TODI STORE



DOCUMENTO DE ARQUITECTURA DE SOFTWARE PARA EL DASHBOARD DE SEGUIMIENTO DE ESTUDIOS DE VIDEOJUEGOS INDEPENDIENTES (DSEVI)



|  |  |  |
| --- | --- | --- |
| HISTORIAL DE CAMBIOS | | |
| Versión | Fecha | Cambios realizados |
| 01 | 11/11/2022 | Emisión inicial, Introducción, Propósito ,alcance del documento, especificación de los atributos de calidad |

|  |  |  |
| --- | --- | --- |
| Elaboró | Revisó | Aprobó |
| Calderón Bermudo | Villacis Alvear | Torres Domenack |

Contenido

[**Introducción**](#_blmed0ybxli) **4**

[**Propósito**](#_ksl37zg5yny1) **4**

[**Alcance**](#_rxx7h56utoqk) **4**

[**Atributos de calidad**](#_5n8ox4gdkfn2) **4**

[**Estilos arquitectónicos**](#_74afzhp13ar5) **4**

**Tecnologías Utilizadas 5**

[**Estructura arquitectónica general**](#_ppgjeb9l5cqm) **6**

**Decisión arquitectónica 6**

## Introducción

El presente documento provee una vista de alto nivel de la arquitectura del Sistema del Dashboard de Seguimiento de Estudios de Videojuegos Independientes (DSEVI) y da una vista general del proceso de desarrollo.

## Propósito

El Documento de Arquitectura de Software presenta la arquitectura a través de ilustraciones, cada una de las cuales ilustra un aspecto en particular del software desarrollado. Se pretende de esta forma que el documento brinde al lector una visión global y comprensible del diseño general del sistema desarrollado.

## Alcance

En el documento se incluyen los aspectos fundamentales de la arquitectura y se omiten aquellos puntos que no se consideren pertinentes como es el caso de los procesos específicos del sistema.

## Atributos de calidad

* **Escalabilidad:**

La aplicación mantendrá su buen funcionamiento y rendimiento tras los cambios o crecimiento que pueda aplicarse a lo largo de su ciclo de vida.

* **Eficiencia:**

El sistema debe ser capaz de operar adecuadamente con hasta 1000 usuarios con sesiones concurrentes. El sistema debe ser capaz de procesar 900 transacciones por segundo.

* **Seguridad:**

El sistema guardará las contraseñas de los usuarios encriptándolas en la base de datos.

* **Portabilidad** :

El sistema podrá ser ejecutado de manera transparente en diferentes plataformas en caso fuera necesaria la migración de sistema operativo, base de datos o tecnología de equipos.

* **Usabilidad:**

El tiempo de aprendizaje del sistema por un usuario debe ser menor a cuatro horas. El sistema debe contar con manuales de usuario debidamente estructurados.

## Estilos arquitectónicos

Las aplicaciones web son un tipo especial de aplicación cliente/servidor, donde el cliente (el navegador o explorador) como el servidor (servidor web) y el protocolo de comunicación (HTTP) están estandarizados y/o han de ser creados por el programador de la aplicación.

La idea principal de este modelo es una arquitectura de dos capas (la capa cliente y la capa de servidor); sin embargo, este modelo se puede extender a una arquitectura de tres capas, donde la interfaz del usuario, la lógica de negocio y las bases de datos se separan por componentes

## Tecnologías Utilizadas

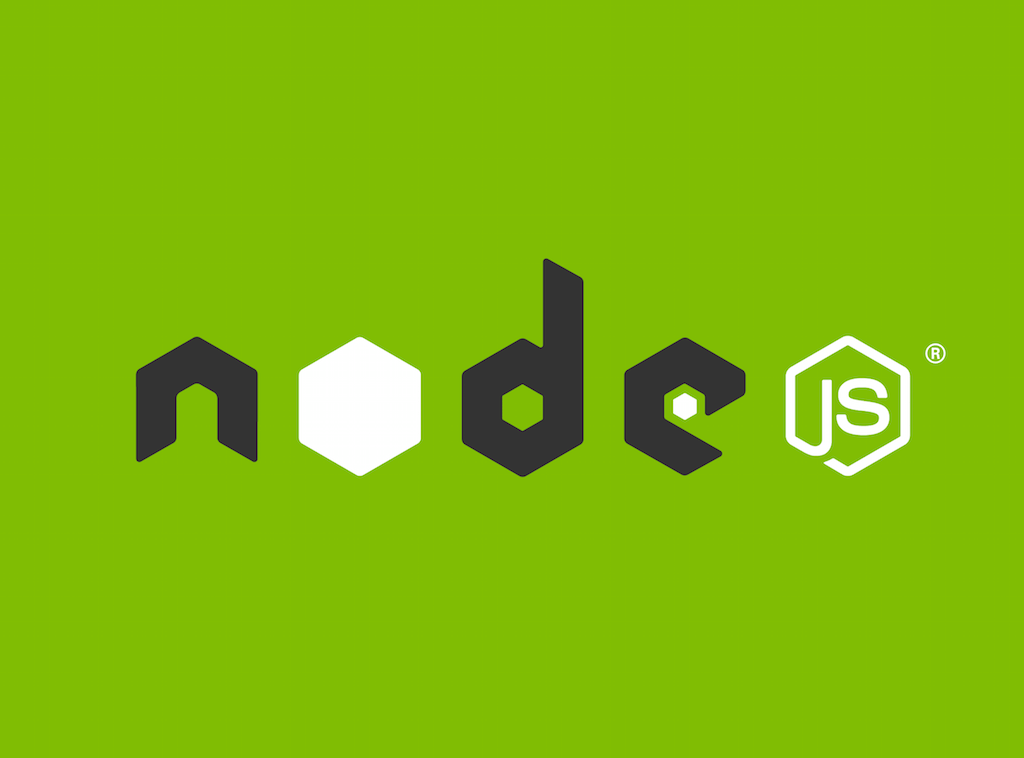
Para el desarrollo del sistema DSEVI se utilizaron las siguientes tecnologías y servicios.

HTML/CSS

HTML es el lenguaje de marcado con el que rodeas contenido, para decirle a los navegadores acerca de los encabezados, listas, tablas, etc. CSS es el lenguaje de hoja de estilos con el cual estilas la página, para decirle a los navegadores que cambien el color, fuente, diseño y más.

Angular

Angular es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página.

NodeJS

Es un entorno de tiempo de ejecución de JavaScript, de ahí su terminación «. js». Este entorno de tiempo es open source, es decir, de código abierto, multiplataforma y que se ejecuta del lado del servidor.

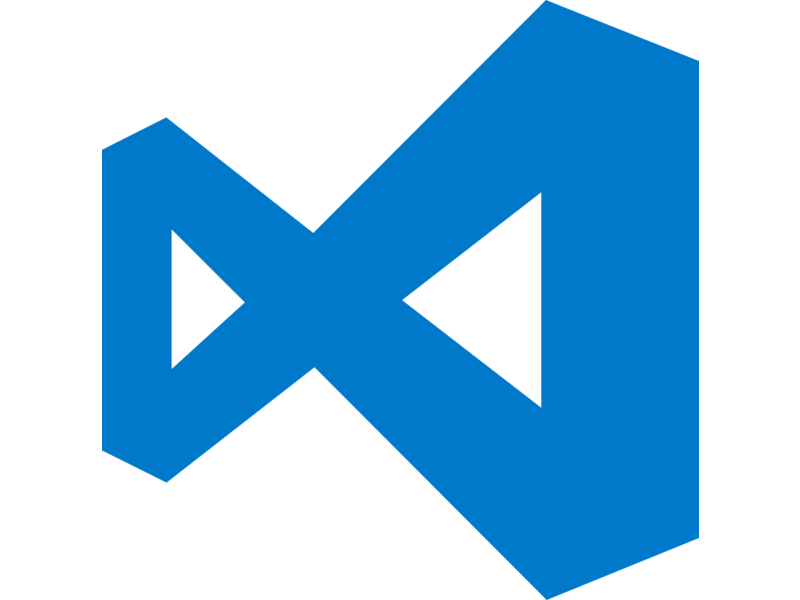


ExpressJS

Es un entorno de trabajo para aplicaciones web para Node.js, de código abierto y con licencia MIT. Se utiliza para desarrollar aplicaciones web y APIs

MongoDB

Es un sistema de base de datos NoSQL, orientado a documentos y de código abierto. En lugar de guardar los datos en tablas, tal y como se hace en las bases de datos relacionales



Visual Studio Code

Editor de código fuente independiente que se ejecuta en Windows, macOS y Linux. La elección principal para desarrolladores web y JavaScript, con extensiones para admitir casi cualquier lenguaje de programación.

TypeScript



Es un lenguaje de programación libre y de código abierto desarrollado y mantenido por Microsoft. Es un superconjunto de JavaScript, que esencialmente añade tipos estáticos y objetos basados en clases

## Estructura arquitectónica general

## Decisión Arquitectónica

|  |  |
| --- | --- |
| **Problema** | Se requiere un sistema de comercio electrónico en el que los clientes de TODI Store puedan comprar productos desde cualquier navegador web. Además, se debe permitir gestionar este e-commerce desde otro sistema web de uso interno y exclusivo para TODI Store. |
| **Decisión** | Arquitectura de tres capas |
| **Estado** | Aprobado |
| Supuestos | Calendario:  -Revisar cronograma del proyecto (DSEVI-CP)  Herramientas usadas (MEAN STACK):  - HTML/CSS  - Angular  - NodeJS  - ExpressJS  - MongoDB  - Visual Studio Code |
| **Restricciones** | Es mucho más difícil programar y probar el software que en una arquitectura de dos niveles porque tienen que comunicarse más dispositivos para terminar la transacción de un usuario, y esto supone también una mayor cantidad de tráfico en la red. |
| **Otras propuestas** | 1. Multicapa de 4 capas      1. Multinivel |
| **Argumento** | La ventaja fundamental de una arquitectura n-capas comparado con una arquitectura de dos niveles (o una tres-capas con una de dos niveles) es que separa hacia fuera el proceso, eso ocurre para mejorar el balance de la carga en los diversos servidores.  Centralización del control: los accesos, recursos y la integridad de los datos son controlados por el servidor de forma que un programa cliente defectuoso o no autorizado no pueda dañar el sistema.  Escalabilidad: se puede aumentar la capacidad de clientes y servidores por separado. Cualquier elemento puede ser aumentado (o mejorado) en cualquier momento, o se pueden añadir nuevos nodos a la red (clientes y/o servidores).  Fácil mantenimiento: al estar distribuidas las funciones y responsabilidades entre varios ordenadores independientes, es posible reemplazar, reparar, actualizar, o incluso trasladar un servidor, mientras que sus clientes no se verán afectados por ese cambio (o se afectarán mínimamente). Esta independencia |
| **Principios relacionados** | Las decisiones anteriormente planteadas son de la creación de los siete integrantes del grupo, estamos dispuestos a abordar el DSEVI cumpliendo con los principios que consideramos más importantes. De esta forma garantizamos la eficiencia y calidad adecuada, interfaces agradables al usuario y costos accesibles para el cliente.  El sistema y las decisiones que hemos planteado van acorde con ello. |