INSTITUTO TECNOLÓGICO DE OAXACA

INGENIERÍA EN SISTEMAS COMPUTACIONALES

MATERIA: INGENIERÍA DE SOFTWARE

ACTIVIDAD 1 T2: “DISEÑO ARQUITECTÓNICO “

ALUMNOS:

DIEGO REVILLA JOSÉ ANTONIO

GARCÍA GARCÍA JOSÉ ÁNGEL

ROMÁN HERNÁNDEZ ESTEBAN DANIEL

MANZANO SÁNCHEZ ISRAEL

GRUPO: 6SB

HORA: 09:00 – 10:00

Índice

Introducción

En el presente documento se decide el diseño arquitectónico e identificación del sistema web HuxGym para la empresa Gym Huatulco Fitness Center en la asignatura de ingeniería de software. Este documento contiene todos los diagramas propuestos, definiciones y protocolos utilizados en el sistema web a desarrollar. El objetivo primordial del documento es poder ya establecer una arquitectura para el sistema a desarrollar y el tener una perspectiva de la implementación de dicha arquitectura.

En la sección de arquitectura podemos encontrar primeramente nuestra identificación del tipo de sistema que vamos a desarrollar, basándonos en las características de este sistema web y también el porqué de la elección del diseño arquitectónico, respaldando los objetivos, proponiendo el plan y las practicas recomendadas que se debe seguir parar diseñar y estructurar el sistema web propuesto, por lo que se puede encontrar una explicación detallada respecto a la arquitectura seleccionada, incluyendo sus componentes, ventajas y desventajas, posteriormente se explica el diseño de esa arquitectura seleccionada, para finalmente explicar nuestros motivos y razones por la cual seleccionamos dicho diseño de tal arquitectura para nuestro sistema HuxGym.

Además que una vez justificada la arquitectura seleccionada se presenta el diagrama de la implementación en donde aparecen todas las tecnologías y servicios que emplearemos para el desarrollo del sistema HuxGym, la idea de esto es tener una mejor visión de lo que vamos a desarrollar, así mismo, ese diagrama es explicado para comprender como se da la implementación con las tecnologías mencionadas y también se da una breve descripción de cada tecnología a emplear, esto para tener un contexto de lo que detalla el diagrama.

Por lo que al finalizar este documento, se tiene claro la arquitectura a emplear, el diseño de la misma y la proyección de cómo sería con nuestras tecnologías, facilitándonos una perspectiva general del funcionamiento de nuestro sistema web.

Arquitectura

Tipo de Sistema

Después de investigar llegamos a la conclusión de que nuestro sistema web HuxGym es un sistema de información basado en transacciones ya que un sistema de información basado en transacciones se caracteriza por recolectar, almacenar, modificar y recuperar toda la información generada por las transacciones producidas y además estos son sistemas basados en la web por lo tanto su acceso es mediante un navegador web. Una transacción se puede decir que es algún evento que genera o modifica los datos que se encuentran almacenados en el sistema, como por ejemplo compras, ventas y pagos. Con esta información nos dimos cuenta que nuestro sistema web HuxGym cuenta con esas mismas características de funcionamiento ya que el acceso a este será por medio de un navegador web y este sistema recolectará información sobre los clientes, productos, membresías, ventas, compras, estos datos los almacenara en una base de datos, y el personal autorizado tendrá la posibilidad de modificar esa información como por ejemplo, cambiar el tipo de membresía de un cliente, eliminar un cliente, registrar una venta o ver las estadísticas de compras y ventas de un periodo de tiempo. Un ejemplo de transacción en nuestro sistema puede ser el cambio de membresía de un cliente, ya que con esto estaríamos modificando los datos almacenados en nuestro sistema. Otra de las razones por la que suponemos que nuestro sistema HuxGym es un sistema de información basado en transacciones es que este va dirigido especialmente al área de ventas y administración lo cual es una de las características principales de este.

Las ventajas que poseen estos sistemas son:

- Permiten manejar grandes volúmenes de información de manera eficaz y eficiente en tiempo real.
- Agilizan y automatizan los procesos realizados por el negocio. (Por ejemplo, al realizar una venta, el sistema se realizará todas las operaciones aritméticas lo cual ahorrará tiempo).

- Transforman tareas complejas para los seres humanos en actividades más sencillas (Como por ejemplo buscar los datos de un empleado).

Y por otra parte, las desventajas

- Una caída del sistema puede provocar parálisis en el negocio.
- Cualquier alteración invalida en la información puede provocar situaciones caóticas llevando a la quiebra o destrucción del negocio.

Tipo de Arquitectura

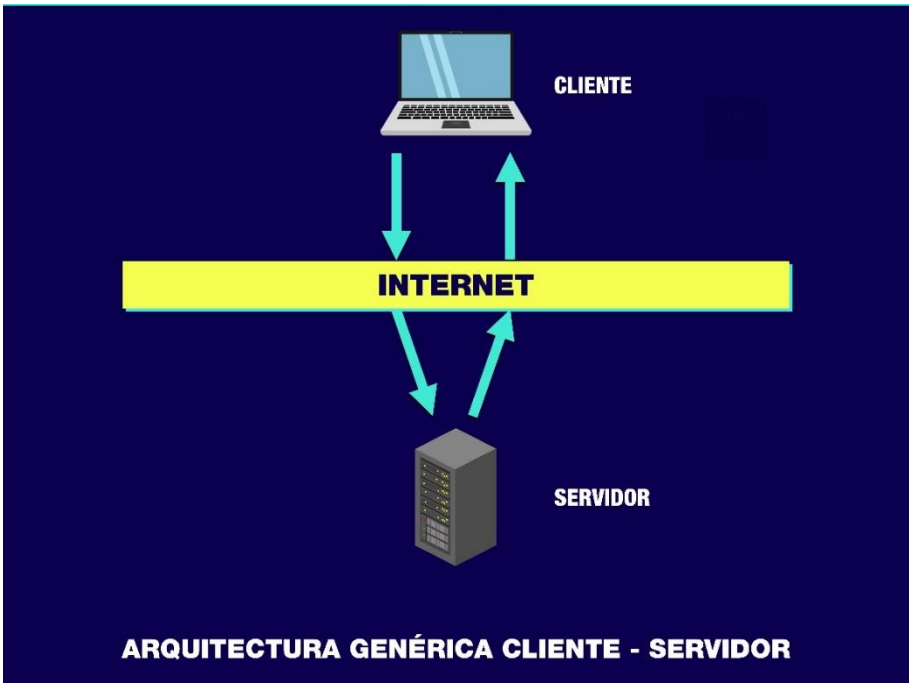
Una vez identificado nuestro tipo de sistema que vamos a desarrollar, el cual corresponde a un sistema de información basado en transacciones, entonces podemos proceder a definir la arquitectura que empleará dicho sistema. Pero primeramente vamos a tener en claro la definición de lo que es una arquitectura y arquitectura de software, esto con la idea de entender a que hace referencia y la importancia de la misma en el desarrollo de nuestro sistema web.

Cuando se define a el término arquitectura se suele pensar en la arquitectura de una construcción, es decir, se considera la forma general de una estructura física, pero va más allá de eso, ya que es la manera en la que los distintos componentes del edificio se integran para formar un todo, es decir, la arquitectura es el arte de pensar, diseñar y realizar construcciones de diferentes edificios. Ahora bien, podemos aplicar este concepto desde el área de software, por lo que se puede definir a la arquitectura del software o sistema como la estructura o estructuras del sistema, lo que comprende a todos los componentes del software, propiedades externas y las relaciones entre ellos. También es importante recalcar que la arquitectura de software es diferente del diseño de software, ya que un diseño de software es una instancia de una arquitectura.

La arquitectura que elegimos para nuestro sistema web es la cliente-servidor con un diseño de tres capas, esto debido a que está separado por el servidor web, el servidor de aplicación y el servidor de la base de datos.

Arquitectura Cliente-Servidor

La arquitectura cliente-servidor es una arquitectura en la que el cliente envía un mensaje solicitando un determinado servicio a un servidor, es decir, el cliente hace una petición al servidor, y este envía uno o varios mensajes con la respuesta al cliente, esto quiere decir, que el servidor provee servicios. Por lo tanto, esta arquitectura permite que se involucren uno o más clientes solicitando uno o más servicios. En esta arquitectura todos los servicios y solicitudes se distribuyen por la red. Su funcionalidad es como un sistema informático distribuido porque en el que todos los componentes están realizando sus tareas de forma independiente entre sí.



El cliente es la aplicación o programa que permite al usuario realizar solicitudes y enviarlos al servidor. Además que el cliente es el que maneja todas las funciones relacionadas con la manipulación de datos, y esto puede ser a través de una interfaz gráfica de usuario.

Algunas de las funciones generales que lleva acabo el cliente son:

- Administrar la interfaz de usuario.
- Generar requerimientos de la base de datos.

- Enviar y recibir información al servidor.
- Formatear resultados enviados por el servidor.

En la parte del cliente se encuentra el Frontend, que es la parte de una aplicación que interactúa con los usuarios, y también es conocida como el lado del cliente.

El servidor es un programa especializado que se encarga de atender las peticiones de múltiples clientes solicitando algún recurso administrador por él, además que se encarga de manejar todas las funciones relacionadas con las reglas del negocio y los recursos de los datos. Las funciones generales que puede realizar son:

- Aceptar y procesa las transacciones de bases de datos que hacen los clientes.
- Formatear datos para transmitirlos a los clientes.
- Proceso solicitudes de acceso a los datos.
- Ejecutar la lógica de la aplicación y realizar las respectivas validaciones.
- Aplicación de las reglas de negocio mediante validaciones o formateo de datos.

En la parte del servidor se encuentra el Backend, que es el interior de las aplicaciones que viven en el servidor y también es conocida como el lado del servidor.

A parte del cliente y servidor como componentes principales de esta arquitectura, también existen otros elementos que aporta a la creación de la misma, estos son:

- Red: Una red es un conjunto de clientes, servidores y base de datos unidos de una manera física o no física en el que existen protocolos de transmisión de información establecidos.
- Protocolo: Un protocolo es un conjunto de normas o reglas y pasos establecidos de manera clara y concreta sobre el flujo de información en una red estructurada.
- Servicios: Un servicio es un conjunto de información que busca responder las necesidades de un cliente.

- Base de datos: Son bancos de información ordenada, categorizada y clasificada que forman parte de la red, son sitios de almacenaje para la utilización de los servidores y también directamente de los clientes.

No está de más mencionar las ventajas y desventajas que existen de esta arquitectura, para posteriormente explicar porque es la que más funcionaría en nuestro sistema HuxGym.

Las ventajas que tendremos al implementar estar arquitectura son:

- La principal ventaja de cliente-servidor es el control centralizado con el que está integrada. Toda la información necesaria se coloca en un solo lugar y además se manipulan para usuarios designados (clientes) a través de un acceso autorizado.
- Es de alta accesibilidad, ya que cada cliente tiene la oportunidad de acceder a la información a través de donde desee, ya sea una interfaz de escritorio, una terminal, etc.
- Son altamente escalables, siempre que el administrador lo necesite puede incrementar el número de recursos como clientes y servidores, además de poder incorporar otras tecnologías.
- Permite que las aplicaciones construidas a partir de esta arquitectura puedan ser independientemente de la plataforma de hardware o los antecedentes técnicos del software de parte del cliente.
- Es fácil reemplazar, reparar, actualizar y reubicar un servidor mientras el cliente no se ve afectado. Este cambio inconsciente se llama como encapsulación.
- Los servidores tienen un mejor control de acceso y recursos para garantizar que solo los clientes autorizados puedan acceder o manipular los datos y que las actualizaciones del servidor se administren de manera eficaz.

Las desventajas que tendremos al implementarla:

- Cuando hay frecuentes solicitudes de clientes simultáneas, el servidor se sobrecarga gravemente, lo que genera congestión de tráfico.

- Dado que está centralizado, si falla un servidor crítico, las solicitudes de los clientes no se cumplen. Por tanto, el cliente-servidor carece de la solidez de una buena red.
- Cuando se implementen los servidores, funcionará sin parar. Lo que significa que se le debe prestar la debida atención.

Una vez detallada la arquitectura cliente-servidor podemos definir ahora el diseño que se empleará para nuestro sistema a partir de dicha arquitectura.

Diseño de Cliente-Servidor

El diseño como ya se mencionó, es el correspondiente al de tres niveles, en los que cada nivel tiene un funcionamiento respectivo, pero se parte de la misma idea de tener un cliente y un servidor, solo que ahora el servidor se divide en dos servicios a parte, para brindar su funcionamiento y así se forman los tres niveles, los cuales son:

1. Nivel de Presentación

   Este nivel corresponde al todo lo relacionado con la presentación e interacción de parte del cliente, incluye al navegador web del cliente y al servidor web, donde el navegador web es el que permite al cliente acceder a Internet para realizar las peticiones y el servidor web es el encargado de formatear dichas peticiones para enviarlas al nivel de aplicación. Este nivel es básicamente la Interfaz de Usuario.

2. Nivel de Aplicación

   En este nivel se implementa la lógica de la aplicación, la manipulación de los datos y también se aplican las reglas del negocio. Principalmente su función es procesar la solicitud enviada por el nivel de presentación y aplicar la lógica necesaria para resolver dicha petición y retornar una respuesta a ello. No resuelve por si sola la petición por lo que hace uso del nivel de datos para complementar su respuesta.

3. Nivel de Datos

   En este nivel se encuentra a los datos que gestiona la aplicación, dichos datos pueden ser información en una base de datos o en documentos, también se puede encontrar a servicios que tienen como función realizar la gestión de los datos y ejecutar las transacciones solicitadas por el nivel de aplicación.

Explicación de la selección de la arquitectura Cliente - Servidor

Como hemos observado, la arquitectura cliente-servidor tiene ventajas que nuestro cliente busca en su sistema pero uno de los factores más relevantes del porque vamos utilizar esta arquitectura es porque se implementa la comunicación mediante protocolos de Internet, y al ser nuestro sistema un sistema web, este está en el mismo entorno. Ya que las aplicaciones o sistemas web siempre realizan el proceso de recoger datos del usuario, enviar los datos al servidor donde este ejecuta un programa o servicio de acuerdo a la solicitud y ocupa esos datos para obtener más datos o datos específicos, formatearlos y enviarlos como respuesta para que finalmente sea entregado al cliente. Por ende, está casi definida que su arquitectura debe ser cliente-servidor si se busca la mejor opción, prácticamente esta arquitectura es una platilla de cualquier sistema web. Además que las ventajas que nos aportará darle a nuestro cliente y que satisfacer las necesidades del mismo, son la accesibilidad, la seguridad y escalabilidad principalmente, y al tener un diseño de tres niveles, podemos obtener más beneficios como los son mejor rendimiento con la integridad de los datos, mejor seguridad comparada a la del diseño de dos niveles y que prácticamente oculta la estructura de la base de datos.

La accesibilidad debido a que el cliente desea acceder a su sistema web desde cualquier dispositivo electrónico que pueda realizar una conexión a Internet, es decir, puede ser desde una computadora, un celular, una Tablet, etc. Además que puede acceder durante cualquier hora del día al sistema, ya que los servidores estarán funcionando las 24 horas del día. También que el dispositivo que se emplee para acceder, no dependa de él completamente la aplicación, solamente se encargará de ejecutar la parte del nivel de representación, evitando la carga de la

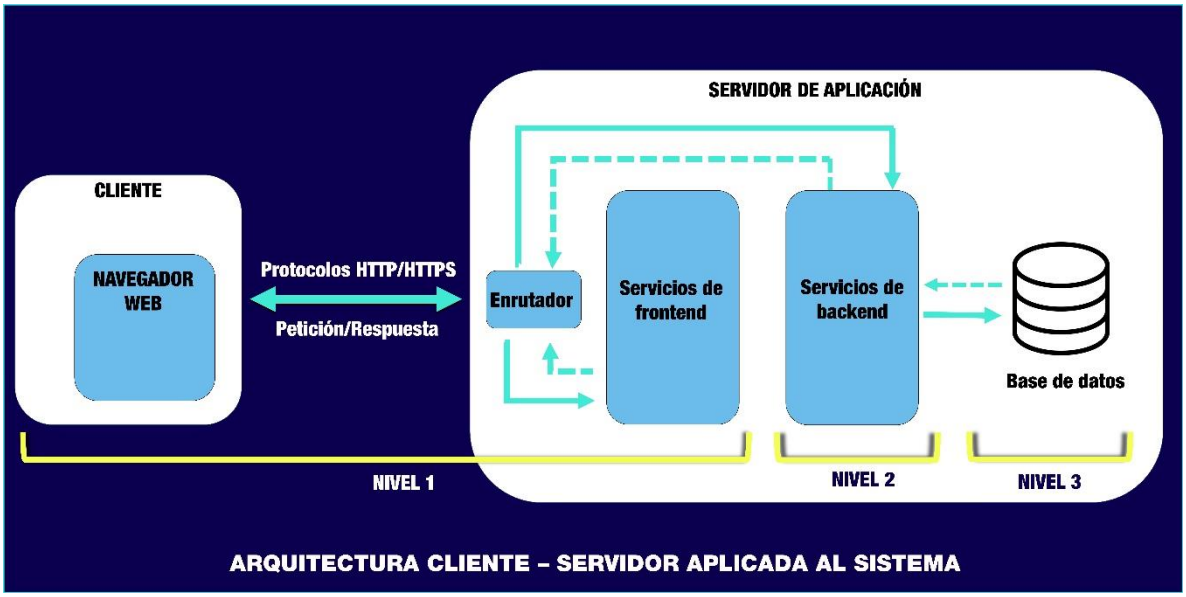
lógica de la aplicación y los datos, además que la arquitectura del dispositivo donde se accede y su software empleado para realizar peticiones, no influyen directamente en el funcionamiento del sistema web.

Respecto a la seguridad, esta se da de forma independiente, al ser de tres niveles, cada nivel realiza su parte de seguridad, esto ayuda a que se tenga una mejor integridad entre los servicios y a su vez se obtenga mayor seguridad en conjunto, y en cuanto a la seguridad de los datos, esta es mejor, ya que hay un nivel dedicado a tratar la gestión de los datos y es un nivel que no está directamente enlazado al cliente, sino hay un previo nivel que valida la autorización a los datos.

Y el otro factor que es relevante para optar por esta arquitectura, es que presenta una flexibilidad para ampliarla, además que aunque el diseño que se presenta es de tres niveles principalmente, se puede ir ampliando y adaptarse a un diseño de multinivel sin problema alguno, esto quiere decir que tiene escalabilidad horizontal que corresponde a añadir más clientes y escalabilidad vertical que indica la ampliación de la potencia de los servidores. Esto permite que nuestro cliente si requiere en un futuro acoplar el desarrollo de una aplicación móvil lo podría sin problemas o si requiere de más servicios se podrían agregar sin tener complicaciones a limitarnos en la cantidad de servicios a agregar.

Por lo mencionado, podemos decir que estos factores fueron los importantes para decidir por esta arquitectura, siendo la naturaleza de nuestro sistema web lo que nos orientó a optar por esta y de forma extra, los beneficios que se obtenían, siendo importantes estos, ya que estaríamos satisfaciendo de forma concreta con lo solicitado por el cliente, es decir, con los requerimientos no funcionales que se debe tener. Y contemplando las desventajas que pueda llegar a tener esta arquitectura, una de ellas se soluciona mediante el diseño con tres capas, y la que respecta a que el servidor falle, no habría problema debido a que el cliente es consciente de las fallas que existan, ya que no suelen ser frecuentes y tiene mucho sentido que si se cae un servicio, el sistema no funcione, ya que si unas partes funcionan, no tendría mucho sentido que lo hagan a medias, es decir, que sea atómico, si falla un nivel que fallen todos.

El diagrama del diseño de nuestra arquitectura es el presentado a continuación, donde se puede apreciar claramente los tres niveles y la relación cliente-servidor.



Implementación de Arquitectura Cliente-Servidor

Una vez que ya se tiene en claro la arquitectura en niveles con la que contará el sistema, se tiene que tener en cuenta que en cada una de estas capas se utilizarán diferentes tecnologías que desempeñan funciones especializadas para el funcionamiento de cada una de las partes del sistema.

Las tecnologías que se implementarán en el nivel de presentación deben:

- Permitir la conexión al servidor donde se encuentra el sistema web
- Mostrar la interfaz al usuario
- Gestionar peticiones
- Brindar elementos dinámicos a la interfaz del usuario
- Llevar a cabo la lógica de las acciones que el usuario realice y que afecten únicamente a la interfaz

Las tecnologías que se implementarán en el nivel de aplicación deben:

- Llevar a cabo la lógica del funcionamiento del sistema
- Manejo de una API para la comunicación entre los servicios de frontend, backend y el cliente
- Brindar el correcto formato de los datos que se reciben o se envían

Las tecnologías que se implementarán en el nivel de datos deben:

- Brindar los servicios típicos de una base de datos, como almacenar, gestionar, facilitar y ordenar datos.

Y respecto a los protocolos de comunicación que se emplean:

- TCP/IP

  Conjunto de protocolos utilizados asignar una dirección a los dispositivos de una red y asegurar el correcto envío de la información entre ellos. El protocolo IP se encarga de proporcionar una dirección IP a todos los equipos conectados a una red, ya sean hosts cliente o servidores. Por otra parte, el protocolo TCP se encarga de establecer y permitir la conexión entre dos hosts y asegurar el intercambio de datos entre ellos. TCP asegura además una comunicación segura entre los dos puntos, ya que cuenta con acuse de recibo.
- HTTP/HTTPS

  Tanto HTTP como HTTPS serán utilizados para las llamadas al API que se implementan como peticiones HTTP. El protocolo HTTP es la base de cualquier intercambio de datos en la Web, y un protocolo de estructura cliente-servidor, esto quiere decir que una petición de datos es iniciada por el elemento que recibirá los datos (el cliente), normalmente un navegador Web. HTTPS tiene la misma función, pero es más seguro, pues es necesario implementar un certificado SSL.
- SSL/TSL

  El protocolo SSL, "Secure Socket Layer" (capa de puertos seguros), es el predecesor del protocolo TLS (Seguridad de la Capa de Transporte). Se trata

de protocolos criptográficos que proporcionan privacidad e integridad en la comunicación entre dos puntos en una red de comunicación. Esto garantiza que la información transmitida por dicha red no pueda ser interceptada ni modificada por elementos no autorizados, garantizando de esta forma que sólo los emisores y los receptores legítimos sean los que tengan acceso a la comunicación de manera íntegra. Es el que se realiza junto al protocolo HTTP, dando lugar al HTTPS

- DNS

  DNS es el sistema de nombres de dominio. Los navegadores web interactúan mediante direcciones de protocolo de Internet (IP). El DNS traduce nombres de dominio a direcciones IP para que los navegadores puedan cargar los recursos de Internet.

- SSH

  SSH o Secure Shell, es un protocolo de administración remota que le permite a los usuarios controlar y modificar sus servidores remotos a través de Internet a través de un mecanismo de autenticación. Proporciona un mecanismo para autenticar un usuario remoto, transferir entradas desde el cliente al host y retransmitir la salida de vuelta al cliente.

Y estas tecnologías se acoplarían en cada nivel, esto se describe a continuación.

- En el primer nivel, se encuentra el cliente que puede acceder a nuestro sistema web mediante un navegador web, también los servicios de un enrutador que ayudarán a redireccionar peticiones y los servicios Frontend para la creación de la interfaz gráfica mediante un framework que facilite la creación de esa parte del sistema.
- En el nivel dos se encuentra la lógica de la aplicación o sistema, esta lógica será implementada mediante una API creada con el lenguaje de programación que se considere apropiado, además de un framework que facilite la creación del backend.
- En el nivel tres estará el servicio de base de datos

Y el funcionamiento o flujo de nuestro sistema de forma general sería el que se describirá a continuación. Se debe aclarar que dicho flujo se puede repetir varias veces, y también puede variar, no siempre es el mismo.

1. El cliente mediante un navegador web acceder a Internet y luego se dirige al dominio de nuestro sistema web, y realiza la petición al servidor,
2. En el servidor de aplicación se ejecuta el servicio de enrutamiento, e identifica a donde debe redireccionar la petición, puede identificar si a los servicios del Frontend o Backend.
3. Si redirecciona la petición a los servicios del Frontend, este lo que hace es analizar la petición y verifica si requiere de información extra de la base de datos, si es así, realiza una petición al enrutador para acceder a los servicios del Backend.
4. El enrutador procesa la petición y se la entrega a los servicios del Backend, el cual recibe y analiza la petición, ejecuta la lógica correspondiente, y si requiere de información, hace uso de los servicios de la base de datos.
5. La base de datos ejecuta las transacciones solicitadas por los servicios del Backend y retorna los datos solicitados.
6. Los servicios del Backend con la respuesta por parte de la base de datos le permite construir su respuesta para luego enviarla al enrutador.
7. El enrutador recibe la respuesta del Backend y se la retorna a los servicios del Frontend, quien ahora ya puede completar la primera petición, entonces procede a devolver una respuesta al enrutador con los componentes construidos con el framework que se encarga de la interfaz de usuario para formar la parte solicitada, ya sea una sola página web o varias.
8. Luego el enrutador retorna la petición al cliente con todo lo proporcionado por los servicios del Frontend.
9. Finalmente, el usuario pude observar la respuesta de lo que solicitó.

Conclusión

Para terminar, podemos concluir con la realización de este trabajo que tuvo como objetivo decidir el diseño arquitectónico y la identificación del sistema a desarrollar por parte de este sistema web HuxGym de la empresa Gym Huatulco Fitness Center. Con la realización de dicho trabajo, pudimos tener una gran mejora al momento de organizar nuestro sistema web respaldando los objetivos que tenemos en mente al desarrollar. Esta capacidad de definir los servicios nuevos, funciones y características que tenemos al momento de identificar qué tipo de sistema pertenece el proyecto presentado no sirve de apoyo para estructurar mejor el sistema. Durante el desarrollo de este trabajo tuvimos unas confusiones respecto a la identificación del sistema, ya que las características que tiene pueden ser semejantes a las de un sistema de procesamiento de transacción, sin embargo, entendimos que esta solo puede realizar ciertos movimientos y no tiene el alcance final del tipo de sistema que en verdad tiene nuestro proyecto. También al momento de definir la arquitectura tuvimos algunos problemas porque la arquitectura cliente-servidor y por capas, parecían ser iguales o semejantes así como estaban descritas en el documento compartido por el catedrático, por lo que tuvimos que realizar una búsqueda más amplia respecto a las arquitecturas y así mismo buscamos ejemplos de aplicaciones o sitios web para tener un mejor panorama de lo que podríamos implementar, hasta finalmente decidir por una arquitectura que vimos correcta.

En el presente trabajo definimos el tipo de arquitectura que mejor se adaptó a nuestro sistema web y en el cual se optó por una arquitectura cliente-servidor, debido a que esta nos ayudaría a satisfacer los requerimientos no operacionales del cliente. También identificamos las ventajas y desventajas que tenemos al implementar este tipo de arquitectura y cómo afectaría el rendimiento del sistema web. Con los diagramas presentados, tenemos una mejor idea de cómo está estructurado el sistema a desarrollar por parte tanto del Backend y Frontend, identificamos y establecimos todos los servicios y tecnologías que se requieren para la construcción y funcionamiento del sistema, además de los protocolos requeridos de nuestro sistema.

Bibliografía

Red Hat, Inc. (2021). *Red Hat*. Recuperado el 08 de Mayo de 2021, de ¿Qué es una API de REST?: https://www.redhat.com/es/topics/api/what-is-a-rest-api