estudiantes }}</h3>  
11  
12 {% for e in estudiantes %}  
13 ....  
14 {{e}}<br/>  
15 <ul>  
16 {% for numero in e.numerotelefonico_set.all%}  
17   <li>  
18     Número telefónico {{numero}}<br/>  
19   </li>  
20 {% endfor %}  
21 </ul>  
22 <br/>  
23 {% endfor %}  
24  
25 {% endblock %}
```

- La primera línea utiliza el tag **extends** junto al nombre de la plantilla base "master.html"; con lo cual se hereda la estructura la plantilla. En el archivo se personaliza los bloques "title" y "content" con el contenido que se requiere (Figura 55).

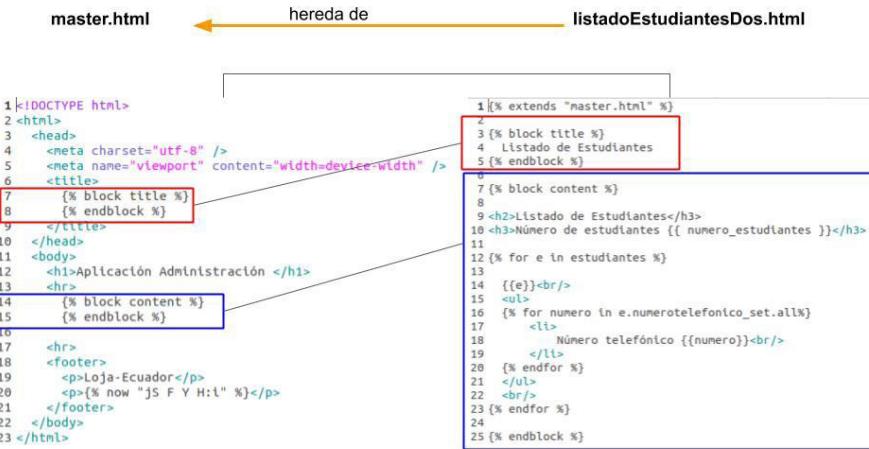


Figura 55. Herencia de plantillas en Django

Si usted inicia-levanta el proyecto y accede a la dirección:
<http://127.0.0.1:8000/administrativo/listado-estudiantes-dos>, observará la siguiente pantalla.

Captura de pantalla de un navegador web mostrando la visualización de la plantilla `listadoEstudiantesDos.html`. La URL en la barra de direcciones es `127.0.0.1:8000/administrativo/listado-estudiantes-dos`. La página muestra el encabezado "Aplicación Administración" y la sección "Listado de Estudiantes" con el título "Número de estudiantes 6". Se detallan los datos de seis estudiantes:

- Rolando Sarango 1100991122
 - Número telefónico 08888776655 alquiler
 - Número telefónico 07712341234 alquiler
- Luisa Tene 1100991133
 - Número telefónico 09988776655 particular
 - Número telefónico 07712341233 público
- Pedro Pérez 1134565678
 - Número telefónico 0550551234 alquiler
- Lucas Moreno 3312345678
- Lucia Andrade 7712345678
- Alex Mendoza Mendoza 2234567891

Al pie de la página se muestra "Loja-Ecuador" y la fecha "16th Julio 2020 20:10".

Figura 56. Visualización de una plantilla que hereda de "master.html"

- Estimado estudiante puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo

5.3.7. Agregar hojas de estilo a plantillas

Ahora que se entiende el proceso de manejo plantillas mediante herencia, es momento de agregar a la aplicación acceso a hojas de estilo - CSS.

Para las hojas de estilo se necesita agregar las rutas de los archivos CSS en la plantilla “**master.html**”. En el directorio de la aplicación se crea una estructura de directorios **static – css**; es ahí donde se debe agregar los archivos de las hojas de estilo. En el archivo “**master.html**” se usa el tag **load** para cargar los archivos del directorio **static** (Figura 57).

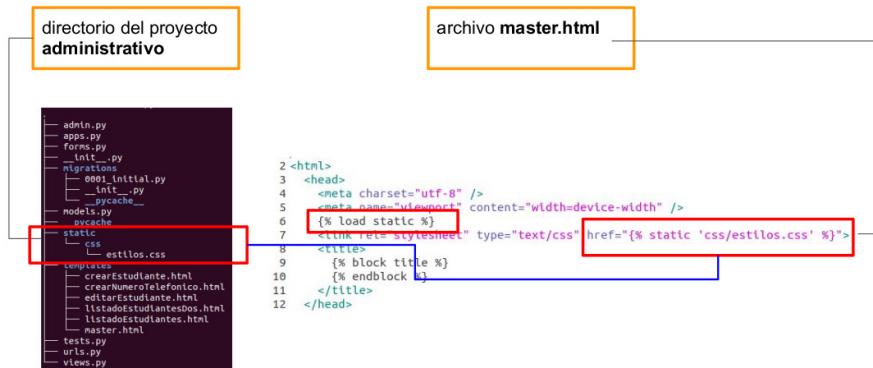


Figura 57. Agregar CSS a las plantillas de una aplicación en Django

Luego de agregar el archivo con las propiedades CSS en el archivo máster, todas las plantillas que heredan de la principal quedan de la manera indicada en la Figura 58.

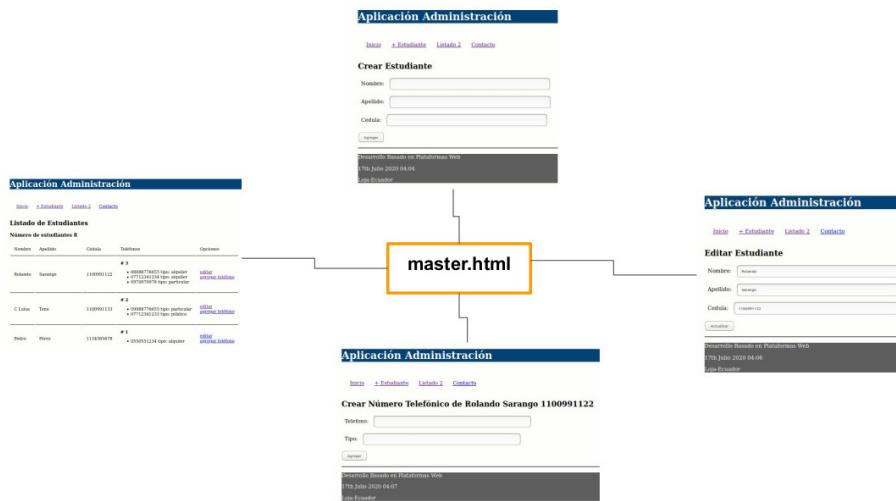


Figura 58. Aplicación de hojas de estilo en plantillas que heredan de master.html

A continuación, se detalla las opciones del pequeño sistema generado:

- Parte administrativa de la aplicación:
 - <http://127.0.0.1:8000/admin>
- Listado de estudiantes:
 - <http://127.0.0.1:8000/administrativo/listado-estudiantes-dos>
 - Ruta del archivo urls.py:
 - `path('listado-estudiantes-dos', views.listadoEstudiantesDos, name='listadoEstudiantesDos'),`
 - Función del views.py
 - `listadoEstudiantesDos`

- Template
 - listadoEstudiantesDos.html
- Agregar nuevo estudiante
 - http://127.0.0.1:8000/administrativo/crear-estudiante
 - Ruta del archivo urls.py:
 - path('crear-estudiante', views.crearEstudiante, name='crearEstudiante'),
 - Función del views.py
 - crearEstudiante
 - Template
 - crearEstudiante.html
- Editar estudiante
 - http://127.0.0.1:8000/administrativo/editar-estudiante/1
 - Ruta del archivo urls.py:
 - path('editar-estudiante', views.editarEstudiante, name='editarEstudiante'),
 - Función del views.py
 - editarEstudiante
 - Template
 - editarEstudiante.html

- Agregar número telefónico
 - <http://127.0.0.1:8000/administrativo/crear-telefono/1>
 - Ruta del archivo urls.py:
 - path('crear-telefono', views.crearTelefono, name='crearTelefono'),
 - Función del views.py
 - crearTelefono
 - Template
 - crearTelefono.html
- Estimado estudiante, puede descargar el ejemplo y ponerlo en funcionamiento en su entorno local.
 - Desarrollo del ejemplo



Actividad de aprendizaje recomendada

Actividad 2

Actividad de aprendizaje: Realizar la autoevaluación 6, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 5 Uso de frameworks de ambiente web, subsección 5.3. En la parte inicial de la actividad se presentan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes.

La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas



Autoevaluación 6

Es momento de medir el entendimiento de algunos conceptos estudiados en la segunda parte de la Unidad 5 Uso de frameworks de ambiente web. Usted puede revisar los siguientes temas:

- Ítems 5.3.4, 5.3.5, 5.3.6 y 5.3.7 de la presente guía.
- Recurso abierto denominado: La guía definitiva de django – Desarrolla aplicaciones web de forma rápida y sencilla
 - Capítulo 7: Procesamiento de formularios
 - Capítulo 4: El sistema de plantillas - apartado Herencia de plantillas

La autoevaluación le permitirá reforzar los conceptos estudiados. Los enunciados tienen una sola opción correcta, lea de forma atenta cada respuesta y seleccione la opción que usted considere correcta.

1. ¿Cuál es el archivo en dónde se maneja la lógica en el framework Django?

- a. Views.py.
- b. Models.py.
- c. Def.py.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

2. Dada la siguiente estructura identifique el nombre de la aplicación.

Líneas de código

```
manage.py  
ministerio  
asgi.py  
__init__.py  
__pycache__  
settings.py  
urls.py  
wsgi.py  
rrhh  
admin.py  
apps.py  
__init__.py  
migrations  
__init__.py  
models.py  
tests.py  
views.py
```

- a. Ministerio.
- b. Apps.
- c. Rrhh.

- 3. Dadas las siguientes líneas de código que forman parte de una función del archivo views.py; identifique que tipo de datos se pasan como contexto al template o plantilla.**

Líneas de código

```
estudiantes = Persona.objects.all()
mensaje = "Listado de Personas"
numeroPersonas = len(estudiantes)
informacion_template = {'personas': estudiantes, 'numero_estudiantes':
numeroPersonas, 'titulo': mensaje}
return render(request, 'listadoEstudiantes.html', informacion_template)
```

- a. 3 variables: 1) listado de objetos de tipo Persona; 2) numero_estudiantes tipo entero; 3) titulo de tipo cadena
 - b. 3 variables: 1) listado de objetos de tipo Estudiante; 2) numero_estudiantes tipo entero; 3) titulo de tipo cadena
 - c. 3 variables: 1) listado de objetos de tipo Persona; 2) numero_estudiantes tipo cadena; 3) titulo de tipo cadena
- 4. En los templates de Django, ¿cómo se representan las variables?**
- a. { variable }
 - b. {{ variable }}
 - c. [[variable]]
- 5. A través del uso de tags en Django, identifique el ciclo repetitivo que se pueden representar en los templates.**
- a. While.
 - b. Do-while.
 - c. For.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

6. ¿Cuál es la clase que se puede usar para crear un formulario a partir de una clase definida en el modelo?
- Form.
 - ModelForm.
 - ModelForms.
7. ¿Cuál es la directiva que se usa en Django para prevenir ataques de sitios que generan solicitudes falsas?
- {% csrf_token%}
 - {{ csrf_token }}
 - {% csrfToken%}
8. Dado el siguiente archivo y en función de una petición generada a través de la URL 127.0.0.1:8000/administrativo/1/reporte-general, identifique la vista a la cual se hace referencia.

Líneas de código

```
url.py del proyecto
urlpatterns = [
    path('administrativo/', include('administrativo.urls')),
]
```

```
url.py de la aplicación
urlpatterns = [
    path("", views.listadoEstudiantesDos,
          name='listadoEstudiantesDos'),
    path('editar-estudiante/<int:id>', views.editarEstudiante,
          name='editarEstudiante'),
    path('<int:id>/reporte-general', views.crearReporte,
          name='crearReporte'),
]
```

- Views.crearReporte.
- Views.
- CrearReporte.

Líneas de código

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="viewport" content="width=device-width" />
6     <title>
7       {% block title %}</title>
8       {% endblock %}
9   </head>
10  <body>
11    <h1>Aplicación Autoevaluación </h1>
12    <hr>
13    {% block content %}</hr>
14    {% endblock %}
15
16
17    <hr>
18    <footer>
19      <p>Loja-Ecuador</p>
20      <p>{% now "jS F Y H:i" %}</p>
21    </footer>
22  </body>
23 </html>
```

- a. {% extends "principal.html" %}
 {% title %}
 Crear Estudiante
 {% endblock %}
 {% content %}
 <h2>Crear Estudiante</h3>
 <form action="" method="post">
 {% csrf_token %}
 {{formulario.as_p}}
 <p><input type='submit' value='Aregar' /></p>
 </form>
 {% endblock %}

Índice

Primer
bimestre

Segundo
bimestre

Solucionario

Referencias
bibliográficas

- b.
- ```
{% extends "principal.html" %}
{% block title %}
Crear Estudiante
{% end %}
{% block content %}
<h2>Crear Estudiante</h3>
<form action="" method="post">
{% csrf_token %}
{% formulario.as_p %}
<p><input type='submit' value='Aregar' /></p>
</form>
{% end %}
```
- c.
- ```
{% extends "principal.html" %}  
{% block title %}  
Crear Estudiante  
{% endblock %}  
{% block content %}  
<h2>Crear Estudiante</h3>  
<form action="" method="post">  
{% csrf_token %}  
{% formulario.as_p %}  
<p><input type='submit' value='Aregar' /></p>  
</form>  
{% endblock %}
```

10. Dado el siguiente modelo de un proyecto en Django; identifique la clase que permita generar un formulario a través de ModelForm.

Líneas de código

```
class Docente(models.Model):
    nombre = models.CharField(max_length=30)
    apellido = models.CharField(max_length=30)
    identificacion = models.CharField(max_length=30, unique=True)
    def __str__(self):
        return "%s %s %s" % (self.nombre,
                             self.apellido,
                             self.identificacion)
```

- a. class DocenteForm(models.ModelForm):
class Meta:
 model = Docente
 fields = ['nombre', 'apellido', 'identificacion']
- b. class DocenteForm(ModelForm):
 model = Docente
 fields = ['nombre', 'apellido', 'identificacion']
- c. class DocenteForm(ModelForm):
 class Meta:
 model = Docente
 fields = ['nombre', 'apellido', 'identificacion']

[Ir al solucionario](#)



Actividades finales del bimestre



Semana 15



Actividades de aprendizaje recomendadas

En la semana 15 de estudio se recomienda volver a revisar, analizar y contestar las preguntas de las autoevaluaciones del segundo bimestre.

Actividad 1

- Actividad de aprendizaje: realice la autoevaluación 1, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 1 Conceptos generales de desarrollo de plataformas web. En la parte introductoria de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes. La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Actividad 2

- Actividad de aprendizaje: Realice la autoevaluación 2, donde se exponen un conjunto de preguntas en función de las temáticas descritas en la unidad 2 *Programación del lado del cliente*. En la parte introductoria de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes. La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica

Actividad 3

- Actividad de aprendizaje: Realice la autoevaluación 3, donde se presentan un conjunto de preguntas en función de las temáticas descritas en la unidad 3 *Acceso a base de datos relacionales mediante Object Relational Mapper (ORM)*. En la parte inicial de la actividad se listan los contenidos de la guía didáctica, texto básico y recursos educativos abiertos que se deben revisar para contestar las interrogantes. La autoevaluación le permitirá identificar el nivel de aprendizaje alcanzado y los ítems que se debe volver a revisar para lograr los resultados de aprendizaje propuestos. Además, usted podrá plantearse interrogantes o ejercicios similares para reafirmar lo analizado y estudiado. Luego de realizar la autoevaluación, puede revisar las respuestas correctas de las preguntas en la sección Solucionario de la guía didáctica.



Semana 16

Distinguidos estudiantes.

Hemos llegado al final de nuestro estudio, la semana 16 que es un período de tiempo destinado para que usted pueda reforzar temáticas no comprendidas en el transcurso del segundo bimestre, con el objetivo de prepararse de la mejor manera para rendir la evaluación presencial del bimestre. Por ello, se plantea el repaso de las siguientes unidades, generando resúmenes, cuadros sinópticos, mapas conceptuales, entre otros.

- Unidad 4. Programación en el servidor web.
- Unidad 5. Uso de *frameworks* de ambiente web.

Además, es importante realizar las siguientes acciones:

- Revisar los recursos de aprendizaje recomendados en el plan docente de la asignatura.
- Plantearse ejercicios y problemáticas similares a las expuestas en los contenidos de la guía.
- Revisar las preguntas que conforman los cuestionarios de repaso del bimestre planteados en el entorno virtual.



4. Solucionario

Autoevaluación 1		
Pregunta	Respuesta	Retroalimentación
1	b	En el punto 1.1. Características deseables de un sitio web de la guía didáctica; se listan las propiedades de calidad de software con el desarrollo de sitios web. En la lista se encuentra la propiedad: corrección y funcionalidad.
2	c	En el punto 1.1. Características deseables de un sitio web de la guía didáctica; se listan las propiedades de calidad de software con el desarrollo de sitios web. La seguridad se encarga de prevenir la inyección SQL maliciosa.
3	c	La W3C es una comunidad internacional que trabaja en la generación de estándares para la web a nivel global. Se enfoca de igual manera en garantizar el acceso a Internet a cualquier persona, independiente de su condición.
4	b	HTTPS (HyperText Transport Protocolo Secure), permite utilizar una conexión con servidores seguros de Internet. El puerto usado por el protocolo es 443.
5	b	Una URL, Uniform Resource Locator está formada por Protocolo: HTTP o HTTPS; Dominio;Ruta.
6	c	En punto 1.3. Servicios en la Nube (cloud) se muestran un conjunto de ideas afirmativas en relación a servicios en la nube. Una de ellas menciona que es un modelo que permite acceder a diversos servicios como: almacenamiento en servidores, acceso a redes de información.

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Autoevaluación 1		
Pregunta	Respuesta	Retroalimentación
7	a	La correcta definición de HTML es: lenguaje de etiquetado de documentos que permite la generación de páginas.
8	a	El texto básico indica algunos complementos para XML: XSL, XPath, XLink, XPointer y XQL.
9	b	En el texto básico se indica que hosting es el proceso de alojar una página web en un servidor.
10	c	Las características de un equipo servidor son muy singulares en cuanto al almacenamiento y procesamiento de información.

Ir a la
autoevaluación

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Autoevaluación 2		
Pregunta	Respuesta	Retroalimentación
1	b	Existe un conjunto de recomendaciones para crear aplicaciones web intuitivas, una de ella es la de generar pruebas con usuarios reales.
2	a	Al inicio de un archivo HTML se establece la versión que será interpretada; por ejemplo: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
3	c	Se recomienda revisar que los contenidos generados en la aplicación web estén bajo las recomendaciones de la W3C; para ello se debe usar la página web https://validator.w3.org/ .
4	c	La forma correcta de establecer como conjunto de caracteres a interpretar: UTF-8 es: <meta charset="utf-8" />.
5	b	En la línea 12 se necesita especificar en la etiqueta p un estilo que permita establecer un fondo azul y letras de color blanco. Para usar el estilo del archivo estilos.css se debe hacer referencia a través del atributo class. De la forma: <p class="tres">Cédula</p>
6	b	En JavaScript cuando se usa el método innerText para establecer un valor; el mismo se representa tal cual fue establecido. Respuesta correcta: 9 10
7	a	La forma correcta de establecer el acceso a un archivo externo JavaScript es de la forma: <script src="https://code.jquery.com/jquery-3.5.0.js"></script>
8	a	Del conjunto de modelos usados en base de datos NoSql: las base CouchDB usa documentos y Neo4j usa grafos.
9	c	La interfaz de acceso web de CouchDB se llama Fauxton y la dirección por defecto de acceso es: http://127.0.0.1:5984/_utils/



Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Autoevaluación 2

Pregunta	Respuesta	Retroalimentación
10	c	<p>La forma correcta de agregar documentos a una base de datos CouchDB es:</p> <pre>{ "_id": "aac322d3ae479d4c3434ae6587001944", "ciudad": "Quito", "Provincia": "Pichincha" }</pre>

Ir a la
autoevaluación

Autoevaluación 3		
Pregunta	Respuesta	Retroalimentación
1	a	De acuerdo al punto 3.1 Introducción de la unidad Acceso a base de datos relacionales mediante Object Relational Mapper (ORM); se considera como afirmativa: Pasar objetos de clases realizadas en lenguajes de programación a registros en base de datos relacionales.
2	b	Una de las características de los ORM es cambiar de motor de base datos en cualquier momento.
3	b	Del listado de ORM's expuestos en la pregunta Doctrine es basado en lenguaje PHP.
4	a	La forma correcta de instalación de un librería en Python a través del gestor de paquetes pip es: <code>pip install nombre-librería</code> La respuesta correcta es: <code>pip install SQLAlchemy</code>
5	a	se crea una variable que haga uso del módulo <code>create_engine</code> , a la cual se envía como parámetro una cadena de texto que indica el motor a usar. La respuesta correcta es: <code>engine = create_engine('sqlite:///demobase.db')</code>
6	b	El proceso correcto es: <code>from sqlalchemy.ext.declarative import declarative_base engine = create_engine(<<cadena_base_datos>>) Base = declarative_base() Base.metadata.create_all(engine)</code>
7	c	En la sentencia: <code>datos = session.query(Hospital).filter(Hospital.numero_camas==80).all()</code> Se filtran todos los objetos de tipo Hospital que tengan el atributo numero_camas igual a 80. La respuesta correcta es: <i>Presentación de Hospitales</i> <i>Hospital: Machala - camas: 80 - pisos: 8</i> <hr/> <i>Hospital: Luis Vernaza - camas: 80 - pisos: 10</i> <hr/>

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Autoevaluación 3

Pregunta	Respuesta	Retroalimentación
8	c	<p>La sentencia:</p> <pre>datos = session.query(Hospital).order_by(Hospital.numero_pisos).all()</pre> <p>Permite obtener todos los objetos de tipo Hospital ordenados de menor a mayor en función del atributo numero_pisos.</p> <p>La respuesta correcta es:</p> <p><i>Presentación de Hospitales</i></p> <p><i>Hospital: Machala - camas: 80 - pisos: 8</i></p> <p>_____</p> <p><i>Hospital: Isidro Ayora - camas: 100 - pisos: 10</i></p> <p>_____</p> <p><i>Hospital: Luis Vernaza - camas: 80 - pisos: 10</i></p> <p>_____</p> <p><i>Hospital: Andrade Marín - camas: 200 - pisos: 20</i></p> <p>_____</p>
9	b	<p>La sentencia:</p> <pre>datos = session.query(Hospital).filter(Hospital.nombre.like("%Ay%")).all()</pre> <p>Permite obtener los objetos de tipo Hospital que contengan en el valor del atributo nombre la cadena "Ay".</p> <p>La respuesta correcta es:</p> <p><i>Presentación de Hospitales</i></p> <p><i>Hospital: Isidro Ayora - camas: 100 - pisos: 10</i></p> <p>_____</p> <p><i>Hospital: Pedro Ayora - camas: 80 - pisos: 10</i></p> <p>_____</p>
10	a	<p>En SQLAlchemy la forma correcta de identificar a una columna como clave foránea es a través de:</p> <pre>persona_id = Column(Integer, ForeignKey('persona.id'))</pre>

Ir a la
autoevaluación

Autoevaluación 4		
Pregunta	Respuesta	Retroalimentación
1	c	Del listado de opciones: Apache y Nginx son considerados como servidores web.
2	c	El código PHP debe ir entre las etiquetas: <?php ?> Por tal razón la opción correcta es la que tiene la porción de código de la siguiente forma: <?php echo "Verificando conocimientos" ; ?>
3	c	La porción de código en PHP <?php function operacion(\$num1, \$num2){ return \$num1 + \$num2; } \$tabla = 3; \$inicio = 1; echo '<h3>Tabla del '. \$tabla, "</h3>"; while(\$inicio <= 5){ \$formato = "<tr> <td>%d</td><td>%s</td><td>%d</td><td class='celda'>%d</td> </tr>"; echo sprintf(\$formato, \$inicio, "+", \$tabla, operacion(\$tabla, \$inicio)); \$inicio++; } ?> de acuerdo a la lógica permite generar una salida como la expuesta en la opción c.
4	b	El proceso repetitivo expuesto en el código PHP junto a la estructura condicional genera una salida como la imagen de la opción b.
5	a	El proceso repetitivo expuesto en el código PHP y el uso de las funciones operación y operacion2, genera una salida como la imagen de la opción b.

Autoevaluación 4		
Pregunta	Respuesta	Retroalimentación
6	c	Para capturar los valores enviados a través de un formulario con el método POST es a través del uso de <code>\$_REQUEST</code> y para acceder a las variables necesarias se lo hace con el nombre del atributo como cadena.
7	b	La forma correcta de crear un objeto que permita crear un enlace a una base de datos MariaDB o Mysql es haciendo uso de la clase: <code>mysqli</code>
8	c	La forma correcta de realizar un redirect en PHP es: <code>header("Location: archivo-destino.php?variable="valor");</code>
9	a	En el archivo se genera un formulario que permite crear un conjunto de tags: <code>input type=radio</code> y <code>label</code> . Se hace uso de un ciclo repetitivo que en cada iteración llama una función llamada <code>información</code> que retorna una cadena.
10	c	Haciendo uso de <code>date('Ymd',strtotime(\$v))</code> ; se presenta una fecha de la forma: Añomesdia – 20200725

Ir a la
autoevaluación

[Índice](#)[Primer bimestre](#)[Segundo bimestre](#)[Solucionario](#)[Referencias bibliográficas](#)

Autoevaluación 5		
Pregunta	Respuesta	Retroalimentación
1	a	En el punto 5.1. Modelo-vista-controlador base de los <i>frameworks</i> , se indican algunas temáticas que se abordan cuando se usa un <i>framework</i> . En base a ello la respuesta que tiene las opciones: Lenguaje de programación, librerías y bibliotecas; es la correcta.
2	b	En el punto 5.2. Clasificación de <i>frameworks</i> de ambiente web según los lenguajes de programación, se listan algunos <i>frameworks</i> con sus respectivos sistemas de plantillas. Twing y Blade son usados por <i>frameworks</i> que se basan en lenguaje PHP.
3	a	La forma correcta para crear un proyecto en Django es haciendo uso del comando django-admin startproject [nombre-proyecto]
4	b	Luego de haber creado el proyecto; el uso del manage.py permite generar aplicaciones, actualizaciones, crear usuarios administradores entre otras tareas.
5	c	El comando python manage.py migrate permite crear o actualizar la base de datos.
6	a	Para crear aplicaciones en Django se usa el comando: python manage.py startapp [nombre-aplicación]
7	b	Dada una entidad llamada Universidad, la forma correcta de obtener los registros guardados en la base de datos es a través del uso de la siguiente sentencia en lenguaje Python. Universidad.objects.all()
8	a	Para verificar si una cadena está dentro de los valor de un atributo se usa: nombreAtributo__contains="cadena" Por tal razón la respuesta correcta es: Universidad.objects.filter(direccion__contains="Loja").all()
9	b	El atributo que permite mostrar los atributos de un modelo de una aplicación en la interfaz de administración es: list_display
10	a	Mediante el comando: python manage.py createsuperuser ; se crea un súper usuario del proyecto de Django

[Ir a la autoevaluación](#)

Autoevaluación 6		
Pregunta	Respuesta	Retroalimentación
1	a	Bajo la filosofía MVT del framework Django, el archivo views.py es el archivo donde agrega la lógica de una aplicación.
2	c	Dada la estructura y asumiendo que se ha creado una aplicación en función del comando correcto; el nombre de la aplicación es: rrhh
3	a	Con base a las líneas de código que forman parte de una vista; la opción correcta es: 3 variables: 1) listado de objetos de tipo Persona 2) numero_estudiantes tipo entero; 3) titulo de tipo cadena.
4	b	La forma correcta para representar el valor de un variable en un template de Django es {{ }}
5	c	En Django a través del sistema de plantillas se puede representar un ciclo repetitivo for {% for %} {% endfor %}
6	b	Django permite crear formularios a partir de los modelos descritos en el archivo models.py; para ello usa la clase ModelForm
7	a	La representación de un tag en un template de Django es bajo la forma: {% tag %} Django usa el tag con nombre csrf_token para prevenir ataques de sitios que generan solicitudes falsas.
8	c	Cuando se realiza una petición, se recibe la misma en el archivo url.py del proyecto; path('administrativo/', include('administrativo.urls')), luego en el archivo url de la aplicación, la petición coincide con el path: path('<int:id>/ reporte-general ', views.crearReporte, name='crearReporte') Por lo tanto el nombre de la vista es: crearReporte
9	c	La forma correcta de indicar un bloque para personalizar es: {% block inicio %} ... {% endblock%}; donde inicio es el nombre del bloque

Índice

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas

Autoevaluación 6

Pregunta	Respuesta	Retroalimentación
10	c	La opción que implementa de forma correcta un formulario a través de ModelForm es la opción c; misma que hace uso de la subclase Meta y atributos model y fields.

Ir a la
autoevaluación



5. Referencias bibliográficas

Colomina Pardo, O., Arques Corrales, P., & Montoyo-Bojo, J. (2011). Tecnologías Web. Tema 4: Frameworks JavaScript. Jquery.

jQuery UI (2020). Recuperado de <http://learn.jquery.com/jquery-ui/gettingstarted/>

Herrera, H. A., & Valenzuela, C. R. (2016). NoSQL, la nueva tendencia en el manejo de datos. *Tecnología Investigación Y Academia*, 4(1), 147-150.

D. Robles, M. Sánchez, R. Serrano, B. Adárraga & D. Heredia. “¿Qué características tienen los esquemas NoSQL?”

Investigación y Desarrollo en TIC, vol. 6, no. 1, pp. 40-44, 2015.

Museros Cabedo, L., & Sanz Blasco, I. (2010). Tema 6: Otros Modelos de Bases de Datos. El modelo orientado a objetos y objeto-relacional.

Riera, F. B., Maimo, E. H. G., & Martín, J. P. OBJECT/RELATIONAL MAPPING.

Viñals, J. T. (2012). Del cloud computing al big data. Barcelona. universitat oberta de catalunya..

Learning sqlalchemy (2020). Recuperado de <https://riptutorial.com/Download/sqlalchemy.pdf>

Índice

Django, la guía definitiva (2020). Recuperado de <https://pythonizame.s3.amazonaws.com/media/Book/guia-definitiva-django-18/file/34ba425e-5985-11e5-964d-04015fb6ba01.pdf>

Gutiérrez González, Á., & López Goytia, J. L. (2016). Desarrollo y programacion en entornos web. México. Alfaomega.

Primer bimestre

Segundo bimestre

Solucionario

Referencias bibliográficas