



Tabla de contenido

Plan de Desarrollo de Software	3
Introducción.....	3
Propósito	3
Alcance	3
VISTA GENERAL DEL PROYECTO	4
Objetivo I.....	4
Suposiciones y restricciones.....	4
Suposiciones:	4
Restricciones:	4
ENTREGABLES DEL PROYECTO	4
Fase 1	4
Fase 2	4
Fase 3	4
Evolución del plan del software	5
ARQUITECTURA EN CAPAS (2, 3 Y N)	5
Los principales beneficios del estilo de arquitectura basado en capas son	5
• Abstracción.....	5
• Aislamiento	5
• Rendimiento.....	5
• Mejoras en Pruebas	5
• Independencia.....	5
Información:.....	5
Logística de presentación.....	6
Tecnologías	13
Herramientas	14
Logística de Negocio.....	22
Presentación distribuida	22
Presentación remota	22
Capa de negocios.....	23
Lógica distribuida	24
Ventajas	24



Desventajas	24
Capa de datos: Back end	25
SISTEMAS DE INFORMACION	25
Administración de datos remota	25
Base de datos distribuida	26
ESTRUCTURA DE MICROSERVICIOS	27
Estructura del sistema	27
Estructura de sistema aplicándole arquitectura de microservicios	27
Estrategias de pruebas de software	29
PLANTILLAS	29
OBJETIVOS	31
VISION GENERAL	31
PRUEBAS DE UNIDAD	31
PRUEBA DE VALIDACION	32
PRUEBA DE SISTEMA	32
CICLO COMPLETO DE PRUEBAS	33
RECOMENDACIONES	34
CONCLUSION	35



Link que lo Direcciona a la Actividad Especifica en GITHUB:

<https://github.com/JonatanR97/PROYECTO-ANALISIS-DE-SISTEMAS-2/wiki/ANALISIS>

Plan de Desarrollo de Software

Introducción

El proyecto en general tiene como fin desarrollar un sistema de web.

Teniendo en cuenta que la realidad organizacional podría ser mejorada considerablemente, el objetivo es dar solución a aquellas tareas de mayor importancia y que soportan una gran cantidad de datos a ser controlados y de esta manera agilizar los procesos dentro de la institución.

Es por ello por lo que se ha considerado un diseño modular y en base a una metodología iterativa-incremental para implementar las tareas de administración y difusión de información de los campeonatos de la institución, pensando en que a futuro este sistema pueda continuar creciendo hasta abarcar a todas las funciones de la asociación nacional de futbol de la liga nacional.

Propósito

Crear un sistema que pueda controlar automáticamente las operaciones para una buena gestión de inventario y el excelente manejo de control, ya que será de gran ayuda para que los usuarios puedan gestionar digitalmente el entorno del sistema.

Alcance

El sistema debe incluir lo siguiente:

- De las ventas se requiere poder ver las ventas realizadas y ejecutar una nueva venta
- En un apartado de productos se solicita registrar un nuevo producto y poder ver los registros de los productos (también en este apartado se pueda eliminar, actualizar y editar)
- Se requiere registrar nuevos clientes, también poder ver los clientes que ya fueron registrados (en este apartado se debe poder editar y eliminar los clientes)
- Se solicita poder registrar proveedores, también que se pueda observar los proveedores que se encuentran en el sistema (se debe de poder editar y eliminar)
- Se debe de poder registrar usuarios, y de ello se debe poder ver los usuarios que se registraron (De ellos se debe de poder editar y eliminar usuarios)

Resumen



En estos tiempos la información se maneja de una forma rápida y confiable, por lo cual tener herramientas que ofrezcan estas dos cualidades se hacen más indispensables, es por esto que los sistemas web son cada día más necesarios para empresarios, comerciantes y para el público en general.

Como un aporte más presentamos una un sistema para gestión de inventario que le permita al público en general realizar sus pedidos desde su hogar o desde cualquier parte de la ciudad capital o cualquiera de los diferentes departamentos del país, teniendo como única exigencia que los usuarios se encuentren registrados en la base de datos para datos de las compras he información.

VISTA GENERAL DEL PROYECTO

Objetivo I

Desarrollar un sistema que satisfaga las necesidades al usuario y en el que podamos realizar un seguimiento de la correcta gestión de inventario con el fin de crear orden y facilitar la gestión de los usuarios que serán los encargados de ser beneficiados con dicho sistema.

Suposiciones y restricciones

Suposiciones:

- Capacitar a los administradores para el manejo correcto del sistema y que su uso sea lo más descifrable para el buen manejo del sistema.
- La información almacenada del sistema será totalmente resguardada por el sistema para poder esquivar el rapto de información y la huida de datos.
- El sistema será proceso eficiente en la gestión para la liga nacional.

Restricciones:

- Solo los administradores pueden realizar cambios.
- Solo los administradores pueden acceder a la información de la base de datos.

ENTREGABLES DEL PROYECTO

Fase 1

- Calendarios de trabajo
- Manual de usuario
- Documento del software
- Presupuesto al Usuario

Fase 2

- Recopilación de información.
- Especificaciones de los diagramas.

Fase 3

- Presentación de los nuevos avances del software.
- Desarrollo de la base de datos.



- Verificación del sistema en el caso de que se presente fallas y modificaciones.

Evolución del plan del software

El plan del desarrollo del sistema se revisará semanalmente testeando y verificando la integridad de la base de datos y ver si el software necesita cambios y modificaciones.

Se verifica la integridad del proyecto para poder refinar detalles del ya mencionado programa.

ARQUITECTURA EN CAPAS (2, 3 Y N)

Los principales beneficios del estilo de arquitectura basado en capas son:

- **Abstracción.** Las capas permiten cambios que se realicen en un nivel abstracto. Usted puede incrementar o disminuir el nivel de abstracción usado en cada capa de la “pila” jerárquica.
- **Aislamiento.** El estilo de arquitectura de capas permite asilar los cambios en tecnologías a ciertas capas para reducir el impacto en el sistema total.
- **Rendimiento.** Distribuir las capas entre múltiples sistemas (físicos) puede incrementar la escalabilidad, la tolerancia a fallos y el rendimiento.
- **Mejoras en Pruebas.** La capacidad de realizar pruebas se beneficia de tener unas interfaces bien definidas para cada capa, así como de la habilidad para cambiar a diferentes implementaciones de las interfaces de cada capa.
- **Independencia.** El estilo de arquitectura basado en capas el requerimiento de considerar el hardware y los problemas de instalación, así como las dependencias de interfaces externas.

Información:

- De las ventas se requiere poder ver las ventas realizadas y ejecutar una nueva venta
- En un apartado de productos se solicita registrar un nuevo producto y poder ver los registros de los productos (también en este apartado se pueda eliminar, actualizar y editar)



- Se requiere registrar nuevos clientes, también poder ver los clientes que ya fueron registrados (en este apartado se debe poder editar y eliminar los clientes)
- Se solicita poder registrar proveedores, también que se pueda observar los proveedores que se encuentran en el sistema (se debe de poder editar y eliminar)
- Se debe de poder registrar usuarios, y de ello se debe poder ver los usuarios que se registraron (De ellos se debe de poder editar y eliminar usuarios)

Logística de presentación

Aquí se muestro la presentación logística del software a desarrollar, es decir todas las formas, del sistema o menús de dialogo, con los cuales interactúa el usuario.

Los servicios de presentación proporcionan la interfaz necesaria para presentar información y reunir datos.

Los servicios de presentación son identificados con la interfaz de usuario, residen en un programa ejecutable localizado en la estación de trabajo del usuario.

Aun así, existen oportunidades para identificar servicios que residen en componentes separados.

Diseño UI

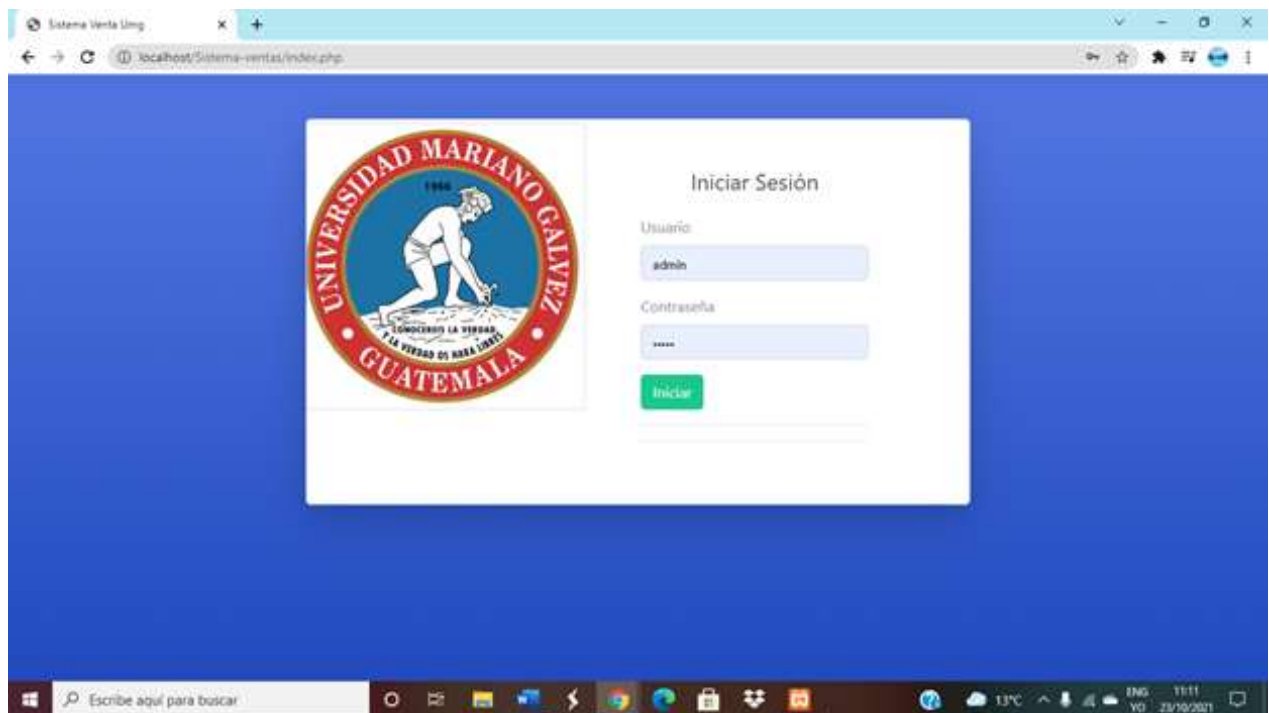
El Diseño que se aplicará en el sistema será UI ya que las interfaces de la aplicación web, se estará ocupando css, y Bootstrap, para darle mejor vista a la aplicación y que el diseño de interfaz sea más dinámico y creativo para el usuario, y que se vea profesional.



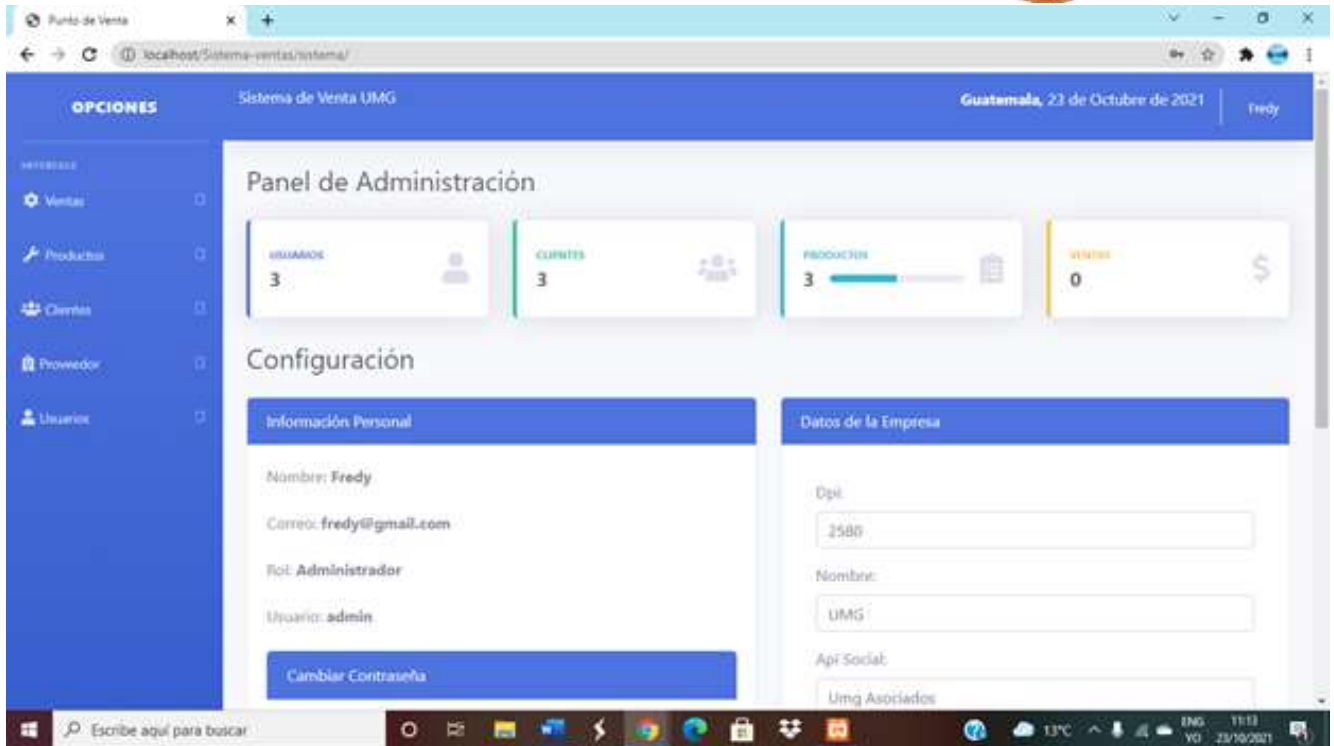


Muestras del Diseño implementado en el sistema.

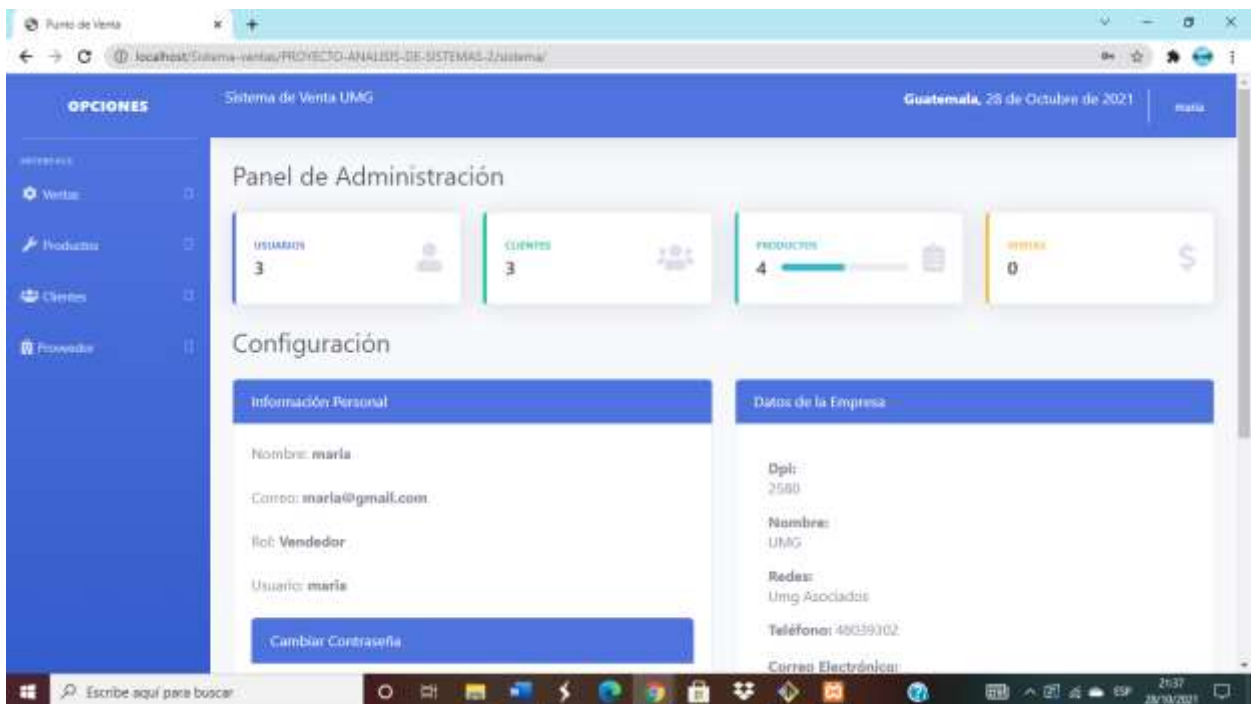
Diseño de Login



Diseño de vista para administrador



Diseño de vista para vendedor



Diseño para vista de ventas



Punto de Venta

localhost/Sistema-ventas/sistema/nueva_venta.php

Datos del Cliente

[Nuevo Cliente](#)

Dpi: Nombre: Teléfono:

Dirección:

Datos Venta

VENDEDOR: FREDY

Código	Des.	Stock	Cantidad	Precio	Precio Total	Acciones
<input type="text"/>	-	-	<input type="text" value="0"/>	0.00	0.00	

Código	Descripción	Cantidad	Precio	Precio Total	Acciones
--------	-------------	----------	--------	--------------	----------

Copyright © Fredy Lopez 2021

Diseño para ver ventas realizadas

Punto de Venta

localhost/Sistema-ventas/sistema/ventas.php

OPCIONES Sistema de Venta UMG Guatemala, 23 de Octubre de 2021 fredy

Registros

Mostrar 10 registros Buscar

Id	Fecha	Total	Acciones
2	2021-10-21 22:17:20	2000.00	Ver

Mostrando 1 a 1 de 1 registros

Anterior **1** Siguiente

Copyright © Fredy Lopez 2021



Diseño para el ingreso de producto

Panel de Registro

Registrar Producto

Proveedor:

Producto:

Precio:

Cantidad:

Diseño para la vista de productos registrados

Productos

Mostrar 10 registros

Buscar:

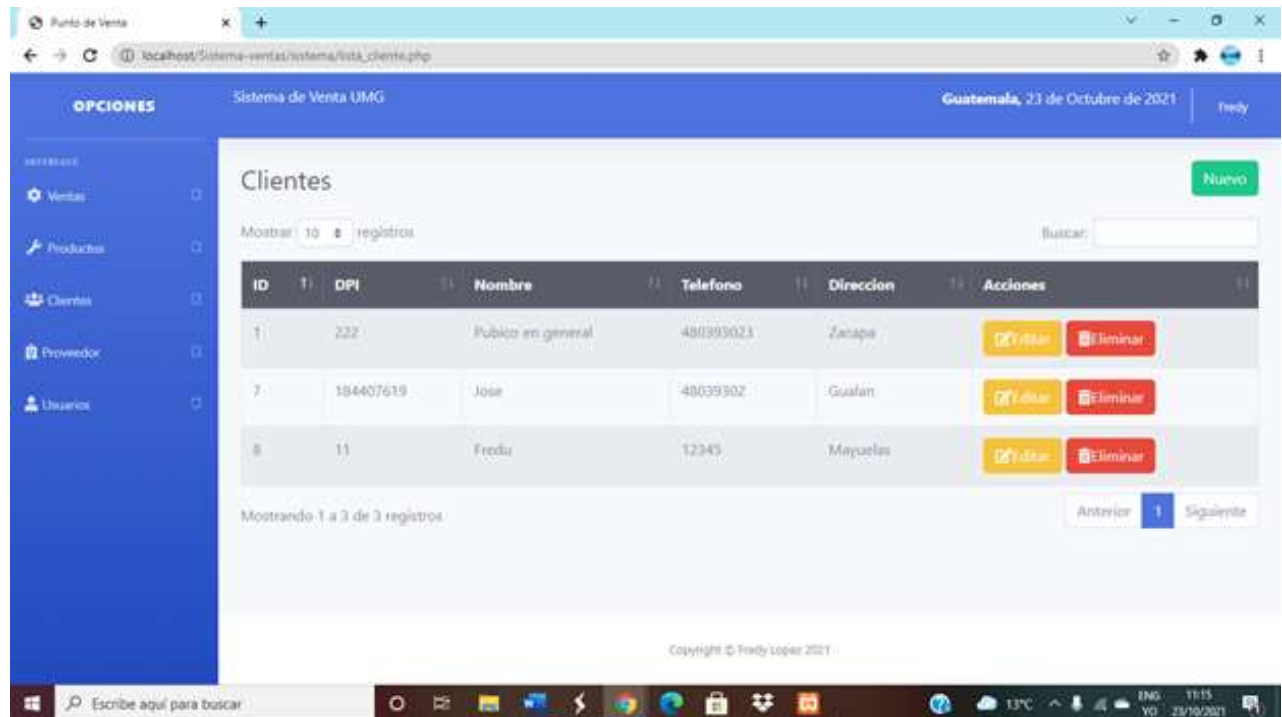
ID	Producto	Precio	Stock	Acciones
1	Laptop	2000.00	196	<input type="button" value="Actualizar"/> <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
6	Impresora Samsung	800.00	0	<input type="button" value="Actualizar"/> <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>
8	Audifonos	1000.00	10	<input type="button" value="Actualizar"/> <input type="button" value="Editar"/> <input type="button" value="Eliminar"/>

Mostrando 1 a 3 de 3 registros

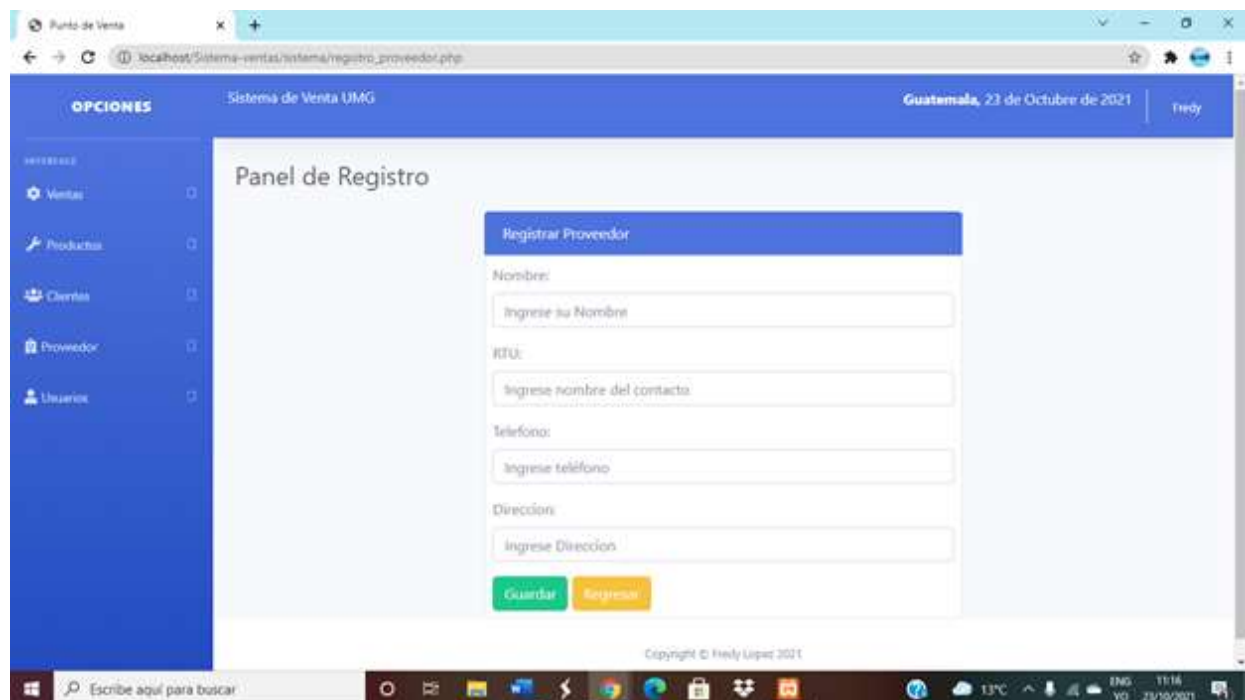
Anterior Siguiente



Diseño de vista para el registro de clientes

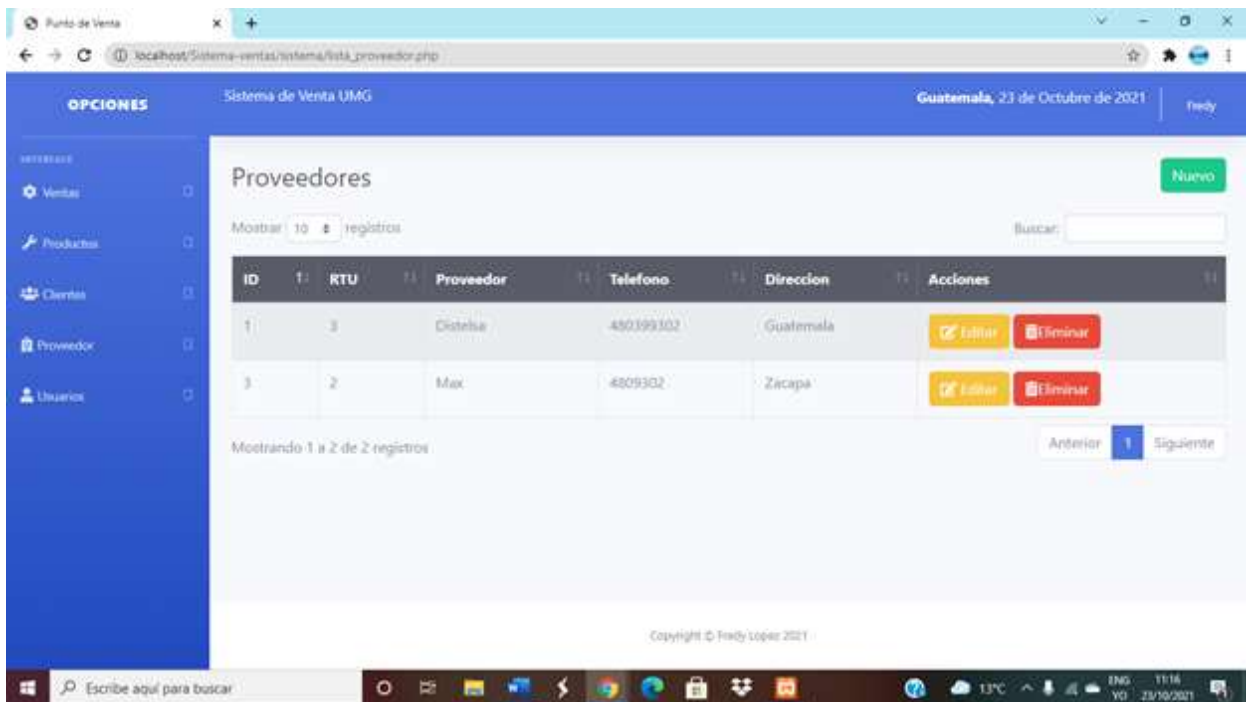


Diseño para registrar proveedores

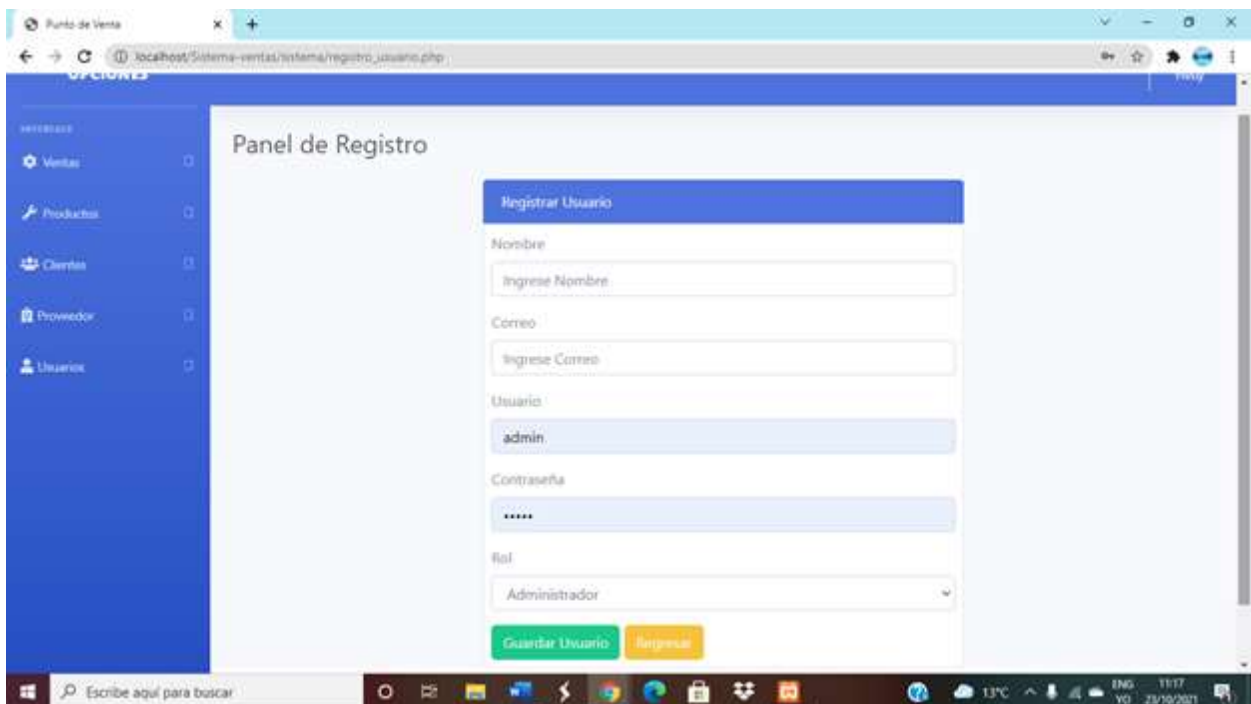




Diseño para la vista de proveedores registrados

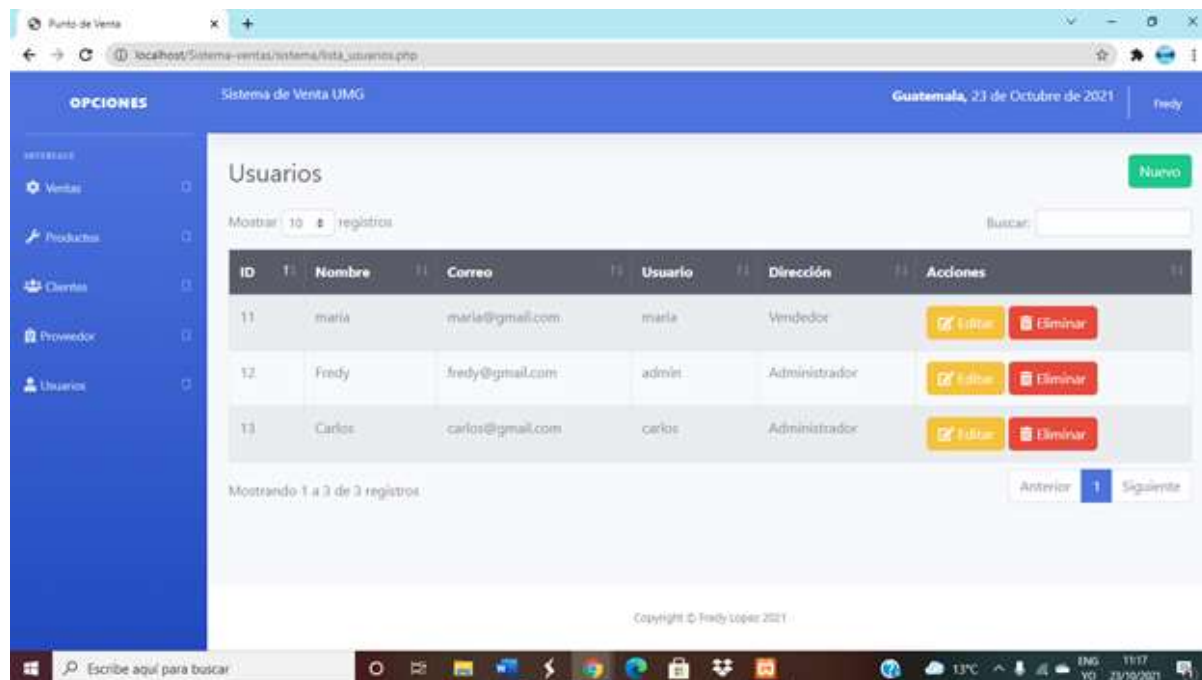


Diseño para registro de usuario





Diseño de vista para usuarios



Tecnologías

En esta sección se definen brevemente las tecnologías a utilizar durante el proceso de desarrollo del proyecto.

PHP:

Es la sigla que representa Hypertext Pre-Processor. Es un lenguaje gratuito y multiplataforma para programar script del lado del servidor, que se incrustan en el código

HTML:

Es la sigla que representa HyperText Markup Language. Es un lenguaje de marcado que permite la elaboración de páginas webs.

JavaScript:

Es un lenguaje de programación interpretado, se utiliza principalmente en el cliente, implementado como parte de un navegador web permitiendo mejoras en la interfaz de usuario y páginas web dinámicas.

CSS:

Es la sigla que representa Cascading Style Sheet. Es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos



definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos, su presentación y es imprescindible para crear páginas web complejas.

Herramientas

En esta sección se definen brevemente las herramientas a utilizar durante el proceso de desarrollo del proyecto.

Visual Studio Code:

es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS. Incluye soporte para la depuración, control integrado de Git, resaltado de sintaxis, finalización inteligente de código, fragmentos y refactorización de código.

MySQL Workbench:

Es un software gratuito distribuido por Oracle Corporation.

Permite la gestión e implementación de un servidor local de base de datos MySQL que servirá para la fase de pruebas.

MySQL:

Es sistema de gestión de base de datos relacional gratuito que es distribuido y mantenido por Oracle.

PhpMyAdmin:

Es una plataforma gratuita y multiplataforma para la gestión de base de datos MySQL. Será utilizado para la implementación de base de datos remota.

Bootstrap:

Es una biblioteca multiplataforma o conjunto de herramientas de código abierto para diseño de sitios y aplicaciones web.

Apache Jmeter:

JMeter es una herramienta de testing cuyas funcionalidades se pueden resumir en tres:

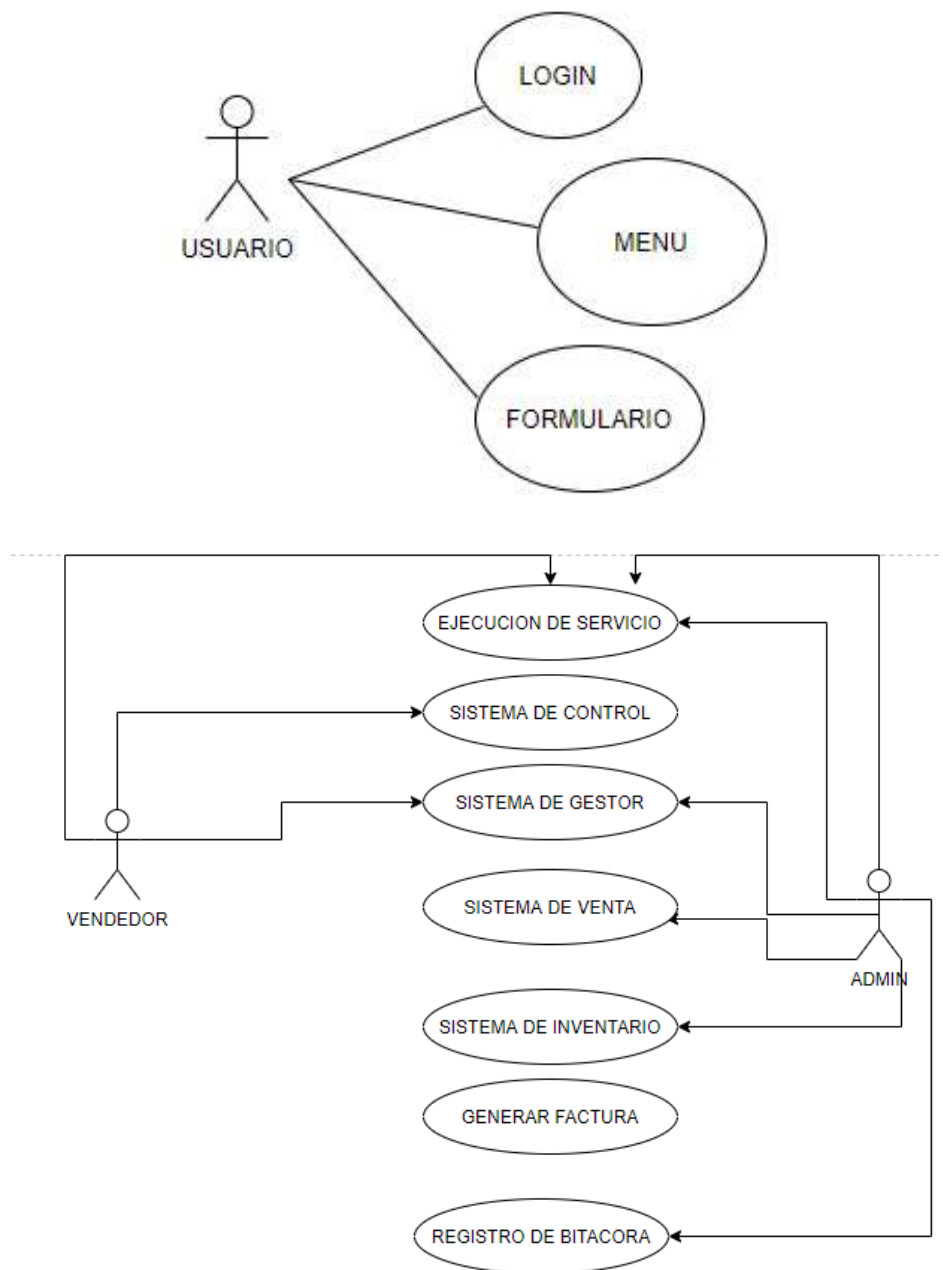
- Diseñar un testplan, esto es, generar un fichero. jmx
- Ejecutar un testplan
- Ver de distintas formas los resultados de la ejecución de un testplan (vía listeners)

Para diseñar un testplan, JMeter dispone de una interfaz GUI a modo de diseñador, en la que el tester puede ir agregando componentes de manera visual, y ejecutar los componentes agregados, viendo el resultado. Una vez finalizado el diseño del testplan, la herramienta permite grabar este como un fichero. jmx.



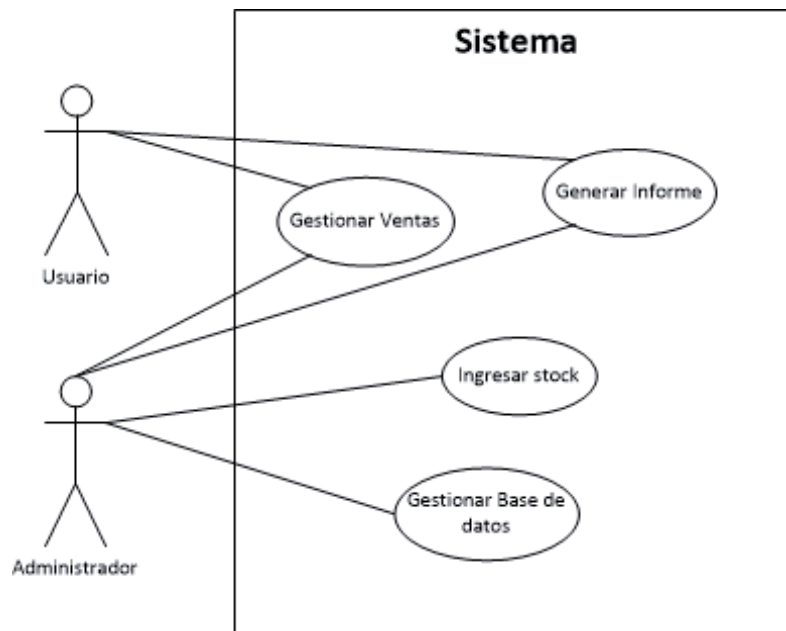
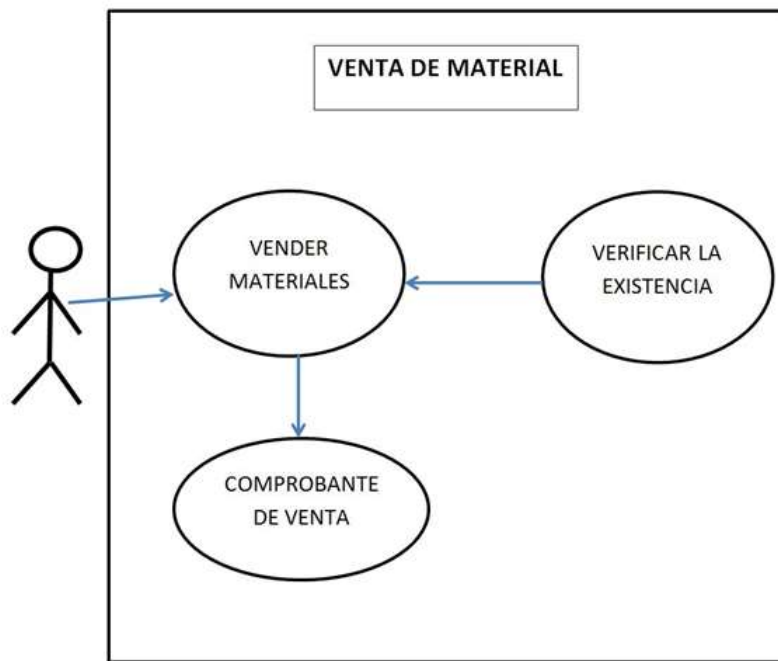
La capa de servicios de presentación es responsable de:

- Obtener información de usuario.
- Enviar la información del usuario a los servicios de negocios para su procesamiento
- Recibir los resultados del procesamiento de los servicios de negocios
- Presentar estos resultados al usuario



Actores primarios:

- Administrador, Vendedor
- Registrarse Venta
- Registro Productos
- Facturación
- Resultados



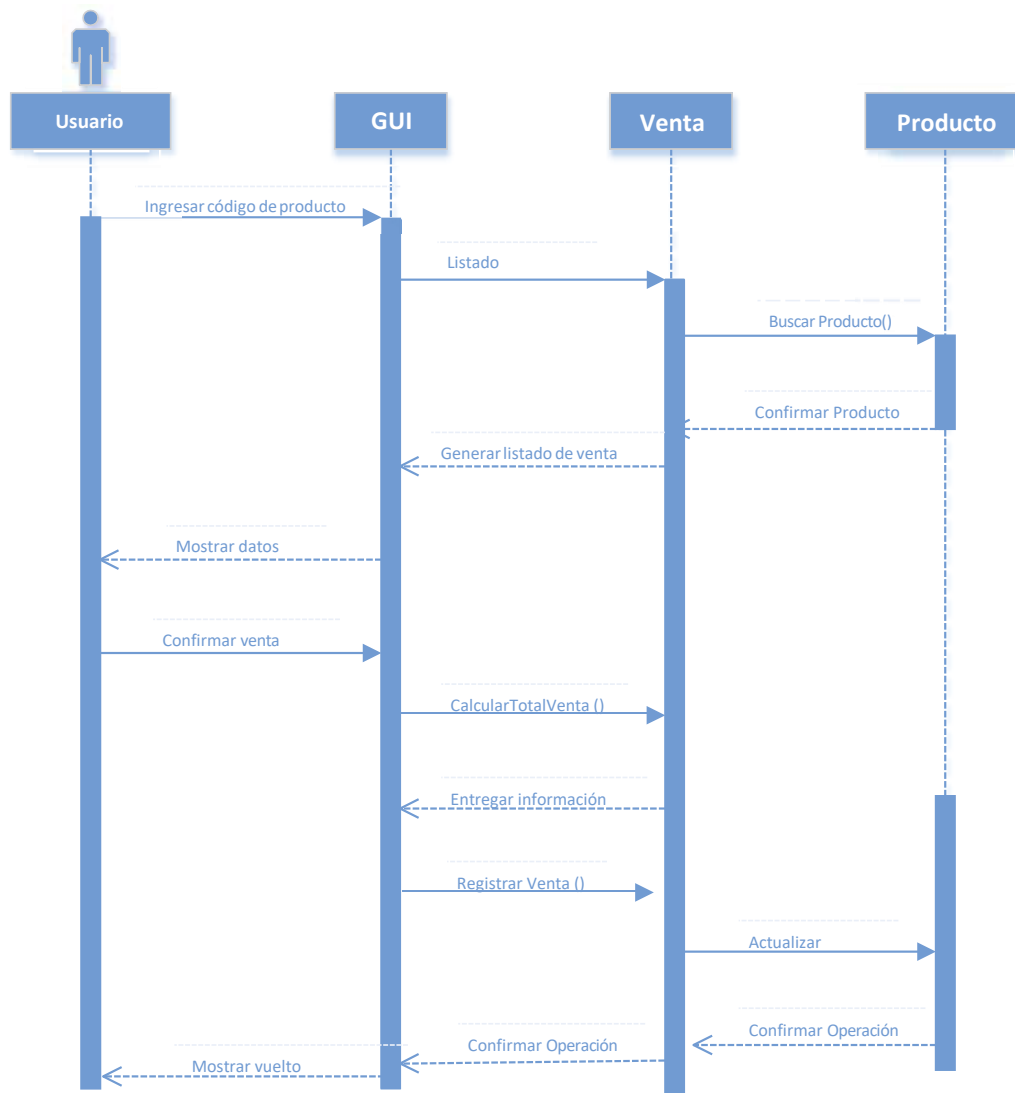


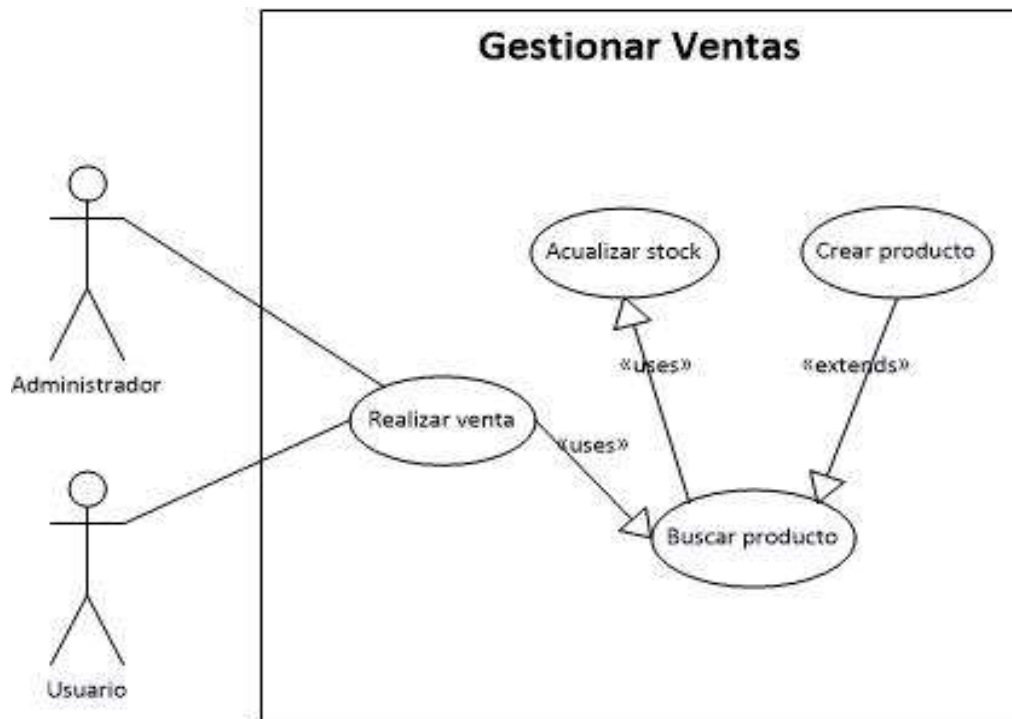
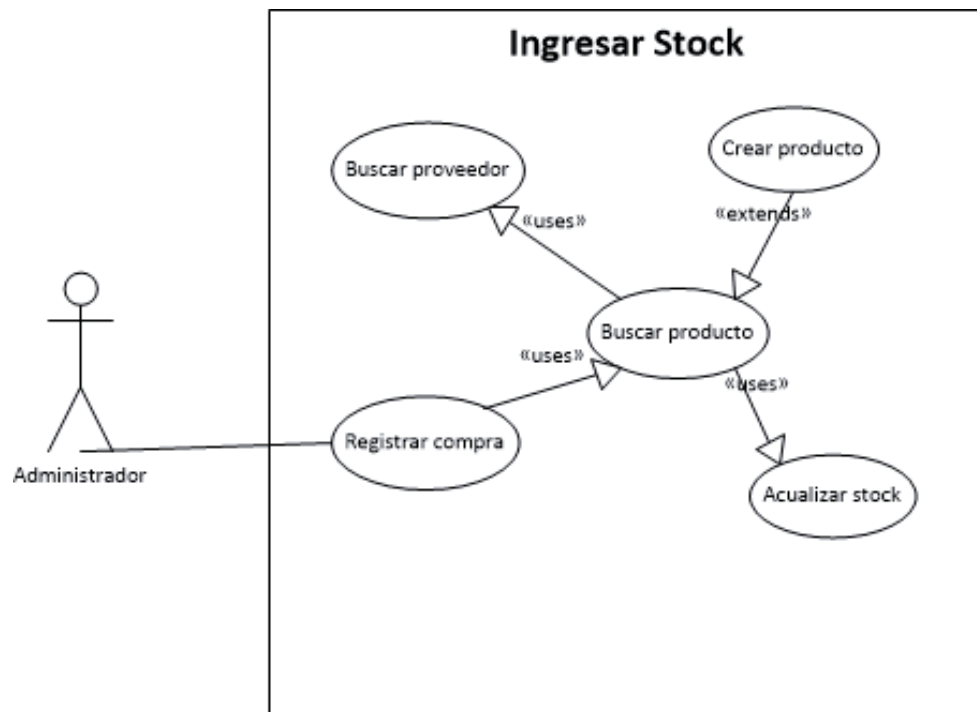
Caso de Uso	Realizar Venta
Actor Principal	Administrador Usuario
Participantes e Intereses	Administrador: Desea realizar una venta. Usuario: Desea Realizar una venta.
Precondiciones	1. El usuario debe estar logueado para realizar la venta.
Postcondiciones	Al finalizar la venta se tendrá un listado de los productos y sus respectivos precios más el precio total.
Escenario principal	1. El Usuario ingresa al sistema 2. El Usuario selecciona la opción realizar venta. 3. El Usuario ingresa los productos que se venderán a través de un lector de código de barra.
Extensiones	1.1 El Usuario ingresa incorrectamente su contraseña. 1.1.1 El sistema le indica al usuario que su ingreso no ha sido exitoso. 1.1.2. El sistema manda un mensaje pidiendo que lo intente de nuevo. 3.1 El Usuario ingresa incorrectamente su contraseña. 3.1.1 El sistema le indica al usuario que su ingreso no ha sido exitoso. 3.1.2. El sistema manda un mensaje pidiendo que lo intente de nuevo. 3.2 El usuario realiza correctamente la venta. 3.2.1 Se muestra por pantalla los productos y sus precios más el total.
Requisitos Especiales	No hay requisitos especiales.
Frecuencia de Ocurrencia	Alta

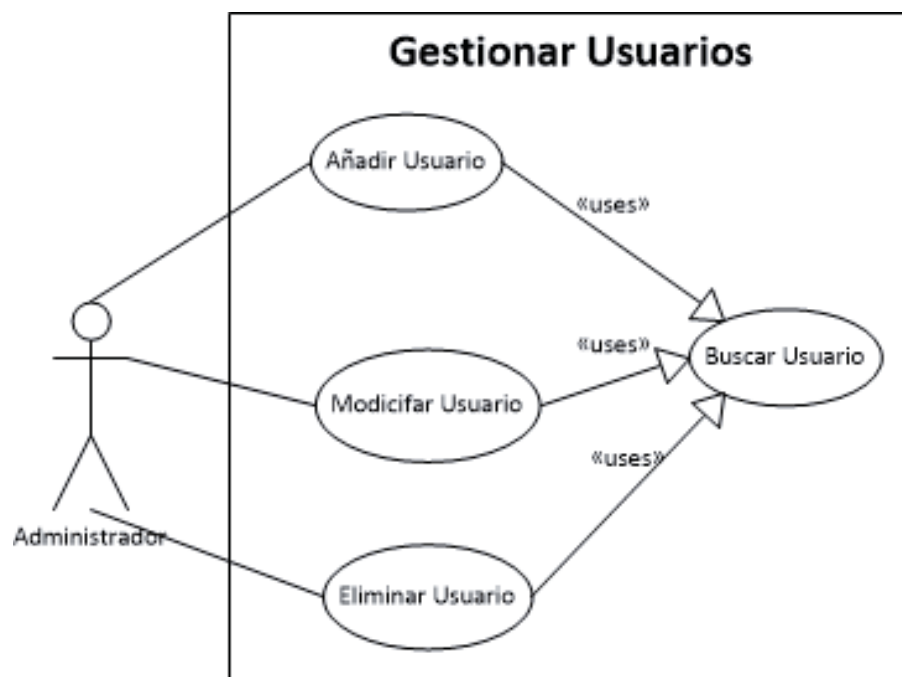
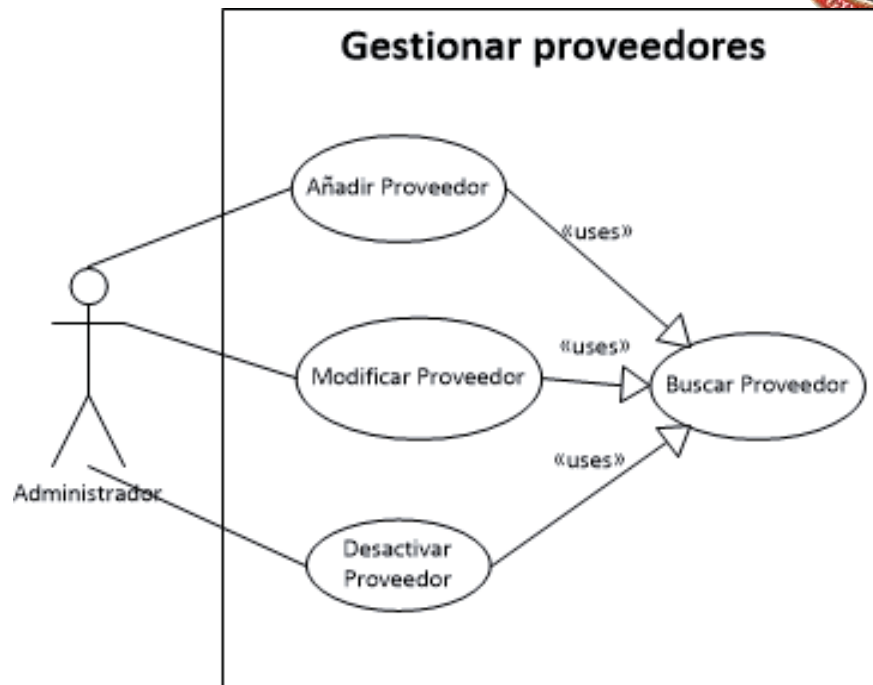
Caso de Uso	Registrar Venta
Actor Principal	Administrador
Participantes e Intereses	Administrador: Desea registrar Venta.
Precondiciones	1. El usuario debe tener privilegios de administrador. 2. El usuario debe haber ingresado en la opción de ingresar stock.
Postcondiciones	Por pantalla se indica que el ingreso de stock fue exitoso.
Escenario principal	El Usuario ingresa los códigos de los productos y la cantidad de cada uno. El sistema verifica si los productos existen con la función buscar productos. El sistema calcula los costos totales de cada producto y muestra el registro por pantalla. 4. El usuario confirma el registro. 5. El sistema guarda el registro en la base de datos.
Extensiones	1.1 El usuario ingresa letras en el espacio de código de producto. 1.1.1 El sistema le pide que ingrese un valor numérico por pantalla.



Requisitos Especiales		El paso 2 del escenario principal debe indicar que exista el producto.
Frecuencia Ocurrencia	de	Alta



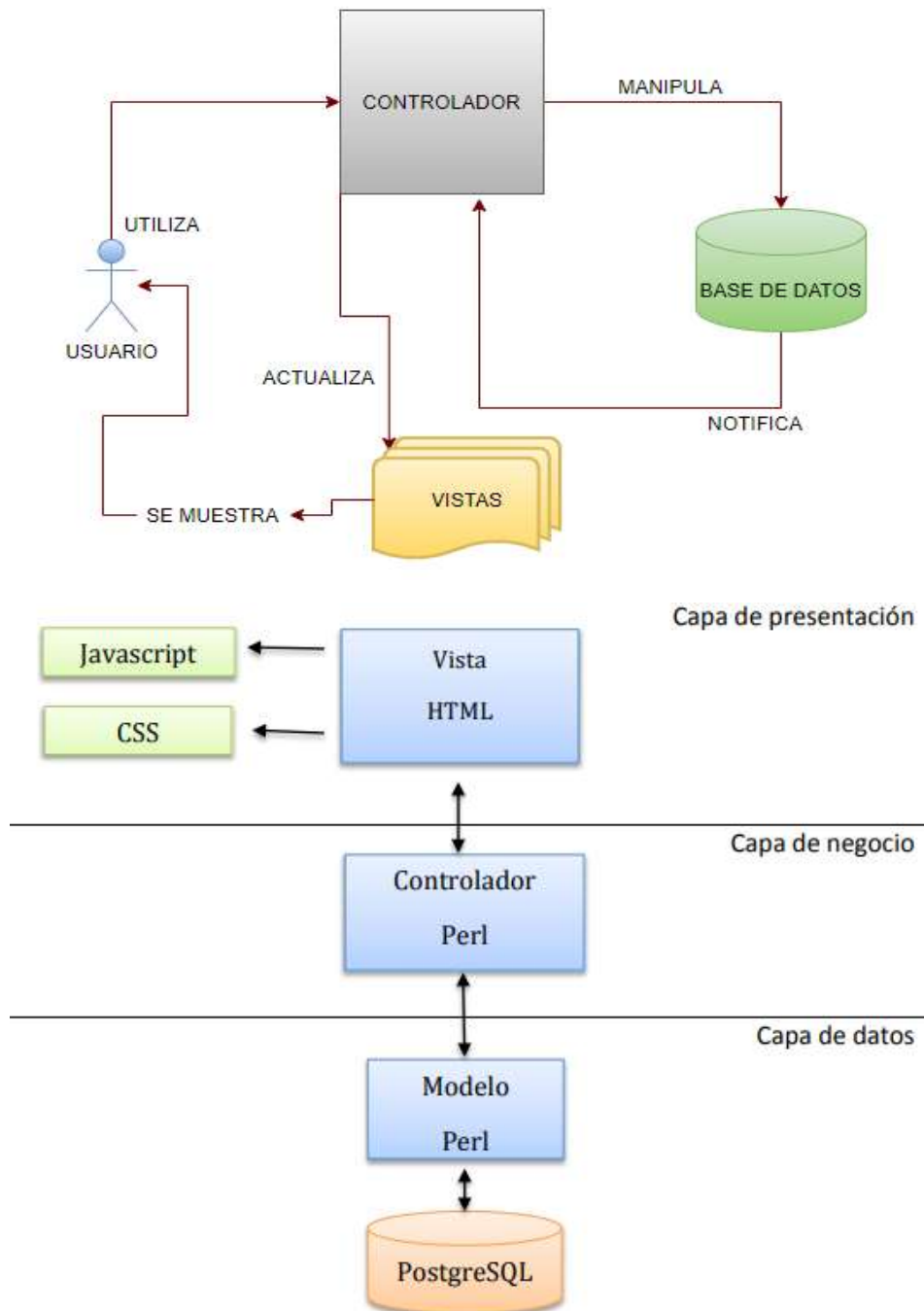


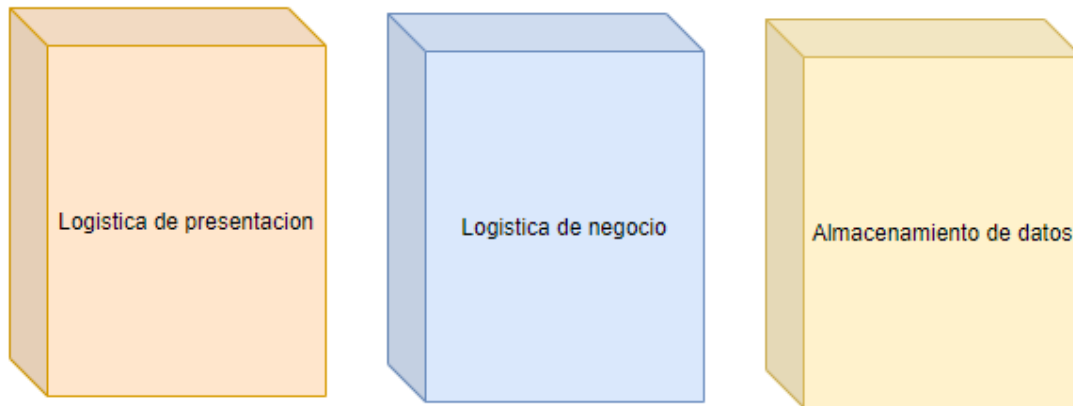




Modelo Vista Controlador: es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de control en tres componentes distintos:

- Modelo: datos.
- Vista: muestra la información del modelo al usuario.
- Controlador: gestiona las entradas del usuario e implementa la lógica de la aplicación.





Logística de Negocio

Presentación distribuida

Se distribuye la interfaz entre el cliente y la plataforma servidora.

La aplicación y los datos están ambos en el servidor. Similar a la arquitectura tradicional de un Host y Terminales. El PC se aprovecha solo para mejorar la interfaz gráfica del usuario.

Ventajas

- Revitaliza los sistemas antiguos.
- Bajo costo de desarrollo.
- No hay cambios en los sistemas existentes.

Desventajas

- El sistema sigue en el Host.
- No se aprovecha la GUI y/o LAN.
- La interfaz del usuario se mantiene en muchas plataformas.

Presentación remota

- La interfaz para el usuario está completamente en el cliente.
- La aplicación y los datos están en el servidor.

Ventajas

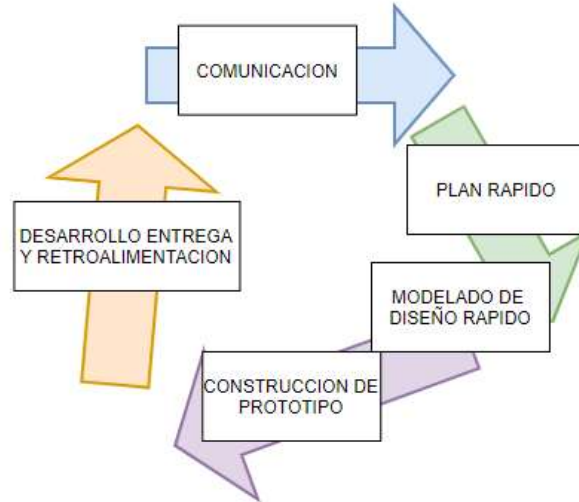
- La interfaz del usuario aprovecha bien la GUI y la LAN.
- La aplicación aprovecha el Host.
- Adecuado para algunos tipos de aplicaciones de apoyo a la toma de decisiones.

Desventajas

- Las aplicaciones pueden ser complejas de desarrollar.

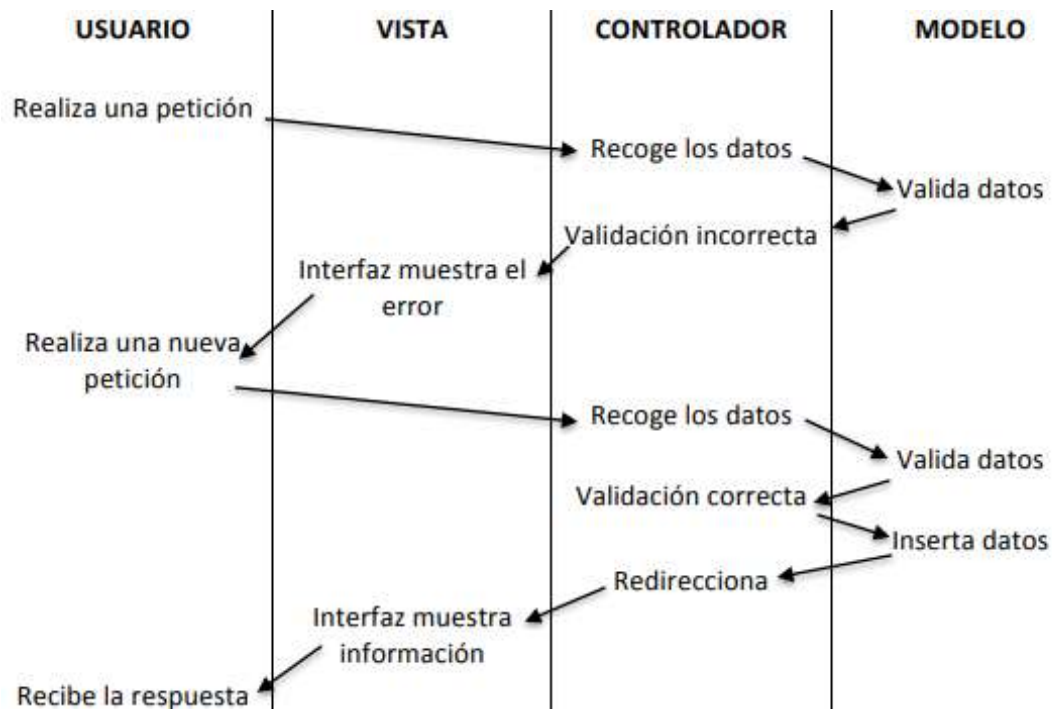


- Los programas de la aplicación siguen en el Host.
- El alto volumen de tráfico en la red puede hacer difícil la operación de aplicaciones muy pesadas.



Capa de negocios

Esta es la encargada de procesar y validar las reglas del negocio, actúa como un servidor para la capa de presentación y traslada las solicitudes de esta a la capa de datos actuando como cliente de esta. Así mismo también funciona de la misma manera de la forma inversa.





Lógica distribuida

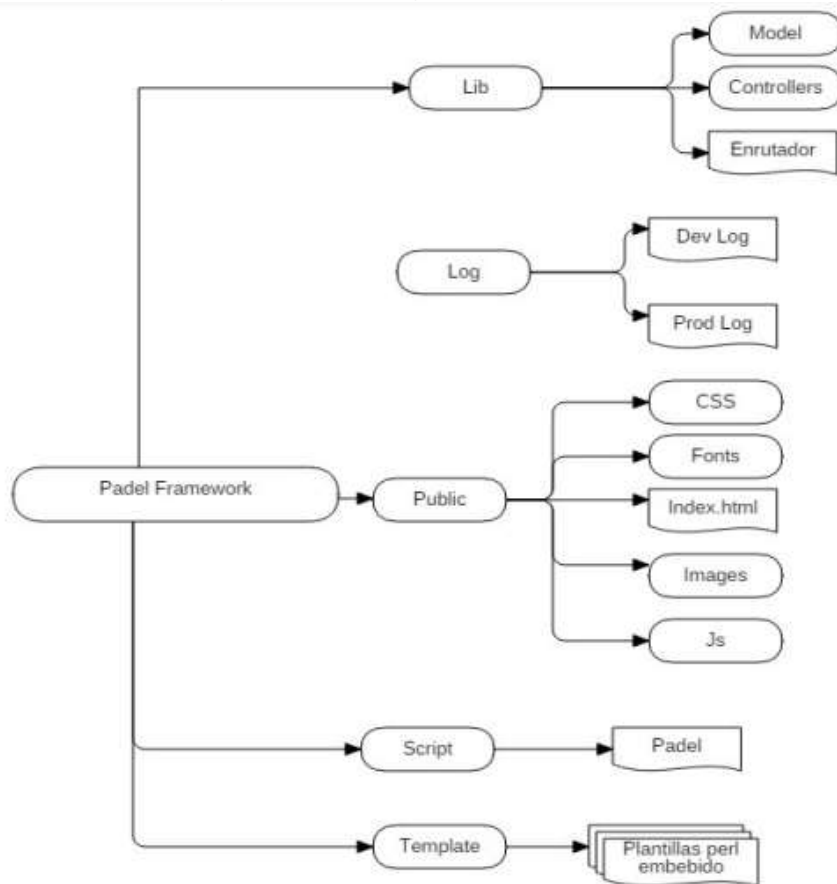
- La interfaz está en el usuario.
- La base de datos está en el servidor.
- La lógica de la aplicación está distribuida entre el usuario y el servidor.

Ventajas

- Arquitectura más simple que puede manejar todo tipo de aplicaciones.
- Los programas del sistema pueden distribuirse al nodo más apropiado.
- Pueden utilizarse con sistemas existentes.

Desventajas

- Es difícil de diseñar.
- Difícil prueba y mantenimiento si los programas del cliente y el servidor están hechos en distintos lenguajes de programación.





Capa de datos: Back end

Esta capa incluye bases de datos y programas que maneja la lectura y escritura sobre las mismas. En esta capa es donde se deben definir los métodos de acceso y políticas de seguridad utilizadas sobre los datos.

SISTEMAS DE INFORMACION

Es el conjunto formal de procesos que, operando sobre una colección de datos estructurada de acuerdo con las necesidades de una empresa, recopila, elabora y distribuye la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando, al menos en parte, los procesos de toma de decisiones necesarios para desempeñar las funciones de negocio de la empresa de acuerdo con su estrategia.



El nivel de servicios de datos es responsable de:

- Almacenar los datos
- Recuperar los datos
- Mantener los datos
- La integridad de los datos

Administración de datos remota

- En el cliente residen tanto la interfaz como los procesos de la aplicación.
- Las bases de datos están en el servidor.
- Es lo que comúnmente imaginamos como aplicación cliente servidor



Ventajas

- Configuración típica de la herramienta GUI 4GL.
- Muy adecuada para las aplicaciones de apoyo a las decisiones del usuario final.
- Fácil de desarrollar ya que los programas de aplicación no están distribuidos.
- Se descargan los programas del Host.

Desventajas

- No maneja aplicaciones pesadas eficientemente.
- La totalidad de los datos viaja por la red, ya que no hay procesamiento que realice el Host.

Base de datos distribuida

- La interfaz, los procesos de la aplicación, y parte de los datos de la base de datos están en el cliente.
- El resto de los datos están en el servidor.

Ventajas

- Configuración soportada por herramientas GUI 4GL.
- Adecuada para las aplicaciones de apoyo al usuario final.
- Apoya acceso a datos almacenados en ambientes heterogéneos.
- Ubicación de los datos es transparente para la aplicación.

Desventajas

- No maneja aplicaciones grandes eficientemente.
- El acceso a la base de datos distribuida es dependiente del proveedor del software administrador de bases de datos.

ESTRUCTURA DE MICROSERVICIOS

Estructura del sistema

Es un software donde se llevará el registro de las ventas emitiendo de esa manera las facturas correspondientes,

A continuación, estructura de lo base de nuestro software:



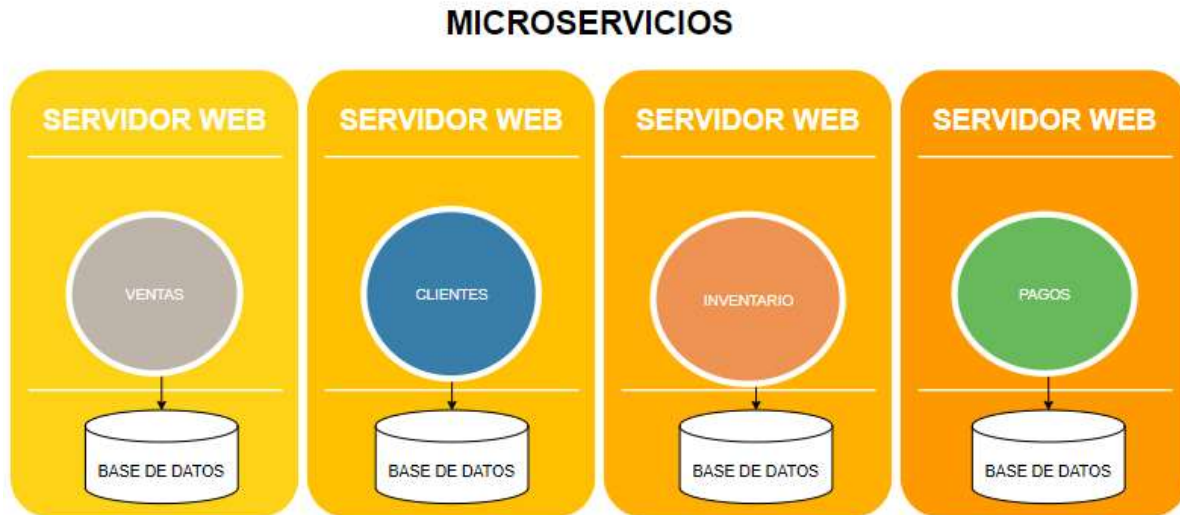
Estructura de sistema aplicándole arquitectura de microservicios:

Cuando hablamos de microservicios decimos que es una aproximación para el desarrollo de software que consiste en construir una aplicación como un conjunto de pequeños servicios, los cuales se ejecutan en su propio proceso y se comunican con mecanismos ligeros.

Es decir a la hora que de tener los servidores montados en un solo lugar se nos cae un servidor el sistema web queda paralizado, entonces ahí es donde entra la arquitectura de multiservicios aplicando y distribuyendo cada servidor y cada funcionalidad por aparte y siempre por medio de la base de datos aplicando cambios como registro, modificaciones, actualizaciones y eliminaciones, de esa manera si está distribuido cada cosa con su servidor por aparte se evita que a la hora de caer un servidor por ejemplo pagos, los demás servidores siguen trabajando normal, y los desarrolladores se enfocan solo donde está el problema, por eso en este análisis se toma en cuenta la importancia de los microservicios para nuestro software ya que de esa manera construimos una aplicación en conjuntos pequeños los cuales se ejecutan en sus propios procesos y se comunican con mecanismos ligeros por medio de todo los datos que se estarán almacenando en la base de datos.



A continuación, miramos la estructura del sistema ya aplicándole la arquitectura de microservicios:

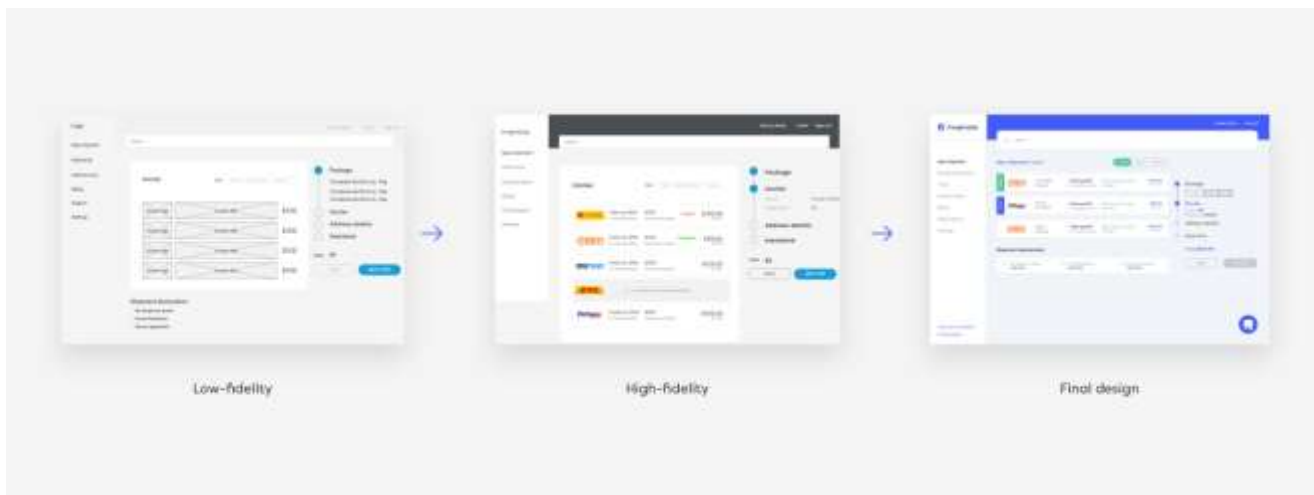
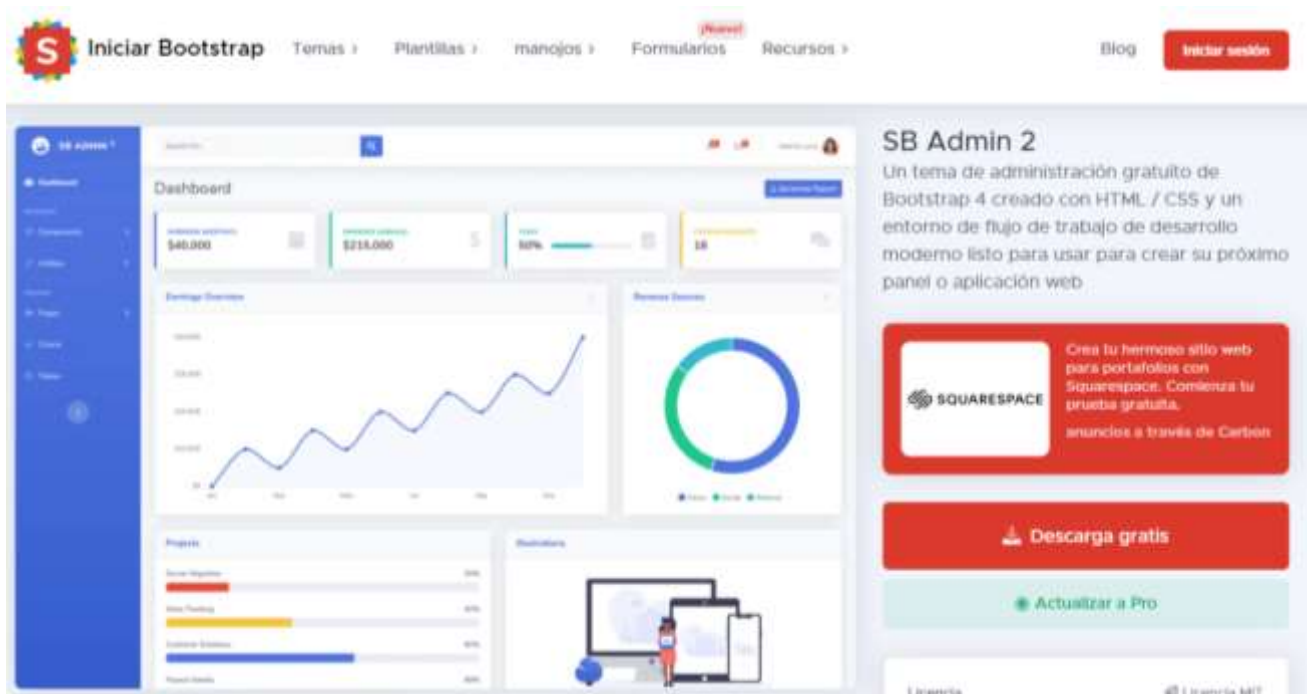


Aquí podemos observar que nuestra sistema cuenta con varios servidores y cada uno de los servidores manejando diferentes procesos y siempre almacenándolos en nuestra base de datos y de esa manera estar comunicados, por ejemplo el usuario a través del servidor del área de ventas él puede entrar al inventario de bodega para realizar una Venta, de esa manera también se lleva el registro en la base de dato de todo los movimientos que se hace y se maneja registros de las facturas emitidas, control de inventario, control de clientes, inventario en bodega.

Estrategias de pruebas de software

PLANTILLAS


En esta sección se verifico y se miro opciones para ver cual plantilla se acoplaba mejor estéticamente a nuestro sistema y que era lo que seria llamativo para el usuario.


SB Admin 2
Un tema de administración gratuito de Bootstrap 4 creado con HTML / CSS y un entorno de flujo de trabajo de desarrollo moderno listo para usar para crear su próximo panel o aplicación web.

Descarga gratis

Actualizar a Pro

Licencia  Licencia MIT




Iniciar Bootstrap

[Temas >](#)
[Plantillas >](#)
[manejos >](#)
[Formularios >](#)
[Recursos >](#)

[Blog](#)
[Iniciar sesión](#)

Dashboard

101.5K

Proyectos

100% completado

12.2K

Proyectos

100% completado

5.3K

Proyectos

100% completado

7

Proyectos

100% completado

Revenue Breakdown

Actual

Target

119%

Seguros

Actual

Target

100%

Privacy Suggestions

Account Storage

Administrador de materiales Pro

Un tema de administración premium de Bootstrap

Comprar ahora - \$ 29

★★★★★
(1 valoración de cliente)

Ventas totales: **\$ 306**

Liberado: **17 de marzo de 2021**

Última actualización: **hace 3 meses**

versión del producto: **1.0.3**



Inicio
 Vaya a Inicio

- Inicio
- Administración
- Tablero
- Productos
- Compras
- Ventas
- Mantenimiento en caja
- Seguros
- Ventas
- Seguros
- Caja de seguridad
- Configuraciones

TABLETO

Con este panel usted podrá administrar su negocio de manera eficiente.

CAJAS
 4 Registrados

CATEGORIAS
 44 Registrados

MARCA
 31 Registrados

PROVEEDORES
 31 Registrados

UBICACIONES
 31 Registrados

USUARIOS
 31 Registrados

CLIENTES
 44 Registrados

SALES & MAYOR
 50 de 1000

PRODUCTOS
 1112 Registrados

COMPRAS
 0 Compras

VENTAS
 0 Ventas

MANTENIMIENTOS
 0 Mantenimientos

DEVOLUCIONES
 0 Devoluciones

SARDEX
 0 Sardeces

REPORTES
 0 Reportes

COPA DE SEGURIDAD
 0 Copias de seguridad

CONFIGURACIONES
 0 Configuraciones



OBJETIVOS

La prueba de software es un elemento crítico para la garantía del correcto funcionamiento del software.

Objetivos:

- Detectar defectos en el software.
- Verificar la integración adecuada de los componentes.
- Verificar que todos los requisitos se han implementado correctamente.
- Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.
- Diseñar casos de prueba que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

VISION GENERAL

PRUEBAS DE UNIDAD

- Se centra los esfuerzos en la unidad más pequeña del diseño de software: el componente o modulo.
- Se enfocan en la lógica de procesamiento interno y de las estructuras de datos dentro de la frontera de un componente.
- Se realiza en paralelo para múltiples componentes.

Las 3 A's del unit testing

Para llevar a cabo buenas pruebas unitarias, deben estar estructuradas siguiendo las tres A's del Unit Testing. Se trata de un concepto fundamental respecto a este tipo de pruebas, que describe un proceso compuesto de tres pasos.

1. Arrange (organizar). Es el primer paso de las pruebas unitarias. En esta parte se definen los requisitos que debe cumplir el código.
2. Act (actuar). Es el paso intermedio de las pruebas, el momento de ejecutar el test que dará lugar a los resultados que deberás analizar.
3. Assert (afirmar). En el último paso, es el momento de comprobar si los resultados obtenidos son los que se esperaban. Si es así, se valida y se sigue adelante. Si no, se corrige el error hasta que desaparezca.

Link que Direcciona a la actividad especifica en GitHub:

<https://github.com/JonatanR97/PROYECTO-ANALISIS-DE-SISTEMAS-2/wiki/PRUEBAS-AL-SISTEMA>



PRUEBA DE VALIDACION

- La validación del software se logra a través de una serie de pruebas que demuestran conformidad con los requerimientos.
- Un plan de prueba subraya las clases de pruebas que se van a realizar y un procedimiento de prueba define casos de prueba específicos que se diseñan para garantizar que: se satisfacen todos los requerimientos de funcionamiento, se logran todas las características de comportamiento, todo el contenido es preciso y se presenta de manera adecuada, se logran todos los requerimientos de rendimiento, la documentación es correcta y se satisfacen la facilidad de uso y otros requerimientos.

Link que Direcciona a la actividad especifica en GitHub:

<https://github.com/JonatanR97/PROYECTO-ANALISIS-DE-SISTEMAS-2/wiki/PRUEBAS-AL-SISTEMA>

PRUEBA DE SISTEMA

- Es una serie de diferentes pruebas cuyo propósito principal es ejercitar por completo el sistema basado en computadora.
- Aunque cada prueba tenga un propósito diferente, todo él funciona para verificar que los elementos del sistema se hayan integrado de manera adecuada y que se realicen las funciones asignadas.

Link que Direcciona a la actividad especifica en GitHub:

<https://github.com/JonatanR97/PROYECTO-ANALISIS-DE-SISTEMAS-2/wiki/PRUEBAS-AL-SISTEMA>



CICLO COMPLETO DE PRUEBAS





RECOMENDACIONES

- ✓ Cada caso de prueba debe definir el resultado de salida esperado.
- ✓ El programador debe evitar probar sus propios programas, ya que desea (consciente o inconscientemente) demostrar que funcionan sin problemas.
- ✓ Se debe inspeccionar a conciencia el resultado de cada prueba, así, poder descubrir posibles síntomas de defectos.
- ✓ Al generar casos de prueba, se deben incluir tanto datos de entrada válidos y esperados como no válidos e inesperados.
- ✓ Las pruebas deben centrarse en dos objetivos (es habitual olvidar el segundo):
 - Probar si el software no hace lo que debe hacer
 - Probar si el software hace lo que debe hacer, es decir, provoca efectos secundarios adversos
- ✓ Se deben evitar los casos desechables, es decir, los no documentados ni diseñados con cuidado.
- ✓ No deben hacerse planes de prueba suponiendo que, prácticamente, no hay defectos en los programas y, por lo tanto, dedicando pocos recursos a las pruebas.
- ✓ La experiencia parece indicar que donde hay un defecto hay otros, es decir, la probabilidad de descubrir nuevos defectos en una parte del software es proporcional al número de defectos ya descubierto.
- ✓ Las pruebas son una tarea tanto o más creativa que el desarrollo de software. Siempre se han considerado las pruebas como una tarea destructiva y rutinaria



CONCLUSION

En Conclusión, un proyecto de desarrollo de un Sistema de Información comprende varios componentes o pasos llevados a cabo durante la etapa del análisis, el cual ayuda a traducir las necesidades del cliente en un modelo de Sistema que utiliza uno más de los componentes: Software, hardware, personas, base de datos, documentación y procedimientos.

En una organización o Empresa, el análisis y Diseño de Sistemas, es el proceso de estudiar su Situación con la finalidad de observar cómo trabaja y decidir si es necesario realizar una mejora; el encargado de llevar a cabo estas tareas es el analista de sistemas.

Antes de comenzar con el desarrollo de cualquier proyecto, se conduce un estudio de Sistemas para detectar todos los detalles de la situación actual de la empresa. La información reunida con este estudio sirve como base para crear varias estrategias de Diseño. Los administradores deciden que estrategias seguir.

Los Gerentes, empleados y otros usuarios finales que se familiarizan cada vez más con el uso de computadoras están teniendo un papel muy importante en el desarrollo de sistemas.

Todas las organizaciones son Sistemas que actúan de manera reciproca con su medio ambiente recibiendo entradas y produciendo salidas. Los Sistemas que pueden estar formados por otros Sistemas de denominan subsistemas y funcionan para alcanzar los fines de su Implantación.

Es por eso por lo que existen varios modelos o métodos para la realización del análisis y diseño de un sistema, lo primero del trabajo fue revisar que es el Análisis y el diseño y posteriormente el autor Kendall, presenta varios modelos que podemos utilizar para la realización y elaboración de un proceso y trabajo exhaustivo y dar solución o respuesta al problema que se ha generado desde la perspectiva del programador y analista.