**DOCUMENTO DE ARQUITECTURA DE SOFTWARE**

# **1.** **Introducción: .**

Este documento proporciona una descripción general de alto nivel y explica la arquitectura Modelo Vista Controlador de la aplicación web de tienda virtual.

El documento define los objetivos de la arquitectura, los casos de uso admitidos por el sistema, los estilos arquitectónicos y los componentes seleccionados. El documento proporciona una justificación para las decisiones de arquitectura y diseño tomadas desde la idea conceptual hasta su implementación.

**HISTORIAL DE VERSIONES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fecha | Versión | Autor | Organización | Descripción |
| 27/05/19 | Primera | Edwin Saavedra | Innova Solutions | Apertura de proyecto y Acoplamiento de casos de uso |
| 28/05/19 | Segunda | Ányelo Gutiérrez | Innova Solutions | Documentación de Vista Lógica Física |
| 03/06/19 | Tercera | Edwin Saavedra y Ányelo Gutiérrez | Innova Solutions | Documentación de Vista Desarrollo y Procesos |

**APROBACIONES**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre y Apellido | Cargo | Organización | Fecha | Firma |
| Edwin Enrique Saavedra Parisaca | Líder | Innova Solutions | 01/04/19 | x |
| Ányelo Carlos Gutiérrez Choque | Arquitecto de software | Innova Solutions | 01/04/19 | x |
| Sergio Rolan Rondón Polanco | Programador | Innova Solutions | 01/04/19 |  |
| María Concepción Quijia Álvarez | Programador | Innova Solutions | 01/04/19 |  |
| Erick David Carpio Hachiri | Ingeniero de pruebas | Innova Solutions | 01/04/19 |  |
| José Luis Monroy Vilcahuaman | Ingeniero de pruebas | Innova Solutions | 01/04/19 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre y Apellido | Cargo | Organización | Fecha | Firma |
| Edwin Enrique Saavedra Parisaca | Líder | Innova Solutions | 02/06/19 | x |
| Anyelo Carlos Gutierrez Choque | Arquitecto de software | Innova Solutions | 02/06/19 | x |
| Sergio Rolan Rondón Polanco | Programador | Innova Solutions | 02/06/19 | X |
| María Concepción Quijia Álvarez | Programador | Innova Solutions | 02/06/19 | X |
| Erick David Carpio Hachiri | Ingeniero de pruebas | Innova Solutions | 02/06/19 |  |
| Jose Luis Monroy Vilcahuaman | Ingeniero de pruebas | Innova Solutions | 02/06/19 |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre y Apellido | Cargo | Organización | Fecha | Firma |
| Edwin Enrique Saavedra Parisaca | Líder | Innova Solutions | 08/06/19 | x |
| Anyelo Carlos Gutierrez Choque | Arquitecto de software | Innova Solutions | 08/06/19 | x |
| Sergio Rolan Rondón Polanco | Programador | Innova Solutions | 08/06/19 | X |
| María Concepción Quijia Álvarez | Programador | Innova Solutions | 08/06/19 | X |
| Erick David Carpio Hachiri | Ingeniero de pruebas | Innova Solutions | 08/06/19 | X |
| Jose Luis Monroy Vilcahuaman | Ingeniero de pruebas | Innova Solutions | 08/06/19 | X |

**1.1.** **Propósito:**

El Documento de Arquitectura de Software (SAD) proporciona una visión general de la arquitectura del Sistema de la Tienda Virtual. Presenta una serie de vistas arquitectónicas diferentes para representar los diferentes aspectos del sistema.

Para representar el software con la mayor precisión posible, la estructura de este documento se basa en la vista de arquitectura "4 + 1" de Philippe Kruchten

**1.2.** **Alcance:**

Este documento describe los diversos aspectos del diseño del sistema que se consideran arquitectónicamente significativos. Estos elementos y comportamientos son fundamentales para guiar la construcción del sistema y para comprender este proyecto como un todo.

## **1.3.** **Referencias:**

Concept: SAD,<http://medbiq.org/std_specs/techguidelines/softwarearchitecture.pdf>

The “4+1” view model of software architecture, Philippe Kruchten, November 1995,<http://www3.software.ibm.com/ibmdl/pub/software/rational/web/whitepapers/2003/Pbk4p1.pdf>

**2. Representación arquitectónica:**

Este documento detalla la arquitectura utilizando las vistas definidas en el modelo “4 + 1” . Las vistas utilizadas para documentar el sistema son:

**2.1 Vista de caso de uso**

* **Audiencia:** Todas las partes interesadas del sistema, incluidos los usuarios finales.
* **Área:** Describe el conjunto de escenarios y / o casos de uso que representan alguna funcionalidad importante y central del sistema. Describe los actores y los casos de uso del sistema, esta vista presenta las necesidades del usuario y se elabora más a nivel de diseño para describir flujos discretos y restricciones con más detalle.
* **Artefactos relacionados:** Modelo de casos de uso, documentos de casos de uso

**2.2 Vista lógica:**

* **Audiencia:** Diseñadores.
* **Área:** Requisitos funcionales: describe el modelo de objeto del diseño. También describe las realizaciones de casos de uso más importantes y los requisitos comerciales del sistema.
* **Artefactos relacionados:** Modelo de diseño

**2.3 Vista de Proceso**

* **Audiencia:** Diseñadores y Partes interesadas
* **Área:** comprende requisitos funcionales y no funcionales como rendimiento, escalabilidad y disponibilidad
* **Artefactos relacionados:** Modelo de proceso

**2.4 Vista de despliegue:**

* **Audiencia:** Gestores de despliegue.
* **Área:** Topología: describe la asignación del software al hardware y muestra los aspectos distribuidos del sistema. Describe las posibles estructuras de implementación, al incluir escenarios de implementación conocidos y anticipados en la arquitectura, permitimos a los implementadores realizar ciertas suposiciones sobre el rendimiento de la red, la interacción del sistema, etc.
* **Artefactos relacionados**: Modelo de implementación.

**2.5 Vista Física:**

* **Audiencia:** Usuarios Finales Diseñadores, Desarrolladores
* **Área:** Distribución de los componentes entre los equipos que integran la solucion,incluyendo los servicios.Se mapea la vista logica en el software y hardware
* **Artefactos relacionados:**Modelo Físico del Sistema

**3. Vista de Casos de Uso:**

El Diagrama debe seguir la notación para casos de uso establecida por UML, incluyendo los elementos del modelo de casos de uso, es decir:

* Actores.
* Casos de Uso.
* Relaciones.

Las relaciones de Actores con casos de usos se denominan “Asociaciones”.

Las relaciones entre casos de uso se denominan “Generalizaciones” y pueden ser de dos tipos, de uso (Uses) o de herencia (Extends).

Según el número de casos de uso se puede usar un diagrama, o varios según los módulos o funcionalidad.

**3.1 Actores:**

Un actor es cualquier entidad externa al sistema modelado que interactúa con él.

No necesariamente coincide con los usuarios, pues un mismo usuario puede desempeñar distintos roles que correspondan con varios actores. Además, un mismo actor puede desempeñar varios papeles según el caso de uso con que interactúa.

Para cada uno de los actores involucrados en el documento y representados en el diagrama, debe completarse la siguiente ficha. Si existe más de un actor, se copia el título (Nombre del Actor) y la ficha tantas veces sea necesario.

* **Actor 1:** Usuario
* **Actor 2:** Sistema Paradise

**3.2 Relaciones:**

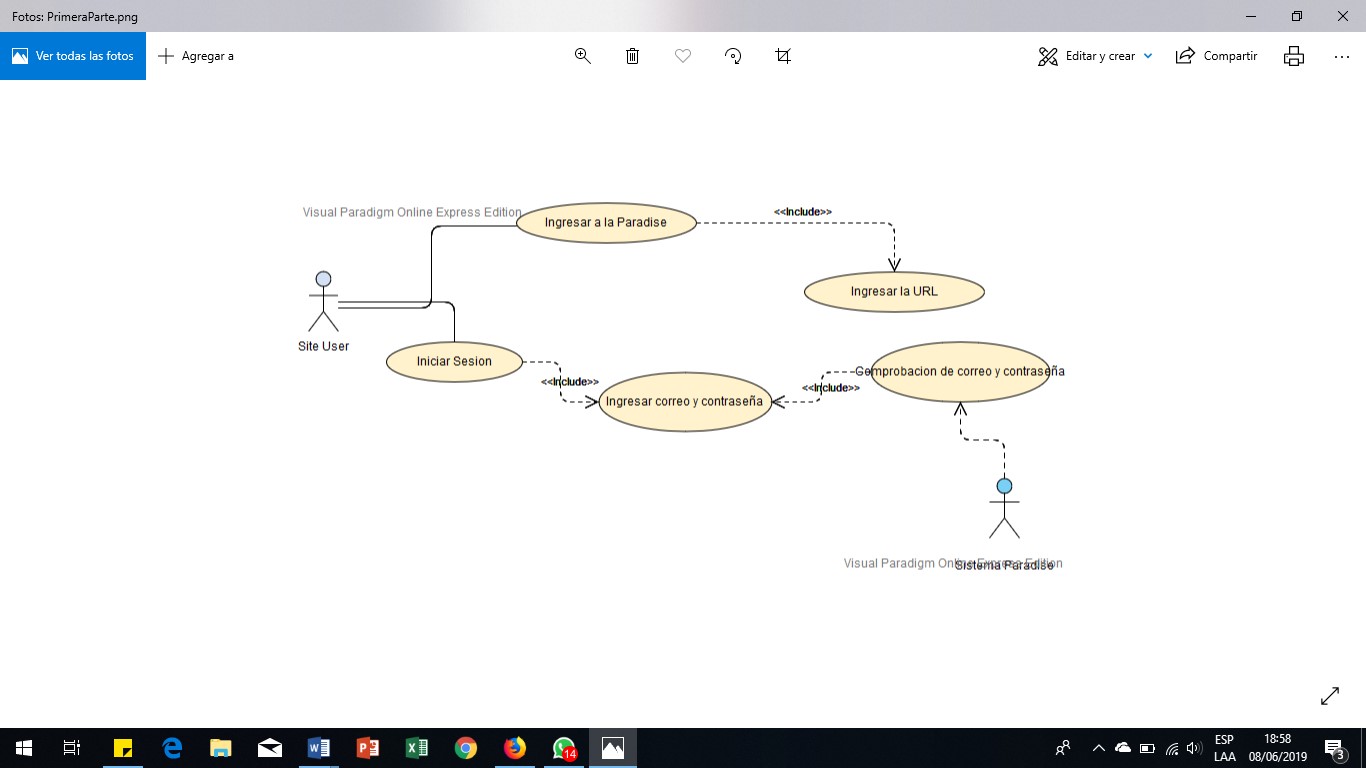
* **Comunicación**: Relación (asociación) entre un actor y un caso de uso. El estereotipo de la relación de comunicación es: <<communicate>> aunque generalmente no se estipula ningún nombre
* **Extensión:** Un caso de uso distinto maneja las excepciones del caso de uso básico
* **Inclusión:** Un caso de uso base incorpora explícitamente el comportamiento de otro en algún lugar de su secuencia. La relación de inclusión sirve para enriquecer un caso de uso con otro y compartir una funcionalidad común entre varios casos de uso, también puede utilizarse para estructurar un caso de uso describiendo sus subfunciones.
* **Especialización y generalización de los casos de uso:** Un caso de uso (subcaso) hereda el comportamiento y significado de otro, es decir las relaciones de comunicación, inclusión y extensión del super-caso de uso.

**3.3 Flujos de Casos de Uso:**

* **Ingresar a Paradise:**

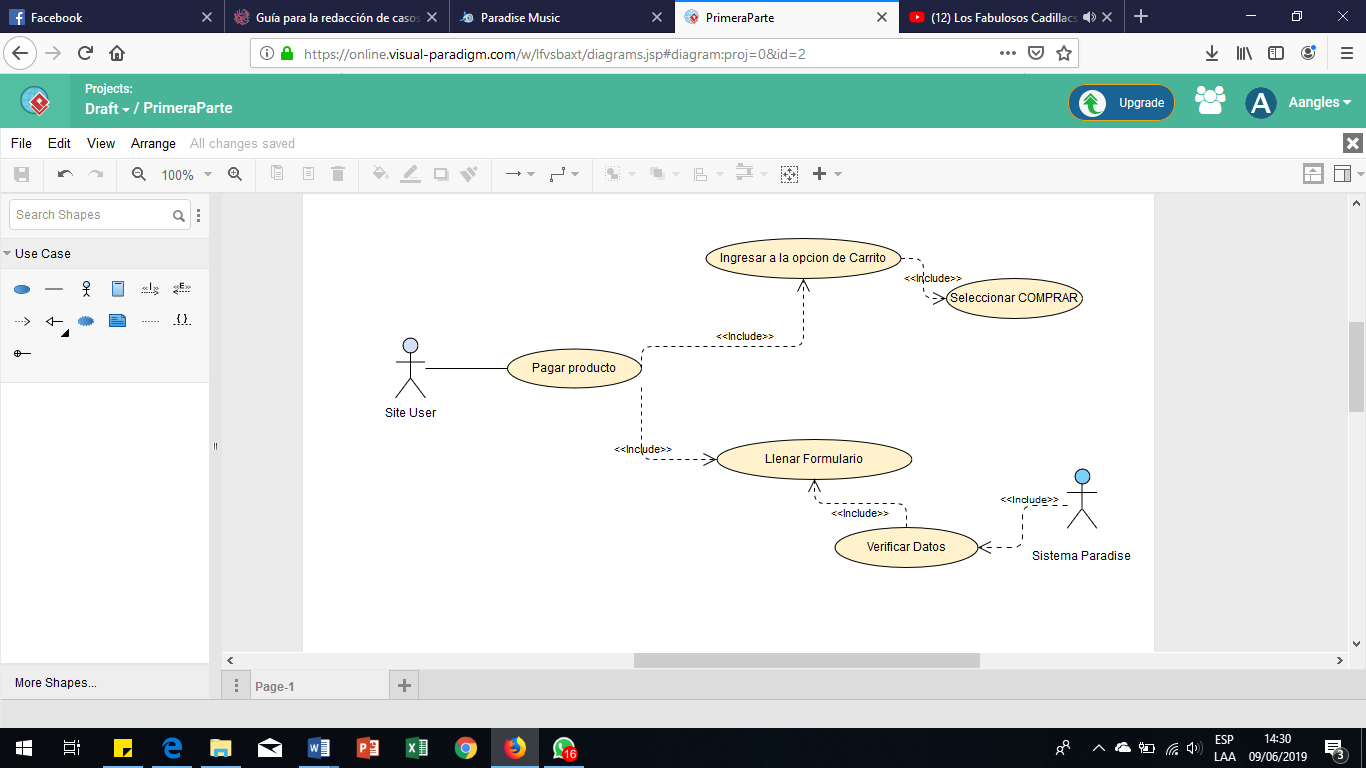
En este primer caso de uso es la primera acción que hace el usuario para poder conocer la Tienda Paradise.

Como lo podemos notar no es necesario, iniciar sesión mediante nuestro correo electrónico para poder visualizar o adquirir algún producto de dicha tienda.

****

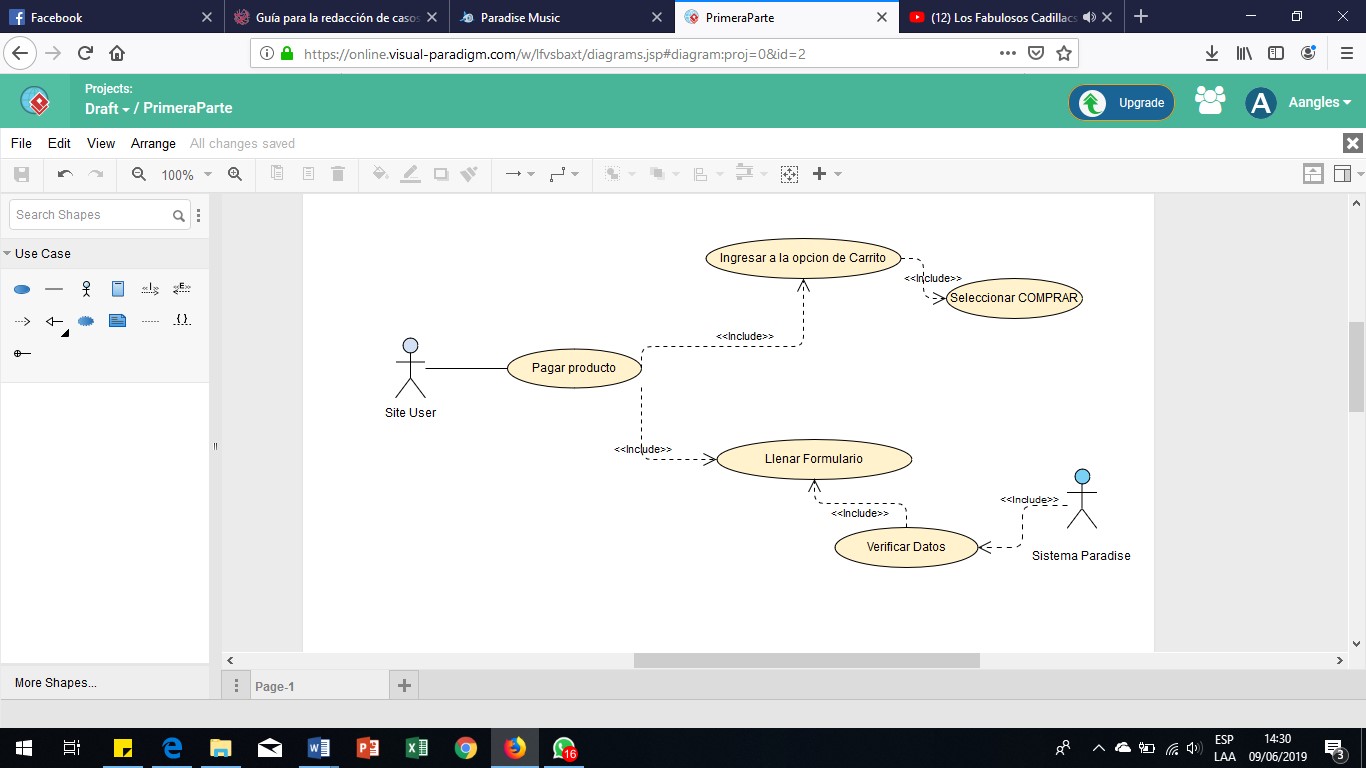
* **Escoger Producto:**

El objetivo de esta tienda es que el usuario compre un producto Paradise, por lo cual para poder realizar esta acción el usuario debe primero navegar por la página Paradise y escoger el producto deseado, o en todo caso ir de frente a la opción Productos escoger la categoría que desee y agregarlo al carrito

****

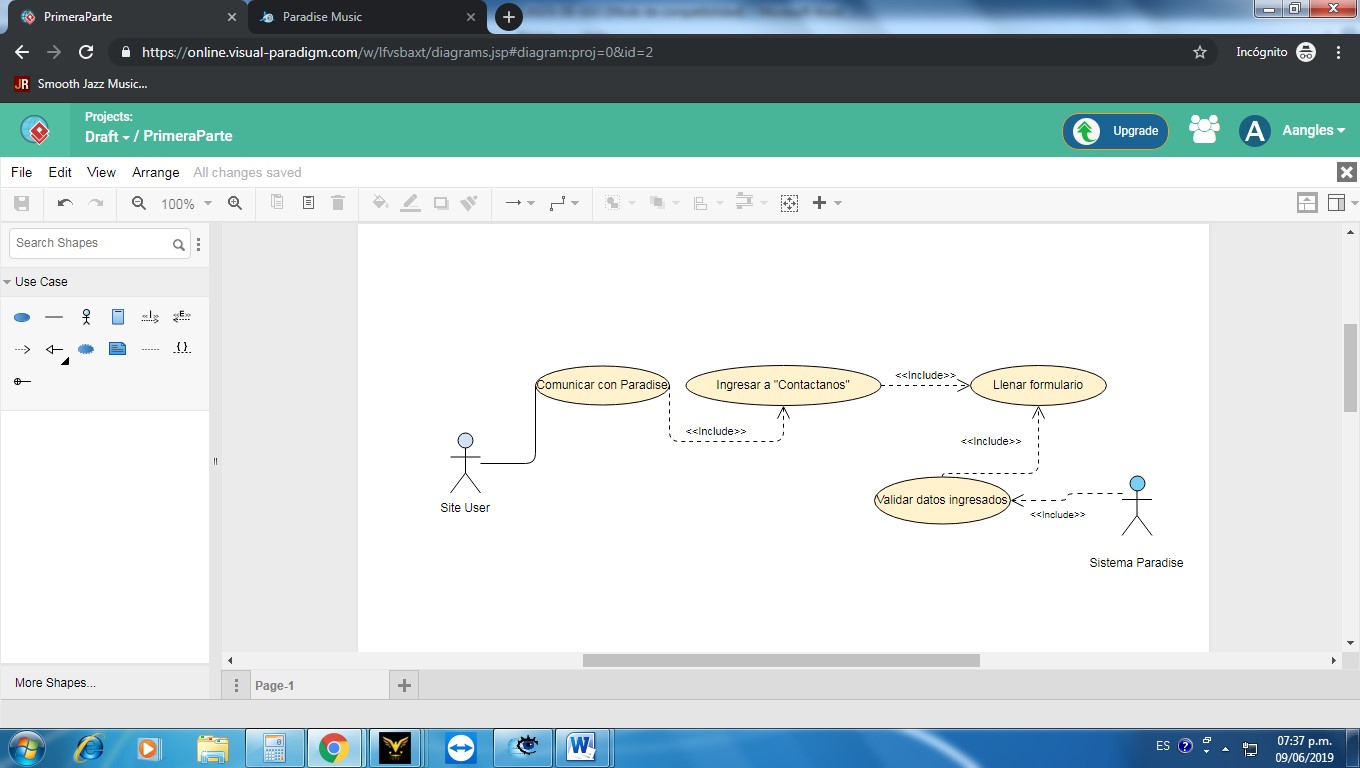
* **Pagar Producto:**

Luego de adquirir el producto que el usuario desea, para poder completar la compra es necesario cancelar el producto. Para pagar el producto se ingresa al carrito y de ahí se selecciona COMPRAR y finalmente se llena un formulario sobre los datos del usuario y los datos requeridos sobre la tarjeta con la que se va a cancelar dicho pedido.

****

* **Comunicar con Paradise:**

Paradise al ser consciente que los usuarios van a querer hacer preguntas o consultas personalmente se ha creado la Opción “Contáctanos” donde el usuario puede ingresar a esta opción y dejar su mensaje.

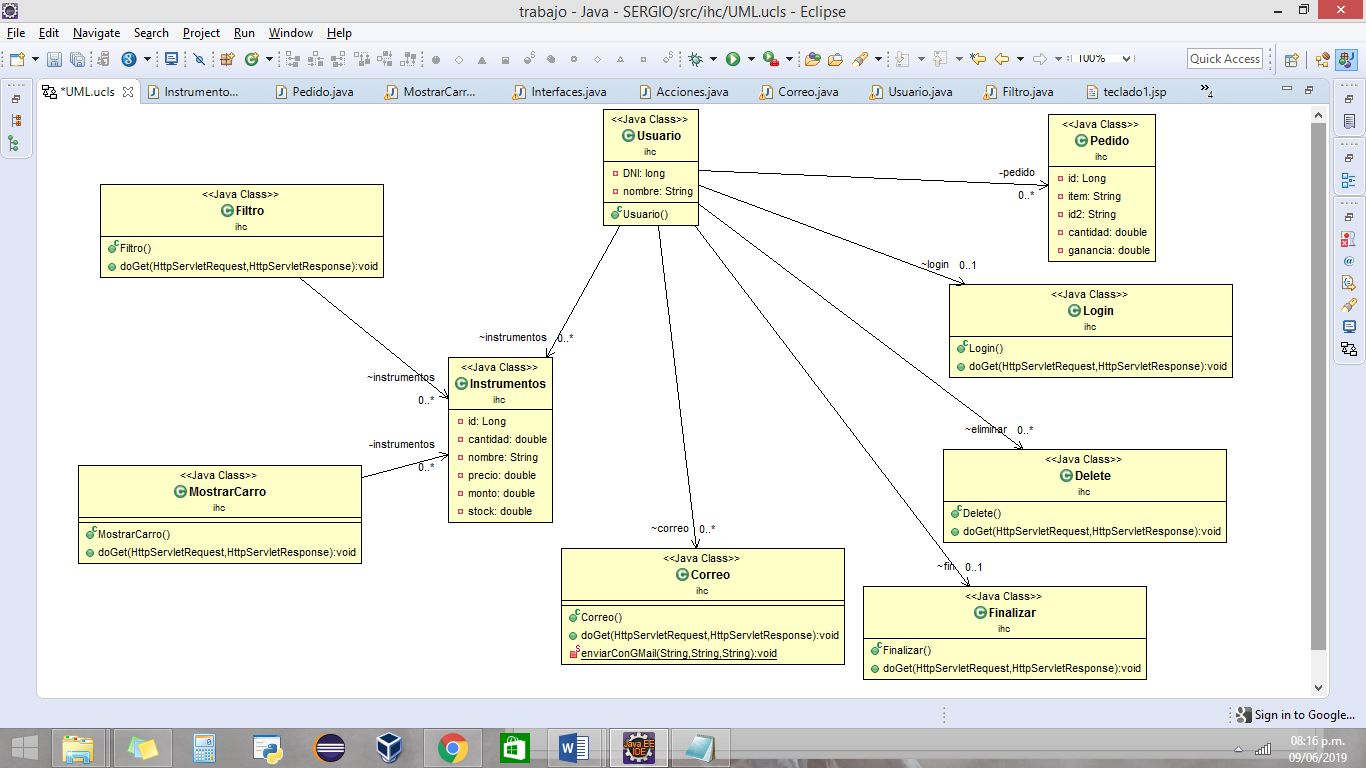
****

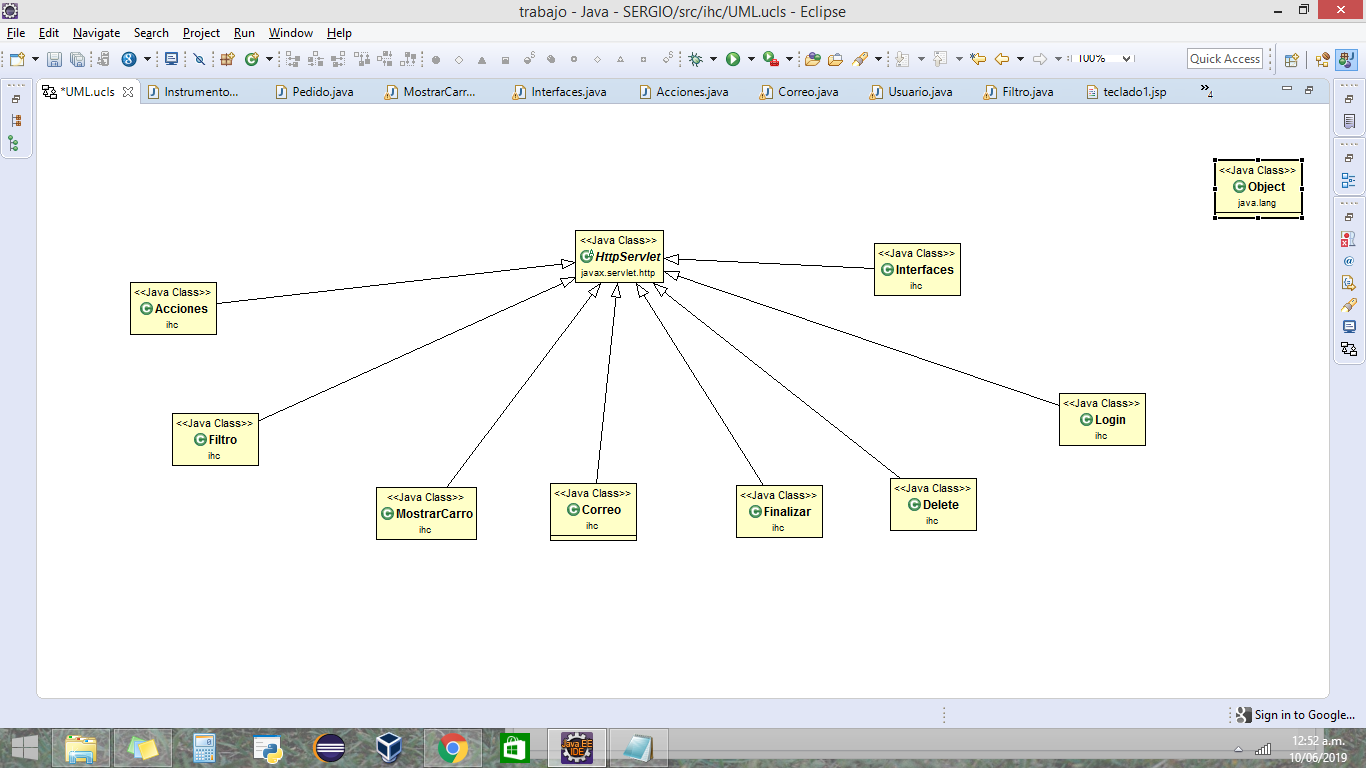
**4. Vista Lógica:**

La aplicación web de tienda virtual es aplicación orientada objetos, por tanto se ha realizado las diversas relaciones de asociación respectivas entre clases implementadas(diagrama de clases UML).Este diagrama de clases fue creada por el arquitecto de software y que asu vez va a permitir al programador, saber qué clases debe crear y qué métodos debe implementar. si en caso hubiese que agregar una clase adicional, entonces se modifica el diseño y se implementa dicha clase.

La arquitectura lógica apoya principalmente los requisitos funcionales –lo que el sistema debe brindar en términos de servicios a sus usuarios. El sistema se descompone en una serie de abstracciones clave, tomadas

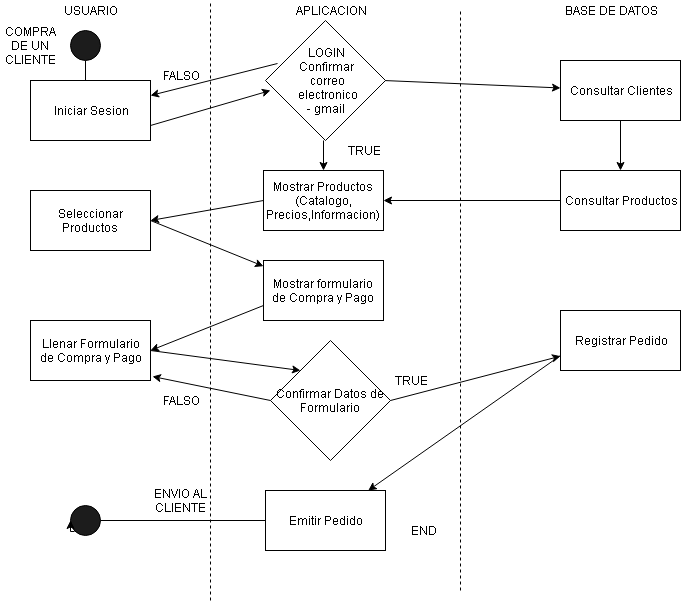
(principalmente) del dominio del problema en la forma de objetos o clases de objetos.

****

****

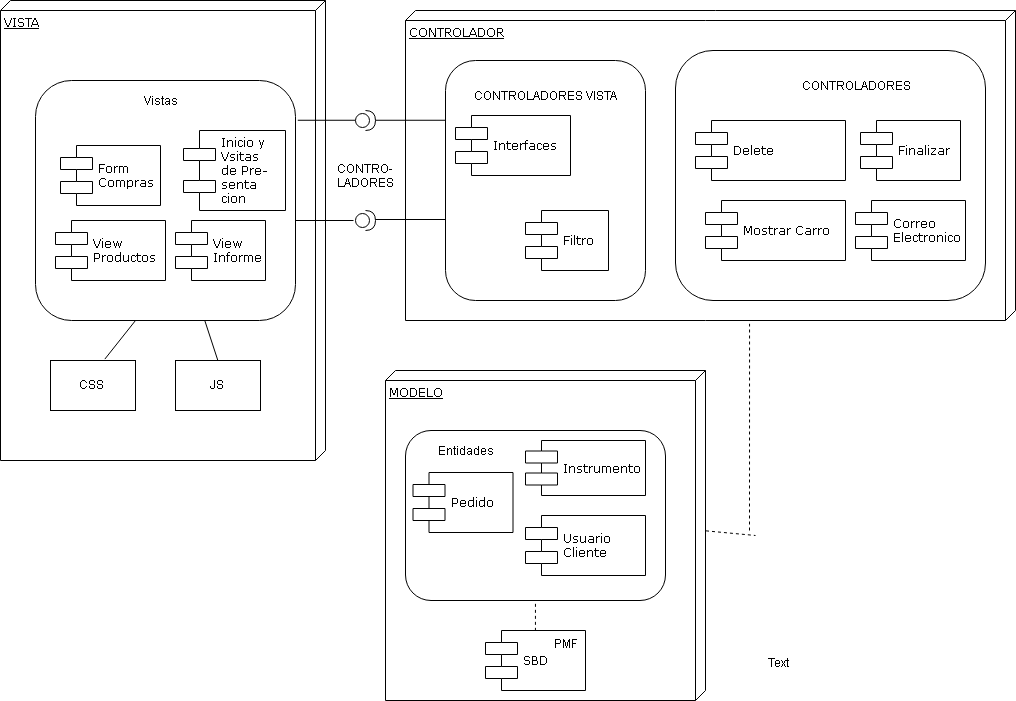
**5. Vista de Proceso:**

La vista de procesos muestra la perspectiva de una parte de la solución, también comprende requisitos funcionales y no funcionales como rendimiento, escalabilidad y disponibilidad .Los procesos son sus principales elementos y se ubican a diferentes niveles de abstracción.



**6. Vista de Despliegue o de Desarrollo:**

lLa vista de despliegue muestra el sistema desde la perspectiva del programador y se ocupa de la gestión de software ; o en otra palabras se va mostrar como esta dividido el sistema de software en componentes y las dependencias que hay entre esos componentes. A continuación se presenta la vista de despliegue, en ella se puede observar que es accedido por dos tipos de clientes desde un navegador web, estos son los usuarios comunes y los administradores



**7. Vista Física:**

Representa como están distribuidos los componentes entre los equipos que integran la solucion,incluyendo los servicios.Se mapea la vista logica en el software y hardware.La arquitectura física toma en cuenta primeramente los requisitos no funcionales del sistema tales como la disponibilidad, confiabilidad (tolerancia a fallas), performance (throughput), y escalabilidad.

