Universidad Autónoma del Estado de México

Centro Universitario UAEM Valle de Chalco

USO DE FRAMEWORKS PARA EL DESARROLLO WEB MODERNO

ENSAYO

QUE PARA OBTENER EL TÍTULO DE

INGENIERO EN COMPUTACIÓN

P R E S E N T A

JHONATAN HERNÁNDEZ BAÑOS

ASESOR:
M. EN C.C. RICARDO BUCIO LÓPEZ

Revisor: L.C.I. HORACIO JESÚS TACUBEÑO CRUZ

Revisor: M. EN C.C. ARMANDO RERGIS RAMÍREZ

VALLE DE CHALCO SOLIDARIDAD, MÉXICO FEBRERO 2023.

CENTRO UNIVERSITARIO UAEM XICO VALLE DE CHALCO

CUVCH

USO DE FRAMEWORKS
PARA EL DESARROLLO WEB MODERNO

ÍNDICE

Índice de Tablas

Índice de Imágenes

1. INTRODUCCIÓN

La industria del desarrollo web está siempre en constante cambio y de forma continua con la invención de nuevas tecnologías y herramientas. Se puede decir que la razón principal de la modificación de la forma de desarrollar sistemas y aplicaciones web, es la amplia gama de nuevos dispositivos móviles, como las tabletas, teléfonos, televisores con acceso a internet que cambiaron completamente el panorama de la web.

La innovación de nuevas tecnologías web, y las actualizaciones de lenguajes como HTML5 y CSS3, es muy significativo. Ya que la combinación de estas tecnologías resuelve problemas anteriores por lo que se adaptan a las nuevas necesidades de navegación y experiencia de usuario. Sin embargo, una de sus más grandes adiciones en esta nueva versión es poder añadir audio y video sin necesidad de usar Flash u otro reproductor multimedia.

Sin embargo, la actualización de HTML5 y CSS3 no son excepción, ya que también trajeron consigo una nueva forma de programar la web, y sirvió para separar el desarrollo en distintas capas, por un lado, tenemos la capa de cliente que es llamada Front-end la cual está conformada por aquellos lenguajes que nos permiten estructurar una página web como es el caso de HTML5 y CSS3, con estos se puede incluir a JavaScript que es un lenguaje de programación que nos permite dar interacción a una página web, este lenguaje es interpretado por el navegador. Por otro lado, tenemos a la capa del servidor que también es llamada Back-end la cual está conformada por lenguajes de programación del lado del servidor tales como PHP, Java, C#, Python y se puede mencionar el mismo JavaScript que puede ser utilizado para correr del lado del servidor, a este lado también incluye la integración de las bases de datos las cuales sirven para almacenar la información de manera persistente.

Es por ello por lo que hoy en día han surgido diferentes herramientas, librerías y estructuras para el desarrollo, a estas herramientas se les conoce como frameworks los cuales nos ayudan a resolver y simplificar una serie de problemas que surgen a la hora de desarrollar algún sitio, página o aplicación web y que además permite agilizar el desarrollo e implementación de estos.

Actualmente existes diferentes tipos de frameworks y cada uno de ellos está especializado en resolver algún tipo de necesidad en específico y la utilización de estos hace que se divida el desarrollo en capas por un lado está la capa de cliente también llamado Front-end y por otro lado está la capa del servidor también conocida como Back-end que en esta capa igual se incluye la conexión y administración de las bases de datos.

Algunos de los cambios más significativos que ha tenido el desarrollo web moderno es la adaptación de las páginas y sitios web a diferentes tipos de entornos y resoluciones de pantalla, ya que hoy en día se tiene una amplia gama de dispositivos móviles como tabletas y teléfonos inteligentes, los cuales nos permiten consultar información y mantenernos conectados a internet. Esto trajo con sigo un término nuevo en el desarrollo web que se denomina como *Responsive Web Design*.

Al hablar de *Responsive Web Design* entendemos como aquel diseño que es adaptable a cualquier dispositivo desde donde es visualizado, como, por ejemplo: tabletas, teléfonos inteligentes, laptops o computadoras de escritorio con variedad de resoluciones de pantalla, etc. El uso de esta filosofía de desarrollo y diseño genera sitios inteligentes, que le dan al usuario final una mejor usabilidad y experiencia (Quesada, 2013).

De acuerdo con (Vega, 2013) el término *Responsive Web Design* concretamente la idea nació en el año 2008 por el consorcio W3C en su recomendación "*Mobile Web Best Practices*" que definieron el concepto "*OneWeb*", que se basa en la idea de realizar un diseño web para todos y accesible desde cualquier dispositivo, buscando que la experiencia del usuario se enriquezca.

Unos de los problemas que surge a la hora de desarrollar aplicaciones web para los distintos dispositivos móviles y de escritorio. Es que cada uno de estos dispositivos cuentan con su propio entorno, esto quiere decir que tanto la interfaz como el tipo de funciones que tiene cada uno de ellos, tienen que ser implementados correctamente para el buen funcionamiento de una aplicación. Esto hace que los desarrolladores se vean obligados a tomar muchas decisiones acerca de los cambios que tendrá la aplicación en diferentes tipos de dispositivos.

Todo esto y aunado al crecimiento exponencial que ha tenido la web a través de los años ha creado la necesidad tanto de empresas como de los mismos desarrolladores de utilizar e implementar diferentes tipos de frameworks que permita acelerar la creación de sitios, páginas o aplicaciones web de una forma muchas más rápida y ágil, manteniendo las mejores prácticas, así como el uso de patrones permitiendo el desarrollo seguro y escalable.

2. DESARROLLO

Actualmente en nuestro planeta existen millones de dispositivos conectados todos los días a internet y debido a la gran demanda de estos como lo son smartphones, tabletas, libros electrónicos, televisores, videoconsolas, etc. Ha traído consigo una evolución constante en cuanto a la infraestructura y los servicios que consumen estos dispositivos.

Debido a la pluralidad de terminales, es los últimos años los diseñadores web han buscado soluciones para adaptar las interfaces de usuario a los distintos tamaños de pantalla y resoluciones de los dispositivos móviles y desktops. A partir del año 2000, en el mundo de los diseñadores comenzaron a realizar páginas web basadas en la resolución, siendo el cliente el que debía de adaptarse a la resolución para la cual se había optimizado el contenido, todo lo contrario de lo que se busca hoy en día, donde el contenido es que se debe adaptarse al dispositivo en concreto.

La web actualmente forma parte de nuestro día a día, aunque va más allá de los diseños agradables y animaciones bonitas que despiertan los sentidos de los usuarios, detrás de la web encontramos una serie de tecnologías que hacen posible todo esto. Estas tecnologías, se encargan de que la web funcione correctamente para cada usuario.

En el campo del desarrollo web, las tecnologías con la que se desarrolla han tenido una gran evolución a través de los años, pasando de simples sitios estáticos con poca información y limitadas por el contenido de estas a complejas páginas donde encontramos libros, revistas, artículos, etc. Así como complejos y grandes e-commerce que son páginas donde nos permite comprar artículos para nuestro hogar y enviar a domicilio. Y también contamos con aplicaciones que almacenan terabytes de información en audio y video que son consumidos por millones de usuarios diariamente.

Todo esto ha causado que exista diferentes conceptos en el término de la web, en donde hoy en día encontramos que existe páginas web, sitios web y aplicaciones web. Las diferencias de estos términos es que una página web es un documento HTML que también incluye CSS y JavaScript para su funcionamiento además de que es pequeño en términos de contenido y desarrollo, por otro lado, un sitio web es el conjunto de páginas web sobre un mismo dominio y estos sitios web son informativos y por último una aplicación web es un tipo de software que puede ser soportado y ejecutado por los navegadores de internet o por una intranet o red local. Las aplicaciones web se ejecutan por medio del navegador y no necesitan ser instaladas en una computadora o smartphone, ya que los datos o archivos utilizados están almacenados en una red o en la nube.

El interés en la web nunca ha sido tan grande. Nuevos frameworks, mejores herramientas y la evolución de los estándares están surgiendo de todas partes y para los desarrolladores a veces es difícil mantenerse al día. La web moderna está en constante crecimiento, más dispositivos que nunca pueden acceder a ella e interactúan entre sí, y las técnicas están en constante crecimiento para abordar este problema. La adición de media queries a CSS permitió que el estilo de las páginas web se orientaran a pantallas con características específicas, en lugar de que la pagina se viera igual en todos los dispositivos. Este pequeño ajuste cambio fundamentalmente la forma en que se diseñaron los sitios, lo que dio lugar a un nuevo conjunto de técnicas para la creación de sitios web conocido como *Responsive Web Design*, en el que el diseño de una sola página web debe responder a las capacidades del dispositivo que se está renderizando.

2.1. Responsive Web Design

Responsive Design, o *Response Web Design* (*RWD*), es una técnica utilizada para diseñar una página web que se adapte a las capacidades específicas de un dispositivo en particular. Esta técnica se habilito principalmente mediante una función de CSS conocida como media queries, y antes de que se generalizara este término, era común crear varias versiones de un sitio, una para escritorio y otra para dispositivos móviles, y luego entregar HTML diferente basado en el user agent (la cadena que envía el navegador para indicar la versión) del navegador.

Esto a menudo requería mantener dos bases de código similares pero separadas y, como resultado, muchos sitios móviles eran simplemente versiones limitadas de un sitio de escritorio y, a medida que los usuarios se pasaban cada vez más a los dispositivos móviles, esto se volvió insostenible. La expansión de los tipos de pantallas en las que podrían aparecer los navegadores web (móviles, tabletas, computadoras, portátiles, relojes, televisores) dio como resultado la práctica de crear una versión que se puede adaptar a muchos dispositivos.

La creación del concepto como tal se le atribuye a Ethan Marcote, en un artículo que escribió en la revista online *"A List Apart"* en el año 2010 y extendida en su libro *Responsive Web Design*, aunque este concepto fue introducido anteriormente por la W3C en el año 2008, bajo el nombre de "*One Web*", en donde se especifica que el mismo contenido y los servicios que ofrece un sitio web estén disponibles, independientemente del dispositivo que se use (Zapata, 2018).

A nivel implementación *Responsive Web Design* tiene tres conceptos claves. El primero de ellos es el uso de las medias queries que nos ofrece CSS3 permitiéndonos aplicar estilos condicionalmente teniendo en cuenta parámetros de la pantalla. El segundo se trata del diseño web fluido, se trata de Layouts definidos en porcentajes que se ajustan a los anchos de la pantalla. Y por último

el tercer concepto se trata de los elementos fluidos dentro de estos Layouts, como son las fuentes, las imágenes o elementos multimedia.

Al crear un sitio con *Responsive Web Design* solo necesitamos una única versión de HTML y CSS que funcionará adecuadamente en cualquier tipo de dispositivo y resolución. Aplicando esta técnica permite reducir el tiempo de desarrollo y evita los contenidos duplicados.

Media Queries

De acuerdo con (Consortium, 2020), una media querie se compone de un tipo de medio (como pantalla) y una expresión lógica que determina la capacidad del dispositivo que el navegador se está ejecutando, como la resolución y la orientación de la pantalla (vertical u horizontal). Cuando la media querie se evalúa como verdadera, la media querie dirige a los navegadores al CSS que ha escrito y configurado específicamente para esas capacidades. Las medias queries son compatibles con las versiones actuales de los principales navegadores.

Las medias queries pueden modificar la apariencia e incluso el comportamiento de un sitio o aplicación en función de un conjunto de coincidente de condiciones sobre la configuración del dispositivo, navegador o sistema del usuario. Estas medias queries son una poderosa herramienta que se implementa dentro de las herramientas de CSS.

Layout Fluido

Los Layouts fluidos son Layouts cuyas dimensiones son determinadas por porcentajes y no por pixeles. Como resultado se obtiene un diseño mucho más manejable. Por ejemplo, podemos dividir un Layout en tres columnas dividiendo el 100% de su espacio en un tamaño de 50% de ancho y 20% para las otras dos respectivamente. Usando este tipo de técnica definimos un Layout para el cual

no es necesario que el usuario emplee un navegador de ancho especifico, si no que el ancho de los elementos se ajustará en forma de acordeón al ancho definido por el navegador del dispositivo ya sea un ordenador de escritorio, una Tablet o un Smartphone.

Este tipo de construcción de diseños evita muchos problemas que aparecían en los Layouts de ancho fijo. Ya que el sitio se adapta al ancho disponible, el diseño puede ajustarse mejor a los espacios disponibles haciendo desaparecer esos espacios en blanco molestos que aparecían en los Layouts de ancho fijo.

Implementando estrategias *Responsive Web Design* y apoyándose en las medias queries de CSS podemos optimizar la interfaz para diferentes resoluciones, haciendo que este tipo de Layouts sea una buena estrategia. Sin embargo, también existen varios problemas que se deben solucionar ya que todo esto no es suficiente para asegurar que el diseño se vea adecuadamente tanto en un teléfono como en una computadora.

Layout Elástico

Los Layouts elásticos se tratan de unos Layouts similares a los Layouts fluidos con la excepción de que estos tienen la restricción de ser controlados por el tamaño de la fuente, típicamente empleando la medida “em”. Un “em” es el tamaño equivalente al tamaño predeterminado definido para la fuente. Por defecto el texto tiene un tamaño de fuente de 16px, por lo tanto, un “em” correspondería con 16 pixeles calculando los tamaños proporcionalmente. Este tamaño por defecto puede cambiar dependiendo del navegador. Este tipo de Layouts proveen un fuerte control sobre la tipografía. En general, los diseñadores recomiendan una longitud de línea para cada párrafo de entre 45 y 70 caracteres para que proporcione una buena experiencia buena al usuario a la hora de leer el texto.

Entre los beneficios de este tipo de Layouts es que los usuarios pueden incrementar y decrementar el tamaño de la fuente escalando los elementos en proporción al tamaño de ésta, aunque, por el contrario, se puede generar exceso de contenido vacío que disminuye el aspecto estético del diseño web.

Layouts Híbridos

La opción de los Layouts híbridos combina las ideas de los Layouts descritos anteriormente. Un ejemplo puede ser mantener una columna con un ancho fijo, pero emplear medidas relativas para el resto de las columnas. Estas técnicas por ejemplo se emplean cuando se diseñan Layouts que necesitan elementos como anuncios con un tamaño especificado en una de sus columnas y ocupar el resto del espacio con porcentajes. Sin embargo, no es una opción muy empleada, por la dificultad de implementación mezclando cálculos relativos con absolutos.

2.2. Front-end

Es la parte de una aplicación que interactúa con los usuarios, es conocida como el lado del cliente. Básicamente es todo lo que el usuario ve en la pantalla cuando accede a un sitio web o aplicación: tipografías, colores, imágenes, iconos, adaptación para distintas pantallas (*RWD*), elementos gráficos, animaciones y, sin lugar a duda, mejora la experiencia de navegación en el lado del cliente.

Tradicionalmente, cuando los desarrolladores web comenzaron a construir sitios web en los años 90, se encargaban de toda la parte de programación. Configuraban el servidor, escribían todo el código del lado del servidor que eventualmente generaría el HTML y finalmente agregaban CSS para darle estilos al sitio.

Pero debido al avance de las tecnologías web y el hardware informático, hoy en día se pueden crear experiencias más sofisticadas que nunca. Debido a esta complejidad, el rol de un desarrollador web ha evolucionado naturalmente y se ha divido en diciplinas separadas. Específicamente, se trazó una línea ente el Back-end y el Front-end (Young, 2015).

El Front-end no debe de confundirse con el diseño web. Aunque algunas personas que se dedican a esta línea del desarrollo puedan poseer habilidades de diseñador, su función no es diseñar un sitio web, sino que es la especialidad que se encarga de implementar un diseño que previamente que se haya realizado, y traduce este diseño utilizando tecnologías del lado del cliente como los son HTML, CSS y JavaScript.

Las personas encargadas de desarrollar la parte Front-end de algún sitio o aplicación web se les denomina desarrollador Front-end, este rol se encarga de programar el código para hacer que la interfaz sea atractiva, intuitiva y que la experiencia de usuario sea agradable para su público objetivo (Devs, 2018).

Por otra parte, el desarrollo Back-end se encarga de programar toda la lógica necesaria del lado del servidor, tales como administración de archivos, consulta y manejos de bases de datos, así como el generar toda la capa de comunicación con la cual el Front-end enviara y recibirá los datos necesarios para desplegar en la interfaz, estos dos roles trabajan siempre por y para el usuario final teniendo en cuenta lo siguientes puntos:

- **Usabilidad:** hace referencia a la facilidad que deben ofrecen las aplicaciones móviles o webs para que el usuario final interactúe sin necesidad de tener un alto grado de conocimientos en la navegación.

- **Accesibilidad:** cuando se habla de accesibilidad se hace en varios términos, en primer lugar, se refiere al acceso a herramientas digitales desde cualquier dispositivo, por otro lado, sería la forma en que se pueden acceder a estas herramientas personas con discapacidad. En definitiva, se refiera a la eliminación de barreras para la equiparación de oportunidades.

- **Experiencia de usuario:** es la interacción que tiene el usuario con el entorno digital, si la experiencia al utilizar diferentes dispositivos de acceso es positiva o por el contrario es compleja y poco satisfactoria.

HTML

Hypertext Markup Language es el lenguaje central de la web. No es necesariamente un lenguaje de programación para crear aplicaciones, sino un lenguaje para escribir documentos. Hay dos enfoques fundamentales para expresar documentos estructurados en ingeniería de software: marcado y distanciamiento. En los lenguajes de marcado, como HTML, la estructura y la anotación están alineadas con el texto y el contenido del documento.

HTML anota secciones de texto con etiquetas, indicadas por corchetes angulares que le dan un significado a la sección correspondiente, o como una forma de incrustar elementos que no son de texto dentro de una página. En cambio, los mecanismos de anotación independiente dejan un documento como esta y luego tienen un segundo archivo que describe la estructura.

El hipertexto es una forma específica de tecnología llamada hipermedia, que fue de gran interés para los investigadores en la década de 1990 cuando se desarrolló HTML. Hipermedia se refiere a cualquier medio no lineal, por ejemplo, un video en que puede navegar en el orden que desee, en lugar de seguir una

sola ruta. La navegación se realiza siguiendo hipervínculos, que son referencias a otros documentos o fragmentos de medios. Tim Berners-Lee pensó en esto como una red de documentos, de ahí el termino de *World Wide Web* (Northwood, 2018).

HTML no es un lenguaje de programación, es un lenguaje de marcado, lo que significa es un sistema para identificar y describir los diversos componentes de un documento como encabezados, párrafos y listas. El marcado indica la estructura subyacente del documento y se puede considerar como un esquema detallado legible por máquina (Robbins, 2018).

CSS

Cascading Styles Sheets, la traducción sería hojas de estilos en cascada. Él es lenguaje utilizado para definir la apariencia o estilo de una web escrita en HTML. Mientras que HTML se utiliza para describir el contenido de una página web, las hojas de estilos en casada (*CSS*) describe como debe verse ese contenido. El aspecto de la página se denomina presentación. Las fuentes, los colores, las imágenes de fondo, el interlineado, el diseño de la página, etc., se controlan con CSS. Incluso pueden agregar efectos especiales y animaciones básicas a un sitio.

Las especificaciones CSS también proporcionan métodos para controlar como se presentarán los documentos en contextos distintos de un navegador, como en forma impreso o leído en voz alta por un lector de pantalla. Con las hojas de estilos manejando la presentación, HTML puede manejar el negocio de definir la estructura y el significado del documento, según lo previsto (Robbins, 2018).

CSS es un lenguaje separado con su propia sintaxis. Permite crear reglas que especifican como el contenido debe aparecer un elemento como, por ejemplo, puede especificar que el fondo de la página es crema, todos los párrafos deben aparecer en gris usando el tipo de letra Arial, o que todos los encabezados de nivel uno deber ser azules, cursiva, tipo de letra Times.

CSS funciona asociando reglas con elementos HTML. Estas reglas gobiernan como se debe de mostrar el contenido de los elementos especificados. Una regla de CSS contiene dos partes: un selector y una declaración. Estas declaraciones se encuentran entre corchetes y una se compone de dos partes: una propiedad y un valor, separados por dos puntos. Puede especificar varias propiedades es una declaración, cada una separado por un punto y coma (Duckett, 2011).

JavaScript

JavaScript fue diseñado originalmente como un lenguaje para manipular páginas web usando una API conocida como Document Object Model (*DOM*). Originalmente creado en Netscape, fue nombrado JavaScript como un intento de montar la ola de publicidad detrás del lenguaje Java cada vez más popular, una decisión que ha causado confusión a algunos nuevos desarrolladores durante años, ya que el lenguaje tiene poco en común con Java.

Aunque se puede lograr cierta interactividad en las páginas web con CSS, JavaScript potencia gran parte de la interactividad en la web. Se considera que JavaScript está basado en objetos porque los utiliza para trabajar con los objetos asociados con un documento de página web: la ventana del navegador, el documento en sí y elementos como formularios, imágenes e hipervínculos (Terry Ann Felke-Morris, 2016).

JavaScript es un lenguaje de scripting que agrega interactividad y comportamientos a las páginas web, entre algunas de sus características podemos mencionar los siguientes puntos:

- Comprobación de entradas de formulario para entradas validas.
- Cambiar estilos para un elemento o un sitio completo.
- Carga de feeds de desplazamiento con más contenido automáticamente.
- Hacer que el navegador recuerde información sobre los usuarios.

A principios de los 2000 Microsoft disfrutando de una cuota de mercado del 95% con Internet Explorer, deja de trabajar con ECMA International en su lenguaje JScript desde que JScript se ha convertido en el scripting predeterminado del lado del cliente. Debido a esto se pone fin a la estandarización continua.

Sin embargo, Mozilla, el suceso de Netscape, lanza el navegador Firefox, esto en septiembre de 2002 que finalmente condujo al surgimiento de JavaScript nuevamente. Firefox fue un navegador popular que comenzó a tomar cuota de mercado de Internet Explorer. En 2004, Mozilla comienza a trabajar con ECMA International en la estandarización, pero no se publican nuevas especificaciones debido a la continua negativa de Microsoft a colaborar.

En febrero de 2005 comienza el resurgimiento de JavaScript. Jesse James Barrett presenta un documento técnico, "*Ajax: A New Approach to Web Applications*", el cual habla sobre la creación de aplicaciones web que permitían la carga en segundo plano en lugar de la recarga de una página completa, y JavaScript era fundamental para ellas. Ajax es una versión abreviada de Asynchronous JavaScript + XML. Esto estímulo a la comunidad JavaScript a comenzar el desarrollo de numerosos frameworks y bibliotecas, por mencionar algunos como:

- **Angular:** marco para crear aplicaciones de una sola página.
- **Dojo Toolkit:** diseñado para reducir el tiempo en el desarrollo de aplicaciones web y sitios multiplataforma.
- **Ember JS:** framework para aplicaciones de una sola página.
- **jQuery:** librería para simplificar HTML DOM.
- **MooTools:** framework para escribir código flexible entre navegadores.
- **Prototipo JS:** framework para desarrollar aplicaciones web.
- **React JS:** librería creada por Facebook para diseñar interfaces de usuario basado en componentes.
- **Vue:** framework para crear interfaces web y aplicaciones de una sola página.

En los primeros días de la web hubo conflictos debido a que Microsoft desarrollo su primera variante de JavaScript, conocida simplemente como JScript en la cual agrego nuevas características, como XMLHttpRequest, lo que permitió a los desarrolladores crear solicitudes a servidores externos para actualizar la información en la página web. La Asociación Europea de Fabricantes de Computadoras (*ECMA*) finalmente decidió fusionar estas implementaciones competidores en un nuevo estándar, llamado ECMAScript.

Por tanto, lo que se llama JavaScript hoy en día es en realidad varias implementaciones de ECMAScript, por lo que es posible escuchar términos como "*ES6*" o "*ES2018*" utilizados en referencia a JavaScript. Esto se refiere a las diferentes especificaciones publicadas por ECMA, que cumplen con los diferentes navegadores (Northwood, 2018).

JavaScript sigue siendo uno de los lenguajes de scripting competentes utilizados para construir proyectos web atractivos. Puede crear características como arrastrar y soltar y componentes como controles deslizantes, todos los

cuales aumentan en gran medida la interfaz y brindan una experiencia elegante al sitio. Los desarrolladores tienen la libertad para escribir fragmentos de JavaScript y ampliar su funcionalidad.

2.3. Back-end

Es la parte del desarrollo que se encarga de la lógica de programación del lado del servidor como la administración de archivos, conexión y consulta a bases de datos, seguridad, creación de APIs que sirven como una capa de comunicación que permite la interacción con el Front-end. Podemos decir que se trata de la parte lógica que el usuario no ve, pero es fundamental para que los elementos se visualicen correctamente. El Back-end trabaja con diferentes lenguajes de programación como los son Java, PHP, .NET, Python, JavaScript, etc.

El Back-end es la capa de acceso a datos de un software o cualquier dispositivo, que no es directamente accesible por los usuarios, además contiene la lógica de la aplicación que maneja dichos datos. El Back-end también accede al servidor, que es una aplicación especializada que entiende la forma como el navegador solicita cosas (Maldeadora, 2019).

El desarrollo Back-end se relaciona directamente con la creación de la arquitectura del software. Es la parte que se ejecuta en un servidor o en la nube, la cual no es visible para los usuarios que consultan el sitio o aplicación web. En esta capa se procesa y almacena la información a partir de los datos obtenidos mediante la interfaz de la aplicación realizada por el Front-end.

Los desarrolladores Back-end son los responsables de desarrollar la lógica y crear soluciones para que todas las acciones solicitadas en un sitio web sean ejecutadas de manera correcta. Algunas de las funciones que se gestionan en la parte del Back-end son:

- Desarrollo de funciones que simplifiquen el proceso de desarrollo.
- Acciones de lógica.
- Gestión con bases de datos.
- Disponibilidad y escalabilidad de la red.
- Ciberseguridad y respaldo de datos.

La capa del Back-end tiene muchos conceptos con los cuales los desarrolladores deben de familiarizarse, algunos de estos conceptos son servidores de aplicaciones, protocolo HTTP, APIs, Web Services, entre otros. El entendimiento y aplicación de estos conceptos es fundamental para el desarrollador Back-end ya que estos son los encargados de relacionarlos unos con otros a la hora de crear un sitio web. Dentro de los lenguajes más utilizados hoy en día para el desarrollo Back-end son los siguientes:

- **ASP.NET:** se trata de una plataforma de desarrollo web propiedad de Microsoft.
- **PHP:** es un lenguaje de código abierto muy popular especialmente utilizado para el desarrollo web y que puede ser incrustado en HTML.
- **Java:** es uno de los lenguajes de programación más utilizados, mayormente en aplicaciones de cliente-servidor para la web.
- **Python:** es un lenguaje de programación interpretado, multiparadigma y multiplataforma.
- **Ruby:** es un lenguaje de programación orientado a objetos que combina una sintaxis inspirada en Python y Perl.
- **Node.js:** permite construir programas de red escalables, es de código abierto y utiliza el mismo lenguaje que en el lado del Front-end, el cual es JavaScript.

Sin embargo, no es suficiente con dominar un lenguaje de programación Back-end, ya que un sitio web debe almacenar datos de manera persistente. Por lo tanto, un desarrollador Back-end también debe aprender y familiarizarse con las bases de datos. Entre las más comunes destacan:

- **SQL Server**
- **MySQL**
- **MariaDB**
- **Oracle**
- **PostgreSQL**
- **MongoDB**

Servidores Web

Un servidor es una computadora que está al servicio de otras computadoras y dispositivos electrónicos como impresoras, móviles, tabletas, etc. Los servidores web tienen como principal función almacenar todos los archivos de un sitio web como los son imágenes, textos, videos, archivos, etc. y los envía a los usuarios a través de los navegadores mediante el protocolo HTTP (Souza, 2019).

La comunicación entre un servidor y sus clientes se hace mediante el protocolo HTTP, es un protocolo de transmisión de información de la World Wide Web. Como características necesarias de un servidor web a nivel de software y hardware, podemos encontrar:

Software

- Sistema operativo
- Sistema de archivos
- Software servidor HTTP
- Virtual hosting
- Monitoreo de red y limites
- Sistema de seguridad

Hardware

- Rack y gabinete
- CPU
- Memoria RAM
- Unidad de almacenamiento
- Puertos de red

Cualquier computadora se puede convertir en un servidor web instalando el software del servidor y conectando la maquina a Internet. Para manejar el tráfico web a velocidades aceptables, los servidores web combinan almacenamiento de alta capacidad con unidades de procesamiento de alta velocidad. Las computadoras con almacenamiento de alta capacidad, procesadores rápidos, alta capacidad de ancho de banda y conexión a Internet de alta velocidad son las más adecuadas para ser servidores web.

Las aplicaciones de software de servidores web, incluido el software de dominio público y los paquetes comerciales, permiten que los sitios se alojen en computadoras. El software le dice al servidor que contenido entregar al cliente, como debe de verse en los navegadores de los clientes y que acciones debe realizar o permitir una vez que se muestra la página. Los servidores web suelen ejecutarse en sistemas basados en Linux o Unix y sistemas operativos Windows.

Existen muchos tipos de servidores web y cada uno de estos tienen configuraciones necesarias para poder desplegar y mostrar ciertos tipos de sitios o aplicaciones web, ya que cuenta con el software necesario para entender el lenguaje con el cual ha sido programado un sitio. Existen gran variedad de servidores web, pero los principales hoy en día en el mercado son:

- **Apache**: servidor web más popular y uno de los más utilizados. Es un software de código abierto que se puede instalar en casi todos los sistemas operativos.
- **Ngnix**: servidor web muy ligero que utiliza el proxy inverso, es decir, que protege la identidad de los servidores, lo que mejora la seguridad de toda la información albergada en ellos.
- **IIS**: servidor web desarrollado por Microsoft que ejecuta Windows a través de la tecnología IIS (*Internet Information Services*). Su gran ventaja es que es compatible con todas las tecnologías de la compañía.

- **Sun Java System**: servidor web creado para soportar una gran carga de trabajo con tecnologías muy específicas como Java, Perl, Python o Ruby. Esta especialmente pensada para programadores.
- **W3C Jigsaw**: nació en el World Wide Web Consortium (*W3C*), un consorcio internacional para establecer los entandares de internet. Es de código abierto y ha sido programado a partir de Java (Rodríguez, 2019).

HTTP

Hypertext Transfer Protocol, es un protocolo de transmisión de información, es utilizado para realizar peticiones de datos y recursos, como pueden ser documentos HTML. Diseñado a principio de la década de 1990, HTTP es un protocolo que establece un criterio de sintaxis y semántica de comunicación entre las diferentes capas de desarrollo.

El protocolo HTTP es basado en el principio de cliente-servidor, cada petición individual se envía a un servidor, el cual la gestiona y response. Entre cada petición y respuesta, hay varios intermediarios, normalmente denominados proxies, los cuales realizan distintas funciones, como: gateways o caches (MDN, 2021).

HTTP es la forma en que los navegadores web solicitan páginas web, aunque hoy en día no solo pueden solicitar necesariamente solo páginas web ya que también existen servicios basados en este protocolo para la transferencia de información entre aplicaciones. Cada comunicación es una transacción HTTP en la que el cliente envía al servidor un mensaje que contienen cabeceras y opcionalmente datos. En este encabezado se define el método con el cual se envía la petición los más utilizados son GET, POST, PUT y DELETE, también se envía la versión del protocolo, así como el host de destino y la URL relativa del recurso solicitado.

Un navegador web utiliza HTTP para comunicarse con un servidor web que aloja un sitio web. Un ejemplo de su funcionamiento lo podemos ver cuando se escribe una URL en la barra de direcciones del navegador, este envía una solicitud GET HTTP al servidor que aloja dicha URL, al igual que cuando se consume una API este envía una solicitud a un servidor de API REST. La respuesta enviada por el servidor contiene un estado HTTP 200 OK seguido de la página HTML correspondiente a la URL.

Los navegadores utilizan el protocolo para recuperar cualquier tipo de recurso como una página HTML, documento CSS, archivo JavaScript, imágenes y cualquier otro documento que necesite el sitio web. Pero es no es su único uso. Cuando se carga alguna foto a un sitio web de una red social, por ejemplo, el navegador use el protocolo HTTP, pero esta vez para enviar un documento a un servidor. En este caso, el navegador envía una solicitud POST con un cuerpo que contiene el archivo de imagen. Por tanto, el protocolo HTTP también se puede utilizar para enviar el contenido de un recurso.

Base de datos

Una base de datos es una colección de información organizada lógicamente, diseñada de tal manera que la información contenida pueda ser accedida para su posterior uso por un programa de computadora. Una base de datos suele estar controlado por un sistema de gestión de base de datos (*DBMS*).

Los datos de los tipos más comunes de bases de datos en funcionamiento hoy en día se modelan normalmente en filas y columnas en una serie de tablas para que el procesamiento y la consulta de datos sean eficaces. La mayoría de las bases de datos utilizan lenguaje estructurado de consultas o por sus siglas en inglés (*SQL*) para escribir y consultar datos (ORACLE, 2021).

El lenguaje SQL permite la comunicación y acceso a bases de datos, permite explotar la flexibilidad y potencia de los sistemas relacionales además que permite gran variedad de operaciones sobre los datos. Los orígenes de SQL están ligados a las bases de datos relacionales, específicamente las que residían en máquinas IBM bajo el sistema de gestión System R, desarrollado por un grupo de la IBM en San José, California. (SQL, s.f.)

Por definición, los datos dentro de una base de datos deben de organizarse de acuerdo con un conjunto coherente y lógico de principios subyacentes. El termino modelos de datos describe la estructura lógica de una base de datos, que determina las reglas de cómo se puede organizar y manipular la información que se encuentra dentro. Hay muchos tipos diferentes de bases de datos, que generalmente se clasifican según sus modelos de datos.

La implementación de un modelo de datos en una base de datos determinada se denomina esquema de base de datos. Este esquema especifica y describe detalles sobre como desea que se implemente la base de datos, como los tipos de datos necesarios u otras restricciones. El esquema de una base de datos es lo que la distingue de una lista o una hoja de cálculo: con un esquema, puede estar seguro de que los datos dentro de una base de datos se organizaran de acuerdo con un cierto conjunto de reglas.

Cabe mencionar, por otra parte, que no importa como este organizada una base de datos, todavía se necesita alguna forma de interactuar con la base de datos para realizar las acciones deseadas. Un sistema de administración de base de datos (*DBMS*) es el software que hace posible que los usuarios finales creen, modifiquen y administren bases de datos, así como definir, almacenar, manipular y recuperar los datos dentro de esas bases de datos.

Administradores de bases de datos

El administrador de bases de datos también conocido por sus siglas en inglés como DataBase Management System (*DBMS*), es una aplicación de software utilizada para acceder, crear y administrar la información en las bases de datos. Un DBMS consta de un grupo de comandos para manipular la base de datos y actúa como una interfaz entre los usuarios finales y la base de datos (IBM, IBM, 2010).

De esta manera el DBMS también tiene como objetivo el facilitar una visión general de las bases de datos, proporcionando una variedad de operaciones administrativas como el ajuste, la supervisión del rendimiento y la recuperación de copias de seguridad. Entre algunas funciones que permite hacer un DBMS podemos mencionar las siguientes:

- **Definir datos:** permite al usuario crear, modificar y eliminar las definiciones que definen la organización de la base de datos.
- **Actualizar datos:** permite al usuario el insertar, modificar y eliminar datos de la base.
- **Recuperar datos:** permite al usuario recuperar datos de la base en función del requisito.
- **Administración de usuarios:** registra usuarios y supervisa las acciones, mantiene la integridad de los datos y supervisa el rendimiento.

El DBMS actúa como intermediario entre el usuario y la base de datos. La estructura de la base de datos en si se almacena como una colección de archivos y es la única forma de acceder a los datos son a través del DBMS este recibe todas las solicitudes de aplicaciones y las traduce en las complejas operaciones necesarias para cumplir con esas solicitudes. El DBMS esconde gran parte de la complejidad interna de la base de datos de los programas de aplicación y los usuarios (Carlos Coronel, 2019).

Los DBMS difieren de la forma en que la información se organiza internamente. La organización interna afecta la rapidez y flexibilidad con la que se puede extraer la información. Si bien hay muchos tipos de DBMS, hay cuatro modelos comunes.

- **Jerárquico:** los datos se modelan en una estructura en forma de árbol. Los datos se almacenan jerárquicamente y se representan mediante una relación padre-hijo. Si bien el padre puede tener muchos hijos, los hijos tienen un solo padre.
- **Red:** este modelo permite que cada hijo tenga varios padres. Esto aborda la necesidad de modelar relaciones complejas. Las entidades se organizan en un gráfico que se puede abordar a través de varias rutas.
- **Relacional:** modelo más utilizado por su facilidad de usar. Este se basa en la normalización de datos en filas y columnas de las tablas y se almacenan en una estructura fija.
- **Orientado a objetos:** los datos se almacenan en forma de objetos. Define una base de datos como una colección de objetos que almacena tanto los valores como las operaciones de los miembros de datos.

Existe una amplia gama de soluciones de software de bases de datos, incluidas soluciones empresariales y de código abierto, disponibles para la administración de bases de datos. Al igual que con los modelos de datos, el DBMS adecuado dependerá de los objetivos, capacidades técnicas y recursos disponibles. Dentro de los DBMS más conocidos podemos mencionar los siguientes:

- **Microsoft SQL Server:** sistema de gestión de base de datos relacional, desarrollado por la empresa Microsoft. El lenguaje de desarrollo utilizado es Transact-SQL, una implementación del estándar ANSI del lenguaje

SQL, utilizado para manipular y recuperar datos, crear tablas y definir relaciones entre ellas. (Microsoft SQL Server, s.f.)

- **MySQL:** sistema de gestión de bases de datos relacional desarrollado bajo licencia dual: Licencia pública general/Licencia comercial por Oracle Corporation y está considerada como la base de datos de código abierto más popular del mundo, y una de las más populares en general junto a Oracle y Microsoft SQL Server, sobre todo para entornos de desarrollo web (MySQL, s.f.).

- **MariaDB:** sistema gestión de bases de datos relacionales de código abierto que es un reemplazo directo compatible para la tecnología de base de datos MySQL. Fue creada como una bifurcación de software de MySQL por desarrolladores que desempeñaron un papel clave en la construcción de la base de datos original, surgió en 2009 en respuesta a Oracle Corp después de la adquisición de MySQL (Foundation, 2022).

- **Oracle Database:** sistema de gestión de base de datos de tipo objeto-relacional, desarrollado por Oracle Corporation (Oracle Database, s.f.).

- **PostgreSQL:** también llamado Postgres, es un sistema de gestión de bases de datos relacional orientado a objetos y de código abierto, publicado bajo la licencia PostgreSQL, similar a la BSD o MIT (PostgreSQL, s.f.).

- **MongoDB:** sistema de base de datos NoSQL orientado a documentos de código abierto. En lugar de guardar los datos en tablas, tal y como se hace en las bases de datos relacionales, MongoDB guarda estructuras de datos BSON con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida (Mongodb, s.f.).

Lenguajes del lado del servidor

El desarrollo de sitios web necesita utilizar algún lenguaje de programación que permita su ejecución desde un servidor web a estos lenguajes se les conoce como lenguajes de programación Back-end, estos son los que se ejecutan inmediatamente antes de que el sitio web se envíe a través de Internet al usuario. Los sitios web que se ejecutan en el servidor pueden realizar un amplio abanico de tareas hasta formar el propio sitio web que va a ver el usuario: acceso a base de datos, conexión a API's o servicios externos.

El desarrollo Back-end, construye la infraestructura digital y la funcionalidad detrás de escena de un sitio web para garantizar que funcione sin problemas. El Back-end está formado por el servidor en el que está alojado el sitio web, una aplicación opera el sitio y una base de datos que almacena los datos del sitio. Los desarrolladores pueden usar una variedad de lenguajes de programación para el desarrollo del Back-end, ya que los servidores se pueden configurar para comprender prácticamente cualquier lenguaje.

Como anteriormente hemos mencionado hay varios lenguajes de programación Back-end bastantes populares y difundidos, dentro de estos podemos encontrar a PHP, Perl, C# .NET, Java, Ruby, Python y JavaScript. Estos lenguajes se encargan de administrar, distribuir y almacenar la información en una base de datos y construir dinámicamente y devolver ficheros HTML y de otros tipos como PDFs, imágenes o datos tipo JSON, XML, etc.

PHP

Fue desarrollado por Rasmus Lerdorf en 1994. Desde su creación, PHP se ha convertido en uno de los lenguajes de programación del lado del servidor lideres en el mundo. Una encuesta de W3Tech muestra que más del 79% de los sitios web a nivel mundial funcionan con este lenguaje.

Dentro de las características que encontramos en PHP es que es un lenguaje multipropósito fácil de usar. Funciona perfectamente con una amplia gama de bases de datos y sistemas operativos. Los marcos modernos, una base de código masiva y la comunidad PHP activa son factores que impulsan la evolución continua de PHP.

También es un lenguaje versátil y de código abierto por lo que existen muchas librerías gratuitas y frameworks que simplifican la programación, eliminando la necesidad de escribir grandes líneas de código repetitivas y consultas a bases de datos tediosas, esto hace que los desarrolladores pueden aprovechar estos recursos para desarrollar más rápidamente el Back-end. Casi todos los sistemas operativos, como Windows y Linux, son compatibles con PHP.

Entre sus principales ventajas podemos mencionar:

- Es de libre distribución.
- Es de código abierto, no se requiere la compra de licencias.
- Extensa comunidad de desarrolladores.
- Amplia disponibilidad de librerías de distribución gratuitas para múltiples propósitos.
- Soporte a distintos motores de bases de datos, como MySQL, PostgreSQL, Oracle, SQL Server, entre otras.
- Fácil de aprender.
- Gran variedad de frameworks.

Desventajas:

- No es un lenguaje fuertemente tipado.
- Brechas abiertas de seguridad si hay malas configuraciones.
- Permite el código espagueti, por lo que genera malas prácticas de programación.

Perl

Practical Extracting and Reporting Language, fue desarrollado por Larry Wall en 1987, es un lenguaje de propósito general desarrollado originalmente para la manipulación de texto y hoy en día se utiliza para una amplia gama de tareas que incluyen administración de sistemas, desarrollo web, programación de redes, desarrollo de GUI y más. Es un lenguaje interpretado, lo que significa que su código se puede ejecutar tal cual, sin una etapa de compilación que cree un programa ejecutable no portátil.

Perl está inspirado a partir de lenguajes como C, sh, awk y sed, que son lenguajes provenientes de los sistemas Unix, pero este está enfocado para ser más práctico y fácil que estos últimos ya que sigue una sintaxis básica para escribir programas.

Entre sus principales ventajas podemos mencionar:

- Toma las mejores características de otros lenguajes, como C, awk, sed, sh y BASIC, entre otros.
- La interfaz de integración de bases de datos de Perls DBI admite bases de datos de terceros, incluidas Oracle, Sybase, PostgreSQL, MySQL y otras.
- Funciona con HTML, XML y otros lenguajes de marcado.
- Admite programación tanto procedimental como orientada a objetos.
- Es extensible. Hay más de 20,000 módulos de terceros disponibles en la Comprehensive Perl Archive Network.
- El intérprete Perl se puede integrar en otros sistemas.

Desventajas:

- Lentitud al inicio de su ejecución.
- Sin control de excepciones.
- Utiliza muchos recursos del equipo donde es desplegado.

C# .NET

Es uno de los lenguajes de programación más populares para el desarrollo Back-end, comúnmente llamado C-Sharp, es un lenguaje fuertemente tipado y orientado a objetivos. Fue desarrollado por el equipo de Andrés Hejlsberg en 1999. C# tiene sus origines en la familia de lenguajes C y su primera versión, tal y como explica Microsoft, tenía mucha similitud a Java. La creación del lenguaje fue hecha para ser una alternativa viable a Java en Windows. Muchas de sus características fueron evolucionando y mejorando hasta llegar a la versión actual.

El lenguaje fue diseñado para su uso en .NET, cuyo objetivo de esta plataforma es crear aplicaciones de forma sencilla. Por tanto, este lenguaje se utiliza para diseñar aplicaciones en esta plataforma. El uso del lenguaje es utilizado para desarrollar aplicaciones en entorno web, así como móviles y de escritorio.

Entre sus principales ventajas podemos mencionar:

- Sistema de tipo unificado, permitiendo realizar operaciones comunes y que los valores de todos los tipos se puedan almacenar, transportar y utilizar de manera coherente.
- Integración con otros lenguajes.
- Multihilo, permite dividir el código en múltiples hilos de ejecución, trabajar en paralelo y sincronizarlos al final.
- Lenguaje orientado a objetos y orientado a componentes.

Desventajas:

- Elevado consumo de recursos, su uso necesita muchos recursos para su correcto funcionamiento y compilación de recursos programados, esto significa más memoria RAM y velocidad de procesamiento del CPU.

- Escaso soporte comunitario, no cuenta con foros de ayuda de colaboración desinteresada lo suficientemente amplios para colaborar en el desarrollo web.
- Plataforma especifica de desarrollo, se necesita un entorno de desarrollo integrado (IDE).

Java

Es un lenguaje de programación y una plataforma informática comercializada por primera vez en 1995 por Sun Microsystems. Se constituye como un lenguaje orientado a objetos, su intención es permitir que los desarrolladores de aplicaciones escriban el programa una sola vez y lo ejecuten en cualquier dispositivo (Oracle, Oracle, 2021).

Una de las principales ventajas de desarrollar software con Java es su portabilidad. Una vez que haya escrito código para un programa Java en una computadora portátil, es muy fácil mover el código a un dispositivo móvil. Cuando el lenguaje fue inventado en 1991 por James Gosling, el objetivo principal era poder “escribir una vez, ejecutar en cualquier lugar”.

Entre sus principales ventajas podemos mencionar:

- Es simple, Java ofrece la funcionalidad de un lenguaje potente, derivado de C y C++, pero sin las características menos usadas y más confusas de estos, haciéndolo más sencillo.
- Orientado a objetos, permite diseñar el software de forma que los distintos tipos de datos que se usen estén unidos a sus operaciones.
- Es distribuido, Java proporciona una gran biblioteca estándar y herramientas para que los programas puedan ser distribuidos.
- Independiente a la plataforma, significa que los programas escritos en el lenguaje pueden ejecutarse en cualquier tipo de hardware, lo que lo hace portable.

Desventajas:

- Java es lento, especialmente en comparación con los lenguajes que se compilan de forma nativa, como C++.
- Requiere una cantidad significativa de espacio en memoria, particularmente en comparación con lenguajes como C y C++.
- La sintaxis puede ser difícil y compleja para nuevos desarrolladores.

Python

Es un lenguaje de programación de alto nivel, de propósito general y muy popular. Python se centra en la legibilidad del código con su uso de sangría significativa. Tipado dinámicamente y recolectado de basura, admite múltiples paradigmas de programación, incluida la programación estructurada (en particular, de procedimiento), orientada a objetos y funcional (Gattinoni, 2010).

El lenguaje de programación Python se utiliza en el desarrollo web, aplicaciones de aprendizaje automática, junto con toda la tecnología de vanguardia en la industria del software. El lenguaje es muy adecuado para principiantes, también para programadores experimentados con otros lenguajes de programación como C++ y Java. Cabe mencionar que Python se ha convertido en un elemento básico en la ciencia de datos, permitiendo a los analistas de datos y otros profesionales utilizar el lenguaje de aprendizaje automático, manipular y analizar datos y completar otras tareas relacionadas con los datos.

De igual forma el lenguaje es muy utilizado a menudo para el desarrollo Back-end de un sitio web o aplicación. El papel de Python en el desarrollo web puede incluir él envió de datos hacia y desde los servidores, el procesamiento de datos y la comunicación con base de datos, el enrutamiento de URL y la garantía de seguridad. Python ofrece varios marcos para el desarrollo web. Los de uso común incluyen Django y Flask (JetBrains, s.f.).

Entre sus principales ventajas podemos mencionar:

- Tiene una sintaxis simple que imita el lenguaje natural, por lo que es más fácil de leer y entender. Esto hace que sea más rápido crear proyectos y más rápido mejorarlos.
- Es versátil. Python se puede utilizar para muchas tareas diferentes, desde el desarrollo web hasta el aprendizaje automático.
- Es de código abierto, lo que significa que es de uso y distribución gratuitos, incluso con fines comerciales.
- Soporte de Big Tech.
- Utilidad y popularidad en áreas de Big Data, Machine Learning y Cloud Computing.

Desventajas

- Falta de patrones de diseño estandarizados.
- No hay sistema de gestión de bases de datos incorporados.
- No es adecuado para aplicaciones grandes.
- Falta de documentación.
- No hay soporte para multihilo.

JavaScript

Este lenguaje de programación que también puede ser utilizado en el Back-end utilizando Node.js el cual es un entorno de tiempo de ejecución de JavaScript, fue construido con el motor V8 de Google, Node.js tiene como propósito principal el ejecutar el lenguaje en el lado del servidor.

Fue escrito originalmente en 2009 y ha tenido un gran impacto desde entonces. Lo mejor de JavaScript es que cuanto ambos se utilizan juntos como tecnologías tanto Front-end como Back-end, JavaScript funciona como una tecnología de desarrollo completo (Shah, 2021).

Una de las principales razones detrás de la popularidad de Node.js es que ayuda en el desarrollo de aplicaciones de red en tiempo real. Además, dado que es de código abierto, también es de fácil acceso. Con Node.js los desarrolladores tienen acceso a las APIs de E/S controladas por eventos. Node.js es similar en diseño e influenciado por sistemas como Ruby's Event Machine y Python's Twisted. Aunque Node.js toma un poco el modelo de eventos mas lejos, presentando un bucle de eventos como unas construcción en tiempo de ejecución en lugar de una librería.

Entre sus principales ventajas podemos mencionar:

- Facilidad de escalabilidad, el entorno de tiempo de ejecución multiplataforma tiene un módulo de clúster y facilita el equilibrio de carga.
- Alto rendimiento para aplicaciones en tiempo real se beneficia enormemente de su capacidad para realizar múltiples tareas.
- Fácil de aprender y rápido de adaptar.
- Mejora el tiempo de respuesta de la aplicación y aumenta el rendimiento.
- Reduce el tiempo de carga en almacenamiento en cache rápido.

Desventajas:

- Reduce el rendimiento en la manipulación de tareas informáticas pesadas, a pesar de sus ventajas de ser de un solo subproceso y controlado por eventos, Nose.js se queda corto en la ejecución de computación pesada basada en CPU por la misma razón.
- Cambios de código intensos debido a API inestable.
- El modelo de programación asíncrona lo hace código difícil de mantener.
- Falta de bibliotecas, el soporte puede poner en peligro el código existente.

2.4. Frameworks

Un framework es un marco o estructura sobre la que se puede crear software, estos tienen una base con un nivel especifico de complejidad que un desarrollador puede extender utilizando su propio código. Los frameworks pueden incluir un conjunto de librerías, compiladores, intérpretes o APIs. En general, proporciona un entorno que facilita un tipo especifico de programación para un proyecto de desarrollo de software.

Los frameworks proporcionan una forma estándar de crear e implementar aplicaciones, en este aspecto consideramos los frameworks de entorno web los cuales permiten la creación de sitios en la World Wide Web. Estos frameworks tienen como objetico la sobrecarga asociada de actividades comunes que se realizan en el desarrollo web, por ejemplo, conexión a bases de datos, marcos de plantillas y administración de sesiones de usuarios, estos a menudo promueven la reutilización de código.

Los frameworks permiten a los desarrolladores tener una base definida por una arquitectura y estructura de carpetas, también incluyen herramientas, librerías, funcionalidades, API's y dictan las reglas de cómo construir un sitio web, aplicación, servicios y otras soluciones. Por lo tanto, este se encarga de crear el esqueleto del sitio web y permite ampliarlo aún más de acuerdo con los requisitos especificados.

Dentro de las ventajas que se tienen el utilizar un framework dentro de un proyecto es que permite el ahorrar tiempo, reducir errores y permite la escalabilidad de este a futuro. Como se menciona, un framework tiene una base definida la cual permite a los desarrolladores seguir una serie de pasos para crear alguna funcionalidad en específico, esto por su parte permite la fácil integración y mejoras de nuevas funcionalidades en los proyectos.

Clasificación de arquitecturas de un framework

La arquitectura de un framework define la relación entre los diferentes componentes del marco. La arquitectura decide cómo las diversas capas interactuarán entre sí. Es importante utilizar una arquitectura bien entendida, ya que define en gran medida cómo funciona la aplicación (Patel J. , 2021).

Model View Controller: muchos frameworks funcionan utilizando esta arquitectura. Permite dividir los modelos de datos con los estándares empresariales de la interfaz de usuario. Es una buena práctica, ya que generalmente fomenta modularizarían y reutilización del código.

Model-View-ViewModel (MVVM): Los frameworks, como KnockoutJS y VueJS, utilizan el modelo de arquitectura Model-Vista-VistaModelo o Modelo-Vista-Carpeta. En esta arquitectura, la capa de vista actúa como controlador y convierte los objetos de datos de la capa Modelo en componentes administrables. Dado que la capa Vista maneja todas las solicitudes de los usuarios directamente, el enlace de datos es mucho más sencillo.

Push-based vs Pull-based: En general, los marcos MVC observan una arquitectura basada en push, también conocida como "*basada en acciones*". Adoptan acciones que realizan el procesamiento necesario y, en consecuencia, empujan los datos a una capa de vista para proporcionar el resultado. Algunos ejemplos de marcos son Spring MVC, Ruby on Rails, Sails.js, Django y CodeIgniter.

Otra opción para esta arquitectura es "*pull-based*", también denominada basados en componentes. Este es un tipo de framework que comienza con la capa de vista, que a su vez puede extraer resultados de diversos controladores según sea necesario. Algunos ejemplos de arquitecturas basadas en pull son JBoss, Tapestry, Lift, Wicket, Micro y JavaServer Faces.

Organización de tres niveles: Las aplicaciones de organización de tres niveles están bien reguladas en tres niveles físicos: aplicación, base de datos y lado del cliente. Esta base de datos suele ser una base de datos relacional.

La aplicación posee lógica de negocio que se ejecuta en un servidor y se corresponde con el cliente que utiliza HTTP. El cliente utiliza un navegador web que ejecuta el código HTML desarrollado por la capa de aplicación.\

Front-end

Un framework Front-end es un paquete de código prescrito que actúa como la estructura base de una aplicación, contienen conjuntos de herramientas, librerías y componentes listos para usar, además permiten añadir nuevos componentes, así mismo como archivos y directorios dentro de estos (Patel T. , 2020).

Los frameworks Front-end se centrar en la creación de la interfaz de usuario de una aplicación, además que permiten desarrollar sitios web más receptivos, crear productos consistentes y mejora la apariencia de la aplicación, estos marcos se basan principalmente en lenguajes de programación como JavaScript, HTML y CSS.

La responsabilidad de un framework Front-end es el desarrollar la parte visual de la aplicación basándose en las capas de diseño como los son, la experiencia de usuario y la interfaz de usuario, también conocidos por sus siglas UX/UI, además un framework permite lidiar con otros aspectos cruciales a la hora de desarrollar una aplicación como puede ser la optimización SEO, plantillas y gestión de la interacción del usuario. Algunas de las ventajas de emplear un framework son:

- **Desarrollo más rápido:** Los frameworks Front-end proporcionan un código base prescrito para varios componentes de la interfaz de usuario. Con la ayuda de esta base y sus componentes, se puede desarrollar el resto del Front-end mucho más rápido.

- **Mejor UX/UI:** Los frameworks proporcionan estilos de base estética y una base sólida para el diseño receptivo. Esto hace que sea más fácil mantener tipografías y otros elementos visuales consistentes en toda la aplicación.

- **Código confiable y mantenible:** La base de código de los frameworks populares es ampliamente utilizada y ampliamente probada. Para la mayoría de los frameworks, el código es de código abierto, fácil de mantener y tiene compatibilidad entre varios navegadores.

- **Mejor colaboración:** Los frameworks siguen los principios de arquitectura y diseño similares. Los desarrolladores Front-end están familiarizados con estos patrones de diseño. Por lo tanto, es fácil para otros comprender y trabajar en el código de la aplicación.

- **Soporte de la comunidad:** Todos los frameworks de desarrollo de aplicaciones web conocidos tienen una gran comunidad de desarrolladores. Con un fácil acceso a las soluciones y una gran cantidad de conocimiento, al trabajar con frameworks se tiene un amplio soporte y documentación.

El uso de frameworks se ha vuelto crítico para el desarrollo y mantenimiento de sitios web. Con muchas opciones disponibles, se vuelve extremadamente difícil decidirse por alguno, ya que cada uno tienes sus pros y contras a la hora de utilizarlos, algunos de los frameworks Front-end más

utilizados hoy en día se encuentra Angular, React, Vue, Svelte y Polymer los cuales son frameworks de JavaScript y Bootstrap, Semantic UI y Tailwind, que son frameworks de CSS.

Angular

Lanzado formalmente en 2016 es un framework completo desarrollado por Google, es una solución ideal para aplicaciones web de mediana a escala empresarial. Sus características avanzadas al igual que la CLI y la arquitectura basada en componentes hacen que el desarrollo web con Angular sea extremadamente fácil y rentable (Team A. , 2022).

A diferencia de React, Angular es exclusivo con su rasgo de enlace de datos bidireccional. Significa que hay sincronización de tiempo real entre la vista y el modelo, donde cualquier alteración en el modelo se replica rápidamente a la vista y viceversa. Angular no es fácil de aprender ya que tiene una amplia curva de aprendizaje. Sin embargo, hay innumerables documentos accesibles y una gran comunidad.

Una de las características clave de Angular es que utiliza TypeScript como lenguaje de programación. Alternativamente, es posible crear aplicaciones Angular utilizando lenguajes como Dart o JavaScript. Sin embargo, TypeScript sigue siendo el lenguaje principal (Sirotka, Flatlogic, 2022).

Angular tiene cuatro tipos principales de desarrollo de aplicaciones web:

- PWA (*Progressive Web Apps*).
- Animaciones de interfaz de usuario.
- Aplicaciones web y móviles.
- Aplicaciones web empresariales.

Anteriormente, existía Angular JS; el equipo de Google lo reescribió desde cero en Typescript y ahora el marco se conoce como Angular 2+ o simplemente Angular. Con Angular, puede crear soluciones multiplataforma que tengan alta velocidad y rendimiento. La desventaja principal de este framework es su gran tamaño, que puede afectar negativamente el rendimiento de las aplicaciones web. Sin embargo, el equipo de desarrollo detrás de este ha ido reduciendo el tamaño con cada nueva versión.

P R O S	Reducción del tiempo de desarrollo.	**C O N T R A S**	Procesamiento más lento.
	El patrón basado en componentes de Angular forma una interfaz de usuario con componentes individuales.		Amplia curva de aprendizaje.
	Gran ecosistema.		La documentación de la CLI no está bien definida.
	El enlace de datos bidireccional está presente.		
	Alto rendimiento.		

Tabla 1 Pros y Contras de Angular.

React

Es una librería de JavaScript y de código abierto desarrollado y creado por Facebook. React funciona como un framework y se utiliza principalmente para desarrollar Single Page Application (*SPA*). También es compatible con el desarrollo de aplicaciones móviles que generalmente no es compatible con la mayoría de los otros frameworks (Team R. , 2022).

React se integra sin mucho esfuerzo con otras librerías para realizar con éxito operaciones de enrutamiento, administración de estado e interaccione con API’s. Con React se pueden crear interfaces de usuario enriquecidas y crear componentes personalizados. También permite el desarrollo de aplicaciones móviles. Es flexible, fácil de aprender y amigable con el SEO.

El SEO (*Search engine optimization*) que significa optimización de motores de búsqueda. En términos simple, significa el proceso de mejorar un sitio web para aumentar su visibilidad cuando las personas buscan productos o servicios relacionados con algún negoció ya sea en Google, Bing y otros motores de búsqueda (Land, 2022).

React no es un framework, es específicamente una librería. La explicación de esto es que React solo se ocupa de la creación de interfaces de usuario y reserva muchas cosas a discreción de proyectos individuales. El conjunto estándar de herramientas para crear una aplicación utilizando React se denomina con frecuencia la pila (Sirotka, Flatlogic, 2022).

Dentro de las características que distinguen a React de otros es por su Virtual Document Object Model (DOM), el DOM es esencialmente un modelo o representación gráfica del documento de nuestra aplicación web creado por el navegador, sobre el cual este aplica los cambios necesarios en cada actualización de estado o evento. En React, para cada Objeto DOM, hay un "objeto DOM virtual" correspondiente, este es una representación de un objeto DOM, como una copia ligera. Un objeto DOM virtual tiene las mismas propiedades de un objeto DOM real, pero carece del poder real para cambiar directamente lo que está en la pantalla.

PROS	Reutilización de componentes.	**CONTRAS**	La curva de aprendizaje es larga.
	Virtual DOM mejora tanto la experiencia de los usuarios (UX) como el trabajo del desarrollador.		Falta de documentación adecuada.
	Una biblioteca de código abierto con una diversidad de herramientas.		Complicaciones con la utilización de JSX.
	El código estable es suministrado por el movimiento de datos de una dirección.		Problemas con SEO.

Tabla 2 *Pros y Contras de React.*

Vue

Es un framework de JavaScript para crear interfaces de usuario (UI) y aplicaciones de una sola página (SPA). Al igual que el mejor de ellos, Vue.js es de código abierto. Utiliza un patrón arquitectónico model-view-viewmodel (MVVM). Fue diseñado por Evan You y lanzado en 2014, el marco es una respuesta directa al tiempo que You paso trabajando con AngularJS en Google. La arquitectura MVVM permite distinguir la lógica de negocio, o modelo, de la interfaz de usuario grafica o vista (You, 2022).

Una de las principales características que tiene este framework son sus directivas. Las directivas son atributos HTML que permiten extender la funcionalidad de HTML, una tecnología fundamental de las páginas web que determinan la estructura del contenido de la página web.

Además de ser un framework progresivo, Vue.js también es incrementalmente adoptable. Esto significa que Vue.js y las aplicaciones que lo utilizan están diseñadas desde cero. La ventaja de esto es que es fácil de comenzar. Y el desarrollo se puede volver complejo cuando lo requiera. La biblioteca principal de Vue.js también se basa en CSS, HTML y JavaScript, las principales tecnologías dentro del desarrollo web (Brewster, 2022).

La naturaleza ligera de Vue.js, cuyo tamaño de paquete es de solo 21KBs, lo hace más rápido que sus competidores. Su DOM virtual, en particular, acelera el renderizado. Sincronizar el DOM virtual es mucho más eficiente que actualizar el DOM real, lo que ralentiza el rendimiento.

PROS	Fácil de comenzar.	**CONTRAS**	Demasiado flexible que puede causar irregularidades.
	Simplicidad y claridad.		Demasiado limitado.
	Renderización rápida.		Demasiado nuevo.
	Curva de aprendizaje fácil.		Reducción de la comunidad de desarrolladores.

Tabla 3 Pros y Contras de Vue

Svelte

Es un framework de JavaScript Front-end de código abierto para crear páginas web interactivas. El concepto general detrás de Svelte es similar a los frameworks preexistentes como React y Vue en que permite a los desarrolladores crear aplicaciones web (Edpresso, 2022).

Es uno de los marcos más nuevo dentro del desarrollo Front-end. Este marco ha hecho un gran cambio al colocar el trabajo en un paso acumulado en lugar de tocarlos en el navegador, a diferencia de los marcos como Vue y React. El funcionamiento de Svelte permite transcribir el código para actualizar el DOM en sincronización con la condición de la aplicación.

Svelte proporciona un enfoque diferente para crear aplicaciones web que algunos de los otros frameworks existentes. Mientras que los marcos como React y Vue hacen la mayor parte de su trabajo en el navegador del usuario mientras la aplicación se está ejecutando, Svelte cambia ese trabajo a un paso de compilación que ocurre cuando construye su aplicación, produciendo JavaScript vainilla altamente optimizado.

El resultado de este enfoque no es solo paquetes de aplicaciones más pequeños y un mejor rendimiento, sino también una experiencia de desarrollador

que es más accesible para las personas que tienen una experiencia limitada en el ecosistema de herramientas moderno (MDN, 2021).

Svelte se adhiere estrechamente al modelo clásico de desarrollo web de HTML, CSS y JS, solo agrega algunas extensiones a HTML y JavaScript. Podría decirse que tiene menos conceptos y herramientas para aprender que algunas de las otras opciones de frameworks.

P R O S	Menos código.	**C O N T R A S**	Falta de apoyo.
	Sin DOM virtual.		Comunidad menor.
	Verdaderamente reactivo.		Herramientas escasas.
	Ligero, simple y utiliza librerías de JavaScript prevalecientes.		

Tabla 4 Pros y Contras de Svelte

Bootstrap

Es un framework CSS de desarrollo web creado por Mark Otto y Jacob Thornton, es gratuito y de código abierto. Está diseñado para facilitar el proceso de desarrollo web responsivo, siguiendo la metodología de mobile-first proporciona una colección de sintaxis para diseños de plantillas.

En otras palabras, Bootstrap ayuda a los desarrolladores web a crear sitios web más rápido, ya que no necesitan preocuparse por los comandos y funciones básicas. El framework consiste en scripts basados en HTML, CSS y JavaScript para varias funciones y componentes relacionados con el diseño web (Team B. , 2022).

El objetivo principal de este framework es la creación de sitios web responsivos, garantizando que todos los elementos de la interfaz de un sitio web funcionen de manera óptima en todos los tamaños de pantalla. Bootstrap es uno de los frameworks más conocidos y utilizados dentro de la comunidad de desarrolladores web y debido a esto el soporte es amplio, además las actualizaciones de este framework siempre añaden nuevos elementos y utilizades que permiten dar mucha más interactividad a la interfaz.

Sin embargo, a pesar de las grandes ventajas que tiene este framework, Bootstrap tiene ciertas limitaciones que no son adecuadas para tipos específicos de proyectos. Dado que Bootstrap tiene un estilo visual consistente, necesita una gran personalización y anulación de estilos para que un proyecto sea diferente de otro. De lo contrario, todos los sitios desarrollados utilizando este framework tendrían los mismos estilos en componentes de navegación, estructura y diseño.

También cabe mencionar que al contar con una gran numero de funciones significa comprender archivos de gran tamaño. El uso de Bootstrap en un proyecto puede ralentizar el tiempo de carga del sitio web y sobrecargar el servidor si no se tiene cuidado, por lo que es necesario que al utilizar este framework solo se agreguen las clases que son necesarias y se utilice la versión minimizada de los archivos.

PROS	Fácil de usar.	**CONTRAS**	Sintaxis confusa.
	Responsive Grid.		Archivos demasiado grandes.
	Compatibilidad con diferentes navegadores.		Estilos pocos personalizados.
	Amplia documentación.		

Tabla 5 Pros y Contras de Bootstrap

Semantic UI

Es un framework CSS de desarrollo web moderno, impulsado por LESS y jQuery. Fue creado por Jack Lukic en 2014. Al igual que Bootstrap nos permite crear plantillas completamente responsivas que se adapten tanto a escritorios como a móviles.

El objetivo de Semantic radica en permitir a los diseñadores y desarrolladores mediante la creación de un lenguaje para compartir la interfaz de usuario. Lo hacen aprovechando un lenguaje semántico y descriptivo para sus clases y convenciones de nomenclatura. En lugar de usar abreviaturas, como lo hacen otros marcos, utiliza palabras reales de una manera más cercana al inglés simple (Team S. , 2022) .

Las interfaces creadas con Semantic UI es únicamente de dos maneras. La primera es la forma en que está estructurado el framework. Este utiliza cinco categorías descriptivas para definir los componentes de la interfaz:

- Un elemento de la interfaz de usuario es un bloque de creación básico. Puede aparecer solo o en grupos uniformes. Por ejemplo, un botón puede ser independiente o colocarse en un grupo de botones.

- Una colección de interfaz de usuario es un grupo de diferentes elementos que son independientes. Por ejemplo, un formulario web puede tener botones, entradas, casillas de verificaciones, iconos, etc.

- Una vista de interfaz de usuario representa una parte común del contenido del sitio web. Por ejemplo, una sección de feeds o comentarios.

- Un módulo de interfaz de usuario es un componente con funcionalidad interactiva basada en JavaScript. Los ejemplos incluyen acordeón, atenuador, modal, etc.

- Un comportamiento de interfaz de usuario es un componente que no puede existir de forma independientes, sino que se usa para inyectar funcionalidad a otros componentes.

Las interfaces creadas con Semantic no solo es significativa y está bien estructurada en términos de nombrar sus clases, sino también de nombrar, definir y describir sus componentes. Esta estructura es mucho más semántica en comparación con la que podemos encontrar en Bootstrap y Foundation.

Y la segunda cosa única de Semantic UI es que proporciona algunas características y componentes exclusivos que no están presentes en otros frameworks. Por ejemplo, feeds y comentarios en la UI, vista de componentes, sidebar and formas de la UI. Además, otra de las características es que utiliza un estilo mínimo y neutral, dejando la personalización abierta a los desarrolladores. Incluye cosas importantes y útiles al tiempo que deja de lado características adicionales que probablemente nunca usara.

PROS	Fácil de usar.	**CONTRAS**	Menor compatibilidad con navegadores.
	Mayor rapidez de diseñar una web.		Diseño menos responsivo.
	Gran variedad de temas disponibles.		Comunidad más pequeña.
			Pocas actualizaciones.

Tabla 6 Pros y Contras de Semantic UI

Tailwind

Es un framework CSS creado por Adam Wathan, a diferencia de otros frameworks CSS como Bootstrap o Foundation, no viene con componentes predefinidos. En cambio, Tailwind CSS utiliza un enfoque diferente. Aporta un nivel de control mucho más bajo mediante el uso de clases basadas en utilidades, mediante el uso de estas clases, se puede crear rápidamente un diseño personalizado con facilidad.

La solución que propone Tailwind es proporcionar una gran variedad de clases CSS que cada una tiene un enfoque propio. De esta forma, cada pequeña clase hace uso de modificadores sobre los elementos HTML que en conjunto permiten estilizar cada elemento de forma que no es necesario ni siquiera tener un gran conocimiento en CSS (Huet, 2022).

Hoy en día al crear un sitio o aplicación web conlleva el realizar distintas tareas y utilizar diferentes herramientas para poder desarrollarlo. Mapear el impacto de los estilos puede volverse tedioso y llevar mucho tiempo, lo que significa un obstáculo en el progreso de la aplicación o sitio web. La implementación de Tailwind elimina estos problemas ya que no se tiene que crear todo un sistema de estilos desde cero y haciendo uso de este framework se podrá invertir menor tiempo en el maquetado y dedicarse a la funcionalidad del sitio.

PROS	Desarrollo más rápido.	**CONTRAS**	El marcado puede ser desorganizado para proyectos grandes.
	Componentes fáciles de personalizar.		No es fácil de aprender.
	Amplia documentación.		No es la mejor opción si se está buscando minimizar el tiempo de maquetado.

Tabla 7 Pros y Contras de Tailwind

Comparación de los frameworks JavaScript desarrollando una aplicación

Para poder tener una referencia del funcionamiento de un framework Front-end, procederemos a crear una aplicación la cual permitirá crear tareas, completarlas y eliminarlas, con este ejemplo podremos ver la estructura y funcionamiento de los distintos frameworks Front-end JavaScript que hemos mencionado anteriormente, la aplicación es sencilla, pero nos permitirá ver la sintaxis que usa cada framework para poder crear aplicaciones.

La aplicación contará con un formulario de una entrada donde el usuario pueda escribir una tarea y luego enviar el formulario para que la tarea enviada aparezca en una lista de tareas, además de que guarde los elementos en el almacenamiento local del navegador, para que cuando el usuario recargue la página los elementos todavía estén presentes.

Sin embargo, aunque el concepto para la creación de esta aplicación es bastante simple, en realidad suceden muchas cosas, como eventos de enlace de datos, administración de estados y el ciclo de vida de la aplicación.

El primer framework que utilizaremos será React, para crear un proyecto nuevo con React necesitaremos ejecutar el siguiente comando desde la terminal, posicionándonos en el directorio en donde se creará la aplicación:

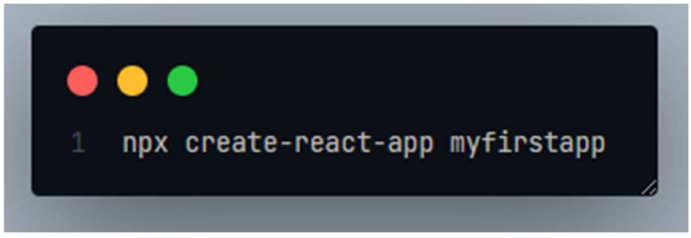


Imagen 1 Comando para crear una nueva aplicación

Este comando nos permite crear la estructura base de nuestro proyecto el cual cuenta con las configuraciones necesarias para levantar un servidor de desarrollo con el cual podemos ver nuestra aplicación y observar los cambios en tiempo de desarrollo.

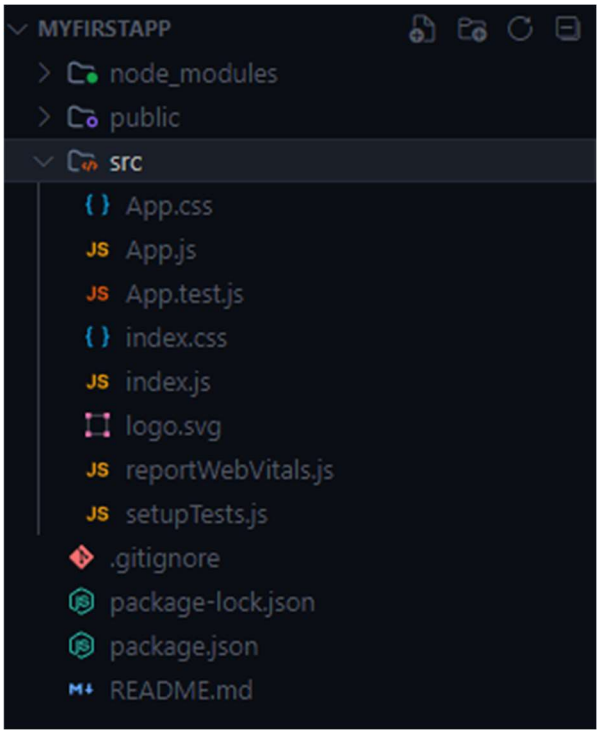


Imagen 2 Estructura base de directorios

El proyecto instala los paquetes de Node necesarios para su funcionamiento, Node es un entorno de tiempo de ejecución de JavaScript, estos paquetes son necesarios para poder transpilar el código creado en React a código nativo generando paquetes mejor conocidos como bundles, con el código de producción que entiende el navegador web.

Imagen 3 Archivo JSON con información del proyecto

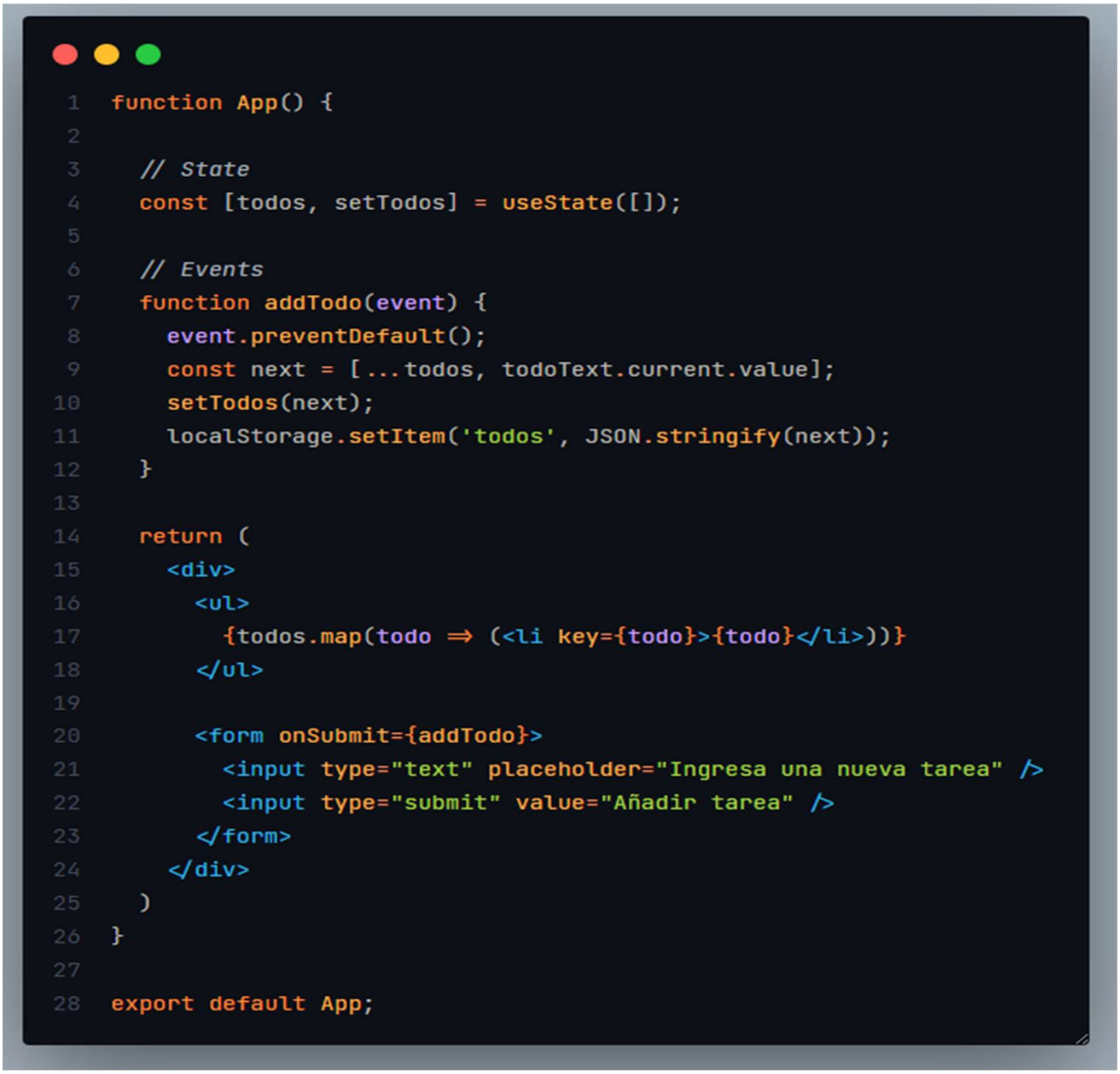
Borraremos el contenido del archivo *App.js* que se encuentra en el directorio *src*, dentro de este declaramos una función llamada App, esa función representa un componente en la interfaz de nuestra aplicación, ahora el valor de retorno de nuestra función es *JSX*, que es similar a *HTML*, pero *JSX* se amplía con una sintaxis adicional que le permite insertar código JavaScript en su *HTML*.

También necesitaremos definir el estado de nuestra aplicación, para esto definimos dentro de nuestro componente creado el estado reactivo con el Hook de estado llamado *useState*, este Hook es una función que nos devolverá dos valores, el primer elemento es el valor de la lista de tareas pendientes como estado reactivo, con esto se quiere decir que cada vez que se actualice, la interfaz de usuario se volverá a procesar para mostrar el estado más reciente y luego el segundo elemento que devuelve *useState* es una función para actualizar el estado.

Imagen 4 Creación de componente App

Ahora continuaremos creando el código *JSX* que retornara nuestra componente el cual se encargara de recorrer los elementos todos que se encuentra en el arreglo y los mostrara directamente en la interfaz de usuario como un elemento de la lista, también tendremos un elemento *form* en cual servirá para que el usuario ingrese nuevas entradas de tareas a la lista, este enviara las entradas a una función llamada *addTodo* que se encargara de actualizar el estado y almacenara el resultado y el almacenamiento en local,

haremos referencia a esta función por medio de su función *onSubmit* del formulario, el cual es activado cada vez que se activa el evento de envió.



```
function App() {

  // State
  const [todos, setTodos] = useState([]);

  // Events
  function addTodo(event) {
    event.preventDefault();
    const next = [...todos, todoText.current.value];
    setTodos(next);
    localStorage.setItem('todos', JSON.stringify(next));
  }

  return (
    <div>
      <ul>
        {todos.map(todo => (<li key={todo}>{todo}</li>))}
      </ul>

      <form onSubmit={addTodo}>
        <input type="text" placeholder="Ingresa una nueva tarea" />
        <input type="submit" value="Añadir tarea" />
      </form>
    </div>
  )
}

export default App;
```

Imagen 5 Entradas de nuevas tareas

Por último, vincularemos nuestra entrada haciendo uso del Hook useRef de React, este permite hacer un enlace del input y obtener el valor actual de este, también necesitaremos usar el Hook *useEffect* que es parte del ciclo de vida de React, *useEffect* tomara los elementos almacenados en local cuando el componente se inicializa por primera vez.

```jsx
import { useState, useEffect, useRef } from "react";

function App() {

  // State
  const [todos, setTodos] = useState([]);

  // Binding
  const todoText = useRef();

  // Lyfecycle
  useEffect(() => {
    const existingTodos = localStorage.getItem('todos');
    setTodos(existingTodos ? JSON.parse(existingTodos) : []);
  }, [])

  // Events
  function addTodo(event) {
    event.preventDefault();
    const next = [...todos, todoText.current.value];
    setTodos(next);
    localStorage.setItem('todos', JSON.stringify(next));
  }

  return (
    <div>
      <ul>
        {todos.map(todo => (<li key={todo}>{todo}</li>))}
      </ul>

      <form onSubmit={addTodo}>
        <input type="text" placeholder="Ingresa una nueva tarea" ref={todoText} />
        <input type="submit" value="Añadir tarea" />
      </form>
    </div>
  )
}

export default App;
```

Imagen 6 Código completo de la aplicación

Con esto tenemos completada la funcionalidad de nuestra aplicación, uno de los puntos fuertes de React es la utilización de la sintaxis de *JSX* la cual

permite ser mucho más descriptivo, con lo cual permite saber exactamente a que elementos están vinculados nuestros datos y eventos.

Ilustración 7 Aplicación React funcionando

El segundo framework a utilizar será Angular, para poder crear una nueva aplicación necesitaremos ejecutar el siguiente comando, para esto debemos estar posicionados sobre el directorio donde queremos crearlo.

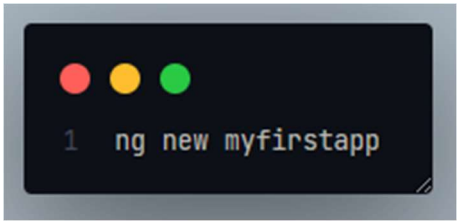


Imagen 8 Comando Angular

Al terminar de ejecutar el comando el CLI de Angular creara la estructura básica de nuestra aplicación, así como las configuraciones necesarias para poder levantar un servidor local, en el cual podemos ver los cambios realizados al momento de desarrollar la funcionalidad de nuestra aplicación, cabe mencionar que Angular instala muchas más librerías y herramientas que necesita para poder desarrollar, esto se debe a que Angular es una plataforma completa de desarrollo de aplicaciones web.

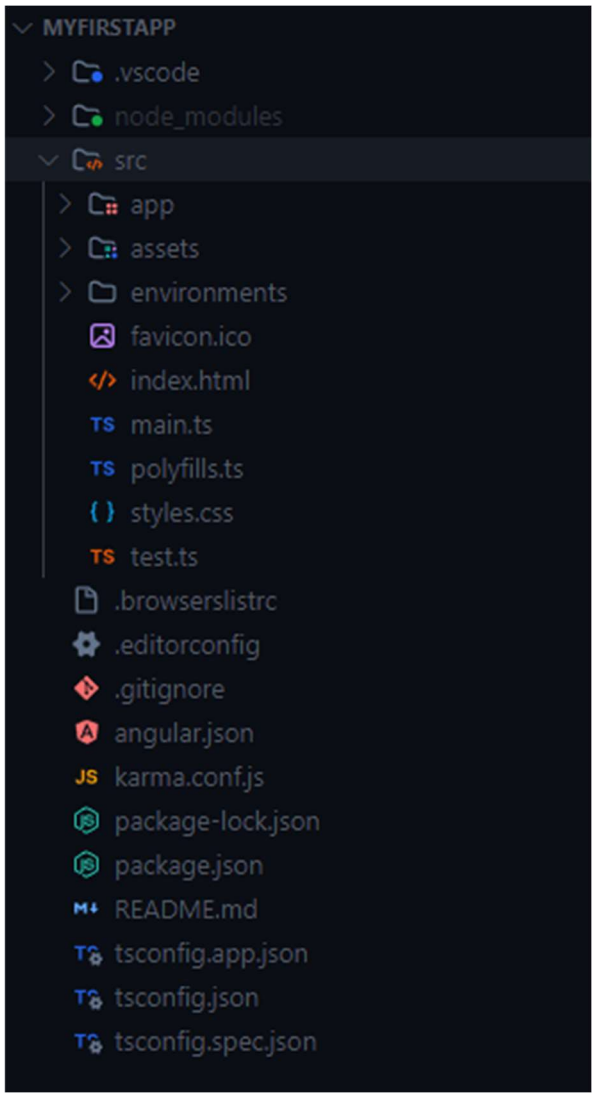


Imagen 9 Estructura de proyecto Angular

La declaración de un componente en Angular se hace por medio de un decorador llamando @Component, este decorador permite definir la lógica del componente, el template y estilos de este componente por separado, Angular utiliza por defecto Typescript para el desarrollo.

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

}

```

Imagen 10 Declaración de un componente en Angular

Para comenzar con el desarrollo de nuestra aplicación primero definiremos dos propiedades una propiedad se llamará *todos* la cual será de tipo *Array*, en Typescript podemos de igual forma definir el tipo de datos que contendrá el *Array* en este caso le indicamos que serán de tipo *String*, la segunda propiedad se llamara *todoText* y solo le asignamos el valor de un *String* vacío, estas se definirán en nuestra clase *AppComponent.*

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {

  // State
  todos: string[] = [];
  todoText = '';

}
```

Imagen 11 Propiedades de componente Angular

Continuamos declarando ahora un método de igual forma nuestro *AppComponent* este se llamara *addTodo*, este método se encargara de actualizar el estado de nuestra componente, además podemos administrar el ciclo de vida de este componente implementando el método especial *ngOnInit*, este método se llamara cada vez que nuestro componente se inicialice por primera vez, dentro de este método comprobaremos si existen *todos* guardados dentro de *localStorage* del navegador y si no es el caso inicializara nuestros *todos* como un Array vacío.

```
import { Component, OnInit } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent implements OnInit {

  // State
  todos: string[] = [];
  todoText = '';

  // Lifecycle
  ngOnInit() {
    const existingTodos = localStorage.getItem('todos');
    this.todos = JSON.parse(existingTodos as string) || [];
  }

  // Events
  addTodo() {
    this.todos.push(this.todoText);
    localStorage.setItem('todos', JSON.stringify(this.todos));
  }
}
```

Imagen 12 Lógica de componente Angular

Ahora para poder crear la interfaz de nuestra aplicación iremos al archivo llamado app.component.html, en este archivo se puede colocar lenguaje HTML la diferencia es que Angular potencia el lenguaje HTML con un lenguaje de plantillas especial que hace posible recorrer un bucle dentro de la propia plantilla,

en este caso la plantilla recorrerá nuestra propiedad *todos* para poder imprimirlos en pantalla, también haremos un *Event Binding* de nuestro formulario esto con la finalidad de enviar los nuevos *todos* que el usuario ingrese por medio de este, para poder obtener el valor del todo ingresaron utilizaremos el enlace de datos que ofrece Angular y que podemos colocarlo directamente en el template de nuestro componente.

```
<ul>
  <li *ngFor="let todo of todos">{{ todo }}</li>
</ul>

<form (ngSubmit)="addTodo()">
  <input type="text" name="todotext" [(ngModel)]="todoText">

  <input type="submit" value="Agregar Todo">
</form>

```

Imagen 13 Template de componente Angular

Con esto finalizamos nuestra aplicación, el funcionamiento de esta es el mismo que la creada con React, la diferencia radica en la forma de generar un componente, así como la lógica de este.

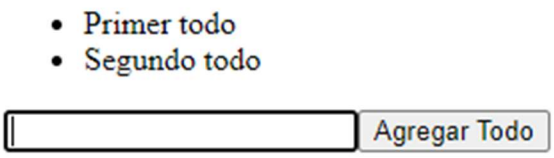


Imagen 14 Aplicación Angular

Para finalizar nuestra comparación de crearemos ahora la misma aplicación, pero utilizando el framework Vue, la sintaxis de este framework es similar a Angular en su primera versión AngularJS, pero en un paquete que es mucho más accesible para desarrolladores independientes ya que cuenta con diferentes paquetes oficiales como, por ejemplo, enrutamiento y administración de estado, así como también paquetes de terceros que permite la extensión de Vue.

Vue al igual que Angular cuenta con un CLI para la terminal el cual utilizaremos para crear una nueva aplicación de Vue, ejecutan el siguiente comando posicionados en el directorio donde se creará la aplicación.

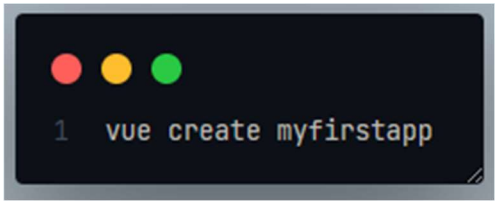


Imagen 15 Comando Vue CLI

Este comando nos creara al igual que los anteriores frameworks la estructura base de directorios, así como las configuraciones necesarias para el funcionamiento de nuestra aplicación, además de incluir las herramientas necesarias para el desarrollo de esta.

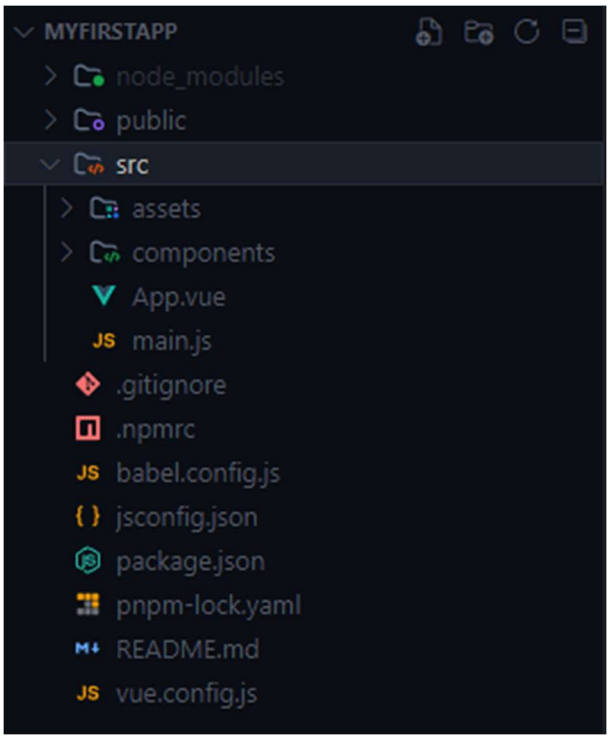
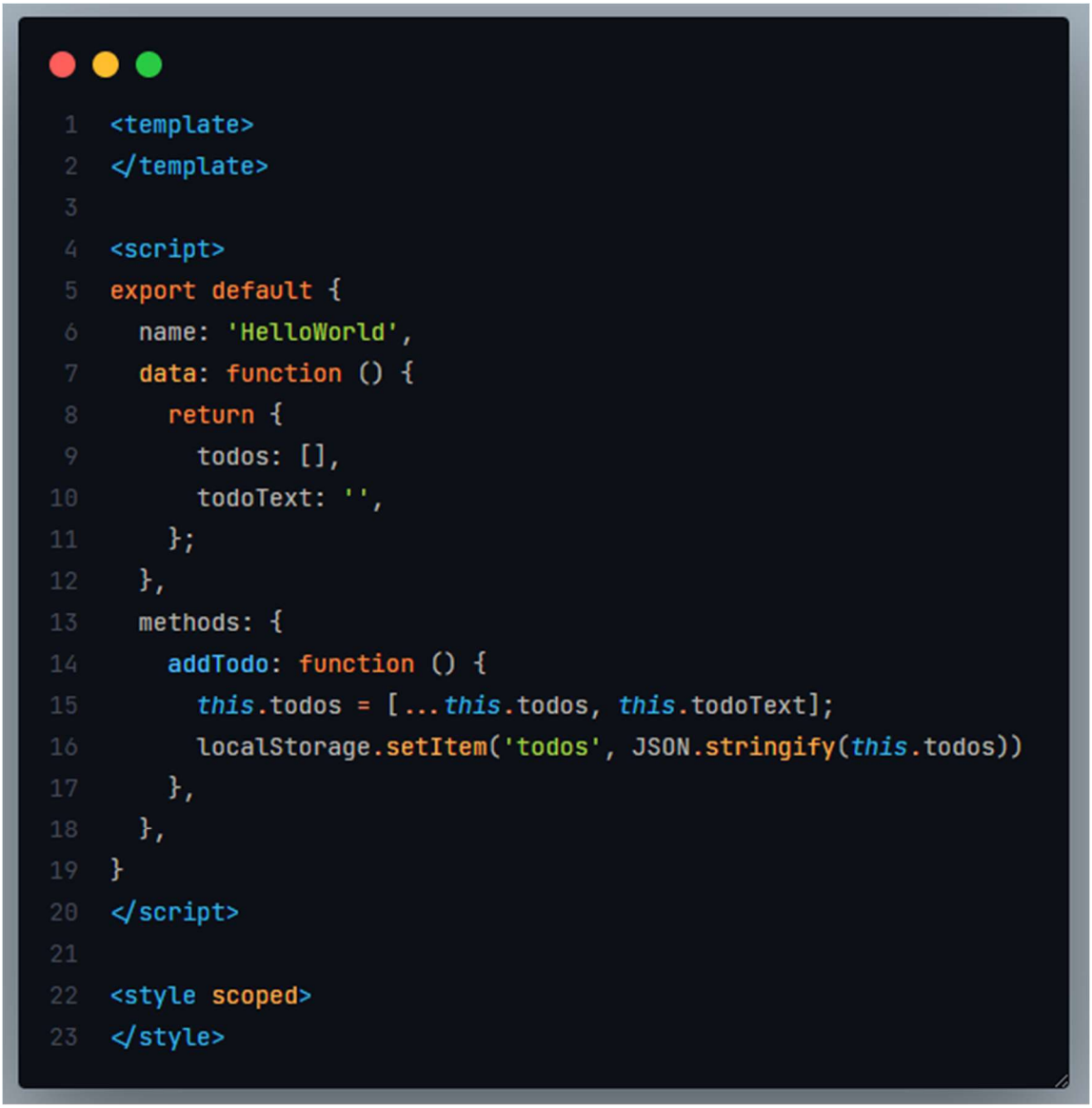


Imagen 16 Estructura base de Vue

Para comenzar debemos primero comprender que los componentes en Vue son definidos en archivos que terminen como *.vue* el código dentro de este se organiza en tres partes, una es de plantilla que se indican por medio de la etiqueta *template*, la segunda es de lógica por medio de las etiquetas *script* y la tercera es de estilos del componente que se indica con las etiquetas *style*.

Primero comenzaremos por definir el estado de nuestro componente, para estos definimos una propiedad llamada data, la cual contiene una función que retorna las variables que manejara nuestro componente, también definiremos un método el cual nos permitirá cambiar el estado de nuestro componente, este se llamara cuando se activan diferentes eventos dentro de Vue.



```
<template>
</template>

<script>
export default {
  name: 'HelloWorld',
  data: function () {
    return {
      todos: [],
      todoText: '',
    };
  },
  methods: {
    addTodo: function () {
      this.todos = [...this.todos, this.todoText];
      localStorage.setItem('todos', JSON.stringify(this.todos))
    },
  },
}
</script>

<style scoped>
</style>
```

Imagen 17 Lógica de componente Vue

Vue también incluye sus propios métodos como parte del cilio de vida del componente, para esto en nuestro componente declaramos un método llamado *mounted*, el cual se ejecutará una vez que el componente se inicialice y se muestre en la interfaz.

```
<template>
</template>

<script>
export default {
  name: 'HelloWorld',
  data: function () {
    return {
      todos: [],
      todoText: '',
    };
  },
  methods: {
    addTodo: function () {
      this.todos = [...this.todos, this.todoText];
      localStorage.setItem('todos', JSON.stringify(this.todos))
    },
  },
  mounted: function () {
    const existingTodos = localStorage.getItem('todos');
    this.todos = JSON.parse(existingTodos) || [];
  }
};
</script>

<style scoped>
</style>
```

Imagen 18 Método del ciclo de vida de componente Vue

Ahora dentro de las etiquetas *template* al igual que los anteriores frameworks Vue tiene una configuración similar a estos, las cuales permite

potenciar el lenguaje HTML, añadiendo nuevas funcionalidades a este, como por ejemplo directivas para recorrer elementos de un arreglo.

```vue
<template>
  <div>

    <ul>
      <li v-for="todo in todos" v-bind:key="todo">{{ todo }}</li>
    </ul>

    <form v-on:submit.prevent="addTodo">
      <input type="text" v-model="todoText" placeholder="Que necesitas hacer?">
      <button type="submit">Añadir Todo</button>
    </form>

  </div>
</template>

<script>
export default {
  name: 'HelloWorld',
  data: function () {
    return {
      todos: [],
      todoText: '',
    };
  },
  methods: {
    addTodo: function () {
      this.todos = [...this.todos, this.todoText];
      localStorage.setItem('todos', JSON.stringify(this.todos))
    },
  },
  mounted: function () {
    const existingTodos = localStorage.getItem('todos');
    this.todos = JSON.parse(existingTodos) || [];
  }
};
</script>

<style scoped>
</style>
```

Imagen 19 Template de componente Vue

Por último, ejecutamos nuestra aplicación para poder verla funcionando en nuestro navegador, el funcionamiento de esta es igual que los anteriores, a diferencia que no hicimos ningún otro cambio en los archivos bases por lo que dentro de esta incluye la imagen de Vue.

Imagen 20 Aplicación Vue

Con esto finalizamos el desarrollo de nuestra aplicación de tareas, como podemos observar cada framework tiene su propio CLI el cual permite crear nuevas aplicaciones, así como nuevos componentes en nuestras aplicaciones, los frameworks Front-end se caracterizan principalmente por permitirnos crear componentes reutilizables, de esta forma una aplicación grande se puede dividir en diferentes componentes y cada uno de estos contiene su propia funcionalidad, así como su ciclo de vida dentro de la aplicación, sin duda la utilización de un framework hoy en día en el desarrollo web moderno es imprescindible para poder

crear una mejor aplicación tanto en sus estructura, mantenimiento y escalabilidad.

También podemos mencionar que algo que caracteriza a React y Vue es la simplicidad y versatilidad a la hora de crear un nuevo componente, así como agregar la lógica al componente, ambos frameworks son bastantes ligeros y versátiles a la hora de crear un nuevo componente, pero de igual forma se observa que toda la lógica esta embebida dentro de los mismos componentes, hablando de sus templates, estilos y lógica de este. Por otro parte en Angular vemos que tanto los templates, estilos y lógica están separados en diferentes archivos, lo que permite una mejor legibilidad del código, además podemos observar que la integración de Typescript para el desarrollo de componentes hace que Angular sea mayormente utilizado por grandes equipos de desarrollo y empresa ya que la estructura que nos da el framework permite tener un estilo de desarrollo.

Back-end

Esta capa engloba toda la parte del servidor, la base de datos y el código que interactúa con ella. También es la capa que se encarga del código que sirve datos dinámicos al Front-end del sitio. Este se puede manejar en la mayoría de los lenguajes de programación. Dentro de estos como describimos anteriormente se encuentran Java, ASP.NET, Python y JavaScript con Node.js, pero igual hay mucho otros lenguajes que permite programar del lado del servidor.

Los framework Back-end no son más que las librerías de módulos y herramientas que ayudan a desarrollar la estructura de cualquier sitio. Si se necesita crear una web o aplicación potente, es bastante útil que ciertas cosas ya estén disponibles (Patel J. , 2021).

Los frameworks Back-end permiten a las empresas optimizar el tiempo, dinero y otros recursos que conlleva a la hora del desarrollo. Estos por su parte ayudan a los desarrolladores a concentrarse en los elementos más importantes y únicos de un proyecto al ayudar en el proceso de desarrollo y cubrir la funcionalidad básica. En pocas palabras, los desarrolladores Back-end no tiene que pasar tiempo trabajando con código estándar, sino que, en cambio, pueden centrarse solo en desarrollar la funcionalidad principal del proyecto.

Dentro del Back-end los framework son mucho más variados, estos están escritos en una variedad de lenguajes de programación y tienen una amplia variedad de características. Los principales frameworks Back-end ofrecen un conjunto diverso de herramientas y tecnologías, como programas de soporte, librerías de código, compiladores y herramientas o API específicas que ayudan a desarrollar aplicaciones y crear sistemas de manera rápida y efectiva (Lekh, 2021).

El utilizar un framework para la parte del servidor tiene muchas ventajas entre ellas podemos mencionar algunas como seguridad, minimizar el riesgo de errores humanos, aumenta la confiabilidad en el lado del servidor, acelera el tiempo de desarrollo y facilita el proceso de prueba. Además, el conocimiento de una base activa de usuarios también contribuye a la mejora continua del desarrollo Back-end.

Sin embargo, al elegir un framework Back-end existen varios factores a considerar que permitirá elegir el mejor framework a utilizar y que se adapte a las necesidades del proyecto, ya que como se mencionó anteriormente existe una gran variedad de frameworks para la parte Back-end y cada uno utiliza un lenguaje de programación distinto por lo que también dependerá tanto de la infraestructura donde estarán alojados y de las necesidades del proyecto.

Algunas consideraciones que también se tendrá que elegir al momento de utilizar un framework Back-end son:

Escalabilidad: es una de las preocupaciones más importantes al momento de desarrollar un proyecto, no importa si está desarrollando una aplicación de cara al público o si está desarrollando algo para uso interno, siempre se querrá asegurar de que la tecnología que se utiliza se adapte bien.

Seguridad: la seguridad es un tema muy importante y por buenas razones. A medida que las violaciones de datos de vuelven más comunes hoy en día, los gobiernos y consumidores están intensificando para exigir que todas las aplicaciones y proyectos sean más seguros. Por lo tanto, dadas las posibilidades implicadas de relaciones públicas, financieras y legales de la mala seguridad, los principales framework Back-end deberán aseguran que la seguridad no sea una idea de último momento, sino que sea una parte central de la infraestructura del framework.

Facilidad de desarrollo: este término abarca la curva de aprendizaje que tienen los desarrolladores, la documentación, el soporte de la comunidad, las librerías, ya sea de código abierto o no, y más. Se tiene que tomar en cuenta que el framework sea fácil de usar. Elegir un framework con documentación deficiente, pocos desarrolladores y pocos paquetes comunitarios conducirá a la frustración a largo plazo. Se termina gastando más tiempo y dinero descubriendo los conceptos básicos que desarrollando todas las características del proyecto.

Django

Es un framework de alto nivel que fomenta el desarrollo rápido y el diseño limpio y pragmático. Construido por desarrolladores experimentados, se encarga de gran parte de la molestia del desarrollo web, por lo que permite concentrarse en

escribir una aplicación sin necesidad de reinventar la rueda. Es gratuito y de código abierto (Django Software Foundation, 2022).

Django permitir el desarrollo web utilizando Python. Hasta la llegada de este framework, no era fácil escribir código para la web utilizando Python, pero Django cambio gran parte de ese pensamiento, su lanzamiento fue en 2005, en esencia, Django se adhiere muy estrictamente al paradigma Modelo Vista Controlador (MVC). Hay un asignador relacional de objetos integrado en Django siento este el Modelo, también integra un sistema para procesar solicitudes HTTP, viene con un motor de plantillas incorporado siento la parte de la Vista y un despachador de URL siento este el Controlador (Lekh, 2021).

Este es el mejor framework Back-end para Python, ya que es altamente personalizable y escalable. Cuenta con una extensa comunidad y documentación y facilita el desarrollo de aplicaciones web. Dentro de los sitios conocidos creados con este framework podemos mencionar Mozilla, Disqus, Pinterest, National Geographic, por nombrar algunos.

P R O S	Velocidad y escalabilidad.	**C O N T R A S**	No es bueno para proyectos simples.
	Soporte de la comunidad.		Puede conducir a sitios web lentos.
	Amplia documentación.		Naturaleza monolítica.
	Seguridad		Falta de convenciones.
	Versatilidad.		Difícil de aprender.

Tabla 8 Pros y Contras de Django

Flask

Flask es un framework web, es un módulo de Python que te permite desarrollar aplicaciones web fácilmente. Tiene un núcleo pequeño y fácil de extender: es un

micro framework que no incluye un ORM *(Object Relational Manager)* o tales características (Basics, 2021).

Fue desarrollado por Armin Ronacher, quien dirigió un equipo de entusiastas internacionales de Python llamado Pocco. Flask se basa en el kit de herramientas Werkzeg WSGI y el motor de plantillas Jinja2. Ambos son proyectos de Pocco.

WSGI: Es la interfaz de puerta de enlace de servidor web, se ha utilizado como estándar para el desarrollo de aplicaciones web de Python. WSGI es la especificación de una interfaz común entre servidores web y aplicaciones web.

Werkzeug: Es un kit de herramientas WSGI que implementa solicitudes, objetos de respuestas y funciones de utilidad. Esto permite construir un framework web sobre él. El marco Flask utiliza Werkzeg como una de sus bases.

Jinja2: Es un motor de plantillas popular para Python. Un sistema de plantillas web combina una plantilla como una fuente de datos específica para representar una página web dinámica. Esto le permite pasar variables de Python a plantillas HTML.

Dentro de lo que distingue a Flask es que pertenece a la categoría del micro framework. Estos son normalmente frameworks con poco o ninguna dependencia a bibliotecas externas. Este por su parte tiene tanto pros como contras.

P	Escalable.	**C**	No hay muchas herramientas.
R	Flexible.	**O**	Difícil de familiarizarse en aplicaciones grandes.
O		**N**	
S	Ligero.	**T**	Coste de mantenimiento.

	Amplia documentación.	**R** **A** **S**	

Tabla 9 Pros y Contras de Flask

Ruby on Rails

Ruby on Rails (*Rails*) es un framework de desarrollo de aplicaciones web de código abierto escrito en el lenguaje de programación Ruby. Este lenguaje fue creado por Yukihiro Matsumoto a mediados de la década de 1990. Al igual que C o Java, Ruby es un lenguaje de propósito general, pero mejor conocido por sus beneficios en el desarrollo web. Por otro lado, Rails es una biblioteca de software, que extiende el lenguaje Ruby.

Rails es utilizado para construir aplicaciones modernas de internet como: Twitter, Scribd, Hulu, Xing, Soundcloud, Basecamp y Github. Fue creado en 2003 por David Heinemeire Hansson y desde entonces ha sido extendido por el Rail score tema, más de 2.100 colaboradores y soportado por una extensa y activa comunidad (Rails, 2022).

La mayoría de los lenguajes de programación web como JavaScript, HTML, CSS y SQL no cubren tanto el Back-end como el Front-end. Sin embargo, podemos decir que Rails si lo hace, este abarca ambos extremos y permite a los desarrolladores construir una aplicación web complete. Basado en el lenguaje de programación Ruby, Rails permite tener todas las funciones necesarias para el desarrollo de aplicaciones web al alcance de la mano. Algunas de estas características son:

Arquitectura MVC: Ruby on Rails utilizo el patrón arquitectónico Model-View-Controller utilizado por muchos otros frameworks, este patrón separa el código de una aplicación web en tres partes interconectadas. El modelo es usado para mantener la relación entre el objeto y la base de datos, la vista son las plantillas

que se muestras al usuario y el controlador se usa para combinar el modelo y la vista.

Active record: El registro activo se introduce en Ruby on Rails. Es una poderosa biblioteca que permite al desarrollador diseñar las consultas interactivas de la base de datos.

Built-in Testing: Proporciona un conjunto de pruebas que se ejecutaran en su código. Ahorrando tiempo y esfuerzo.

Convention over configuration: Ruby on Rails evita los archivos de configuración para ahorrar convenciones, reflexión y extensiones dinámicas de tiempo de ejecución. La idea detrás de esto es asignar valor automáticamente sin la intervención del usuario.

PROS	Bibliotecas.	**CONTRAS**	Velocidad de ejecución.
	Calidad del código.		Falta de flexibilidad.
	Automatización de pruebas.		Velocidad de arranque.
	Amplia comunidad.		Poca documentación.

Tabla 10 Pros y Contras de Ruby on Rails

Symfony

Symfony es un framework PHP de código abierto y una colección de componentes PHP reutilizables lanzados por primera vez en 2011 por Fabien Potencier. Los componentes de Symfony son librerías desacopladas para aplicaciones PHP que a su vez conforman una potente plataforma Back-end utilizada para crear grandes aplicaciones (Symfony, 2022).

Además de estos componentes estándar, hay módulos adicionales que se pueden usar para extender el marco por separado. El hecho de que Symfony pueda ser utilizado como base de desarrollo para proyectos web de todo tipo se debe a su estructura modular. Cada módulo tiene su propia función, pero no depende de otros componentes, lo que permite al framework tener un alto grado de flexibilidad y capacidad de expansión.

Dentro de las características que tiene Symfony es que puede ser utilizado como un framework Full Stack si se requiere una amplia gama de funciones para el proyecto. Además, se puede crear un propio conjunto de librerías de Symfony o crear una versión ligera del framework. De esta manera, se pueden adaptar componentes a los requisitos precisos de la aplicación web, ya sea un proyecto empresarial complejo o un sitio web simple.

Algunos de los componentes estándar que se pueden elegir son:

- **Asset:** modulo para generar URL's y control de versiones de archivos de imagen, hojas de estilos CSS y JavaScript.
- **ClassLoader:** garantiza que sus propias clases PHP se carguen automáticamente.
- **Debug:** proporciona herramientas para depurar código PHP con el fin de localizar y clasificar errores.
- **DependencyInjection:** permite definir estándares para la creación de objetos que serán utilizados en el proyecto.
- **EventDispatcher:** componente elemental que controla la comunicación de módulos individuales en forma de eventos.
- **Form:** contiene herramientas que puede utilizar fácilmente para crear formularios HTML reutilizables.
- **Templating:** herramientas para crear un sistema de plantillas.
- **Translation:** modulo para interiorizar el proyecto.

- **Validator:** permite validar las clases creadas.
- **Yaml:** carga y guarda archivos .yml.

Symfony permite la implementación del patrón Model View Controller (MVC). Este patrón de arquitectura divide la aplicación en tres áreas. El concepto MVC proporciona un código claro y bien estructurado y con un alto grado de flexibilidad para diferentes componentes, que pueden intercambiarse y reutilizarse según el concepto, ya que no están vinculados a datos de entrada concretos.

Sin embargo, cabe mencionar que Symfony no pertenece a la línea proclamada de frameworks MVC y no contiene un componente de modelo integrado. Por lo que, al crear nuevos proyectos, estos tienen la libertad de elegir los componentes necesarios para la arquitectura y de acuerdo con los requisitos de este.

P R O S	Flexibilidad.	**C O N T R A S**	Mecanismo de seguridad complejo.
	Rapidez.		Dificultad con actualización de versiones.
	Fácil de usar.		Desarrollo costoso.
	Escalabilidad.		

Tabla 11 Pros y Contras de Symfony

Laravel

Laravel es un framework de aplicaciones web con sintaxis expresiva y elegante. Creemos que el desarrollo debe ser una experiencia agradable y creativa para ser verdaderamente satisfactorio. Laravel intenta eliminar el dolor del desarrollo al facilitar las tareas comunes utilizadas en la mayoría de los proyectos web (Laravel, 2022).

Este framework es uno de los mejores para el desarrollo del Back-end. Tiene una excelente construcción del lenguaje, la capacidad de adaptarse a grupos gigantes y la efectividad de su caja de herramientas avanzadas. Está basado en el lenguaje PHP, Laravel sigue la arquitectura de diseño MVC.

Laravel es uno de los frameworks Back-end más completos, ya que tiene herramientas para la implementación, el desarrollo, las pruebas, la automatización y más. Se podría argumentar que Laravel no solo es un framework, es todo un ecosistema para construir sitios rápidos y eficientes con PHP (Lekh, 2021).

El lanzamiento al público de Laravel fue en el año 2011 y desde entonces es uno de los frameworks de PHP más populares. Hay alrededor de 1,140,640 sitios web activos de Laravel. Laravel es un framework de desarrollo simple y elegante, debido a su sintaxis limpia y completa, que combina la arquitectura MVC para desarrollar aplicaciones web.

Con una sintaxis expresiva y elegante, Laravel permite a los desarrolladores web ser flexibles y creativos mientras cuidad los principales detalles de fondo, a diferencia de otros. Algunas empresas que utilizan este framework son Tarjetas MasterCard, Razorpay, Kmong, Bitpanda.

También podemos enlistar algunas de las características que tiene Laravel como lo son:

Artisan Console: es una de las mejores características de Laravel. Artisan, una herramienta de línea de comandos incorporada en el marco de Laravel ayuda a automatizar la mayoría de los tediosos procesos de programación repetitivos.

Librerías y modularidad: con su marco orientado a objetos, Laravel es uno de los mejores paquetes disponibles. Viene con una serie de librerías preinstaladas que son totalmente compatibles con su modularidad.

Eloquent ORM: Eloquent Object Relational Mapping (ORM) es una característica de Laravel que contiene una implementación simple de PHP Active Record. En lugar de escribir código SQL, los desarrolladores pueden usar la sintaxis PHP para escribir consultas de bases de datos.

Unit-Testing: el framework puede realizar una gran cantidad de pruebas para garantizar que los nuevos cambios de los programadores no rompan nada en la aplicación web inesperadamente.

Motor de plantillas: Laravel es bien conocido por sus plantillas ligeras, que se pueden usar para crear diseños impresionantes con siembra de contenido dinámico.

P R O S	Se puede utilizar nuevas características de PHP.	**C O N T R A S**	Soporte integrado insuficiente.
	Amplia documentación.		Actualizaciones problemáticas.
	Integración con servicios de correo.		Difícil de dominar.
	Enrutamiento inverso.		
	Gestión de colas.		

Tabla 12 Pros y Contras de Laravel

Codeigniter

Codeigniter es un potente framework PHP ampliamente confiable, para la creación de sitios web dinámicos con una huella pequeña. Creado para desarrolladores que necesitan un kit de herramientas simple y elegante para crear aplicaciones web con todas las funciones.

Es uno de los mejores marcos de desarrollo más rápido utilizado para crear aplicaciones web dinámicas y sitios web en PHP. Este framework se basa en sistema débilmente acoplados y hace uso del ampliamente popular MVC, es decir, el patrón Modelo, Vista, Controlador. Es un framework construido para desarrolladores que necesita hacer uso de un kit de herramientas simple y elegante para crear aplicaciones web funcionales.

Cabe recalcar que la naturaleza de Codeigniter es ligera ya que el sistema central requiere el uso de muy pocas librerías, lo que contrasta con mucho otros frameworks presentes en la actualidad, lo que requiere el uso de muchos más recursos. El uso de estas librerías se carga dinámicamente en el tiempo de ejecución al dar una solicitud particular, lo que hace que le sistema sea bastante rápido y ágil.

Al trabajar con Codeigniter permite centrarse en la parte creativa del proyecto y disminuir la cantidad de código que se utiliza para las configuraciones y tareas repetitivas de cada nuevo proyecto. El uso efectivo de la arquitectura MVC, donde hace uso del modelo y la vista junto con el controlador, siempre ha hecho las cosas más fáciles e interesantes (India, 2022).

Algunas de las características y funcionalidad de Codeigniter que lo diferencia de otros frameworks PHP de la actualidad son:

- **Huella diminuta:** A medida que el tamaño del código aumenta dentro de los 2MB, puede implementar y actualizar el código fuente de Codeigniter muy rápidamente.
- **Super rápido:** Permite la carga de archivos en menos de 50ms en comparación con otros frameworks basados en la misma arquitectura.
- **Arquitectura MVC:** Además de utilizar la arquitectura Modelo-Vista-Controlador, Codeigniter utiliza un diseño de máquina de estado. En las aplicaciones, el modelo separa los datos, la logia de negocio y la vista.
- **Componentes integrados:** Los desarrolladores pueden enviar fácilmente un correo electrónico, administrar bases de datos, administrar sesiones y más con los componentes específicos de la aplicación.

P R O S	Seguro.	**C O N T R A S**	Pocas librerías.
	Simple.		No hace hincapié en el mantenimiento de código.
	Flexible.		
	Rápido.		

Tabla 13 Pros y Contras de Codeigniter

Spring

Java Spring Framework (Spring Framework) es un framework popular, de código abierto y de nivel empresarial para crear aplicaciones independientes de nivel de producción que se ejecute en la máquina virtual Java (JVM) (IBM Cloud Education, 2020).

Java Spring Boot (Spring Boot) es una herramienta que hace que el desarrollo de aplicaciones web y microservicios con Spring Framework sea más rápido y fácil a través de tres capacidades principales:

- Configuración automática
- Un enfoque obstinado de la configuración
- La capacidad de crear aplicaciones independientes

Estas características trabajan juntas para proporcionar herramientas que permitan configurar una aplicación basada en Spring con una configuración y configuraciones mínimas. Spring Framework proporciona un modelo integral de programación y configuración para aplicaciones empresariales modernas basadas en Java, en cualquier tipo de plataforma de implementación (Spring, 2022).

Un elemento clave se Spring es el soporte de infraestructura a nivel de aplicación: Spring se centra en la “plomería” de las aplicaciones empresariales para que los equipos puedan centrarse en la lógica empresarial a nivel de aplicación, sin vínculos innecesarios con entornos de implementación específicos.

Cabe mencionar que Spring Boot es una framework adecuado en el que los desarrolladores tienen la intención de que desarrollen API RESTful. Y algunas otras características que tiene este framework podemos mencionar son:

- **Aplicación independiente:** puede simplemente compilar el JAR de la aplicación y ejecutar la aplicación sin necesidad de personalizar la implementación.
- **Servidores integrados:** viene con servidores de aplicaciones como Tomcat, Jetty y Undertow prediseñados que no requieren más instalaciones para su uso.
- **Configurable automáticamente:** Spring y otros marcos de 3rd party se configuran automáticamente.
- **Características similares a producción:** comprobaciones de estado, métricas y configuraciones externalizadas.

P R O S	Fácil desarrollo.	**C O N T R A S**	Falta de control.
	Aplicaciones independientes.		Curva amplia de aprendizaje.
	Tomcat, Jetty o Undertow incluidos.		
	Sin configuración XML.		
	Menos código fuente.		No apto para proyectos a gran escala.
	Configuración y gestión sencillas.		
	Comunidad.		

Tabla 14 Pros y Contras de Spring

JavaServer Faces

Es un framework de interfaz de usuario basado en componentes del lado del servidor para crear aplicaciones web basadas en tecnología Java. Proporciona un modelo de programación bien definido y consta de librerías de etiquetas y API enriquecidas (Oracle, Oracle, 2013).

Es una de las tecnologías Java más avanzadas para el desarrollo de interfaces de usuario. JavaServer Faces (JSF) se puede utilizar para crear cualquier tipo de interfaz de usuario, pero principalmente está dirigido al desarrollo de interfaces web. La tecnología de JSF proporciona una separación perfecta de los componentes de acuerdo con el principio de Modelo, Vista, Controlador (MVC).

Con respecto al desarrollo web, la tecnología JSF se integra bien con JSP, Servlet, HTML, JavaScript y otras tecnologías. Las páginas web creadas con este framework utilizan etiquetas JSF conectadas a las páginas JSP. Las etiquetas se utilizan para agregar componentes en las páginas web y conectar componentes con objetos en el servidor. También contiene controladores de etiquetas que implementen la etiqueta del componente (SAP, s.f.).

Algunas de las fortalezas clave de JSF son:

- **Basado en estándares:** crea un entorno no propietario del framework, por lo que garantiza a los desarrolladores que sus aplicaciones se puedan ejecutar en múltiples implementaciones de JSF, además crea las condiciones necesarias para añadir futuras mejoras.

- **Extensible:** cada proveedor de JSF suministra un conjunto básico de componentes de interfaz de usuario, pero se pueden desarrollar fácilmente componentes adicionales en caso de no satisfacer los requisitos de la aplicación.

- **Personalizable:** el tiempo de ejecución es altamente flexible. Los aspectos del tiempo de ejecución, ciclo de vida, mecanismos de generación de respuesta y los enfoques de ahorro de estado, se pueden adaptar fácilmente a los detalles de una aplicación.

ASP.NET

En 2002, Microsoft introdujo .NET como un marco de software. Se abrevia como .NET, El framework .NET es una plataforma de desarrollo compuesta por herramientas, lenguajes de programación y librerías para crear muchos tipos diferentes de aplicaciones. ASP.NET amplia la plataforma de desarrollo .NET con herramientas y librerías específicas para crear aplicaciones web (Microsoft, 2022).

ASP.NET es un framework web gratuito para crear excelentes sitios y aplicaciones web utilizando HTML, CSS y JavaScript. También se puede utilizar para crear APIs y utilizar tecnologías en tiempo real como Web Sockets. Cuando se usa ASP.NET el código Back-end, como la lógica empresarial y el acceso de datos, se escribe con C#, F# o Visual Basic.

Dado que ASP.NET extiende de .NET, se puede usar el gran ecosistema de paquetes y librerías disponibles para todos los desarrolladores de .NET. También se pueden crear propias librerías que se comparten entre cualquier aplicación escrita en la plataforma .NET (Microsoft, 2022).

A continuación, se enlistan algunas características que ASP.NET agrega a la plataforma de .NET:

- **Framework base para procesar solicitudes web en C# o F#.**
- **Sintaxis de plantillas de páginas web**, conocida como Razor, para crear páginas web dinámicas utilizando C#.
- **Librerías para patrones web comunes**, como Modelo Vista Controlador (*MVC*).
- **Sistema de autenticación** que incluye librerías, una base de datos y páginas de plantillas para manejar inicios de sesión, incluida la autenticación multifactor y la autenticación externa con Google, Twitter y más.
- **Extensiones de editor** para proporcionar resaltado de sintaxis, finalización de código y otras funcionalidades específicas para el desarrollo de páginas web.

ASP.NET ofrece tres framework para crear aplicaciones web: Web Forms, ASP.NET MVC y ASP.NET Web Pages. Los tres frameworks son estables y maduros, y se pueden crear excelentes aplicaciones web con cualquiera de ellos. No importa que framework se elija, se tendrán todos los beneficios y características de ASP.NET en todas partes.

Cada framework apunta a un estilo de desarrollo diferente. El que se elija uno de otro depende del tipo de aplicación que se creara y el enfoque de desarrollo con el que se trabajara, a continuación, se muestra una descripción general de cado unos de los frameworks que ofrece ASP.NET y algunas ideas sobre como poder elegir entre ellos.

Framework	Si se tiene experiencia en:	Estilo de desarrollo
Web Forms	Win Forms	Desarrollo rápido utilizando una gran variedad de librerías de controles que encapsulan el marcado HTML.
	WPF	
	.NET	
MVC	Ruby on Rails	Control total sobre el marcado HTML, código y marcado separados, pruebas fáciles e escribir. La mejor opción para aplicaciones móviles y de una solo página.
	.NET	
Web Pages	ASP clásico	Marcado HTML y el código juntos en el mismo archivo.
	PHP	

Tabla 15 Frameworks que ofrece ASP .NET

Web Forms: Permite crear sitios web dinámicos mediante un modelo familiar de arrastras y colocar controlado por eventos. Una superficie de diseño y cientos de controles y componentes permiten crear rápidamente sitios sofisticados y potentes basados en la interfaz de usuario con acceso a datos (Anderson, 2022).

MVC: Brinda una forma poderosa y basada en patrones de crear sitios web dinámicos que permite una separación limpia de las preocupaciones y brinda control total sobre el marcado para un desarrollo agradable y ágil. Incluye muchas características que permiten un desarrollo con TDD para crear aplicaciones sofisticadas que utilizan los últimos estándares web (Anderson, 2022).

Web Pages: La sintaxis de Razor proporcionan una forma rápida, accesible y ligera de combinar el código del servidor con HTML para crear contenido web dinámico. Conectarse a bases de datos, agregar videos, enlaces a sitios de redes sociales e incluye más características que permite la creación de sitios web con los últimos estándares web (Anderson, 2022).

P R O S	Soporte técnico de Microsoft.	**C O N T R A S**	Opción costosa.
	Mantenimiento sencillo.		Amplia curva de aprendizaje.
	Compatibilidad con APIs.		Herramientas de desarrollo basadas en Windows.
	Escalado y Dockerización.		Documentación no tan buena.
	Código abierto.		
	Rendimiento.		

Tabla 16 Pros y Contras de ASP .NET

Express

Es un framework para Node.js, el entorno de tiempo de ejecución JavaScript. Express es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto solido de características para las aplicaciones web y móviles (Express, 2022).

Muchas aplicaciones web que usan Node.js también usan Express debido a lo potente pero liviano que es. En esencia, Express es un framework de aplicaciones web que proporciona una capa delgada de fundamentos de aplicaciones web y funciona bien con middleware. Con miles de métodos de programa de utilidad HTTP y middleware a su disposición, la creación de una API solida es rápida y sencilla.

A pesar de que Express es en sí mismo bastante minimalista, los desarrolladores han creado paquetes de middleware compatibles para abordar

casi cualquier problema de desarrollo web, Hay librerías para trabajar con cookies, sesiones, inicios de sesión de usuario, parámetros URL, datos POST cabeceras de seguridad y mucho más (MDN Web Docs, 2022).

Hay muchos sitios construidos con Express, incluidos MuleSoft, Accenture, Uber, Cozy, Apiary, MySpace y más. Por supuesto, esto no cuenta los innumerables sitios de alto perfil que usan Node.js como su tecnología subyacente. Dentro de las principales características que encontramos en Express son:

Manejo de Solicitudes: Express posee métodos para especificar que función ha de ser llamada dependiendo del vero HTTP usado en la petición (GET, POST, SET, PUT, etc.) y la estructura de la URL. También tiene los métodos para especificar que plantilla ("view") o gestor de visualización utilizar, donde están guardadas las plantillas de HTML que han de usarse y como generar la visualización adecuada para cada caso (MDN Web Docs, 2022).

Middleware: El middleware se puede aplicar tanto a nivel de aplicación como de ruta, así como encadenarse entre sí. Se puede insertar casi cualquier middleware compatible que se desee en la cadena de manejo de solicitudes (Codecademy, 2022).

P R O S	Fácil de aprender.	**C O N T R A S**	No es tan seguro.
	Documentación.		Problemas con devoluciones de llamada.
	Soporte de middlewares.		
	Velocidad y escalabilidad.		
	Gran comunidad.		

Tabla 17 Pros y Contras de Express

Conclusiones

La constante evolución de la tecnología ha traído consigo varios cambios, si lo vemos desde el punto de vista de los entornos web, este ha revolucionado la forma en que nos relacionamos, comunicamos y aprendemos. Desde la creación de la web hasta nuestros días se han desarrollado nuevas herramientas y creado nuevos lenguajes que han permitido programar la web, esto ha permitido cambiar la forma en que las personas consulta la información desde internet, desde la llegada de los teléfonos inteligentes trajo consigo un cambio drastico en como se creaban las paginas y sitios web, ya que ahora estos no solo serian consultados desde una laptos o PC, si no que ahora se podría consultar desde un dispositivo pequeño, portátil y que hoy en día es tan común para todos.

Esto permitió crear sitios web adaptables, los cuales dependiendo el dispositivo desde donde eran consultados, cambiaban su estructura y la forma de como se mostraban los elementos en pantalla, con la popularización de los teléfonos inteligentes y tabletas, permitió que mas dispositivos se comunicaran entre si por medio de internet, esto permitió mejorar la forma en como estos dispositivos se comunicaban y transferían información mediante internet, trayendo consigo nuevos conceptos como la Nube, la cual permite subir, almacenar y compartir información a cualquier dispositivo que tenga acceso a internet.

Con estas mejoras se empezó a revolucionar la web ya que ahora un sitio web no solo servía y transmitía información a los dispositivos, si no que con la evolución de la web también acarreo mejoras y actualizaciones en los lenguajes que se utilizan para programar esta, implementado nuevas APIs como para el manejos de archivos multimedia, comunicación en tiempo real con el servidor y geolocalización por mencionar algunas.

Estas actualizaciones permitieron la evolución de como se creaba la web, ya que ahora los lenguajes permitían crear sitios mucho mas potentes en cuanto a características así como de visualización de contenido. Cosas y características que solo se podían hacer en las aplicaciones de escritorio, en aquellos días tal vez no imaginaríamos que podíamos correr Microsoft Office desde una web gracias a la potencia de la nube, o tener un centro comercial repleto de productos simplemente accediendo a la web desde un dispositivo portátil y desde la palma de la mano.

Pero, así como la tecnología avanzo, igual cambio la forma en como se programa esta, ya que hoy en día si se quiere hacer un producto o sitio web que realmente impacte a los usuarios, se tiene que hacer una serie de estudios y estadísticas para que esto funcione ya que no solamente basta como hace unos años atrás teniendo a una sola persona programando todo el sitio web comúnmente llamados web master, si no que ahora se necesitan personas especializadas en las diferentes tecnologías que ocupa la web, desde servidores web, lenguajes Back-end, lenguajes Front-end, ciencias de datos e incluso inteligencia artificial, conceptos que antes solamente escuchábamos o veíamos en películas de ciencia ficción hoy en día se han hecho realidad desde relojes inteligentes que permiten hacer llamadas, monitorear el estado de salud y geolocalización en tiempo real, hasta inteligencias artificiales que crean imágenes a partir de descripciones que el usuario les da, y hasta permite crear aplicaciones completas simplemente especificando algunas descripciones de lo que quieres hacer.

Todo esto se tiene que mantener, actualizar y escalar en base al crecimiento de usuarios, así mismo se tuvo que crear nuevos paradigmas de programación y soluciones web que permitieran la interacción desde dispositivos móviles, tabletas y computadoras. Estas web ya no solo se encargarían de enviar y mostrar información estática al cliente, si no que ahora estos datos podrían

actualizarse, crearse y hasta eliminarse, estas operaciones repetitivas dentro de los sitios web trajo consigo una solución, estos son los frameworks de desarrollo, los cuales por su definición es un conjunto de herramientas y funciones predefinidas listas para usarse, desde una conexión a base de datos, hasta creación de canales de comunicación en tiempo real con el cliente, sin duda hoy en día es impensable el crear un sitio como un e-commerce o aplicación web como lo son Word, Excel, PowerPoint, Figma e incluso algunas aplicaciones de la paquetería de Adobe, que sean creados por una sola persona y desde cero, al final el resultado no sería tan optimo además de que requeriría un tiempo prolongado de desarrollo, incluso teniendo a un desarrollador o más trabajando, para esto los frameworks ayudan a mejorar el desarrollo web, permitiendo tener esa estructura base de la arquitectura que tendrá nuestro sitio, como por ejemplo tener una capa de comunicación y exponiendo una API a internet para poder consultar, actualizar y eliminar información de un sitio.

Hoy en día el uso de un framework en el desarrollo web moderno es imprescindible ya que permite ahorra mucho tiempo de desarrollo, además de que ofrece diferentes capas de seguridad y se tienen reglas predefinidas que permite tener una base para el desarrollo de nuevos elementos, rutas o servicios que tendrá el sitio. No importa si se esta trabajando en el Back-end o en el Front-end el uso de un framework de desarrollo permitirá un desarrollo más ágil, además de que permitirá que tu sitio o aplicación sea escalable en un futuro.

Bibliografía

Anderson, R. (12 de 5 de 2022). *Docs Microsoft*. Obtenido de https://docs.microsoft.com/en-us/aspnet/overview

Basics, P. (2021). *Python Basics*. Obtenido de https://pythonbasics.org/what-is-flask-python/

Brewster, C. (2022). *Trio.dev*. Obtenido de https://www.trio.dev/blog/why-use-vue-js

Carlos Coronel, S. M. (2019). *DATABASE SYSTEMS DESIGN, IMPLEMENTATION & MANAGEMENT.* CENGAGE.

Codecademy. (2022). *codecademy*. Obtenido de https://www.codecademy.com/article/what-is-express-js

Consortium, W. W. (21 de Julio de 2020). *W3C*. Obtenido de https://www.w3.org/TR/mediaqueries-4/

Devs, Q. (1 de Diciembre de 2018). *Quality devs*. Obtenido de https://www.qualitydevs.com/2018/12/12/que-es-un-desarrollador-frontend/

Django Software Foundation. (2022). *Django project*. Obtenido de https://www.djangoproject.com/

Duckett, J. (2011). *HTML & CSS.* Indianapolis: John Wiley & Sons, Inc.

Edpresso. (2022). *Educative.io*. Obtenido de https://www.educative.io/edpresso/what-is-svelte

Express. (2022). *Express*. Obtenido de https://expressjs.com/es/

Foundation, M. (2022). *Mariadb*. Obtenido de https://mariadb.org/

Gattinoni, A. (2010). *Python BaseHTTPServer: un servidor HTTP en unas pocas líneas*. Recuperado el 2 de 11 de 2021, de http://tail-f.com.ar/programacion/python/python-basehttpserver-un-servidor-http-en-unas-pocas-lineas.html/comment-page-1

Huet, P. (21 de 01 de 2022). *OpenWebinars*. Obtenido de https://openwebinars.net/blog/que-es-tailwind-css-y-por-que-deberias-usarlo/

IBM. (2010). *IBM*. Obtenido de https://www.ibm.com/docs/en/zos-basic-skills?topic=zos-what-is-database-management-system

IBM. (2010). *IBM*. Obtenido de https://www.ibm.com/docs/en/zos-basic-skills?topic=zos-what-is-database-management-system

IBM Cloud Education. (25 de 3 de 2020). *IBM*. Obtenido de https://www.ibm.com/cloud/learn/java-spring-boot

India, I. S. (13 de 09 de 2022). *Linkedin*. Obtenido de https://www.linkedin.com/pulse/what-codeigniter-features-framework-how-does-work-it-services-india?trk=pulse-article_more-articles_related-content-card

JetBrains. (s.f.). *JetBrains*. Obtenido de https://www.jetbrains.com/lp/devecosystem-2020/python/

Land, S. E. (2022). *Search Engine Land*. Obtenido de https://searchengineland.com/guide/what-is-seo

Laravel. (2022). *Laravel*. Obtenido de https://laravel.com/

Lekh, A. (21 de 1 de 2021). *impressit*. Obtenido de https://impressit.io/blog/best-backend-frameworks

Maldeadora, N. (1 de Enero de 2019). *Platzi*. Obtenido de https://platzi.com/blog/que-es-frontend-y-backend/

MDN. (24 de Marzo de 2021). *MDN Web Docs*. Obtenido de https://developer.mozilla.org/es/docs/Web/HTTP/Overview

MDN Web Docs. (2022). *Developer Mozilla*. Obtenido de https://developer.mozilla.org/es/docs/Learn/Server-side/Express_Nodejs/Introduction

Microsoft. (2022). Obtenido de https://dotnet.microsoft.com/en-us/apps/aspnet

Microsoft SQL Server. (s.f.). Recuperado el 3 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/Microsoft_SQL_Server

Mongodb. (s.f.). Recuperado el 3 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/MongoDB

Mullins, C. s. (07 de 2022). *TechTarget*. Obtenido de https://www.techtarget.com/searchdatamanagement/definition/database-management-system

MySQL. (s.f.). Recuperado el 3 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/MySQL

Nikita, R. (2015). *Responsive Web Design.*

Node. (2022). *Node JS*. Obtenido de https://nodejs.org/es/about/

Northwood, C. (2018). *The Full Stack Developer.* Manchester, UK: Apress.

Oracle. (2013). *Oracle*. Obtenido de https://docs.oracle.com/javaee/6/tutorial/doc/bnaph.html

Oracle. (2021). *Oracle*. Obtenido de https://www.oracle.com/mx/java/

ORACLE. (21 de 12 de 2021). *ORACLE*. Obtenido de https://www.oracle.com/database/what-is-database/

Oracle Database. (s.f.). Recuperado el 3 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/Oracle_Database

Patel, J. (9 de Noviembre de 2021). *Monocubed*. Obtenido de https://www.monocubed.com/blog/most-popular-web-frameworks/

Patel, T. (26 de Mayo de 2020). *ThirdRock Techkno*. Obtenido de https://www.thirdrocktechkno.com/blog/does-your-web-application-need-a-front-end-framework/

PHP. (s.f.). Recuperado el 5 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/PHP

PostgreSQL. (s.f.). Recuperado el 3 de 10 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/PostgreSQL

Quesada, S. (21 de Noviembre de 2013). *Maestros del Web*. Recuperado el 04 de Septiembre de 2016, de http://www.maestrosdelweb.com/que-es-responsive-web-design/

Rails, R. o. (2022). *Ruby on Rails*. Obtenido de https://rubyonrails.org.es/

Robbins, J. N. (2018). *Learning Web Design.* Gravenstein Highway North: O'Reilly.

Rodríguez, X. (30 de Mayo de 2019). *OpenWebinars*. Obtenido de https://openwebinars.net/blog/tipos-servidores-web/

SAP. (s.f.). *SAP*. Obtenido de https://help.sap.com/doc/saphelp_nw73ehp1/7.31.19/en-US/4a/c2f76922a922b0e10000000a42189b/content.htm?no_cache=true

Shah, K. (28 de 07 de 2021). *ThirdRock Techkno*. Obtenido de https://www.thirdrocktechkno.com/blog/pros-and-cons-of-node-js-web-app-development/

Sirotka, A. (18 de 02 de 2022). *Flatlogic*. Obtenido de https://flatlogic.com/blog/what-is-react/

Sirotka, A. (02 de 03 de 2022). *Flatlogic*. Obtenido de https://flatlogic.com/blog/what-is-angular/

Souza, I. d. (14 de Junio de 2019). *rockcontent*. Obtenido de https://rockcontent.com/es/blog/que-es-un-servidor/

spring. (2022). *spring*. Obtenido de https://spring.io/projects/spring-framework

Spring. (2022). *spring*. Obtenido de https://spring.io/projects/spring-framework

SQL. (s.f.). Recuperado el 8 de 7 de 2021, de Wikipedia, la enciclopedia libre: http://es.wikipedia.org/wiki/SQL

StrongLoop. (2022). *Express*. Obtenido de https://expressjs.com/es/resources/glossary.html

Symfony. (2022). *Symfony*. Obtenido de https://symfony.com/what-is-symfony

Team, A. (28 de 02 de 2022). *Angular.io*. Obtenido de https://angular.io/

Team, B. (2022). *Bootstrap*. Obtenido de https://getbootstrap.com/

Team, R. (29 de 03 de 2022). *Reactjs.org*. Obtenido de https://es.reactjs.org/

Team, S. (2022). *Semantic UI*. Obtenido de https://semantic-ui.com/

Terry Ann Felke-Morris, E. (2016). *Basics of Web Design HTML5 & CSS.* Hoboken: Pearson.

Vega, A. A. (2013). Responsive Web Design: Interfaces Web Adaptables a dispositivo Empleando HTML5 y CSS3. Madrid, España.

You, E. (2022). *Vuejs.org*. Obtenido de https://vuejs.org/

Young, J. F. (2015). *Front-End Fundamentals.* Leanpub.

Zapata, J. F. (13 de Junio de 2018). *DevCode.* Obtenido de https://devcode.la/blog/que-es-responsive-web-design/