INSTITUTO SUPERIOR
TECNOLÓGICO
VICENTE LEÓN

Guía general de estudio de la asignatura

TENDENCIAS ACTUALES DE PROGRAMACIÓN

Christian Mauricio Tigse Pilla

Carrera de Tecnología Superior en Desarrollo de Software
Asignatura: Tendencias Actuales de Programación
Código de la asignatura: DS28-5T202
Quinto nivel

www.istvicenteleon.edu.ec

INSTITUTO SUPERIOR
TECNOLÓGICO
VICENTE LEÓN
Belisario Quevedo #501 / Latacunga – Cotopaxi
Campus Matriz

TENDENCIAS ACTUALES DE PROGRAMACIÓN

Autor: Christian Mauricio Tigse Pilla

MSc. Ángel Velásquez Cajas Editor

Directorio editorial institucional

Mg. Omar Sánchez Andrade Rector
Mg. Fabricio Quimba Herrera Vicerrector
Mg. Milton Hidalgo Achig Coordinador de la Unidad de Investigación

Diseño y diagramación

Mg. Alex Zapata Álvarez
Mtr. Leonardo López Lidioma

Revisión técnica de pares académicos

– Andrés Marcelo Salinas Copo
Instituto Superior Tecnológico Bolívar
a.salinas@institutos.gob.ec
– Luis Gonzalo Borja Almeid
Universidad de las Fuerzas Armadas ESPEL
lgborja2@espe.edu.ec

ISBN: 978-9942-676-41-2
Primera edición
Agosto 2024

RIMANA EDITORIAL

Índice

DESARROLLO GUÍA DE ESTUDIO

DESARROLLO GUÍA DE ESTUDIO

1. Datos informativos

Christian Mauricio Tigse Pilla, Ingeniero en informática y Sistemas Computacionales y Maestría en Ingeniería en Software, he trabajado en empresas públicas y privadas en el área de desarrollo de software y soporte técnico de hardware y software, arquitecto de software, programador freelance, consultor software freelance y catedrático del Instituto Superior Vicente León.

2. Presentación de la Asignatura

En la asignatura es indispensable las bases técnicas como un Web Service o Servicio Web, es un método de comunicación entre dos aparatos electrónicos en una red. Es una colección de protocolos abiertos y estándares usados para intercambiar datos entre aplicaciones o sistemas. Las aplicaciones escritas en varios lenguajes de programación que funcionan en plataformas diferentes pueden utilizar web services para intercambiar información a través de una red. La interoperabilidad, por ejemplo, entre Java, Python o Windows y Linux se debe al uso de estándares abiertos. La computación en la nube son varios servidores desde Internet encargados de atender las peticiones en cualquier momento. Se puede tener acceso a su información o servicio, mediante una conexión a internet desde cualquier dispositivo móvil o fijo ubicado en cualquier lugar. Sirven a sus usuarios desde varios proveedores de alojamiento repartidos frecuentemente por todo el mundo. Esta medida reduce los costos, garantiza un mejor tiempo de actividad y que los sitios web sean invulnerables a los delincuentes informáticos, a los gobiernos locales y a sus redadas policiales pertenecientes.

3. Introducción de los Temas

– La guía presenta los siguientes temas:

Unidad I: Tratara de aspecto relevantes referentes web service, como conceptos básicos, su arquitectura, la creación mediante framework que se

usan en la actualidad, el funcionamiento y la manera adecuada de usar un web server en la actualidad, así como las distintas seguridades que se aplica a un web service al poner en producción las mismas.

4. Objetivos de Aprendizaje

Desarrollar aplicaciones en BackEnd con framework basados en JavaScript (API Rest).

Administrar el versionamiento de aplicaciones utilizando un sistema de control de versiones.

5. Unidad y Subunidades

1.Introducción a SOA

1.1. Conceptualización y generalización de SOA

1.2.Características y tipos de SOA

1.3.Creación de una API Rest.

1.4.Uso de framework basados en JavaScript

1.5.Gestión de Base de datos.

1.6.Creación de un CRUD (API Rest).

6. Resultados de Aprendizaje

– Desarrollar aplicaciones en BackEnd con framework basados en JavaScript (API Rest).

– Administrar el versionamiento de aplicaciones utilizando un sistema de control de versiones.

7. Estrategias Metodológicas

Tabla 1

Estrategias metodológicas

Estrategias Metodológicas	Finalidad	Técnicas
Experiencia Concreta	Explora los saberes empíricos con los que llegan sus participantes, a través de lluvias de ideas, preguntas –respuestas, relato de anécdotas, conversatorios, diario comunitario, entre otros; en relación con la temática a ser tratada durante la clase.	investigaciones, juegos, canciones, observación directa, visitas técnicas, collage, proyecciones, viaje imaginario sustentado en la práctica real docente, Experimentación, acertijos.
Reflexión	Desde una situación comunicativa contextualizada a su realidad, plantea el tema utilizando, lecturas científicas o informativas, leyendas, mitos, amorfinos, videos, gráficos o situaciones problemáticas, debates, con el fin de inducir a los participantes a conectar sus conocimientos previos con la nueva información que se les provee.	lluvia de ideas, diálogos, discusiones, foros, conversatorios, rueda de atributos.
Conceptualización	La mediación del docente debe estar dirigida a actividades como la presentación de la nueva información (contenidos curriculares)	organizadores gráficos, cuadros comparativos, resúmenes, esquemas sintéticos, ilustraciones, análisis, síntesis, procedimientos, protocolos, exposiciones, etc.

Aplicación	La concreción del aprendizaje debe reflejar la adquisición de los nuevos contenidos conectados con los saberes y experiencias anteriores.	organizadores gráficos, cuadros comparativos, resolución de ejercicios, elaboración de informes, producción de textos, construcción y solución de cuestionarios, elaboración de carteles, maquetas, afiches, debates, dramatizaciones, teatro, exposiciones, etc.
Los Recursos Didácticos		
Materiales Convencionales	-Impresos (textos): libros, fotocopias, periódicos, documentos, etc. -Tableros didácticos: pizarra y franelógrafo. -Materiales manipulativos: recortables, cartulinas. -Juegos: arquitecturas, juegos de sobremesa. -Materiales de laboratorio.	
Materiales Audiovisuales	-Imágenes fijas proyectables (fotos): diapositivas y fotografías. -Materiales sonoros (audio): casetes, discos programas de radio.t -Materiales audiovisuales (vídeo): audiovisuales, películas y vídeos. y montajes -Programas de televisión.	
Nuevas Tecnologías	-Programas informáticos (CD u on-line). -Educativos: videojuegos, lenguajes de autor, actividades de aprendizaje, presentaciones, multimedia, enciclopedias, animaciones y simulaciones. -Servicios telemáticos: páginas web, weblogs, tours, virtuales, webquest, cazas del tesoro, correo electrónico, chats, foros, unidades didácticas. -TV y vídeo interactivo.	

8. Criterios de Evaluación

A continuación, se detalla en la guía como es el método de evaluación:

Fases	Instrumentos	Primer Parcial %(puntos)	Segundo Parcial %(puntos)	Promedio %(puntos)
Fase 1: Trabajos Prácticos	Trabajos Individual	2	2	2
	Trabajo de clase o colaborativo	2	2	2
	Exposiciones	2	2	2
Fase 2: Lecciones	Escritas	2	2	2
Fase 3: Evaluación	Cuestionario	2	2	2
Total:		10	10	10

Horas Docencia= 90
Horas de Trabajo Autónomo= 40
Horas de prácticas experimentales= 72

9. Desarrollo de las Subunidades

Introducción a SOA

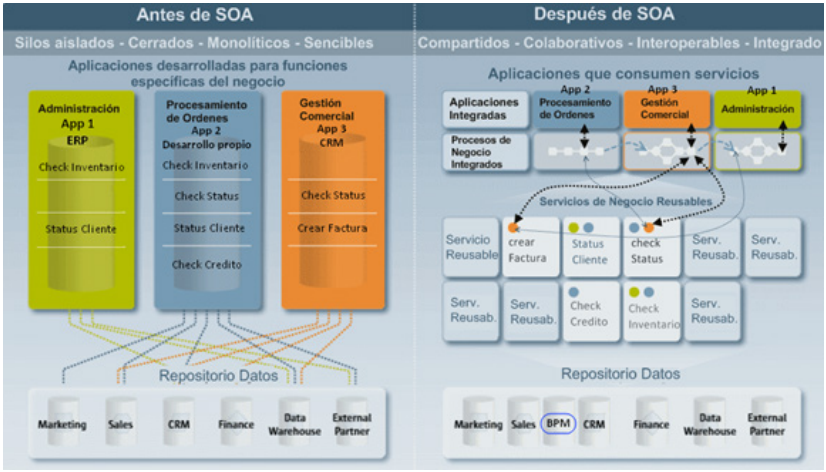
Según (Jesús David Calle, 2021) menciona que SOA es un paradigma de arquitectura de software moderno que sirve para el diseño de aplicaciones como un conjunto de entidades dentro de los sistemas llamados servicios. Los servicios SOA son autónomos, modulares y relativamente independientes.

El autor lo que menciona es que es un enfoque que permite desarrollar aplicaciones descentralizadas complejas donde se puede actualizar componentes individuales en lugar de la aplicación monolítica completa.Según (Hernández, 2023) menciona que “La arquitectura orientada a servicios o

SOA consiente interconectar y acceder a los servicios a través de una interfaz frecuente independiente de la tecnología con la que se ha desarrollado el servicio (ver figura 1)."

Figura 1

Arquitectura Orientada a Servicios



Nota. Estructura de una arquitectura Orientada en Servicio o SOA.Tomado marcosmilanesio.blogspot.com [figura], ¿Que es SOA?,2010,https://marcosmilanesio.blogspot.com/2010/04/bpm-que-es-soa.html

Beneficios de SOA

Para (Holley, 2020) dice que "SOA pone especial énfasis en la reutilización de sus componentes, las arquitecturas o tecnologías orientadas a servicios se puede reutilizar en el desarrollo de nuevos servicios realizado y así obteniendo más servicios".Los servicios que creamos o diseñamos deben ser consistentes con un conjunto de premisas o estar diseñados de tal manera que podamos identificar quién puede acceder a ellos y cómo se puede acceder a ellos para que nuestro catálogo sea fácil de entender y reutilizable.

Entonces, el resultado es que tendemos a alejarnos de las infraestructuras basadas en silos, hacia entornos separados que operan de

manera completamente independiente, y hacia sistemas que son la suma de múltiples servicios que brindan un servicio de extremo a extremo que nos permite responder a las necesidades de la respectiva empresa. (Jesús David Calle, 2021)

Ejemplos Arquitectura SOA

Según (Vaca, 2022) menciona que:

HTTP (Hypertext Transfer Protocol). Los protocolos se utilizan ampliamente en la World Wide Web y es la base para la comunicación en SOA. HTTP se la utiliza para transferir datos en forma de solicitudes y respuestas entre un cliente y servidor.

SOAP (Simple Object Access Protocol). Un protocolo basado en XML que proporciona una estructura estándar para intercambiar mensajes entre servicios en SOA.

REST (Representational State Transfer). Aunque REST no es un protocolo en sí mismo, es un estilo arquitectónico que utiliza los principios y la funcionalidad del protocolo HTTP. Se basa en el concepto de recursos identificados por URL y realiza operaciones sobre estos recursos utilizando métodos HTTP (GET, POST, PUT, DELETE).

JSON (JavaScript Object Notation). Aunque JSON no es un protocolo, se suele utilizar como formato de intercambio de datos en SOA, especialmente en servicios web RESTful.

XML-RPC (XML Remote Procedure Call). es otro protocolo basado en XML que permite invocaciones de procesos remotos entre sistemas que utilizan HTTP como transporte.

AMQP (Advanced Message Queuing Protocol). Aunque no es tan común como antes, AMQP es un protocolo de mensajería utilizado para facilitar la comunicación asincrónica entre servicios en SOA.

1.1.Conceptualización y Generalización de SOA

SOA (Arquitectura Orientada a Servicios) es un método para reutilizar componentes de software a través de interfaces de servicios. Estos servicios utilizan estándares de interfaz y patrones arquitectónicos habituales para que puedan incorporarserápidamente a nuevas aplicaciones. Esto evita la repetición de tareas que requerían los desarrolladores de aplicaciones para no replicar las funcionalidades existentes. (IBM, 2022)

"Cada servicio SOA contiene el código y la integración de datos necesarios para realizar una función comercial completa y discreta, como verificar el crédito de un cliente, calcular el pago mensual de un préstamo o procesar una solicitud de hipoteca". (Vaca, 2022)

Según (Ana Virginia Flores, 2013) menciona que Es un contrato de servicio entre un proveedor de servicios y un consumidor de servicios. La aplicación detrás de la interfaz de servicio puede escribirse en Java, Microsoft .Net, Cobol o cualquier otro lenguaje de programación disponible como una aplicación de software empaquetada por el proveedor (como SAP), una aplicación SaaS (como Salesforce CRM) o fuera de él. estante.

Como una aplicación de código abierto. Las interfaces de servicio generalmente se definen utilizando el lenguaje de descripción de servicios web(WSDL), una estructura de marcado estándar basada en el lenguaje de marcado extensible (XML).

Exponga el servicio utilizando protocolos de red estándar como elProtocolo simple de acceso a objetos (SOAP)/HTTP o Restful HTTP (JSON/HTTP) para enviar solicitudes de lectura o modificación de datos. La gestión de servicios controla el ciclo de vida del desarrollo. Cuando es necesario, los servicios se publican en un registro, lo que permite a los desarrolladores encontrarlos y reutilizarlos rápidamente para crear nuevas aplicaciones o procesos comerciales.

Según (Arsaute, A., Zorzán, FA, Daniele, M., González, A., & Frutos, M., 2018) menciona que "Estos servicios se pueden diseñar desde cero, pero

normalmente se crean exponiendo la funcionalidad de registro heredada como una interfaz de servicio".

Por lo tanto, SOA representa un paso importante en el desarrollo e integración de aplicaciones en las últimas décadas. Antes de la llegada de SOA a finales de la década de 1990, conectar una aplicación a datos o funciones alojadas en otro sistema requería una compleja integración punto a punto que los desarrolladores tenían que reconstruir completa o parcialmente para cada nuevo proyecto de desarrollo.

Al exponer estas capacidades a través de servicios SOA, los desarrolladores pueden simplemente reutilizar las capacidades existentes y conectarlas usando la arquitectura SOA ESB (ver más abajo).

Tenga en cuenta que, si bien SOA y las arquitecturas de microservicios más nuevas comparten muchos términos (es decir, "servicios" y "arquitectura"), sólo están vagamente relacionadas y en realidad, operan a diferentes escalas, como se explica más adelante en este artículo. (Vacas, 2009)

1.2.Características y Tipos de SOA

Para (Beservices, 2021) menciona las siguientes características:

Los servicios son autónomos. Cada servicio SOA se mantiene y desarrolla de forma independiente.

Los servicios son distribuibles. Se pueden ubicar en cualquier parte sobre la red siempre que este soporte los protocolos de comunicación requeridos.

Los servicios se pueden descomponer. Cada servicio SOA es independiente de los otros y puede ser remplazado o actualizado sin romper con las aplicaciones que conecta.

Los servicios no comparten clases. En una arquitectura SOA, los servicios comparten y contratos y esquemas cuando se comunican, no clases internas.

Los servicios son compatibles con políticas. Entiendo políticas como la definición de características como el transporte, protocolo o seguridad.

1.2.1.Principios de SOA

Según (Vilchez, 2022) dice que "Después de comprender algunos de los beneficios de SOA, debemos comprender, o al menos tener una comprensión básica, los principios detrás de este marco de diseño.":

Para (Vilchez, 2022) menciona las siguientes Principios:

Contratos estandarizados. Su propósito es garantizar que todos los servicios cumplan con los estándares de diseño y estén en la misma lista de servicios.

Bajo acoplamiento. Esto consiente que sean reutilizados por los diferentes consumidores.

Abstracción. Contienen únicamente información básica y no es necesario profundizar en lo que se divulga en el contrato.

Reutilización. Se ajusta a todos los servicios con lógica y puede reutilizar fácilmente en diferentes proyectos

Autonomía. Autocontrol del entorno básico sin dependencias.

Ausencia de estado. Esto indica si el estado de la información del servicio es activo o inactivo.

Descubribilidad. Todos estos son metadatos que permiten descubrir e interpretar servicios.

Composición. Le permite crear una lógica más compleja a partir de múltiples servicios.

1.2.2.Tipos de Servicios

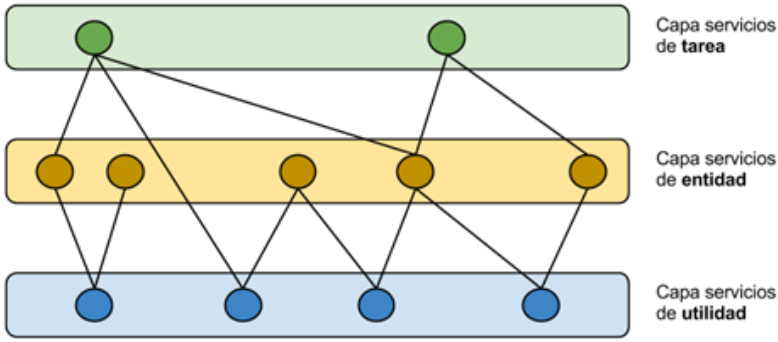
Según (Ballinger, 2001) "El principal promotor de estos modelos de servicio es Thomas Erl, reconocida figura en el campo de la Orientación a Servicios y Cloud Computing".

Según (Ballinger, 2001) habla Distinguiremos 3 tipos de servicios:

Servicios de utilidad.
Servicios de entidad.
Servicios de tarea.

Figura 2

Capas de los Servicios.



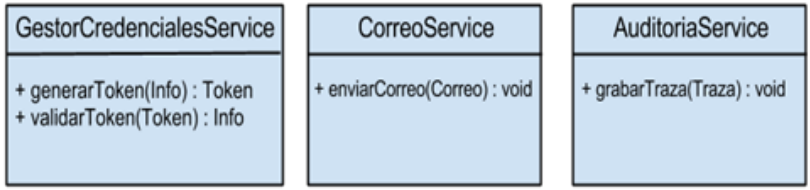
Nota. Capas de los servicios SOA.Tomado https://www.adictosaltrabajo.com [figura], SOA y los tipos de servicios,2013, https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

Servicios de Utilidad

Según (Beservices, 2021) habla que "Los servicios adicionales son aquellos que incluyen multifuncionalidad. Se trata de servicios que no satisfacen necesidades empresariales específicas. Estos servicios son altamente reutilizables (uno de los principios fundamentales del diseño de servicios)".

Figura 3

Ejemplo como se compone un Servicios de utilidad



Nota. Componente de servicios de utilidad. Tomado https://www.adictosaltrabajo.com [figura], SOA y los tipos de servicios,2013, https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

"Algunos ejemplos de dichos servicios podrían ser los servicios para administrar tokens de seguridad para acceder a las aplicaciones o servicios en las plataformas o servicios de correo electrónico". (Vaca, 2022)

En algunos lugares, segmentan los servicios complementarios en función de si proporcionan valor a nivel empresarial (como el correo electrónico) o a nivel de plataforma (como los servicios de conversión de mensajes). Como mencionamos anteriormente, existen diferentes variaciones de esta clasificación. (Ana Virginia Flores, 2013)

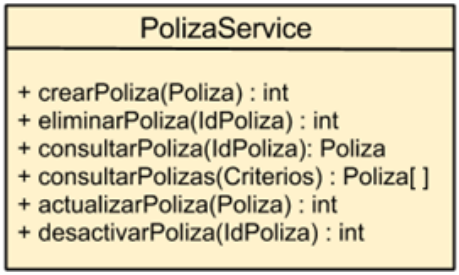
Servicios de Entidad.

Según (Hernández, 2023) habla que los servicios de entidad son aquellos que se centran en el contexto de una entidad comercial. Al igual que las utilidades, no son servicios diseñados para resolver un problema específico y, por tanto, son altamente reutilizables. Muchas de las operaciones expuestas por dichos servicios son CRUD clásicas (crear, leer, actualizar, eliminar).

"Estos servicios, por supuesto, variarán según la empresa en particular y la organización que representan las actividades. Algunos ejemplos serían: Pólizas (Departamento de seguros), cuentas Corrientes (Departamento de Banca), Clientes, etc." (Holley, 2020)

Figura 4

Ejemplo como se compone un Servicios de entidad póliza



Nota. Componente de servicios de utilidad. Tomado https://www.adictosaltrabajo.com [figura], SOA y los tipos de servicios,2013, https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

Servicios de Tarea.

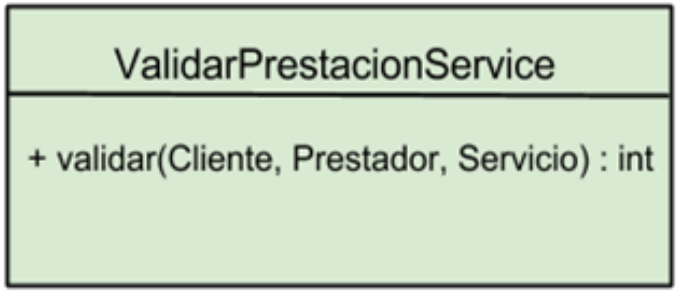
Según (Ballinger, 2001) habla que Los servicios de tarea son servicios que incluyen procesos comerciales y normalmente dependen de servicios de nivel inferior, como servicios de servicios públicos o de dispositivos. Suelen consistir en una serie de acciones para realizar una tarea concreta. Estos últimos son muy interesantes porque normalmente no son servicios que tengan un nivel de reutilización tan alto como los servicios empresariales o de servicios públicos debido a sus necesidades comerciales muy específicas.

Según (Ballinger, 2001) Los servicios de tarea son generalmente menos sólidos que los servicios de entidades o de servicios públicos. Esto se debe a que incluso si surge en respuesta a necesidades comerciales específicas, su funcionalidad cambiará en función de los cambios en el propio negocio.

Un ejemplo podría ser un el servicio brindan servicios autorizados (un caso muy típico entre las compañías de seguros). Veamos la situación cuando un cliente de seguro visita a un médico. Podemos proporcionar un servicio para comprobar a qué médico puede acudir el cliente para una consulta. (Beservices, 2021)

Figura 5

Ejemplo como se compone un Servicios de tarea

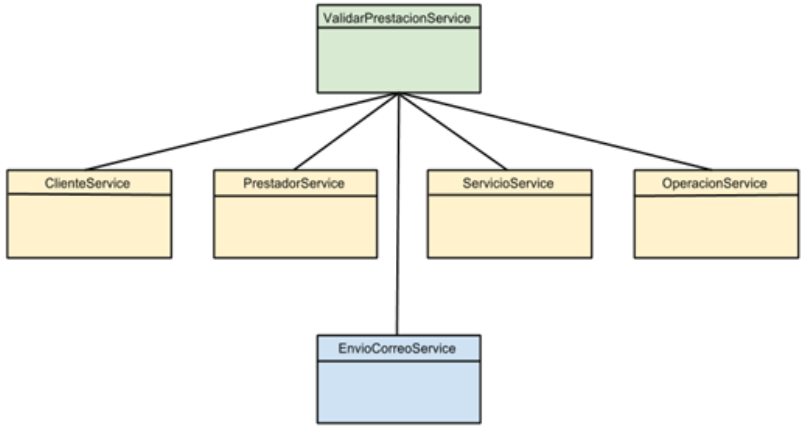


Nota. Componente de servicios de utilidad. Tomado https://www.adictosaltrabajo.com [figura], SOA y los tipos de servicios,2013, https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

Según (Rodríguez, 2015) menciona que "La verificación de los servicios probablemente debería depender de varias entidades, especialmente de comerciales involucradas en el proceso, y posiblemente algunos servicios útiles como enviar un correo electrónico con los resultados de la verificación".

Figura 6

Ejemplo como se compone varios servicios



Nota. Componente de servicios de utilidad. Tomado https://www.adictosaltrabajo.com [figura], SOA y los tipos de servicios,2013, https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

"Al igual que ocurría con los servicios de utilidad, es frecuente encontrar en determinados sitios diferentes «subcategorías» de los servicios de tarea." (Rodríguez, 2015)

1.3.Creación de una API Rest

Los autores (Arsaute, A., Zorzán, FA, Daniele, M., González, A., & Frutos, M., 2018) hablan que los propósitos de las API REST es proporcionar una capa de abstracción para acceder a los datos necesarios para respaldar las operaciones proporcionando un conjunto de funciones y servicios a componentes de nivel superior. Esta API gestionará la información almacenada en la base de datos.

Según los autores (Arsaute, A., Zorzán, FA, Daniele, M., González, A., & Frutos, M., 2018) nos presenta un ejemplo como realizar un Api Rest.

1.3.1.Creando la API REST

"Primero, debemos agregar la dependencia de MVC y realizar la recuperación de dotnet nuevamente. Para agregar una nueva dependencia, abra project.json y agregue la siguiente línea de dependencia debajo de la dependencia agregada anteriormente". (Camilletti, 2016)

Figura 7

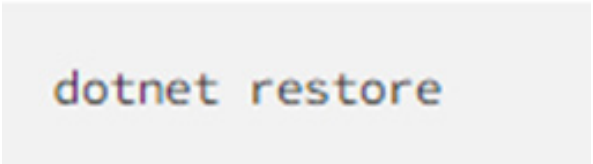
Agregación de la dependencia MVC

*Nota.*Dependencia MVC. Tomado https://www. https://medium.com/ [figura], Creando una API REST con ASP.NET Core desde cero,2016, https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd

"Ahora necesitamos instalar el paquete localmente para lo cual ejecutamos las siguientes líneas. Esto es similar a usar npm install para node.js." (Camilletti, 2016)

Figura 8

Instalación de paquetes de dependencias



*Nota.*Paquetes localmente de JS. Tomado https://www. https://medium.com/ [figura], Creando una API REST con ASP.NET Core desde cero,2016, https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd

"Después de recuperar dotnet, el siguiente paso es actualizar Startup.cs para registrar las solicitudes MVC en proceso. Para hacer esto, reemplazaremos el contenido del archivo con el siguiente contenido": (Camilletti, 2016)

Figura 9

Procesamiento de los requests

```
using Microsoft.AspNetCore.Builder;
using Microsoft.Extensions.DependencyInjection;

namespace aspnetcoreapp
{
  public class Startup
  {
    public void ConfigureServices(IServiceCollection services)
    {
      services.AddMvc();
    }

    public void Configure(IApplicationBuilder app)
    {
      app.UseMvc();
    }
  }
}
```

Nota. Procesamiento de los requests. Tomado https://www. https://medium.com/ [figura], Creando una API REST con ASP.NET Core desde cero,2016, https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd

"Finalmente,se crea el controlador. Se crea una nueva carpeta llamada Controlador y se agrega el archivo ValuesController.cs con el siguiente contenido": (Camilletti, 2016)

Figura 10

ValuesController.cs

```
using System.Collections.Generic;
using Microsoft.AspNetCore.Mvc;

namespace aspnetcoreapp.Controllers
{
  [Route("api/[controller]")]
  public class ValuesController : Controller
  {
    [HttpGet]
    public IEnumerable<string> Get()
```

```
        {
            return new string[] { "value1", "value2" };
        }

        [HttpGet("{id}")]
        public string Get(int id)
        {
            return "value";
        }

        [HttpPost]
        public void Post([FromBody]string value)
        {
        }

        [HttpPut("{id}")]
        public void Put(int id, [FromBody]string value)
        {
        }

        [HttpDelete("{id}")]
        public void Delete(int id)
        {
        }
    }
}
```

Nota. ValuesController . Tomado https://www. https://medium.com/ [figura], Creando una API REST con ASP.NET Core desde cero,2016, https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd

"Ahora todo lo que queda por hacer es probar la aplicación ejecutando dotnet run nuevamente y luego navegando hasta el punto final de la API (en este caso http://localhost:5000/api/values). Después de revisar, verá los siguientes resultados en su navegador": (Camilletti, 2016)

Figura 11

Ejecución de la API

Nota. Resultado de una Api Ret . Tomado https://www. https://medium.com/ [figura], Creando una API REST con ASP.NET Core desde cero,2016, https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd

"La Creación de API REST con ASP.NET es muy sencillo y se puede trabajar de una manera muy sencillo y se puede trabajar desde cualquier plataforma (Windows, Mac o Linux) sin ningún problema". (Camilletti, 2016)

1.4. Uso de Framework Basados en JavaScript

1.4.1. Framework

La autora Sandra Cabello Jurado lo define como un conjunto de bibliotecas desarrolladas en un lenguaje específico que se utilizará para desarrollar páginas web que contengan contenido web preprogramado y reutilizable, como bibliotecas de imágenes (Cabello, 2014, p. 115).

1.4.2. JavaScript

La autora (Alba, Purificación Ribes, 2011) menciona que "JavaScript es uno de los llamados lenguajes de scripting. Un script (traducido literalmente como script) es un archivo de comandos, normalmente un programa sencillo. Por tanto, no podemos definir JavaScript como un lenguaje de programación en sentido estricto, pero sí nos permite crear páginas dinámicas con efectos muy interesantes que pueden mejorar notablemente su apariencia. Esto nos permite realizar determinadas interacciones con el usuario del sitio web, identificar ciertos eventos que puedan ocurrir y responder en consecuencia. Incluso podemos crear algunos programas más complejos para manejar estructuras de datos. Como lenguaje de programación, los programas que escribimos no necesitan ser compilados. Los lenguajes de scripting son lenguajes traducidos. Esto significa en la práctica que cuando usemos JavaScript, escribiremos nuestro programa y podremos ejecutarlo directamente sinhacer nada más. (pág. 17)

Basado en el Estándar ECMAScript [ECMASC]

Le permite acceder y manipular el árbol DOM de archivos HTML, generar contenido dinámicamente, realizar solicitudes HTTP asincrónicas y más. Soporta estructuras de programación similares a lenguajes como C o Java. La principal diferencia es que el alcance de una variable no es el bloque en el que se define, sino la función en la que se declara. Los documentos HTML pueden

contener código JavaScript ejecutable utilizando nodos de etiquetas de script. Estos nodos pueden contener código directamente ejecutable o pueden hacer referencia a archivos externos que contienen código. (Alba, Purificación Ribes, 2011)

Según (Alba, Purificación Ribes, 2011) menciona que "Los navegadores web tradicionales incluyen varios elementos predefinidos relacionados con la ejecución de JavaScript":

Tipos de datos y objetos predefinidos de uso general.
Objetos que permiten manipular el árbol DOM del documento HTML (por ejemplo: el objeto window o el objeto document).
Objetos que permiten realizar otro tipo de operaciones de más alto nivel (por ejemplo, peticiones AJAX).

(Alba, Purificación Ribes, 2011) dice que "Los principales objetos disponibles para el contexto de ejecución de JavaScript en los navegadores tradicionales son":

El objeto window. representa la ventana actual y además, también es el contexto de ejecución de más alto nivel.

El objeto document. representa al documento que está cargado en una ventana. Desde este objeto se puede acceder a todos los elementos del árbol DOM. Tanto el objeto document, como los otros objetos del contexto JavaScript, proporcionan una serie de funciones y métodos que permiten manipular el árbol DOM de la página web.

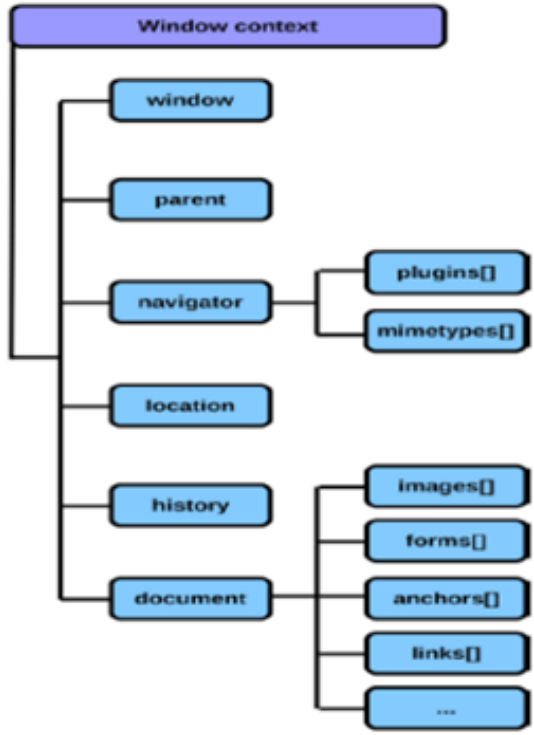
Según (Alba, Purificación Ribes, 2011) menciona que, "Por ejemplo, el objeto document permite, entre otras muchas funcionalidades, las siguientes operaciones":

Acceder al nodo raíz del árbol utilizando la llamada document. documentElement.
Obtener los nodos de un tipo determinado utilizando la función getElementsByTagName.

Obtener nodos por su identificador utilizando la función getElementbyId

Figura 12

Principales objetos accesibles desde el contexto JavaScript



Nota. Esquematiza estos objetos. Tomado https://www. https://um/docencia.com/ [figura], JavaScript,2018, https://www.um.es/docencia/barzana/DAWEB/2017-18/daweb-tema-20-js.html

Ejemplo «¡Hola Mundo!»

Según (Camilletti, 2016) menciona que "La parte anterior suena realmente emocionante y debería serlo. JavaScript es una de las tecnologías más interesantes de la web y, una vez que se sienta cómodo con ella, su sitio entrará en una nueva dimensión de energía y creatividad."

Pero sentirse cómodo con JavaScript es un poco más difícil que aprender HTML y CSS. Debe comenzar poco a poco y avanzar en pasos pequeños y consistentes. Primero, le mostraremos cómo agregar JavaScript básico a su página creando un ejemplo "¡Hola mundo!" (Ballinger, 2001)

Según (Ballinger, 2001) nos mencia los siguientes puntos:

Primero, vaya a su área de preparación y cree una carpeta llamada "scripts". Luego cree un nuevo archivo llamado main.js en la nueva carpeta del script y guárdelo.

A continuación, abra el archivo index.html e ingrese este código en una nueva línea antes de la etiqueta de </body>:

<script src="scripts/main.js"></script>

Es básicamente lo mismo que hace el elemento <link> para CSS : aplica código JavaScript a la página para que pueda interactuar con HTML (y CSS o cualquier otra cosa en la página).

Ahora agregue el siguiente código al archivo main.js

```
const miTitulo = document.querySelector("h1");
miTitulo.textContent = "¡Hola mundo!";
```

Finalmente, asegúrese de guardar sus archivos HTML y JavaScript y abrir index.html en su navegador. Debería ver algo como esto:

Figura 13

Ejemplo Javascript

Nota. Ejemplo de Javascript. Tomado https://www.https://um/docencia.com/ [figura], JavaScript,2001, https://blog.hubspot.es/website/que-es-javascript

1.5.Gestión de Base de datos.

Según (Shekhar, S. & Chawla, S, 2022), Un sistema de gestión de bases de datos (DBMS1) consta de un conjunto de datos interrelacionados y un conjunto de programas que acceden a ellos. Esta definición es en realidad la misma que la dada anteriormente para los sistemas de información; de hecho, un DBMS suele ser la base de un SI.

La situación con los SIG es un poco diferente, porque las bases de datos espaciales, en principio, no son adecuadas para su gestión con un DBMS tradicional.

Sin embargo, cuando se analiza el desarrollo de tecnologías relacionadas con SIG desde la década de 1970 hasta la actualidad, una de las tendencias más obvias es la creciente importancia del uso de DBMS para la gestión de datos temáticos (por ejemplo, soporte SIG).

Originalmente se utilizaban para almacenar atributos temáticos asociados a un conjunto de unidades espaciales almacenadas en formato vectorial; hoy en día, también se utilizan para almacenar información geométrica (conjuntos de coordenadas) de unidades espaciales. Aunque se ha intentado almacenar información en formato ráster en el DBMS, esta opción no es efectiva.

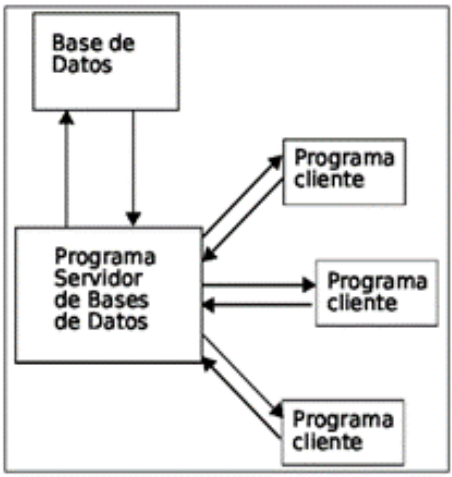
Características Fundamentales de un Sistema de Gestión de Base de Datos (SGBD)

Según (Shekhar, S. & Chawla, S, 2022), "Un DBMS permite el almacenamiento, manipulación y recuperación de datos pertenecientes a la base de datos de una organización en uno o más archivos. En el modelo más amplio (base de datos relacional), la base de datos aparece ante el usuario como un conjunto de tablas con relaciones entre ellas.

A pesar de las similitudes (ambos administran conjuntos de tablas), existen varias diferencias clave entre los DBMS y los programas de hojas de cálculo, la más importante de las cuales es que un DBMS le permite:

Figura 14

Esquema cliente-servidor en una base de datos



Nota. Para qué sirve un gestor de base de datos. Tomado universidadeuropea [figura], gestor de base de datos ,2022, https://universidadeuropea.com/blog/para-que-sirve-gestor-base-datos/

Según (Shekhar, S. & Chawla, S, 2022) menciona 3 principios que se ocupa en un gestor de bases de datos:

El objetivo principal no es priorizar la visualización de toda la información, sino permitir la expresión de consultas complejas cuya resolución se optimiza mediante un lenguaje formal.

El almacenamiento de datos se realiza de manera eficiente, aunque está oculto para el usuario y, a diferencia de las hojas de cálculo, generalmente tiene poco que ver con la estructura de los datos presentados al usuario.

Acceso simultáneo a los datos por parte de múltiples usuarios autorizados para realizar operaciones de actualización y consulta, sin garantizar la seguridad (debido al acceso no autorizado) o la integridad (pérdida de datos debido a que múltiples usuarios intentan acceder a los datos). El mismo archivo al mismo tiempo.

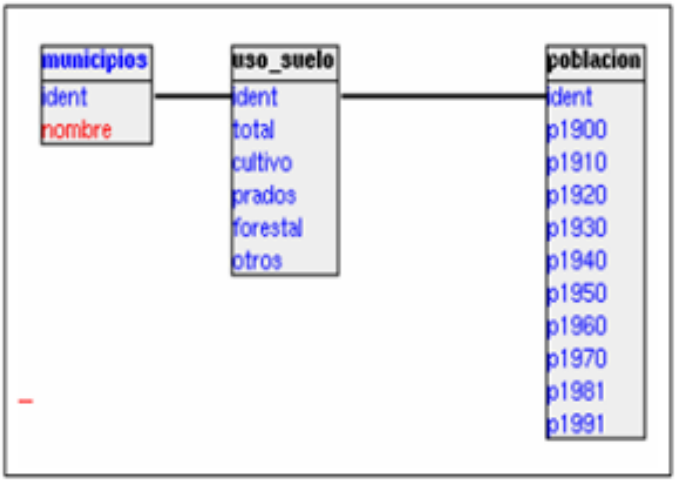
Bases de Datos Relacionales

Este es el modelo más utilizado en la actualidad. Una base de datos relacional es Básicamente un conjunto de tablas que parecen una hoja de cálculo, inclusive los registros y campos. Las entradas representan cada objeto descrito. Los campos de tabla y los campos representan propiedades de objetos variables. de cualquier tipo. (Holley, 2020)

"En el modelo de base de datos relacional, los campos se comunican entre tablas. Estos campos iguales se utilizara para las relaciones entre tablas y permitan consultas complejas entre tablas". (Holley, 2020)

Figura 15

Esquema de base de datos relacional



Nota. Entidad relación. Tomado universidadeuropea [figura], gestor de base de datos ,2022, https://universidadeuropea.com/blog/para-que-sirve-gestor-base-datos/

Esta imagen muestra tres tablas de información del municipio, la primera tabla muestra el nombre del municipio, la segunda tabla muestra el porcentaje de diferentes tipos de uso del suelo en cada ciudad y la tercera tabla muestra todo el siglo XX de cada ciudad. El campo común es el identificador, que es un identificador numérico único para cada ciudad. (Camilletti, 2016)

Ventajas e Inconvenientes de un SGBD

Según (Luis, 2023) menciona "Las ventajas del uso de un SGBD cuando han de manipularse grandes cantidades de datos son enormes":

Se eliminan las inconsistencias en los datos debido al fuerte control que se establece.

Se comparten los datos entre diferentes aplicaciones sin complicaciones, permitiendo una rápida adaptación a nuevas aplicaciones.

Se ahorra espacio de almacenamiento.

Se accede a los datos con extraordinaria rapidez.

Se asegura la protección de los datos frente a malos usos o desastres.

Permiten la creación de entornos personalizados de alta disponibilidad.

Según (Luis, 2023) menciona "Las desventajas del uso de un SGBD"

Desventajas:

La puesta en funcionamiento es larga, ya que se necesita una planificación muy detallada de la estructura de datos.

Se necesita personal especializado para su administración y mantenimiento.

Modelos de Datos

Según (Mheducación) menciona que "Uno de los objetivos más importantes de un DBMS es proporcionar a los usuarios una visión abstracta de los datos, es decir, un usuario que utilizará los datos, pero no será propietario de ellos".

El modelo de datos es la herramienta principal para proporcionar esta abstracción. Se utilizan para la representación y resolución de problemas. Se focaliza el problema en tres niveles de abstracción asociados a la arquitectura ANSI-SPARC de tres niveles. (Mheducación)

Según (Mheducación) menciona que existen 3 nivel de sistemas de gestión de bases de datos:

Nivel físico. Es el nivel más bajo de abstracción; Describe cómo se almacenan realmente los datos.

Nivel lógico o conceptual. describe los datos almacenados en la base de datos y sus relaciones, es decir. objetos en el mundo real, sus atributos y propiedades, y la relación entre ellos.

Nivel externo o de vista. describe las partes de la base de datos a las que los usuarios pueden acceder acceso.

Según (Mheducación) menciona que Un nivel físico es un conjunto de bytes almacenados en un archivo. Un dispositivo magnético que puede ser un disco o una pista de un sector específico. Una jerarquía lógica incluye descripciones y relaciones con otros registros. Un registro en un programa que utiliza un lenguaje de programación.

El último nivel de abstracción, la abstracción externa, es la vista de los datos que tiene el usuario al ejecutar la aplicación en la que se ejecuta, sin que el usuario conozca los detalles de los datos. a veces funciona con unos datos y a veces con otros datos dependiendo de la aplicación. (Rodríguez, 2015)"Si cambiamos el paradigma a un repositorio relacional concreto, como en el caso anterior, habrá un nivel interno y un nivel lógico o conceptual, pero también puede haber múltiples niveles". (Camilletti, 2016)

Diagramas de Estructuras de Datos en el Modelo E-R

Según (Hernández, 2023) menciona que "Un diagrama de entidad-relación representa de alguna manera la estructura lógica de una base de datos gráficos. Los símbolos utilizados son los siguientes":

Rectángulos para representar a las entidades.

Elipses para los atributos. El atributo que forma parte de la clave primaria va subrayado.

Rombos para representar las relaciones.

Las líneas, que unen atributos a entidades y a relaciones, y entidades a relaciones. Si la flecha tiene punta, en ese sentido está el uno, y si no la tiene, en ese sitio está el muchos. La orientación señala cardinalidad.

Si la relación tiene atributos asociados, se le unen a la relación.

Cada componente se etiqueta con el nombre de lo que representa.

Grado y Cardinalidad de las Relaciones

Según (Hernández, 2023) menciona que "El grado de una relación se define como el número de conjuntos de entidades que participan en el conjunto de relaciones o, de manera equivalente, el número de entidades que participan en el conjunto de relaciones".

"Participan en la relación. Una relación que involucra dos entidades es una relación binaria o secundaria. Si son tres, serán de triple o 3er grado. Un conjunto de relaciones puede ser de cualquier grado, idealmente relaciones binarias." (Ana Virginia Flores, 2013)Una relación en la que sólo participa una entidad se llama relación en anillo o de primer nivel; aquellas que relacionan una entidad consigo misma se llaman relaciones reflexivas. Por ejemplo, un EMPLEADO de una empresa puede tener una relación de JEFE que consigo misma: el empleado es el JEFE Hay muchos empleados y el jefe también es un empleado. (Beservices, 2021)

"En el modelo E-R, se reflejan ciertas restricciones que los datos deben obedecer. BD incluido. Estas son restricciones de cardinalidad de asignación, Representa el número de entidades que se pueden asociar con el uso de otra entidad. Conjunto de relaciones". (Hernández, 2023)

Otros Conceptos sobre Bases de Datos

Según (Arsaute, A., Zorzán, FA, Daniele, M., González, A., & Frutos, M., 2018) manifiestan que "a modo de aclarar algunos de los componentes que se pueden encontrar en una base de datos, y que se verán en las próximas unidades, se definen los siguientes conceptos:

Tabla: Es un conjunto de filas y columnas con el mismo nombre que representan un conjunto de valores almacenados como una matriz de datos. Por ejemplo, la información sobre todos los clientes en la base de datos se almacenará en una tabla llamada

Registro: corresponde a cada fila de la tabla. También se les llama tuplas. Por ejemplo, en la tabla Clientes a continuación, observamos dos registros correspondientes a los clientes Juan García y Fernando Martínez."

Tipo de datos. El tipo de datos indica la naturaleza del campo. Por lo tanto, se pueden tener datos numéricos, es decir, datos que se pueden utilizar para realizar cálculos aritméticos (suma, resta, multiplicación). Los datos alfanuméricos son datos que contienen caracteres alfabéticos y números.

Consulta. Es una instrucción para enviar una solicitud a la base de datos. Índice: es una estructura que almacena campos clave en una tabla y los organiza para que sea más fácil encontrar y ordenar registros.

Ver. Disponible al guardar una consulta en una o más tablas de datos. Esto crea una tabla virtual, es decir, aunque se guarda su definición, no se almacena en el dispositivo de almacenamiento del ordenador.

Descripción general. esta es una lista organizada de campos y registros seleccionados en un formato fácil de leer. Por ejemplo, un informe de facturas pendientes de enero ordenadas por nombre de cliente.

Scripts o Scripts. Son un conjunto de instrucciones que se ejecutan de forma secuencial para realizar operaciones adicionales o de mantenimiento sobre los datos almacenados en la base de datos.

Procesos. Son un tipo especial de scripts que se almacenan en una base de datos y forman parte de su esquema.

1.6.Creación de un CRUD (API Rest).

Según (Vilchez, 2022) explica que una Api Rest , Es una arquitectura de software back-end establecida en modelos HTTP que permite crear aplicaciones y servicios que pueden consumirse desde cualquier dispositivo o cliente que utilice HTTP.".

Las operaciones más importantes es el denominado CRUD (Crear, Consultar, Modificar, Eliminar)): Según (Vilchez, 2022)

GET (Leer y consultar los registros)
POST (Crear nuevos registros)
PUT (Editar y modificar los registros)
DELETE (Eliminar los registros)

Según (Shekhar, S. & Chawla, S, 2022) manifiesta que La mayor ventaja de una API REST es donde podemos desarrollar en el backend y en cualquier dispositivo que ahorre mucho tiempo de desarrollo. Capacidad para responder llamadas desde diferentes URL con formato JSON y también pueden recibir datos en formato JSON para gestionar la información proporcionada.

Postman

Según (Shekhar, S. & Chawla, S, 2022) manifiesta que "Base de datos en el siguiente ejemplo requerimos una base de datos con tablas para poder realizar un CRUD (Crear, Consultar, Modificar, Eliminar))."

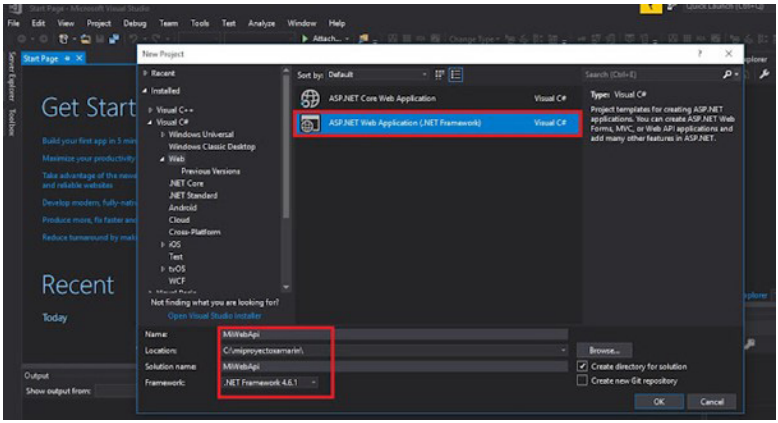
Api Rest: Creación de un Api Rest CRUD en C#

Según (Vaca, 2022) nos ayuda con un ejemplo práctico de una creación de api mediante en CRUD

1.Abrimos Visual Studio y comenzamos a la creación del proyecto con C#.

Figura 16

Crearemos un proyecto Web con C#

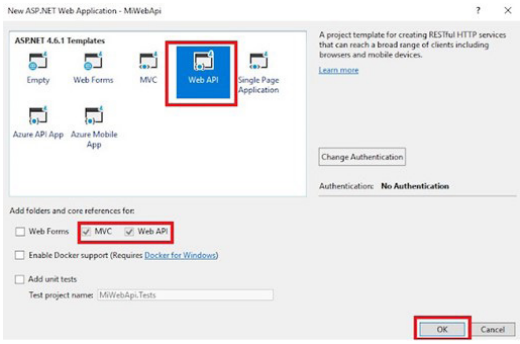


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

2.Se escoge el témplate Web Api, y se escoge las referencias MVC y Web Api.

Figura 17

Crearemos un proyecto Web con C#



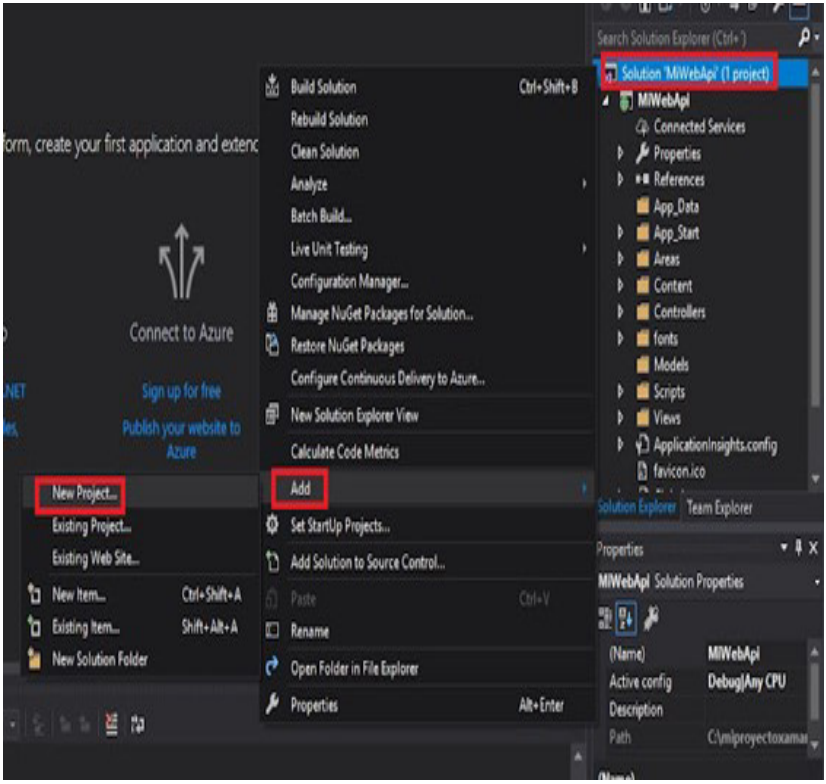
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

3.En el proyecto se escoge un nuevo proyecto donde se realiza la conexión con la base de datos.

Se escoge sobre el proyecto y se añade un Nuevo Proyecto.

Figura 18

Creación de un nuevo proyecto

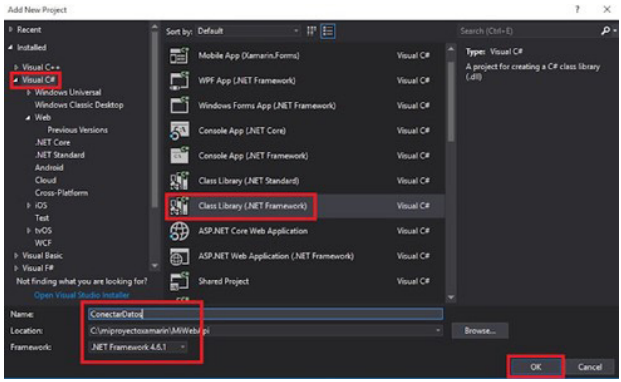


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

4.La creación del nuevo proyecto es en Visual Studio C# y Class Library (.NET Framework), colocamos el nombre y escogemos la ubicación donde se va a guardar el proyecto.

Figura 19

Creación de un nuevo proyecto

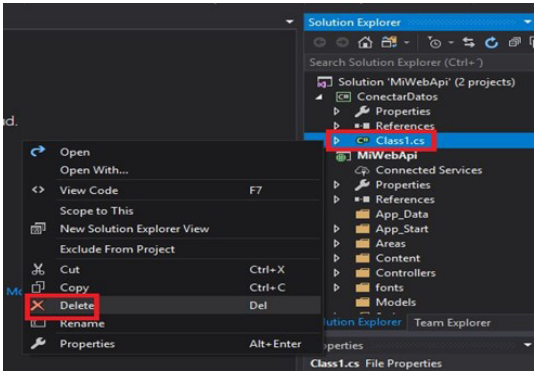


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

5. Una vez ubicados en la Class1.cs procedemos a la eliminación porque ya nos sirve.

Figura 20

Creación de un nuevo proyecto

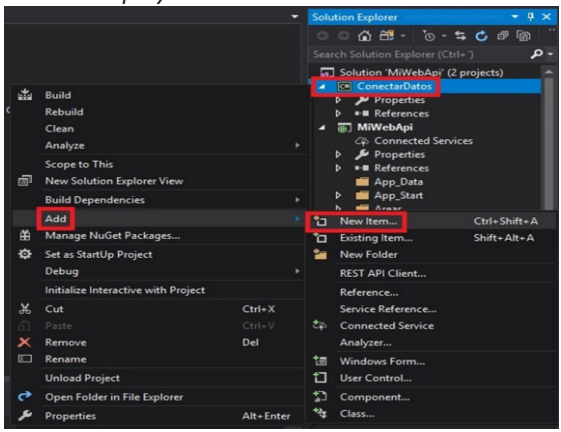


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

6.Añadimos un nuevo ítem dentro de ConectarDatos.

Figura 21

Creación de un Ítem en un proyecto C#

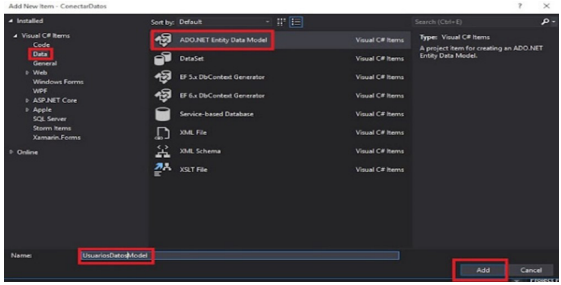


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

7.En visual C#, Data, ADO.NET Entity Data Model, digitamos el nombre y presionamos añadir.

Figura 22

Modelo de datos de entidad

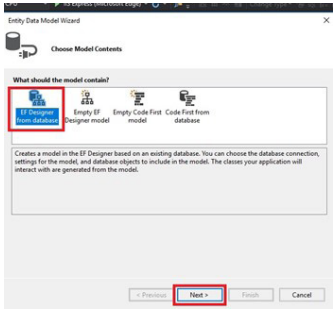


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

8.Cuando estamos en el Entity Data Modeles cogemos la opción EF Designer Form database y presionamos añadir.

Figura 23

Selección del Modelo de datos de entidad

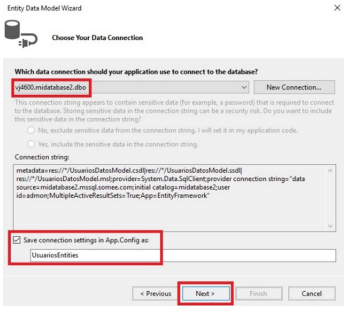


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

9.Escogemos una nueva base de datos, y escogemos la opción para guardar la conexión de la base de datos en el archivo app, en la configuración y colocamos el nombre y guardamos.

Figura 24

Buscar la base de datos

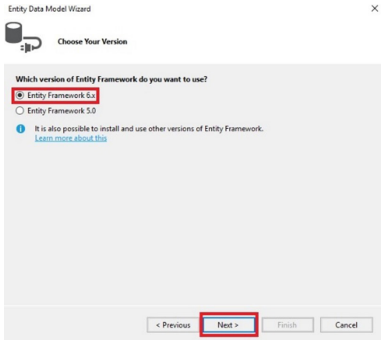


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

10. Escogemos Entity Framework 6.x.

Figura 25

Entity Framework 6.x.

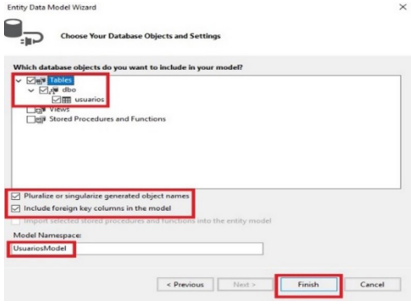


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

11. Escogemos la base de datos que se va a utilizar, escogemos la opción Pluralize or singularize generated object names, Include foreign key columns in the model y colocamos el nombre.

Figura 26

Selección de Base de datos

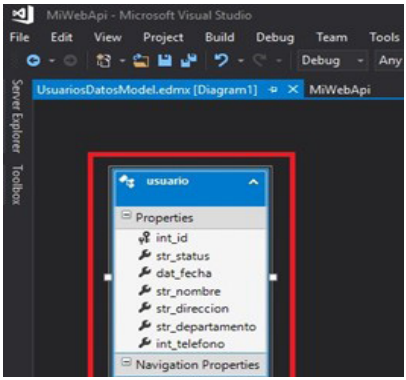


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

12.Una vez finalizado el proyecto se le saca el diagrama con las tablas de la base de datos donde se colocan los campos primarios es int_id.

Figura 27

Diagrama de las tablas de las bases de datos

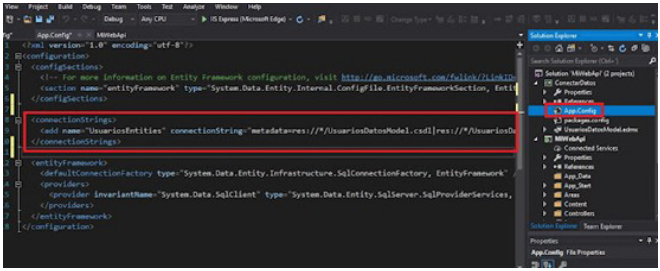


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

13. Abrimos el archivo app.Config, y escogemos la cadena de conexión de la base de datos.

Figura 28

Conexión de la Base de datos

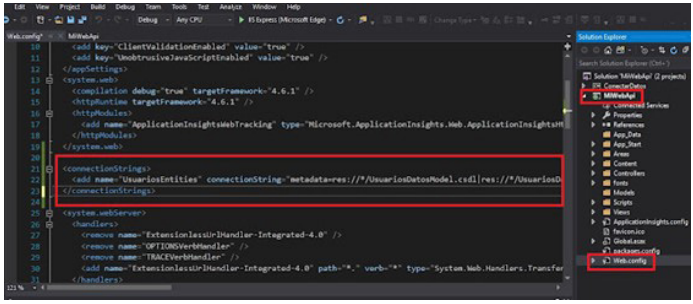


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

14. En el archivo web.Config los editamos y colocamos la cadena de conexión a la base de datos.

Figura 29

Conexión de la Base de datos web.Config

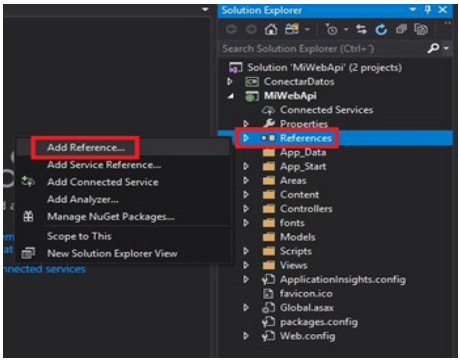


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

15. En el proyecto que estamos trabajando se agrega la referencia principal.

Figura 30

Agregación de referencias

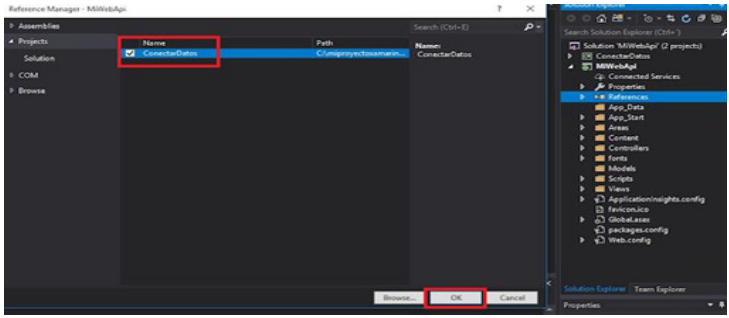


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

16. Seleccionamos el nuevo proyecto que creamos donde se encuentra la configuración de la base de datos.

Figura 31

Configuración de la base de datos

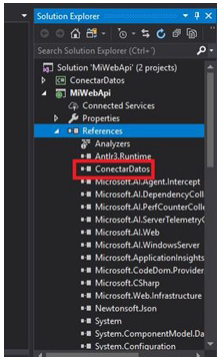


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

17. Se escoge una nueva referencia y se selecciona nuevo para crear la referencia.

Figura 32

Configuración de la base de datos

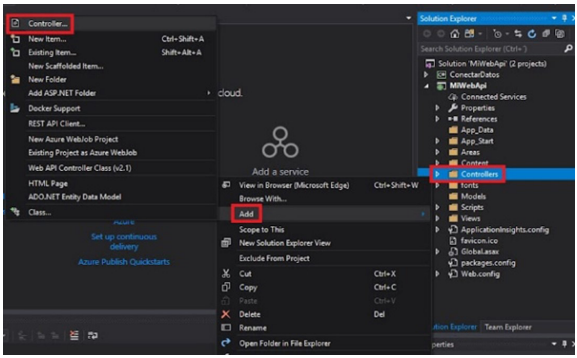


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

18. Se escoge Controller y de ahí añadimos un nuevo control

Figura 33

Agregacion Controller

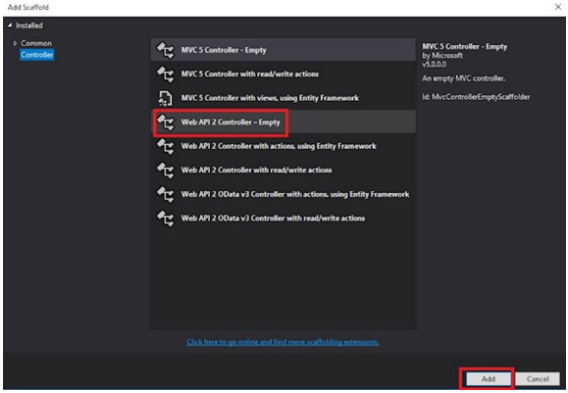


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

19. Escogemos Controller-Empty Web Api 2.

Figura 34

Controller-Empty

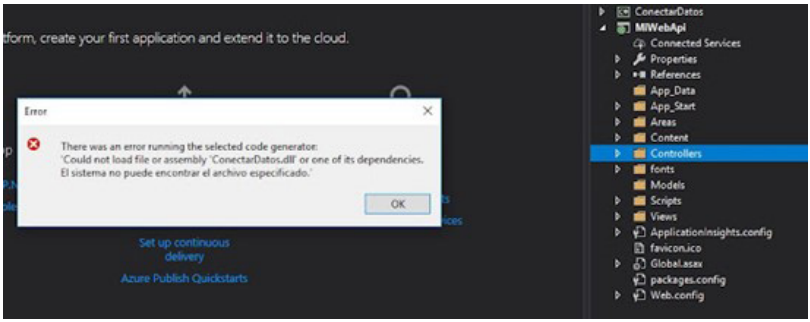


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

20. Si les sale este mensaje de error.

Figura 35

Error al generar un api

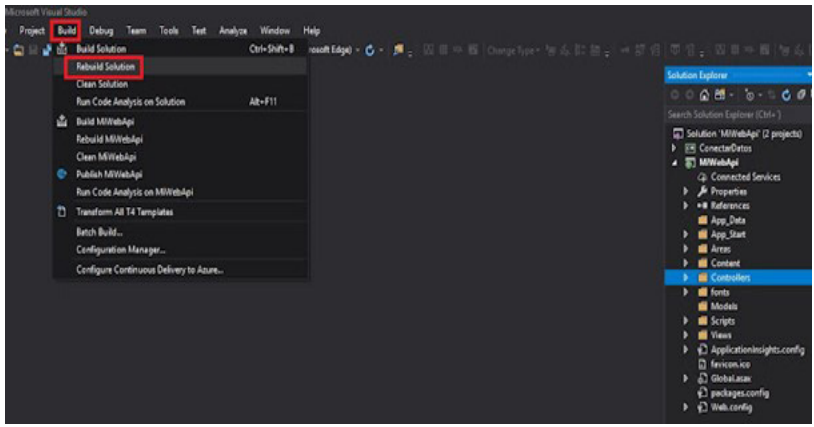


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

21. Escogemos Build/Rebulid Solution.

Figura 36

Seleccionar Build Rebulid Solution

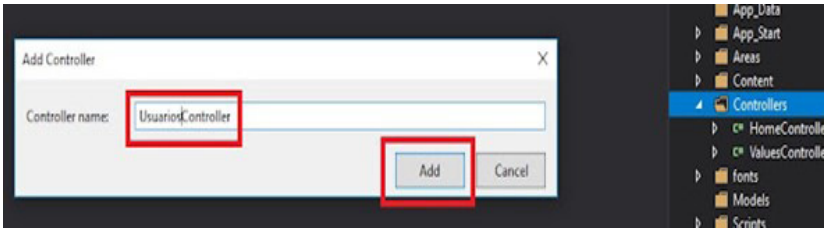


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

22. Se Reintenta y se crea un nuevo controlador y le añadimos.

Figura 37

Creamos un Controller

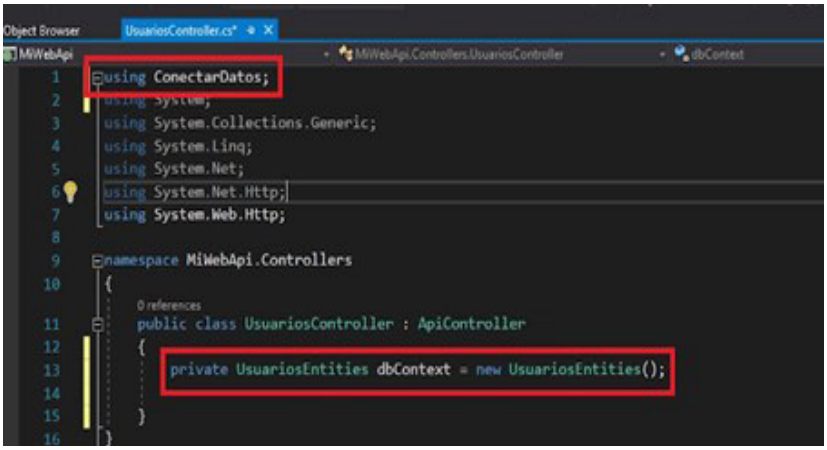


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

23. Se Agrega la conexión a la base de datos y se escribe la siguiente línea.

Figura 38

Creamos la Conexión a la base de datos

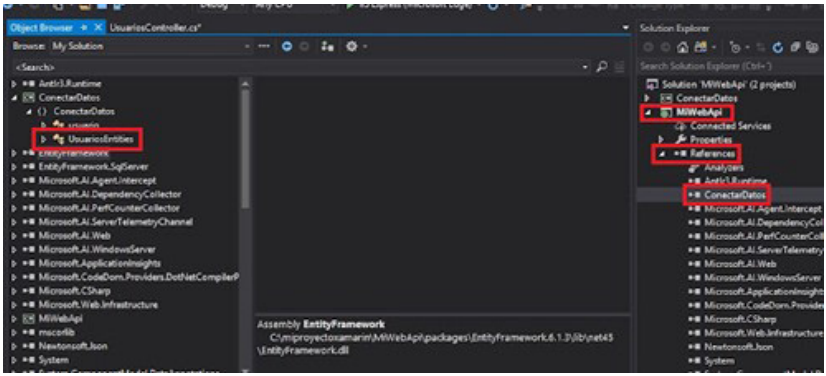


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

24. Los datos se toma del archivo de conexión a la base de datos.

Figura 39

Referencia la Conexión a la base de datos

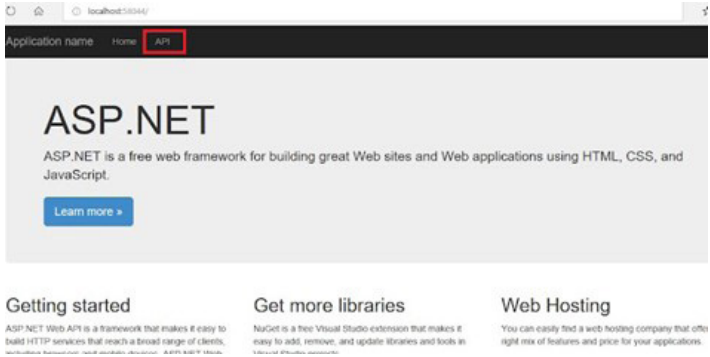


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

25. Finalmente se ejecuta la API y vemos si funciona o no.

Figura 40

Ejecución de la API



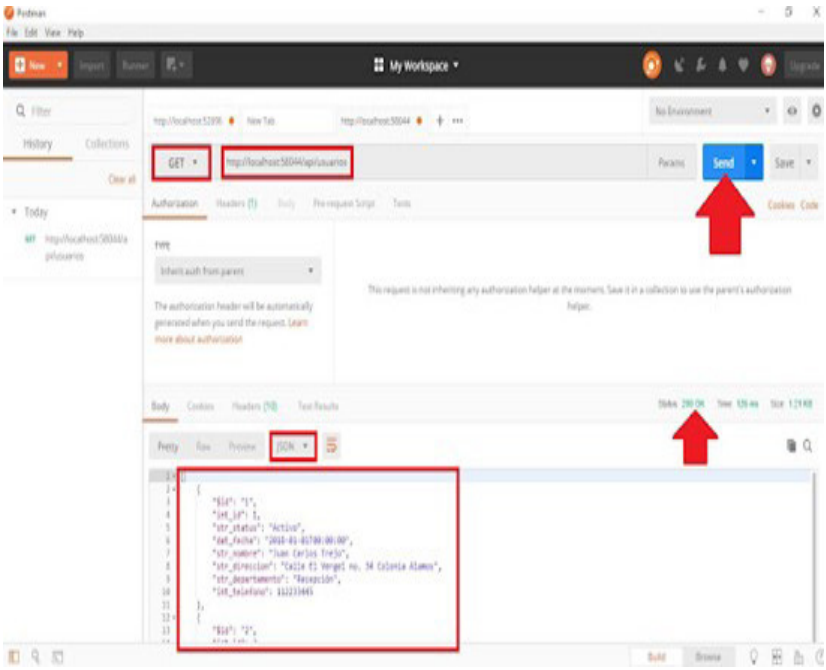
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

METODO GET (Todos los registros)

"Elegimos el método GET, ingresamos la URL y presionamos ENVIAR para mostrar todos los registros en la tabla en la parte inferior (Body)". (Shekhar, S. & Chawla, S, 2022)

Figura 41

Método Get



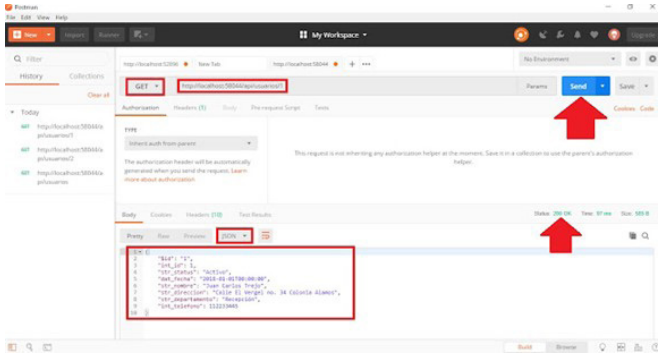
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

METODO GET (Consulta por registro)

"Elegimos el método GET, ingresamos la URL y agregamos no al final. Para ver las entradas, presione enviar, solo las entradas seleccionadas se muestran en la parte inferior (Body)." (Shekhar, S. & Chawla, S, 2022)

Figura 42

Método Get- Consulta por registro



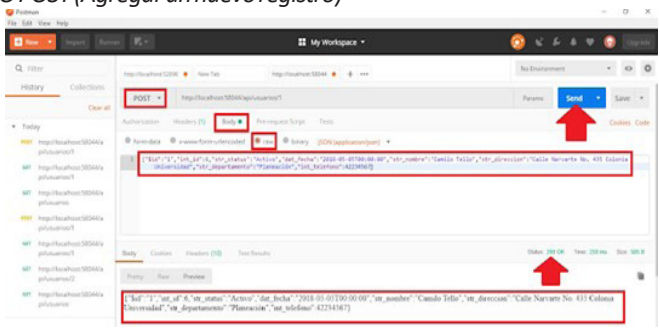
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

METODO POST (Agregar un nuevo registro)

"Elegimos método POST, Body, Raw e ingresamos nuevos datos en formato Json, presionamos ENVIAR, el nuevo registro se mostrará en la parte inferior." (Shekhar, S. & Chawla, S, 2022)

Figura 43

METODO POST (Agregar un nuevo registro)

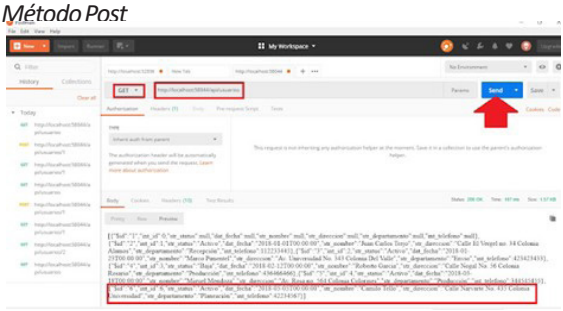


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

"Para verificar que los registros se hayan agregado correctamente, ejecute una consulta utilizando el método GET". (Shekhar, S. & Chawla, S, 2022)

Figura 44

Método Post Método Post



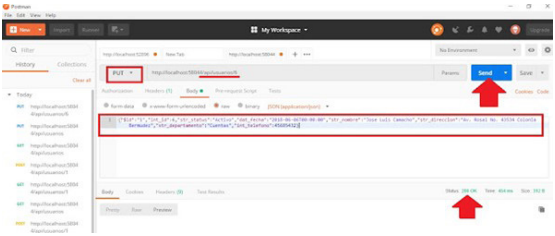
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

METODO PUT (Modificación e un registro)

"Seleccionamos el método PUT y luego escribimos la URL que representa el número. El registro a cambiar ingresaremos los datos a cambiar en formato Json presione ENVIAR nos enviará el resultado 200". (Shekhar, S. & Chawla, S, 2022)

Figura 45

Método Put

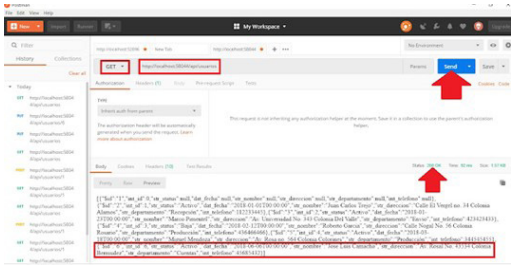


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

Para consultar los cambios se vuelve ejecutar un método GET. (Shekhar, S. & Chawla, S, 2022)

Figura 46

Modificación de registros



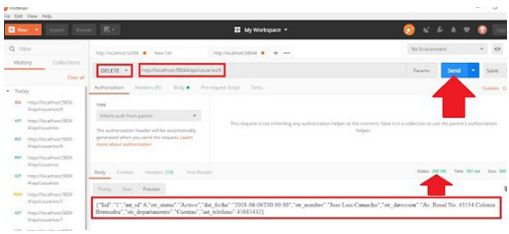
Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

METODO DELETE (Eliminar registros)

"Elegimos el método DELETE y luego escribimos la URL con el número del puerto. El número de registros a eliminar, se pulsa ENVIAR el resultado será 200 y en la parte inferior se indicarán los datos eliminados". (Shekhar, S. & Chawla, S, 2022)

Figura 47

Método Delete

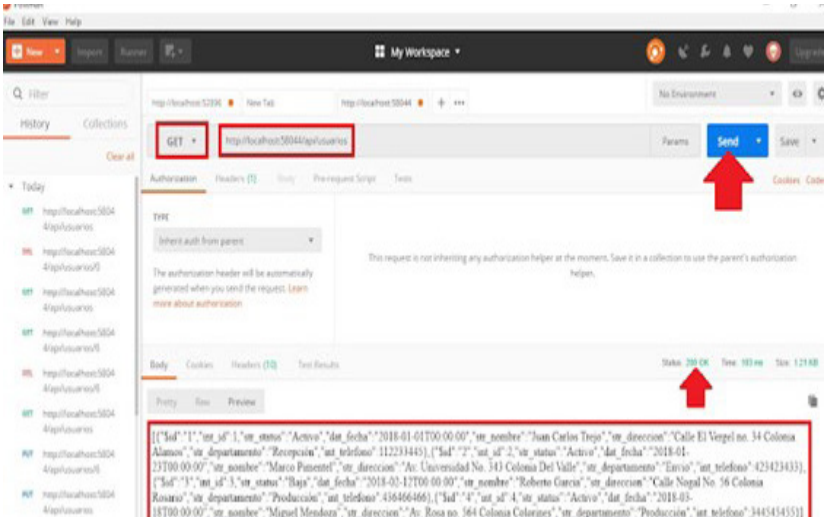


Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

"Para confirmar que el registro ha sido eliminado ejecutaremos nuevamente el método GET. Ahora nuestra Rest Api se puede utilizar en cualquier aplicación utilizando el método HTTP.". (Shekhar, S. & Chawla, S, 2022)

Figura 48

Eliminación de registros



Nota. Proyecto en C#. Tomado erickmarcia.github.io [figura], Creando nuestro proyecto Web ,2022, https://erickmarcia.github.io/blog/Crear-una-Api-Rest-con-AspNet-MVC-y-CSharp/

10. Actividades de Aprendizaje

Tema: Introducción a SOA

– ¿Qué es un SOA?

Tema: Conceptualización y generalización de SOA

– ¿Qué función cumple el SOA?

Tema: Características y tipos de SOA

– Enumere los tipos de servicios

– Menciones cada característica del SOA

Tema: Creación de una API Rest

– Realice una creación de API Rest mediante código .NET

Tema: Uso de framework basados en JavaScript

– ¿Qué framework JavaScript Front-End es mejor para aplicaciones grandes o pequeñas?

Tema: Creación de una API REST con CRUD

– Elaborar un ejemplo de api Rest que contenga el CRUD

Tema: Gestión de Base de datos.

– Se quiere diseñar una base de datos relacional para almacenar información sobre los asuntos que lleva un gabinete de abogados. Cada asunto tiene un número de expediente que lo identifica, y corresponde a un solo cliente. Del asunto se debe almacenar el período (fecha de inicio y fecha de archivo o finalización), su estado (en trámite, archivado, etc.), así como los datos personales del cliente al que pertenece (DNI, nombre, dirección, etc.). Algunos asuntos son llevados por uno o varios procuradores, de los que nos interesa también los datos personales.

11. Autoevaluación

Seleccione la respuesta correcta

1.¿Qué es un SOA?

a)SOA es un paradigma de arquitectura de software moderno para diseñar aplicaciones como un conjunto de entidades de software modulares llamadas servicios.
b)SOA es arquitectura de servicios que comparten contratos y esquemas cuando se comunican entre sí.
c)SOA Permite saber si el estado de la información de un servicio está activa o pasiva.
d)Todas las anteriores

2. ¿ Seleccione las características de un SOA ?

a)Servicios de utilidad, Servicios de entidad, Servicios de tarea

b)servicios son autónomos, servicios son distribuibles, servicios se pueden descomponer, servicios no comparten clases, servicios son compatibles con políticas
c)Bajo acoplamiento, Contratos estandarizados, Abstracción, Ausencia de estado
d)Servicios de utilidad, Servicios de entidad, servicios son autónomos, servicios son distribuibles, Ausencia de estado

3.¿Qué es JavaScript?

a)Conjunto de librerías desarrolladas en un lenguaje determinado que se utilizar para el desarrollo
b)Es un lenguaje de los denominados lenguajes de scripting
c)Servicios de tarea que no suelen ser tan estables como los de entidad o utilidad
d)Ninguno de las anteriores

4.Selecciones los tipos de SOA

a)Servicios de utilidad, Servicios de entidad, Servicios de tarea
b)servicios son autónomos, servicios son distribuibles, servicios se pueden descomponer
c)Bajo acoplamiento, Contratos estandarizados, Abstracción, Ausencia de estado
d)Ninguno de los anteriores

5.¿Qué es un API REST?

a)Arquitectura orientada a servicios o SOA permite interconectar y acceder a los servicios a través de una interfaz común independientemente de la tecnología con la que se ha desarrollado el servicio y de su localización
b)Un método para reutilizar componentes de software a través de interfaces de servicios
c)Es proporcionar una capa de abstracción para acceder a los datos necesarios para operaciones auxiliares proporcionando un conjunto de funciones y servicios a componentes de nivel superior.
d)Son aquellos que están centrados en el contexto de las entidades de negocio para componentes de nivel superior

12. Evaluación final

La evaluación final se realizará presencialmente. El alumno deberá resolver 1 ejercicio planteado aplicando toda su destreza para la Creación de una API REST aplicando el CRUD. La evaluación final será sobre 10 puntos.

13. Solucionario de las Autoevaluaciones

Autoevaluación 1

1. ¿Qué es un SOA?

a)SOA es un paradigma de arquitectura de software moderno para diseñar aplicaciones como un conjunto de entidades de software modulares llamadas servicios.

b)SOA es arquitectura de servicios que comparten contratos y esquemas cuando se comunican entre sí.

c)SOA Permite saber si el estado de la información de un servicio está activa o pasiva.

d)Todas las anteriores

2.¿Seleccione las características de un SOA?

a)Servicios de utilidad, Servicios de entidad, Servicios de tarea

b)servicios son autónomos, servicios son distribuibles, servicios se pueden descomponer, servicios no comparten clases, servicios son compatibles con políticas

c)Bajo acoplamiento, Contratos estandarizados, Abstracción, Ausencia de estado

d)Servicios de utilidad, Servicios de entidad, servicios son autónomos, servicios son distribuibles, Ausencia de estado

3.¿Qué es JavaScript?

a)Conjunto de librerías desarrolladas en un lenguaje determinado que se utilizar para el desarrollo

b)Es un lenguaje de los denominados lenguajes de scripting

c)Servicios de tarea que no suelen ser tan estables como los de entidad o utilidad

d)Ninguno de las anteriores

4.¿Selecciones los tipos de SOA ?

a)Servicios de utilidad, Servicios de entidad, Servicios de tarea

b)servicios son autónomos, servicios son distribuibles, servicios se pueden descomponer

c)Bajo acoplamiento, Contratos estandarizados, Abstracción, Ausencia de estado

d)Ninguno de los anteriores

5.¿Qué es un API REST?

a)Arquitectura orientada a servicios o SOA permite interconectar y acceder a los servicios a través de una interfaz común independientemente de la tecnología con la que se ha desarrollado el servicio y de su localización

b)Un método para reutilizar componentes de software a través de interfaces de servicios

c)Es proporcionar una capa de abstracción para acceder a los datos necesarios para operaciones auxiliares proporcionando un conjunto de funciones y servicios a componentes de nivel superior.

d)Son aquellos que están centrados en el contexto de las entidades de negocio para componentes de nivel superior

14. Glosario

A

Acceso: La manera en la cual los archivos o conjunto de datos son referenciados por la computadora.

Aplicación: Programa informático que proporciona servicios de alto nivel al usuario, generalmente utilizando otros programas más básicos que se sitúan por debajo.

Arquitectura de software: Es un conjunto de patrones y abstracciones que proporcionan el marco de referencia para guiar la construcción del software para un sistema de información. Esta define los componentes, sus interfaces y la comunicación entre estos.

Arquitectura Orientada a Servicios: SOA consiste en una forma de modularizar los sistemas y aplicaciones en componentes de negocio que pueden combinarse y recombinarse con interfaces bien definidos para responder a las necesidades de las empresas.

B

Base de Datos: Es una serie de datos organizados y relacionados entre sí, los cuales son recolectados y explotados por los Sistemas de Información de una empresa o negocio en particular.

C

Calidad de software: Es el grado en el que un sistema, componente o proceso cumple con los requisitos especificados y las necesidades o expectativas del cliente usuario.

Campo: Unidad básica de una base de datos, un campo puede ser, por ejemplo, el nombre de una persona. Los nombres de los campos, no pueden empezar con espacios en blanco y caracteres especiales. No pueden llevar puntos, ni signos de exclamación o corchetes. Si pueden tener espacios en blanco en el medio.

Componente: Es el término que se utiliza para describir una unidad independiente que encapsula una serie de funcionalidades. Adicionalmente un componente puede ser utilizado junto con otros para crear sistemas más completos.

I

Interfaz: Es el elemento de software que permite el flujo de información entre varias aplicaciones o entre el software o el usuario.

R

Requisito: En informática se refiere a las necesidades que tiene el usuario para el funcionamiento de un proceso.

S

Servicios de infraestructura: Son servicios requeridos o brindados por el departamento del IT para soportar la operación de los servicios del negocio.

Sistemas de información: Conjunto de elementos físicos, lógicos, de comunicación, datos y personal que, interrelacionados permiten el almacenamiento, transmisión y proceso de la información.

W

WSDL: Estándar para intercambio de datos más utilizado en integración. Debido a la diversidad de plataformas que pueden intervenir durante el proceso de integración y dado que la integración se fundamenta en intercambio de información, XML representa una capa fundamental para representar dicho intercambio.

15. Referencias Bibliográficas

– Alba, Purificación Ribes. (2011). JavaScript. Obtenido de https://www.um.es/docencia/barzana/DAWEB/Lenguaje-de-programacion-JavaScript-1.pdf
– Ana Virginia Flores, E. O. (2013). ARQUITECTURA ORIENTADA AL SERVICIO SERVICE ORIENTED ARCHITECTURE (SOA). Obtenido de https://revistas.uta.edu.ec/erevista/index.php/dide/article/download/54/47/114
– Arsaute, A., Zorzán, FA, Daniele, M., González, A., & Frutos, M. (2018). Arquitectura SOA.
– Ballinger, K. (2001). Tipos de Servicios. Obtenido de https://www.um.es/geograf/sigmur/sigpdf/temario_9.pdfg
– Beservices, E. P. (2021). ¿Qué es la arquitectura orientada a servicios (SOA)? ¿Cuáles son los beneficios en la empresa? Beservices.es. Obtenido de https://blog.beservices.es/blog/que-es-la-arquitectura-orientada-servicios-soa-cuales-son-los-beneficios-en-la-empresa
– Camilletti, N. B. (2016). Creando una API REST con ASP.NET Core desde cero. Obtenido de https://medium.com/nbellocam-es/creando-una-api-rest-con-asp-net-core-desde-cero-fc58924395fd
– Hernández, T. (2023). Introducción al SOA. Obtenido de https://gredos.usal.es/bitstream/handle/10366/139697/BISITE_Calle_Trujillo_SOA.pdf;jsessionid=8659832290FA989482E5B3D5054A72BF?sequence=1
– Holley, C. a. (2020). What Is Service-Oriented Architecture (SOA)? Obtenido de https://es.mathworks.com/discovery/soa.html
– IBM. (2022). B2B Solutions Group, artículo. Migrating to a Service Oriented Architecture, by Kishore. Obtenido de https://www.ibm.com/es-es/topics/soa
– Jesús David Calle, J. T. (2021). Introducción al SOA. Obtenido de https://gredos.usal.es/bitstream/handle/10366/139697/BISITE_Calle_Trujillo_SOA.pdf;jsessionid=8659832290FA989482E5B3D5054A72BF?sequence=1
– Luis, Q. (2023). Tipoos de Gestores de Bases de Datos. Obtenido de https://blog.powerdata.es/el-valor-de-la-gestion-de-datos/que-es-un-gestor-de-datos-y-para-que-sirve
– Mheducación. (s.f.). Gestores de base de datos: qué son y qué tipos existen. Obtenido de https://mexico.unir.net/ingenieria/noticias/gestores-de-base-de-datos/
– Rodríguez, M. A. (2015). Arquitectura Software. Obtenido de https://www.adictosaltrabajo.com/2013/09/23/soa-tipos-servicios/

–Shekhar, S. & Chawla, S. (2022). Gestor de Bases de Datos. Obtenido de https://universidadeuropea.com/blog/para-que-sirve-gestor-base-datos/
Vaca, A. (2022). Ejemplos de arquitectura SOA. Obtenido de https://revistacloud.com/ejemplos-de-arquitectura-soa/
– Vacas, F. S. (2009). Complejidad y Tecnologías de la Información. España: Fundación Rogelio Segovia. doi:978-84-7402-365-7
–Vilchez, E. D. (2022). Arquitectura Orientada a servicios (soa). Linkedin.com. Obtenido de https://es.linkedin.com/pulse/arquitectura-orientada-servicios-soa-eber-daniel-or%C3%A9-vilchez

16. Anexos o Recursos

Anexo 1

–https://revistas.uta.edu.ec/erevista/index.php/dide/article/view/54/47
–https://d1wqtxts1xzle7.cloudfront.net/43461896/introduccion_apis_rest-sample-libre.pdf?1457378275=&response-content-
–libre.pdf?1457378275=&response-content-
–disposition=inline%3B+filename%3DIntroduccion_apis_rest_sample.pdf&Expires=1704245940&Signature=WcvYzXh9WvB9zRgLnSWUyaZ97pVyneO-
–KY0GGI8H5gzl~aT5~aZEODBPW2cxECZarr5Xmsd8zhnQROuQ8~0fLZElb-fDhZOy41fMObRnUpZJ4foBp6YlQ7iwLf50PS0MMAa-
–6OGLO45SA1KDbTn79c9Un4GhBKtgYxgbzfWKgk0MrEDZ~CigfU~u2GlqXP-BO44ZXpmzV~NwJErBx7klZubkSwxwIbhn7Nibz~DE7nTnvePBmrNR3qso6b2h-DK7v0mzl0x1n52kJ024ngqNZgzKtI6SiRkzTsmv22DFT-SN4f4~nXBotAgTgz1q-qPPf-K-gXyUrl1MIjnxOI5Cs-EV6Q__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA
–https://books.google.es/books?hl=es&lr=&id=szDMlRzwzuUC&oi=fnd&pg=PA1&dq=javascript+libro&ots=0DlIUYvETg&sig=jCUraORD6iRJZsdPX6HyQHNoCmk#v=onepage&q=javascript%20libro&f=false
–https://books.google.es/books?hl=es&lr=&id=eJqFGM_TgCoC&oi=fnd&pg=PA1&dq=javascript+libro&ots=t1J4UHf2wi&sig=deSHjE3OvUVEDVtYniOI-tTnpnds#v=onepage&q=javascript%20libro&f=false
–https://books.google.es/books?hl=es&lr=&id=XjbeDwAAQBAJ&oi=fnd&pg=PR5&dq=gesti%C3%B3n+de+base+de+datos+libro&ots=DJv_yKKXMP&sig=MGtqnHcRCpmuegF2kK7cZn8WdRU#v=onepage&q=gesti%C3%B3n%20de%20base%20de%20datos%20libro&f=false

–https://books.google.es/books?hl=es&lr=&id=H0VBDwAAQBAJ&oi=fnd&pg=PA7&dq=gesti%C3%B3n+de+base+de+datos+libro&ots=fYd_UDAK2w&sig=eo6mlnwazZbsStXIsU0XliB8aCY#v=onepage&q=gesti%C3%B3n%20de%20base%20de%20datos%20libro&f=false
–https://openaccess.uoc.edu/bitstream/10609/69205/1/Bases%20de%20datos_M%C3%B3dulo1_Sistemas%20de%20base%20de%20datos.pdf

Banco de imágenes libres

–https://unsplash.com/

INSTITUTO SUPERIOR
TECNOLÓGICO
VICENTE LEÓN

Guía general de estudio de la asignatura

Agosto 2024

ISBN: 978-9942-676-41-2