**ANEXO**

**DOCUMENTO DE ARQUITECTURA**

Contenido

[1. ALCANCE DEL DOCUMENTO 3](#_Toc17210271)

[2. RESUMEN DEL PROYECTO 3](#_Toc17210272)

[3. DEFINICIÓN DE RECURSOS (HARDWARE) 4](#_Toc17210273)

[4. DIAGRAMA DE ARQUITECTURA FISICA 6](#_Toc17210274)

[5. DEFINICIÓN DE RECURSOS (SOFTWARE) 7](#_Toc17210275)

[6. ARQUITECTURA DE SOFTWARE. 10](#_Toc17210276)

[7. DIAGRAMA DE DESPLIEGUE. 11](#_Toc17210277)

## ALCANCE DEL DOCUMENTO

El presente documento tendrá por finalidad explicar la estrategia abordada por el equipo de desarrollo, basando la decisión de acuerdo a los elementos observados y direccionando el análisis a la solución del proyecto.

## RESUMEN DEL PROYECTO

En presente proyecto nace bajo la necesidad de poder satisfacer a un rango de la población que busca obtener clases particulares, de la misma manera que entregar una propuesta a los profesionales que buscan obtener ingresos por sus servicios. Es así que la solución propuesta implica un sistema de comunicación basado en servicios API y una plataforma web progresiva, entregando beneficios para los dos sectores implicados y logrando satisfacer sus necesidades.

De esta manera se establecen los alcances que el proyecto tendrá que cumplir, la metodología implicada y los riesgos presentes en todo el desarrollo del mismo.

## DEFINICIÓN DE RECURSOS (HARDWARE)

A continuación, se detallan las características técnicas de la infraestructura en la cual se construyó y probó la solución; se dispuso la siguiente estructura para cada una de las máquinas, donde se levantaron los distintos componentes:

**Máquinas de desarrollo:**

Equipo 1: HP EliteBook

Sistema Operativo: Windows 10 Pro

Memoria RAM: 32 GB

Almacenamiento: SSD 500 GB

Procesador: Intel Core i7 8400U / 2.1 GHz

Equipo 2: Asus

Sistema Operativo: Windows 10

Procesador: Intel Core i5-7200 CPU @ 2.50 GHz

Memoria RAM: 8 GB.

Almacenamiento: SSD 256 GB.

**Equipamiento de Pruebas:**

Equipo: HP EliteBook

Sistema Operativo: Windows 10 Pro

Memoria RAM: 32 GB

Almacenamiento: SSD 500 GB

Procesador: Intel Core i7 8400U / 2.1 GHz

Equipo: Asus

Sistema Operativo: Windows 10

Procesador: Intel Core i5-7200 CPU @ 2.50 GHz

Memoria RAM: 8 GB.

Almacenamiento: SSD 256 GB.

Hosting ON Cloud Amazon:

Equipo: Máquina virtual.

Sistema Operativo: Linux

Procesador: virtual 1 Core

Memoria RAM: 16 GB.

## DIAGRAMA DE ARQUITECTURA FISICA

A continuación, se muestra en la figura No 01 el diagrama de la estructura física para el despliegue de la aplicación, mostrando los servicios que se utilizaran para la base de datos y el servidor web en Amazon AWS, equipo descrito en el punto anterior.

Se considera como host cualquier equipo del cliente que tenga como navegador web Google Chrome/Firefox/Edge.

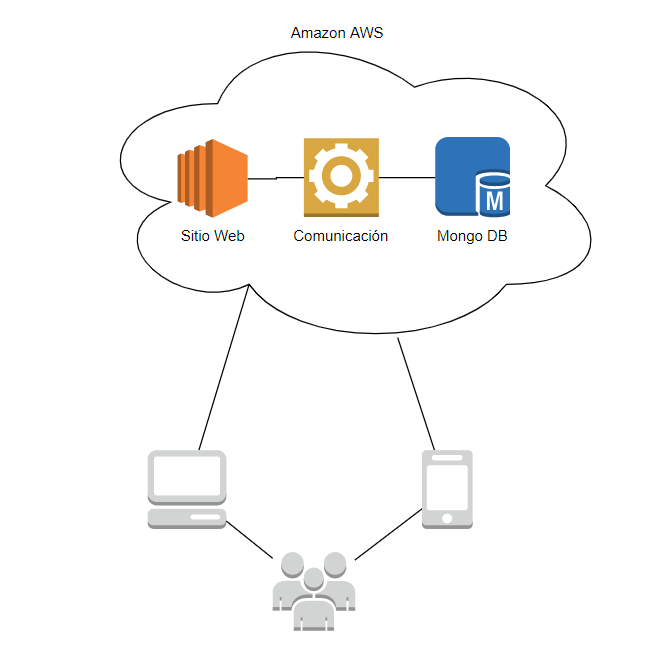


Figura No 01 “Diagrama de arquitectura física”

## DEFINICIÓN DE RECURSOS (SOFTWARE)

A continuación, se detallan las características técnicas del software que se utilizó en la construcción de la solución, informamos la siguiente lista de software necesarios para la creación y levantamiento de los distintos productos:

* 1. **PWA:**

Aplicación web progresiva son aplicaciones que están a medio camino entre las aplicaciones nativas y aplicación web: son básicamente páginas web, pero mediante el uso de Service Workers y otras tecnologías se comportan más como aplicaciones normales que como aplicaciones web.

Mediante los Service Workers y otras tecnologías las aplicaciones web progresivas pueden seguir ejecutándose en segundo plano sin tener que vivir dentro del navegador. En el móvil es posible instalarlas como una aplicación más y también en Windows mediante la mediación de Google Chrome y Mozilla Firefox.

* 1. **Node.js:**

Node.js es un ambiente de ejecución de JavaScript. Utiliza un modelo de entrada y salida sin bloqueo controlado por eventos, de esta manera lo hace un entorno ligero y eficiente.

Node.js cambió la forma en que programamos JavaScript ya que ahora todo nuestro código debe funcionar de manera asíncrona a partir de eventos.

Gracias a que está construido en base a V8, motor el cual está escrito en C++, prácticamente podemos correr JavaScript en cualquier sistema operativo, desde servidores hasta dispositivos móviles.

Los motivos de su elección y posterior utizacion :

* La compilación de Node.js se realiza en tiempo de ejecución, Just In Time (JIT), esto trae consigo una mayor optimización a las funciones que más veces sean requeridas.
* Mediante clústeres permite tener una alta escalabilidad.
* Podemos expandir nuestro código añadiendo módulos de forma fácil gracias al Node Package Manager (NPM).
* Un alto rendimiento en proyectos donde se necesite ejecución en tiempo real.
* En equipos pequeños podremos realizar front-end, back-end y hasta una aplicación móvil con un mismo lenguaje.
* Es la base de construcción más utilizada en sitios web PWA (webs progresivas)
  1. **MongoDB:**

Como motor de base de datos utilizamos MongoDB, la cual es una base de datos de tipo no relacional, esto es, que en ella no existen tablas, si no que la información se guarda como documentos en formato BJSON, el cual es un formato de json, ya que está construida en JavaScript.

se eligió la base de datos por los siguientes motivos:

* Velocidad. Si una aplicación necesita almacenar o acceder a mucha información en poco tiempo, se necesita una base de datos que soporte gran velocidad en las consultas. Las bases de datos documentales son capaces de ser mucho más rápidas que las relacionales, pudiendo atender clientes que necesiten realizar muchas operaciones por segundo.
* Volumen. En cuanto al tamaño de la base de datos, si tenemos gran cantidad de información, podríamos tener la necesidad de soportar importantes cantidades de volumen. Las bases de datos relacionales tienen tendencia a funcionar más lentamente cuando en una tabla se encuentran cantidades muy grandes de registros (del orden de un millón para arriba). Situaciones así obligan a los administradores a buscar soluciones, como dividir las tablas en diversos segmentos, produciendo un coste en el acceso a los datos y la operativa. Este no es un problema en las bases de datos NoSQL, que son capaces de administrar volúmenes gigantescos de datos en sus entidades.
* Variabilidad. Las necesidades enormes de velocidad y volumen suelen darse juntas y afectan a muchas aplicaciones actuales. Sin embargo, hay otra característica de la información que es todavía más representativa para decantarse por las NoSQL, como es la variabilidad. En bases de datos relacionales el esquema de la información está minuciosamente definido de antemano. Por ejemplo, no puedes inventarte campos en los registros sobre la marcha. En las bases de datos documentales, como MongoDB, no hay problema en que cada documento almacene campos distintos, pudiendo ser flexibles en cuanto al esquema de la información.
  1. **GitHub:**

Se utiliza como repositorio para el control de versiones del código de la herramienta web.

* 1. **AWS (Amazon Web Services Cloud):**

Se utilizó como ambiente Cloud para el levantamiento del servidor que tiene contenida la herramienta web y la base la base de datos, se seleccionó por las siguientes razones:

* Reducción de costos: Si bien es cierto no es gratuito, tiene una versión de prueba que entrega un servidor para todo uso gratis por un año. En comparación con otras nubes como azure (de microsoft), el precio por características es bastante más bajo.
* Seguro: Aplica un enfoque integral para proteger y reforzar la infraestructura, incluidas medidas físicas, operativas y de software.

## ARQUITECTURA DE SOFTWARE.

Para la construcción de la web se empleó un patrón MVP (Modelo, Vista, Presentador), éste es una derivación del clásico patrón MVC (Modelo, Vista, Controlador), en el patrón original MVC, el controlador, la vista y el modelo tienen comunicación entre sí, lo que permite que la vista consulte al modelo directamente sin pasar por el controlador siempre, como muestra la imagen a continuación.

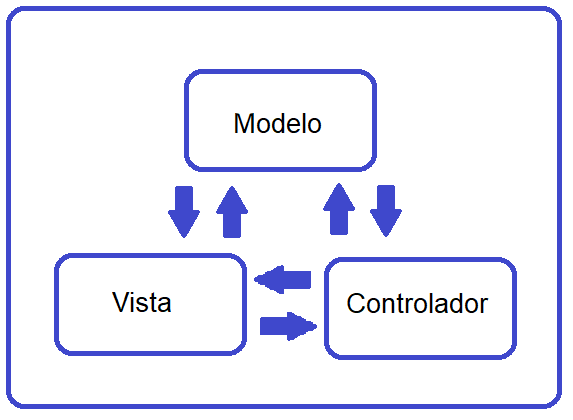


Figura No 02

Por otro lado, el Patrón MVP obliga a que toda la comunicación de las demás capas tenga que pasar por la capa de presentador, por lo que esta última asume la responsabilidad de intermediario, lo que aumenta los estándares de seguridad de la aplicación, ya que no permite consultas de la vista directamente al modelo, que no pasen por la lógica del presentador como se puede apreciar a continuación.

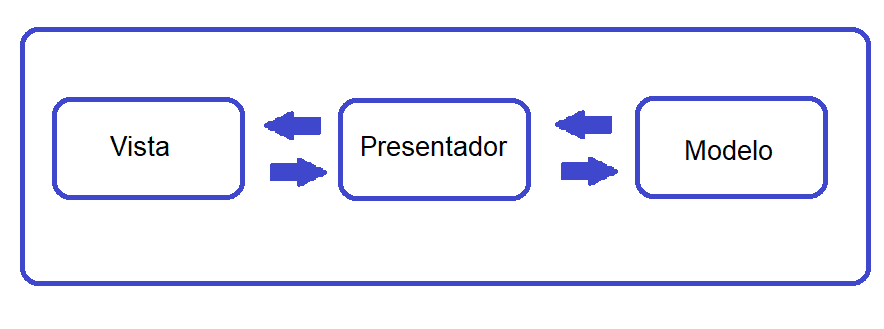


Figura No 03

## DIAGRAMA DE DESPLIEGUE.