UNIVERSIDAD PRIVADA DE TACNA

FACULTAD DE INGENIERIA

Escuela Profesional de Ingeniería de Sistemas

Proyecto "Implementación de una tienda online especializada de productos y servicios para mascotas PETLORD"

Curso: Programación III

Docente: *Ing. Elard Rodríguez Marca*

Integrantes:

Caxi Calani, Luis Eduardo ***(2018062487)***

Castillo Delgado, Jesus Angel ***(2018000491)***

Gonzalez Franco, Daniel Alejandro ***(2015052599)***

Tacna – Perú

2022

CONTROL DE VERSIONES					
Versión	Hecha por	Revisada por	Aprobada por	Fecha	Motivo
1.0	MPV	ELV	ARV	10/10/2020	Versión Original

Sistema *"Implementación de una Tienda online de productos y servicios para mascotas PETLORD"*
Documento de Arquitectura de Software

Versión *1.0*

DOCUMENTO DE LA ARQUITECTURA DEL SOFTWARE

1. Introducción

La base y lo más importante en cualquier desarrollo de software es la arquitectura, que es la que provee una estructura sólida y organizada al sistema y por años los diseñadores de software han estado desarrollando sistemas sin tomar en cuenta este aspecto y por lo tanto los sistemas resultantes no son lo que los clientes desean o en el mejor de los casos no cumple con todos los objetivos establecidos, por lo tanto se realizan sistemas sin calidad y es debido a esto que se requiere de más tiempo y esfuerzo para repararlos. No es si no hasta 1990 cuando el termino de "Arquitectura de Software" comenzó a tomar aceptación tanto en la comunidad de investigadores como en la industria. Las bases sobre las cuales éste campo de la Ingeniería de Software subyace fueron creadas por sus fundadores, David Parnas, Fred Brooks, Edsger Dijkstra, entre otros.

Por ello, el presente documento hace una descripción y brinda una visión general de la arquitectura del Sistema de la PetShop, el cual es el software a desarrollar por el grupo de trabajo.

1.1 Propósito

El presente documento brindara una descripción detallada de la arquitectura del software de la website para la empresa de PetLord a través de diferentes vistas arquitectónicas, las cuales ilustran un aspecto en particular del software a desarrollarse. De esta forma, se pretende brindarle al lector una visión global y comprensible del diseño general del tema presentado.

1.2 Alcance

Ofrece soluciones funcionales ajustadas a las necesidades específicas de la empresa, se podrán acceder a los datos de la empresa en tiempo real con herramientas fáciles de usar que permitan la adecuada administración y control del PetShop.

1.3 Definición

Termino	Definición
Actor	Usuario del sistema que puede participar De un caso de uso.
Arquitectura de software	conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente como al comportamiento del software en tiempo de ejecución. Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto
Caso de Uso	Secuencia de acciones que el sistema realiza, la cual proporciona un resultado de valor observable.
Diseño	Actividad creativa que proyecta objetos para después fabricarlos.
Escenario	Especifica el comportamiento y limita el interés de un área específica del sistema para 1 o varios stakeholders.
Paquetes	Agrupaciones de casos de uso y actores por funcionalidad que proveen.

1. **Análisis de Requerimientos**

4.1 Requerimientos funcionales

Requerimientos funcionales			Sistema	Prioridad
RF-1	Usuarios	Ingresar al Sistema	Sistema Escritorio	Alta
RF-2	Usuarios	Salir del Sistema	Sistema Escritorio	Alta
RF-3	Administrador	Gestionar Cuentas	Sistema Escritorio	Alta
RF-4	Administrador	Gestionar Productos	Sistema Escritorio	Alta
RF-5	Administrador	Gestionar Compras	Sistema Escritorio	Alta
RF-6	Administrado	Generar reportes	Sistema Escritorio	Alta
RF-7	Visita	Registrarse	Sistema Escritorio	Alta
RF-8	Cliente	Buscar Producto	Sistema Web	Alta
RF-9	Cliente	Comprar Producto	Sistema Web	Alta
RF-10	Cliente	Escoger formas de pago	Sistema Web	Alta
RF-11	Cliente	Visualizar Historial de compras	Sistema Web	Alta

4.2 Requerimientos no funcionales

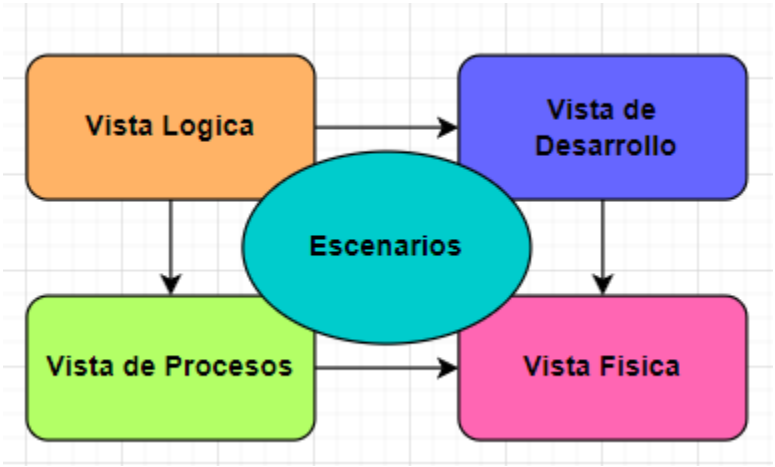
ID	Nombre del Requisito	Descripción
RNF-1	Disponibilidad	Estará disponible el sistema tanta apertura hasta el cierre del negocio y solo el encargo tendrá acceso las 24 horas.
RNF-2	Rendimiento	Dará respuesta inmediata a cualquier solicitud de pedido y se emitirá un comprobante de pago una vez finalizado
RNF-3	Usabilidad	El software debe tener una gráfica de fácil uso e intuitiva, además debe permitir la rapidez de la operación
RNF-4	Fiabilidad	Emitirá información veraz a tiempo real.
RNF-5	Seguridad	Los datos registrados serán confidenciales
RNF-6	Facilidad	Los empleadores tendrán un manual de fácil entendimiento para el uso del sistema.

2. Representación Arquitectónica

Para representar adecuadamente la arquitectura de un sistema es necesario contar con varios diagramas o vistas. Dada la cantidad de características y de elementos que tiene un sistema de software no es posible incluirlos todos en un solo diagrama y que sirva, además, para todas las personas que participan en el desarrollo.

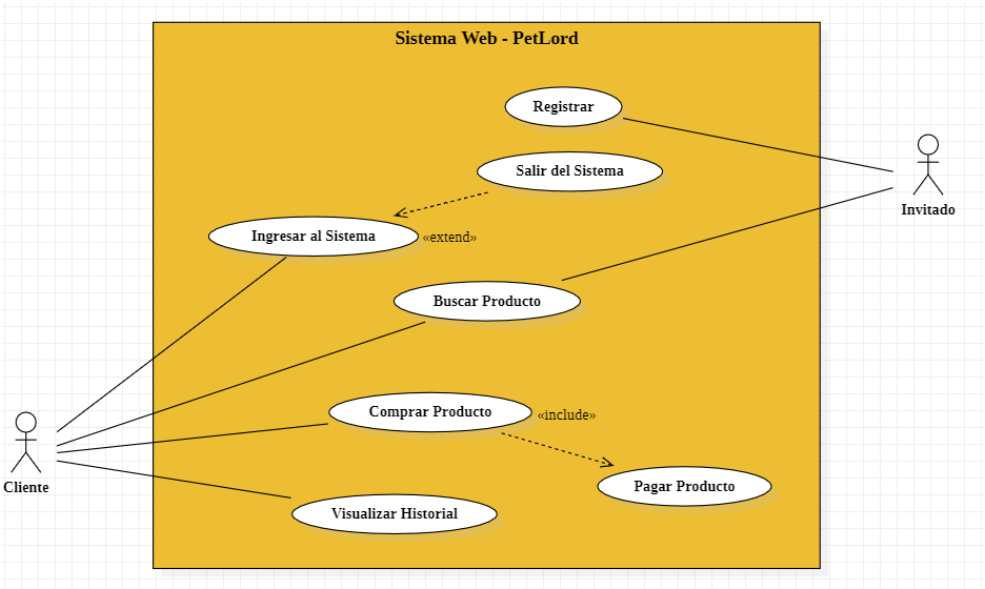
MODELO DE 4+1 VISTAS

Es un modelo de vistas diseñada por el profesor Philippe Kruchten y que encaja con el estandar “IEEE 1471-2000" que se utiliza para describir la arquitectura de un sistema de software intensivo basado en el uso de múltiples puntos de vista



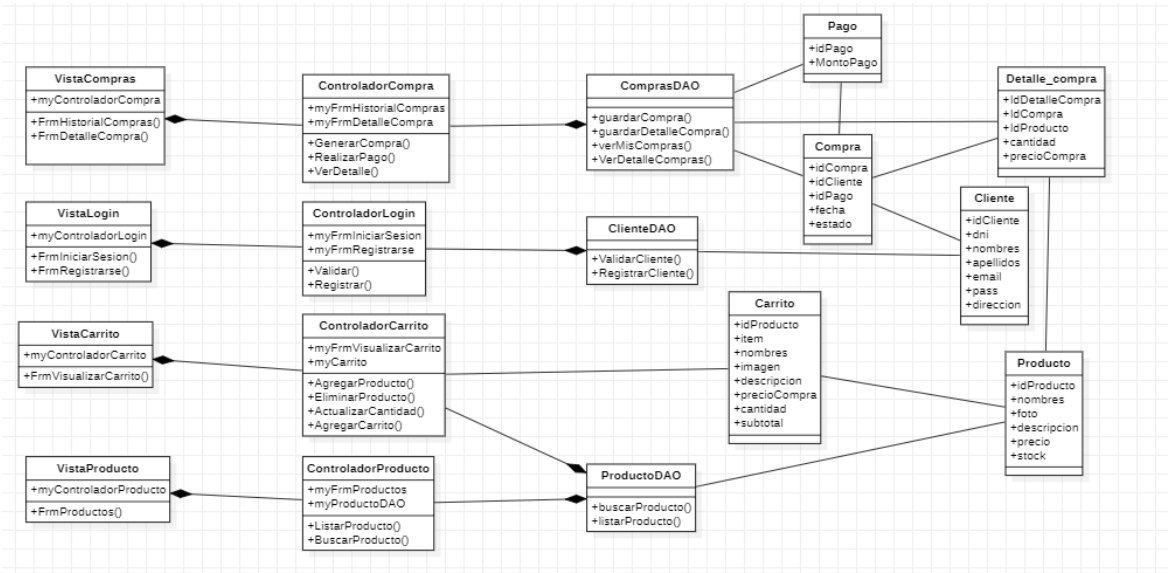
SISTEMA WEB:

2.1 Escenarios

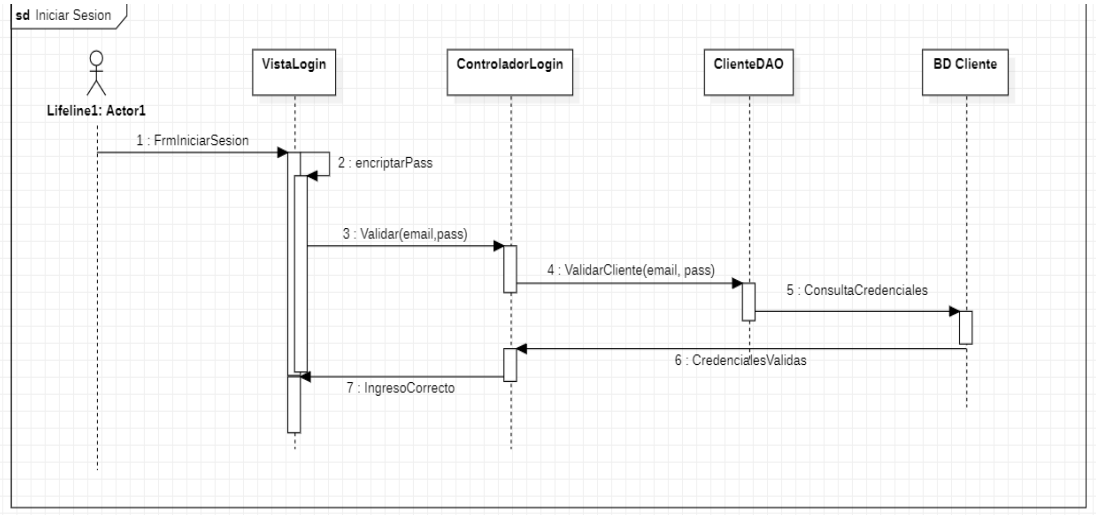


2.2 Vista Lógica

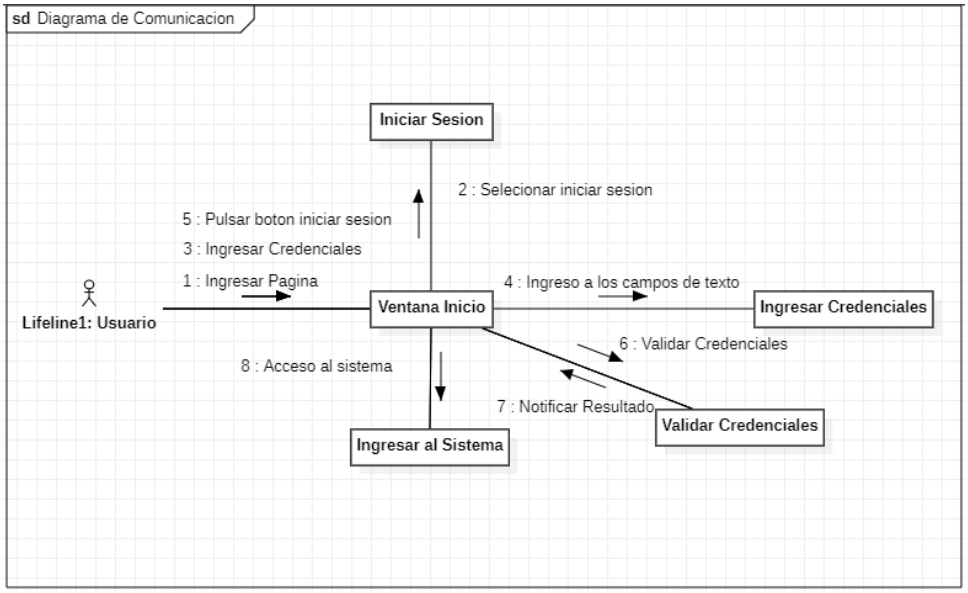
- Diagrama de clases



- Diagrama de Secuencia

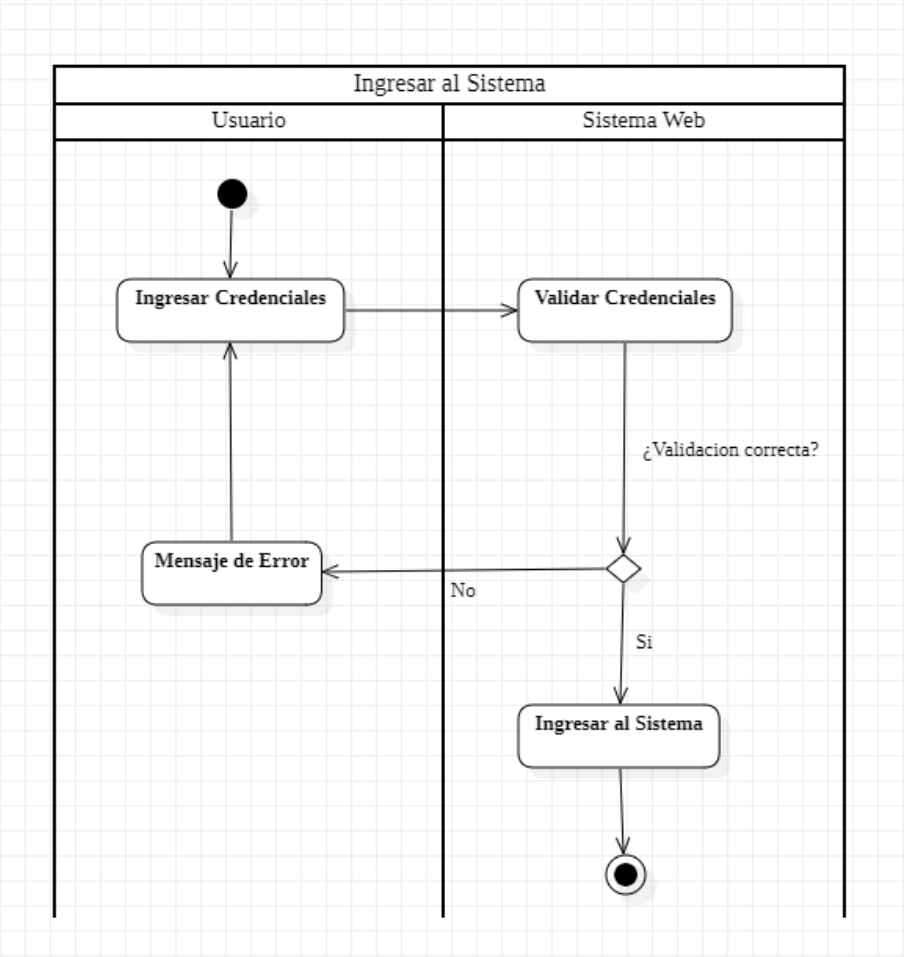


- Diagrama de Comunicación



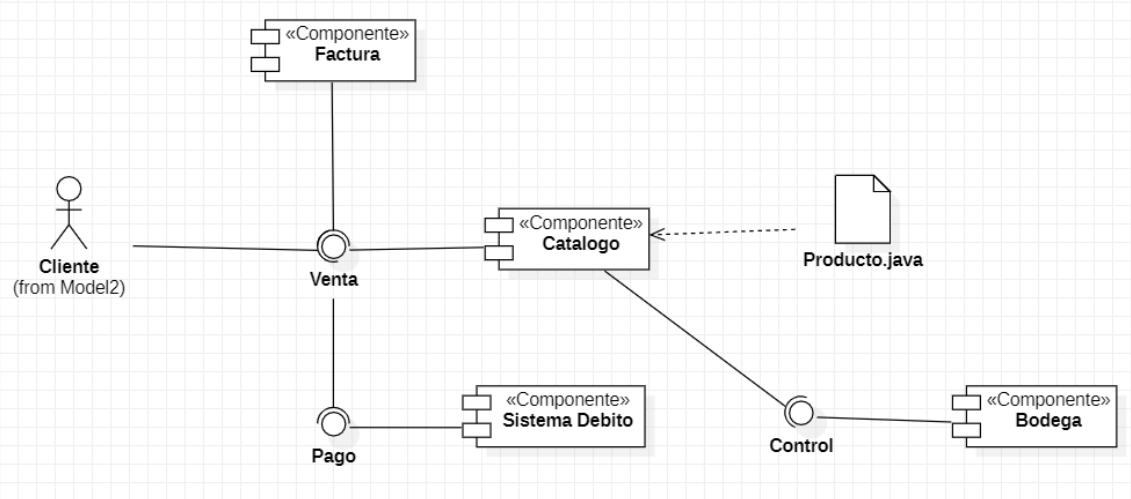
2.3 Vista del Proceso

- Diagrama de Actividades

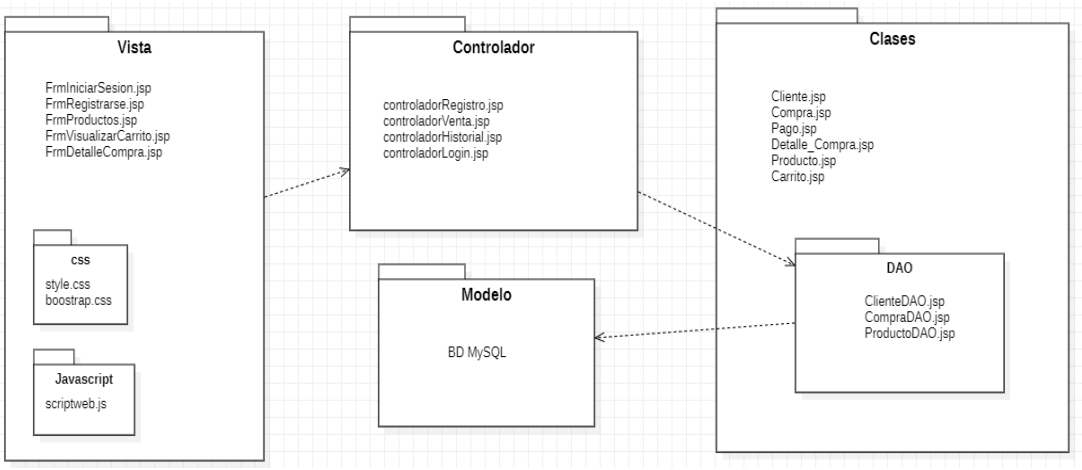


2.4 Vista del Desarrollo

- Diagrama de Componentes

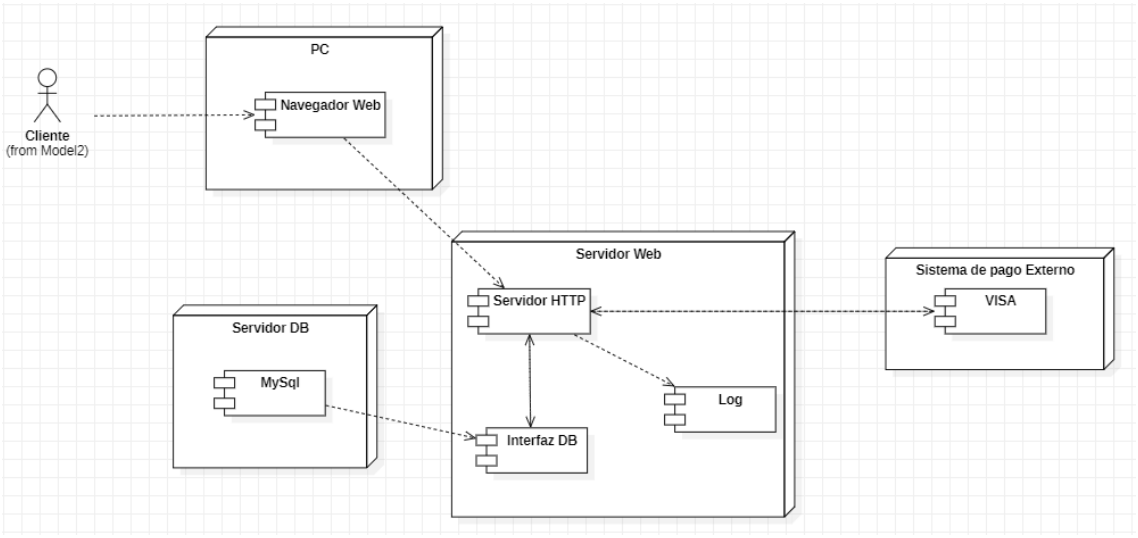


- Diagrama de Paquetes



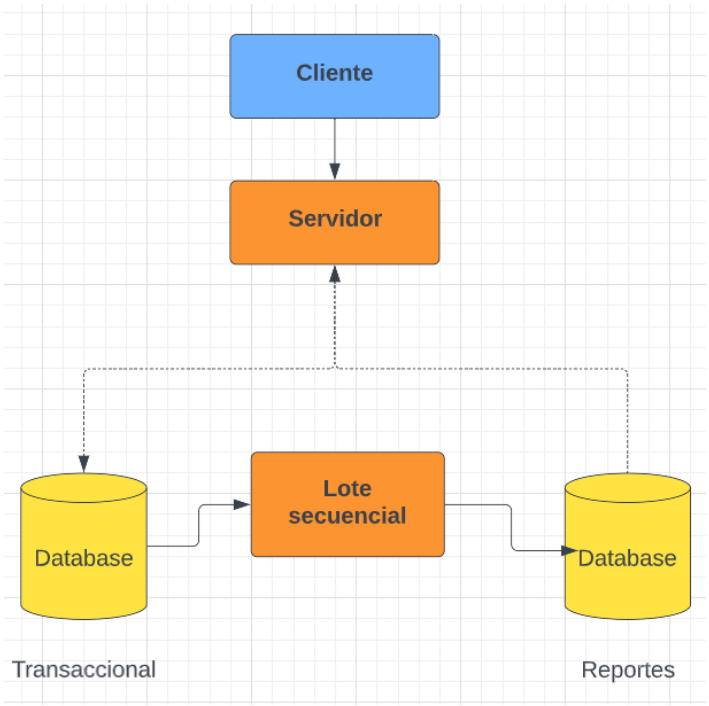
2.5 Vista Física

- Diagrama de Despliegue

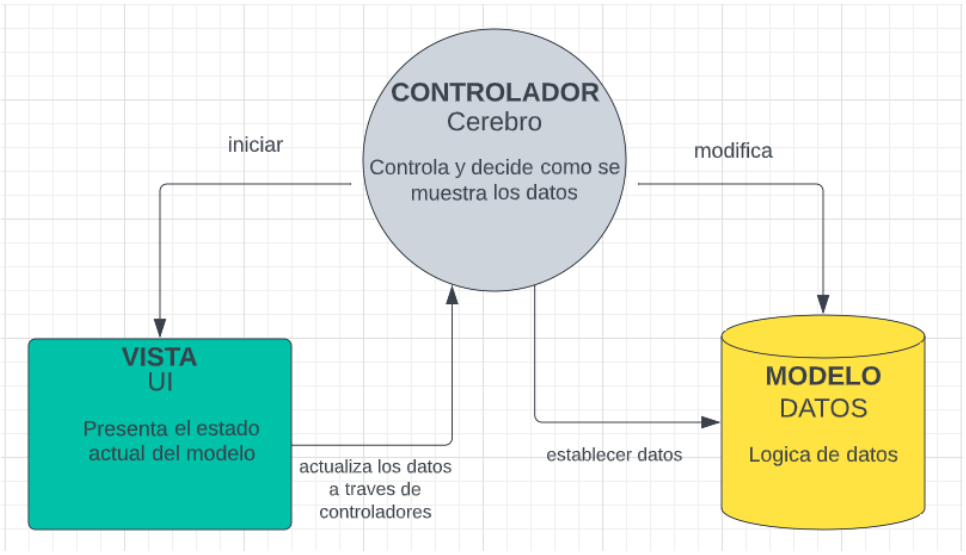


2.6 Estilos y Patrones Arquitectónicos

- Arquitectura Cliente/Servidor



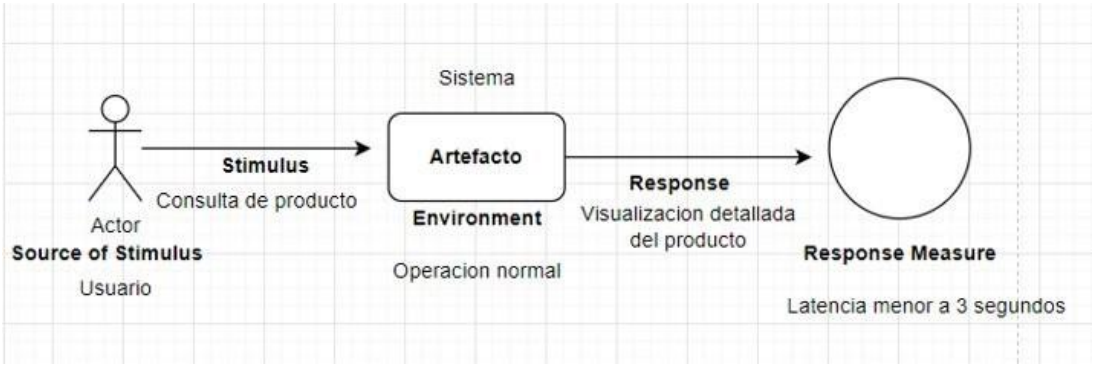
- Patrón de Arquitectura MVC



3. Objetivos y limitaciones arquitectónicas

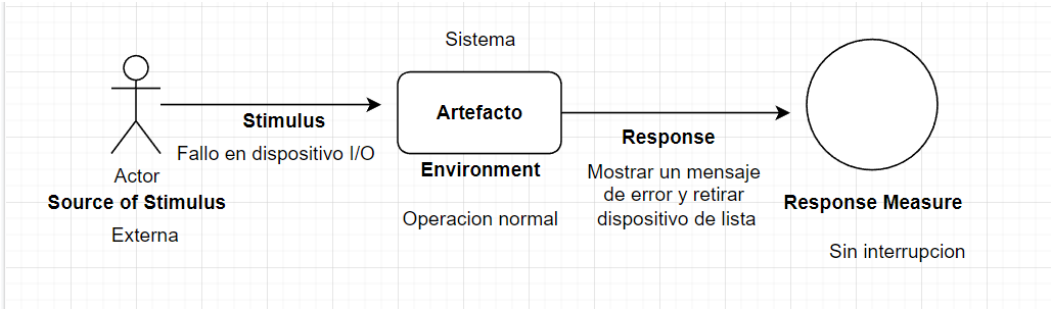
3.1 Desempeño

- Un usuario realiza una consulta al catálogo de productos en un momento normal de operación del sistema. El sistema muestra el resultado de la consulta en un tiempo no mayor a 3 segundos.



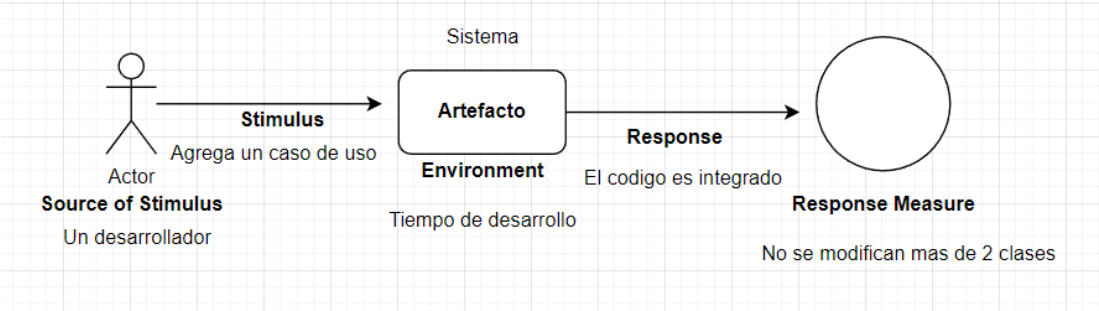
3.2 Disponibilidad

- Una falla en un dispositivo I/O causa que un driver del sistema deje de funcionar en un momento normal de la operación. El sistema operativo deberá mostrar un mensaje de error retirando al dispositivo de la lista de dispositivos disponibles y continuar operando sin interrupción.



3.3 Modificabilidad

- Un desarrollador agrega un caso de uso al esqueleto ejecutable de la arquitectura en tiempo de desarrollo. El código del caso uso es integrado y no requiere de modificaciones en más de 2 clases de la arquitectura.



3.4 Rendimiento