**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| PROGRAMA DE FORMACIÓN | Técnico en procesamiento de pruebas de *software* |
| --- | --- |

| COMPETENCIA | 220501123. Construir sitios web según técnicas de interoperabilidad y protocolos  técnicos. | RESULTADOS DE APRENDIZAJE | 220501123-4. Construir aplicación *software* de acuerdo con los requerimientos técnicos. |
| --- | --- | --- | --- |

| NÚMERO DEL COMPONENTE FORMATIVO | 003 |
| --- | --- |
| NOMBRE DEL COMPONENTE FORMATIVO | Implementar un sistema de información web |
| BREVE DESCRIPCIÓN | Se abordarán los conceptos de configuración del entorno de desarrollo como espacio de trabajo para el desarrollo de la API Rest. Seguidamente se revisan las buenas prácticas de estándares de codificación para el desarrollo de la API que será construida con las tecnologías de Node JS, Express, y MongoDB, asimismo, se abordará *tenting* de la API desarrollada con Postman*.* |
| PALABRAS CLAVE | *Framework,* Bootstrap, Node JS, Postman, MongoDB |

| ÁREA OCUPACIONAL | 2 - Ciencias naturales, aplicadas y relacionadas |
| --- | --- |
| IDIOMA | Español |

1. **TABLA DE CONTENIDOS:**

**Introducción**

**1. Diseño y codificación de *frontend***

1.1. Prototipo *(mockups,* Sketch*, wireframe)*

1.2. Maquetación del sitio web

1.3. Uso de estilos

1.4. Validación de interfazde usuario

**2. Implementación de *backend***

2.1. CRUD

2.2. Definición de la base de datos

2.3. Instalación *Node JS*

2.4. Api CRUD con *Node JS*

2.5. API de acceso a la base de datos

2.6. Codificación

2.7. Pruebas unitarias

**3. Manuales técnicos y de usuario**

1. **DESARROLLO DE CONTENIDOS:**

**Introducción**

Como regla general no es suficiente conocer y dominar las tecnologías que se involucran en la creación de una aplicación web, pues es preciso emplear técnicas y herramientas tecnológicas para agilizar el uso de esas tecnologías. En tal sentido, para todo proyecto *software* es recomendable alistar las herramientas necesarias y mínimas para iniciar con un desarrollo y, dependiendo del alcance, la tecnología y/o lenguaje, se sugiere definir muy bien este tipo de recursos para tal ejercicio; también se pueden adoptar marcos de trabajo que son una base de uso de buenas prácticas de código. A continuación, se podrán conocer los conceptos generales que se abordarán en este componente formativo.

Interfaz de usuario gráfica

Descripción generada automáticamente

* + - 1. **Diseño y codificación de *frontend***

Antes de poner en marcha un sistema de información, es buena práctica prototipar el sistema para reducir la brecha entre lo que el cliente espera y lo que finalmente recibe. Este proceso también sirve para refinar los requerimientos justo antes de codificar un sistema informático. Para tal finalidad existen varias herramientas de prototipado y, precisamente para cumplir con este propósito, se abordará el prototipo (*mockups, sketch, wireframe*), la maquetación del sitio web y el uso de estilos.

* 1. **Prototipo *(mockup,* Sketch*, wireframe)***

El proceso de prototipado consiste en elaborar una representación visual de los componentes del *software* y este proceso da como resultado la creación de subproductos según el nivel de fidelidad visual respecto al producto final, de esta forma podemos definir los prototipos de la siguiente forma: *mockups* (alta fidelidad) y *wireframe* (es una representación de baja o media fidelidad). En algunos equipos de trabajo los desarrolladores usan el nombre “prototipado” para referirse a una simulación que ofrece algún tipo de interacción con el usuario, para todos los casos existen muchas herramientas que permiten la creación de alguno de estos subproductos tal vez el más popular para crear *mockups, wireframe* o prototipos con interacción, es Sketch, aunque solo es para Mac, es la que mejor calidad de prototipados ofrece actualmente.

Debido a que Sketch no es de libre distribución y no todos disponen de un Mac para realizar un prototipo, a continuación, se presentan unos pasos sencillos para elaborar un *mockup,* para lo cual de manera inicial se deberá crear una cuenta en el siguiente enlace: <https://balsamiq.cloud/>

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Además, cada vez aparecen más sistemas para realizar estos prototipos como Canva, Adobe XD, entre otros.

* 1. **Maquetación del sitio web**

El proceso de maquetación web es la transformación del diseño o prototipo a un producto web en un conjunto de archivos (HTML, CSS, JS) de manera que serán desplegados (renderizados) por un navegador.

| La maquetación web es el proceso que consiste en tomar el prototipo gráfico también denominado *layout* (con los requisitos estructurales y estéticos definidos y aprobados en un análisis inicial) pasa a transformarse en código HTML, CSS y JS (estándares web) para que los navegadores puedan interpretarlo correctamente. | Maquetación Web - Nivel inicial - Capacitaciones |
| --- | --- |

* 1. **Uso de estilos**

El uso apropiado de propiedades de hojas estilos CSS sobre elementos del documento HTML permite realizar la maquetación o definición de un diseño, y es una de las propiedades fundamentales que ofrecen los *frameworks* (marcos de trabajo)de CSS. A lo largo de las últimas décadas los marcos de trabajo han tenido varios enfoques que se discutirán a continuación.

***Movie First Design* vs. *Responsive Web Design***

En un principio, los *frameworks* CSS creaban estructurados en clases, con atributos para diseño o maquetación de páginas web que tuvieran la posibilidad de redimensionar sus contenidos a pantallas más pequeñas, a estos se les llama diseños adaptativos, es decir, al reducir la resolución se reduce el tamaño del contenido.

| *Mobile first* «los móviles primero», es la estrategia de diseño que refiere a una maquetación que tenga en cuenta, en primera instancia, el despliegue en dispositivos móviles. Pantallas reducidas en relación con las tablets o los monitores que usamos comúnmente con los ordenadores, y tras tener la maqueta preparada para móviles, realizar un escalado y hacer otra dimensión de pantallas, es decir, aumentar el tamaño y adaptarlo a una pantalla de escritorio. | que es mobile firts posicionamiento |
| --- | --- |

**Bootstrap CSS**

Bootstrap es un conjunto de herramientas de estilos CSS de código abierto, para que los desarrollos web se puedan usar con HTML, CSS y JavaScript. Con él se puede dar forma al sitio o páginas web usando sus librerías CSS y JavaScript. Incluye diferentes componentes: ventanas modales, menús, cuadros, botones, formulario, es decir, los elementos que se necesitan para maquetar una página. A continuación, se presenta la maquetación de un formulario.

| Bootstrap permite crear interfaces de usuario limpias y totalmente adaptables a dispositivos y pantallas de cada tamaño, algunas de sus principales características son:   * Documentación y soporte amplio con HTML5 y CSS3, que permite un uso flexible. * Un sistema grillas o (GRID) que permite diseñar usando unas grillas de 12 columnas donde se debe plasmar el contenido. Así permite diseñar y desarrollar responsive de forma mucho más fácil e intuitiva. * Es compatible con la mayoría de los navegadores actuales. |  |
| --- | --- |

**Material Desing**

Material Design es un estilo de diseño creado por Google. Fue anunciado el 25 de junio del 2014 en la conferencia Google I/O. Se ha implementado en el sistema operativo para móviles Android desde la versión Lollipop y también en otros servicios gratuitos de Google como Docs, Calendar, Drive, etc.

Google ha dispuesto al público una guía sobre el Material Design para que los diseñadores se animen a implementarlo en aplicaciones móviles y sitios web (disponible en <https://material.io/design>). Los objetivos de esta propuesta son:

* Emplear un lenguaje visual combinando los principios de un buen diseño y las opciones disponibles en las nuevas tecnologías.
* Ser un sistema que posibilite una experiencia uniforme en diferentes plataformas y dispositivos (como pantallas táctiles, control de voz, etc.).

Google junto con los objetivos del Material Design, presenta también sus tres principios fundamentales:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Maquetación**

A continuación, se explicará cómo maquetar empleando Bootstrap, que es actualmente el más común de los *frameworks* de CSS.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

**Estructura**

Como se mencionó antes, existen etiquetas que definen la estructura de una página como lo son <body></body>, <header></header>, <main></main>, <section></section>, <aside></aside>, <footer></footer>, por mencionar los principales, la idea general es poder diseñar para diferentes pantallas como se muestra en la siguiente imagen:

| Si a los componentes de la imagen anterior se definen de la siguiente forma los elementos:   * Bloque azul: <header></header> * Bloque rojo: <aside></aside> * Bloque verde: <main></main> * Bloques naranjas: <section></section> * Bloque violeta: <footer></footer> |  |
| --- | --- |

El resultado, aplicando las clases del Boostrap para los cuatro (4) tamaños presentado, seria:

|  | <div class="container">          <div class="row">              <header class="col-12 col-md-12 col-lg-12 col-xl-12" style="height: 30px; background-color: cornflowerblue;">              </header>          </div>          <div class="row">              <aside class="col-12 col-md-3 col-lg-3 col-xl-3" style="background-color: darkred; min-height: 30px;"></aside>              <main  class="col-12 col-md-9 col-lg-9 col-xl-9" style="background-color: olivedrab; min-height: 150px;">                  <div class="row m-1">                      <section class="col-12 col-md-10 col-lg-6 col-xl-4 mb-1" style="background-color: orange; min-height: 30px;"></section>                      <section class="col-12 col-md-10 col-lg-6 col-xl-4 mb-1" style="background-color: orange; min-height: 30px;"></section>                      <section class="col-12 col-md-10 col-lg-6 col-xl-4 mb-1" style="background-color: orange; min-height: 30px;"></section>                    </div>              </main>          </div>          <div class="row">              <footer class="col-12" style="background-color: mediumpurple; min-height: 60px;">              </footer>          </div>        </div> |
| --- | --- |

**Formularios**

Los formularios también son susceptibles a ser maquetados, de manera que se dimensionen adecuada y automáticamente de acuerdo con el tipo del dispositivo.

| <div class="container m-2">          <form>            <div  class="row mb-2">                <label class="col-12 col-sm-3 col-md-4 form-label" for="nombre" >Nombre</label>                <div class="col-12 col-sm-9 col-md-8">                     <input type="text" class="form-control" id="nombre">                </div>            </div>            <div  class="row mb-2">                <label class="col-12 col-sm-3 col-md-4 form-label" for="correo" >Correo</label>                <div class="col-12 col-sm-9 col-md-8">                     <input type="email" class="form-control" id="correo">                </div>            </div>            <div  class="row mb-2">                <label class="col-12 col-sm-3 col-md-4 form-label" for="asunto" >Asunto</label>                <div class="col-12 col-sm-9 col-md-8">                     <input type="text" class="form-control" id="asunto">                </div>            </div>            <div  class="row mb-2">                <label class="col-12 col-sm-3 col-md-4 form-label" for="asunto" >Mensaje</label>                <div class="col-12 col-sm-9 col-md-8">                <textarea class="form-control" rows="5" id="asunto"></textarea>                </div>            </div>            <div>                <button class="btn btn-primary float-end">Enviar</button>            </div>          </form>        </div> |  |
| --- | --- |

El formulario proyecta dos tipos de pantalla, una pequeña y la otra extrapequeña, tal como se muestra a continuación:

| Formulario para pantalla pequeña | Formulario para pantalla extrapequeña |
| --- | --- |
|  |  |

En conclusión, dimensionar los elementos es importante para lograr una óptima visualización en diferentes dispositivos que se usan actualmente.

* 1. **Validación de interfaz de usuario**

La validación y verificación de la interfaz del usuario es recomendable que se realicen tres puntos del proceso que son:

**En el análisis**

Las interfaces deben responder a los requisitos funcionales definidos en el proceso de captura y análisis de requisitos.

**En el diseño**

Las interfaces diseñadas deben garantizar la calidad en la presentación, es decir, que sea fácilmente identificable la funcionalidad de cada componente presente, en especial aquellos que están dispuestos para lograr la interacción con el usuario.

**En la funcionalidad**

Durante las pruebas donde, básicamente, se ensayan ejecutando la aplicación.

Teniendo en cuenta lo anterior, los objetivos que se obtienen con las pruebas de interfaz son:

* Las características que tiene la interfaz deben probarse para asegurar que las reglas del diseño, la estética y el contenido visual estén a disposición del usuario sin lugar a errores.
* Los elementos individuales de la interfaz deben probarse de forma unitaria (por ejemplo, se prueba HTML dinámico, CGI, carrito de compras, etc.) y con especial cuidado aquellos que manipula la presentación de datos y componentes visuales.
* La interfaz debe probarse frente a cada historia de usuario, o caso de uso, para descubrir errores de semánticos, y facilidad de uso por parte del usuario.
* La interfaz debe probarse dentro de una diversidad de ambientes para asegurar su compatibilidad (celular, tablets, pantallas, smart-tv, etc.).

| Pruebas semánticas: mientras se prueba cada historia de usuario o caso de uso, se revisa la información en pantalla o *display* que el usuario obtendrá (especialmente en los menús y formularios), que sea entendible y coherente con el usuario y el contexto en que este empleado el sistema. | Ejemplo: el usuario-operador no vea información solo permitida para los administradores.  Una vez probada cada historia de usuario o caso de uso de forma unitaria para cada mecanismo de la interfaz, se continúa con la prueba semántica, esta evalúa cuán bien el diseño se preocupa de los usuarios y su interacción con el sistema, ofreciendo una dirección clara al siguiente paso del proceso informático u opciones disponibles dentro del flujo de operaciones del sistema y manteniendo consistencia de lenguaje y enfoque. |
| --- | --- |

* + - 1. **Implementación de *backend***

Es muy común dar bastante importancia a los aspectos de las aplicaciones web que son visibles, como las imágenes o el diseño, pero también existen otros elementos muy importantes para que una aplicación funcione de forma correcta y que no son visibles a simple vista.

| El *backend* es esa parte invisible pero esencial en todo proyecto web, y a la que es de extrema importancia darle toda la trascendencia del caso, porque de él depende en gran parte los procesos informáticos que realiza un sistema. |
| --- |

En el desarrollo web, el *backend*, es la parte del sistema que se encarga de todos los procesos necesarios para que la aplicación web funcione de forma correcta y estos procesos o funciones no son visibles. Algunas de estas acciones que controlan el *backend* son la conexión con la base de datos o la comunicación con el servidor de y los procesos o algoritmos que debe realizar la aplicación.

A continuación, se presenta cómo desarrollar un *backend* sencillo que permita interactuar con la base de datos sin prestar mucho énfasis en el desarrollo de algoritmos informáticos. Esto debido a que en los procesos de pruebas es importante comprender cómo funcionan internamente las aplicaciones web.

* 1. **CRUD**

CRUD hace referencia a un acrónimo, que consiste en las primeras letras de las cuatro operaciones fundamentales que las aplicaciones realizan en sistemas de bases de datos:

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

En otras palabras, CRUD se denomina a las funciones requeridas por un usuario, en una aplicación para crear y gestionar datos: Create (crear registros), Read (leer registros), Update (actualizar registros) y Delete bzw. Destroy (borrar registros).

* 1. **Definición de la base de datos**

Para este caso de prueba se implementará una base de datos para un Sistema de Peticiones, Quejas, Reclamos y Sugerencias (PQRS), que es una herramienta que permite conocer las inquietudes y manifestaciones que tienen los visitantes a un sitio web.

La siguiente estructura puede ser la de un documento para una base de datos en MongoDB.

{

"peticionario": "Funanito de Tal",

"asunto": "Asunto de la queja",

"fecha": "2022-11-01",

"pqrs": "Texto detalla del la PQRS",

"estado": "sin respuesta",

"respuesta": null,

"fecha\_respuesta": null

}

Se llamará a la colección PQRS y ella contendrá registros o documentos con una estructura similar a la ya mostrada.

* 1. **Instalación Node JS**

A continuación, se instalará un intérprete de lenguaje de scripting como Node JS para JavaScript, para que sea ejecutado desde el sistema operativo, que no es la forma más común, ya que este lenguaje nació para navegadores. Para iniciar el proceso se debe ingresar a la página de Node JS para descargar la última versión, y seguir las instrucciones para instalar este intérprete.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

Con estas herramientas instaladas se puede proceder a construcción un api que permita la comunicación de una aplicación web con una base de datos a través de un servidor web creado con el intérprete del lenguaje JavaScript.

* 1. **Api CRUD con Node JS**

Es preciso instalar un servidor web que permita recibir las solicitudes de servicios CRUD desde un navegador u otro sistema de pruebas. Para tal fin, se emplea un conjunto de librerías JavaScript llamado *express*, para lo cual se debe en el siguiente orden, el comando correspondiente.

| Instructivo Api CRUD con Node JS  Ingresa al instructivo para conocer el paso a paso para la instalación de Api CRUD con Node JS |
| --- |

* 1. **API de acceso a la base de datos**

A continuación, se usará la dependencia de MongoDB para la conexión; por ser un ejemplo práctico se empleará MongoDB instalandolo con el siguiente comando.

| Instructivo API de acceso a la base de datos  Ingresa al instructivo para conocer el paso a paso para la instalación de Api desde la base de datos. |
| --- |

* 1. **Codificación**

Es la escritura de las sentencias del algoritmo en un lenguaje de programación. A continuación, se presentarán los códigos que contendrá el api REST.

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

* 1. **Pruebas unitarias**

Una prueba unitaria es una forma de comprobar el correcto funcionamiento de una unidad o porción de código. Para el caso que se viene trabajando, una función o un procedimiento es cada elemento del api correspondiente a las operaciones GRUD. Esto es útil para asegurar que cada unidad funcione correcta y eficientemente por separado y también para validar que el código hace lo que tiene que hacer y verificando que sea correcto.

| Descargar e Instalar Postman - Probar petición GET  Conoce como se pueda realizar pruebas unitarias en un sistema Postman, |
| --- |

| Pruebas unitarias  Visualice los siguientes videos de Instalación y preparación Postman y Pruebas unitarias de *software*, para conocer sobre el proceso de ejecución de las pruebas unitarias. |
| --- |

* + - 1. **Manuales técnicos y de usuario**

Una de las cosas menos valoradas o probadas es la validación de la documentación del proyecto *software*, esto es respecto al manual de usuario, el manual técnico y la documentación presente en la codificación de los algoritmos. Es importante que los procedimientos informativos descritos en el manual de usuario correspondan al flujo de los procesos y al procesamiento de datos confirme se han validado; es común que sean los mismos *testers* quienes realicen los manuales de usuario. Para el manual técnico, se ha probado es un poco más difícil, ya que requiere más de las competencias de administración de servicios (servidores, bases de datos, sistemas operativos, etc.).

El manual técnico es un documento de un determinado sistema que intenta dar asistencia a sus usuarios y va dirigido a la dirección de IT, al administrador de los sistemas o servidores y a otros desarrolladores de *software* para que puedan darle mantenimiento en caso de que se requiera. Un manual técnico debe contener como mínimo:

* Objetivo y alcances del sistema.
* Normas, políticas y procedimientos de la organización en las que se basa el sistema para su implementación.
* Requisitos de la infraestructura y de dependencias de otros *softwares* o plataformas.
* Descripción de bases de datos y diagramas de relación.
* Diseño de reportes y pantallas.
* Diccionario de datos.
* Lista de directorios a los que debe hacer copias recurrentes para asegurar la no pérdida de información.
* Procedimientos.
* Procedimiento de mantenimiento del sistema.
* Procedimiento de instalación y configuración del sistema y sus dependencias.

En cuanto al manual de usuario, debe ser entendido por cualquier usuario principiante y serles de utilidad a los usuarios avanzados; así va dirigido a un público más general que no necesariamente debe tener conocimientos específicos en el área de interés.

Así, este manual tiene como objetivo instruir al usuario en el uso del sistema y la solución de los problemas que puedan suceder en la operación. A continuación, se indica la información que debe tener.

* Un índice.
* Una guía rápida sobre cómo cada una utiliza las funciones principales del sistema.
* Una sección para la resolución de problemas.
* Una FAQ.
* Información de contacto y canales de soporte.
* Un glosario.

El manual de usuario tiene como objetivo instruir al usuario en el uso del sistema y la solución de los problemas que puedan surgir en la operación. Este resuelve nuestras dudas como usuario, todas las posibles opciones que puede realizar el usuario con estas aplicaciones de manera detallada y mediante el uso de capturas de pantalla (imágenes).

| En conclusión, el componente formativo posibilita abordar los conceptos de configuración de un entorno de desarrollo de *software*, a partir del conocimiento de API Rest, integrado con las tecnologías de Node JS*, express,* y MongoDB, desarrollando la API desarrollada a través de la herramienta Postman. |
| --- |

1. **ACTIVIDADES DIDÁCTICAS (OPCIONALES SI SON SUGERIDAS)**

| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| --- | --- |
| Nombre de la Actividad | Emparejamiento entre término y definición |
| Objetivo de la actividad | Afianzar algunos de los conceptos más importantes asociados a la construcción de una API REST. |
| Tipo de actividad sugerida | Arrastrar y soltar el término con la definición que corresponde. |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | Actividad didáctica 1 |

1. **MATERIAL COMPLEMENTARIO:**

| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| --- | --- | --- | --- |
| Introducción | Visual Studio Code. (s.f.) *Visual Studio Code. Instalación.* | Instructivo | Carpeta de anexos con el nombre: Instalación. Visual Studio Code |
| Bootstrap | Pinchao, P. (2021).  *Maquetación en Bootstrap 3 columnas*. <https://codepen.io/peter-emerson-pinchao/pen/xxLwPWJ> | Página web | <https://codepen.io/peter-emerson-pinchao/pen/xxLwPWJ> |
| Estructura | Pinchao, P. (2021).  *Estructura de una página*. <https://codepen.io/peter-emerson-pinchao/pen/xxLwQOp> | Página web | <https://codepen.io/peter-emerson-pinchao/pen/xxLwQOp> |
| Instalación *Node JS* | Gabriel Coding. (2020). *Cómo instalar Node JS en Windows 10.* [Video]. YouTube. <https://www.youtube.com/watch?v=BgtB31gXkoA> | Vídeo | <https://www.youtube.com/watch?v=BgtB31gXkoA> |
| API de acceso a la base de datos | Katie DiSimone. (2020). *mLab to Atlas for existing NS users.* [Video]. YouTube. <https://www.youtube.com/watch?v=hXSLBACjYQo> | Vídeo | <https://www.youtube.com/watch?v=hXSLBACjYQo> |
| Pruebas unitarias | Domini Code. (2018). *Como usar Postman, en español.* [Video]. YouTube. <https://www.youtube.com/watch?v=qsejysrhJiU> | Vídeo | <https://www.youtube.com/watch?v=qsejysrhJiU> |

1. **GLOSARIO:**

| TÉRMINO | SIGNIFICADO |
| --- | --- |
| Identación | dentro del contexto del desarrollo *software*, identar es un término que significa mover un bloque de código hacia la derecha insertando espacios o tabulaciones, para así separarlo del margen izquierdo y distinguirlo mejor del texto adyacente, es decir de organizar mejor el código. La identificación se considera como principio fundamental de buena práctica de la programación *software*. |
| JSON | JavaScript Object Notation (JSON). Es un formato basado en texto estándar que se utiliza para representar datos estructurados en la sintaxis de objetos de JavaScript. Se utiliza principalmente para transmitir datos en aplicaciones web (como es el caso de enviar algunos datos desde el servidor al cliente, por lo tanto, estos datos pueden ser mostrados en páginas web, o viceversa). |
| *Localhost* | en el contexto de las redes de computadoras, *localhost* es una computadora que ejecuta un programa. La computadora que se tiene al frente funciona también como un servidor virtual; este modo es comúnmente utilizado para realizar pruebas en el desarrollo web. |
| Manejadores o *handlers* de eventos | en la programación con JavaScript un evento por sí mismo carece de utilidad. Para que los eventos resulten útiles o aplicables, se debe asociar funciones o código JavaScript a cada evento. De esta forma, cuando se produce un evento de manera inmediata se ejecuta el código indicado, por lo que la aplicación responde ante cualquier evento que se produzca durante su ejecución. Las funciones o código JavaScript que se realizan para cada evento se denominan "manejador de eventos”. |
| *Middleware* | bloque de código que se ejecuta entre la petición que hace el usuario (*request*) hasta que esa petición llega al servidor. De esta manera son funciones que tienen acceso al objeto de solicitud (req), el objeto de respuesta (res) y la siguiente función de middleware en el ciclo de solicitud-respuesta de la aplicación. |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Choque, J. C. (2016). *Adición de estándares de codificación NodeJS (1ff84b6a) · Commits · Adrian / bolivia-libre*. GitLab. <https://gitlab.agetic.gob.bo/dbarra/bolivia-libre/commit/1ff84b6ae70f0e9436f6e6ddc7f3e060bc9556ad>

Chrome Web Store. (2021). *Postman*. <https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=es-419>

Colaboradores de los Proyectos Wikimedia. (2020). *Framework para aplicaciones web*. <https://es.wikipedia.org/w/index.php?title=Framework_para_aplicaciones_web&oldid=128980356>

Express. (2021). *Direccionamiento básico de Express*. <https://expressjs.com/es/starter/basic-routing.html>

Fundéu BBV. (2012). *Edición y maquetación de documentos web*. <https://www.fundeu.es/escribireninternet/edicion-y-maquetacion-de-documentos-web>

Gómez, D. (2017). *Nodemon. vortexbird.* <https://vortexbird.com/nodemon/#:%7E:text=Nodemon%20es%20una%20utilidad%20que,desarrollo%20de%20aplicaciones%20en%20nodojs>

Lázaro, G. (2015). *Efecto Ripple de Material Design [HTML+CSS+JS].* <https://gersonlazaro.com/tutorial-efecto-ripple-de-material-design-html-css-js>

López, A. (2021). *Qué es Postman y para qué sirve*. OpenWebinars.net. <https://openwebinars.net/blog/que-es-postman/>

Material Design. (2021). *The color system.* <https://material.io/design/color/the-color-system.html#color-usage-and-palettes>

# MDN Web Docs. (2021). *Introducción a Express/Node.* <https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction>

Mozilla. (2021). *Await - JavaScript | MDN*. MDN Web Docs. <https://developer.mozilla.org/es/docs/Web/JavaScript/Reference/Operators/await>

NodeJS.org. (2021). *Acerca de Node.js*. Node.js. <https://nodejs.org/es/about/>

Npm. (2021). *npm About*. <https://www.npmjs.com/about>

Red Hat. (2021). *El concepto de IDE*. <https://www.redhat.com/es/topics/middleware/what-is-ide>

Whyte, A. (2021). *Google JavaScript Style Guide*. Google JavaScript Style Guide. <https://google.github.io/styleguide/javascriptguide.xml>

1. **CONTROL DEL DOCUMENTO**

|  | Nombre | Cargo | Dependencia | Fecha |
| --- | --- | --- | --- | --- |
| Autor (es) | Peter Emerson Pinchao | Experto Temático | Regional Cauca, Centro de Teleinformática y Producción Industrial | Octubre 2021 |
| Paula Andrea Taborda Ortiz | Diseñadora instruccional | Regional distrito capital - Centro de diseño y metrología. | Octubre 2021 |
| Ana Catalina Córdoba Sus | Revisora metodológica y pedagógica | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica. | Noviembre 2021 |
| Rafael Neftalí Lizcano Reyes | Asesor pedagógico | Regional Santander - Centro Industrial del Diseño y la Manufactura. | Noviembre 2021 |
|  | José Gabriel Ortiz Abella | Corrector de estilo | Regional Distrito Capital – Centro para la Industria de la Comunicación Gráfica. | Noviembre del 2021. |

1. **CONTROL DE CAMBIOS**

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