[StockWise]

(DAS) Documento Arquitectura Sistema

Versión 1.0

# **Identificación de Documento**

| **Identificación** | StockWise |
| --- | --- |
| **Proyecto** | StockWise |
| **Versión** | 1.0 |

| **Documento mantenido por** | Martin Salazar |
| --- | --- |
| **Fecha de última revisión** | 30-09-2024 |
| **Fecha de próxima revisión** | 18-10-2024 |

| **Documento aprobado por** | Fabian Saldaño |
| --- | --- |
| **Fecha de última aprobación** | No registrado |

# Historia de Revisiones

| **Fecha** | **Versión** | **Descripción** | **Autor** |
| --- | --- | --- | --- |
| 30-09-2024 | 1.0 | Inicio Documento DAS | Martin Salazar |
| 16-10.2024 | 1.1 | Cierre Documento DAS | Jorge Muñoz |
|  |  |  |  |
|  |  |  |  |

# **Tabla de Contenidos**

[**Identificación de Documento 0**](#_heading=h.tepr40fqxw94)

[**Historia de Revisiones 1**](#_heading=h.eo4fwldli4t7)

[**Tabla de Contenidos 1**](#_heading=h.iphpg58g5ct0)

[**1 Introducción 2**](#_heading=h.gjdgxs)

[1.1 Contexto del Problema 3](#_heading=h.30j0zll)

[1.2 Propósito 3](#_heading=h.1fob9te)

[1.3 Ámbito 3](#_heading=h.3znysh7)

[1.4 Definiciones, acrónimos y abreviaciones 3](#_heading=h.2et92p0)

[1.5 Resumen ejecutivo 4](#_heading=h.3dy6vkm)

[1.6 Representación 4](#_heading=h.1t3h5sf)

[**2 Metas y Restricciones de la Arquitectura 6**](#_heading=h.4d34og8)

[2.1 Metas de la arquitectura 6](#_heading=h.2s8eyo1)

[2.2 Restricciones de la Arquitectura 6](#_heading=h.17dp8vu)

[2.3 Otros antecedentes y consideraciones 7](#_heading=h.3rdcrjn)

[**3 Vista de Escenarios 7**](#_heading=h.26in1rg)

[3.1 Modelo de Casos de Uso 8](#_heading=h.lnxbz9)

[3.2 Casos de Usos Extendidos 8](#_heading=h.35nkun2)

[3.3 Especificación de los Escenarios de Calidad Relevantes 9](#_heading=h.1ksv4uv)

[**4 Vista de Procesos 12**](#_heading=h.44sinio)

[DA Autenticar 12](#_heading=h.ts0ar1sa111b)

[DA Administrar Usuario 13](#_heading=h.dsw7aglb6n3l)

[DA Visualizar Historial Movimientos 14](#_heading=h.i4yknv5gjqzi)

[DA Auditar Acciones y Actividades 14](#_heading=h.myj65eg2gyb9)

[DA Generar Reporte 15](#_heading=h.utrk4013l1xk)

[DA Visualizar Dashboard 16](#_heading=h.m2386kp4nkau)

[DA Gestionar Inventario 17](#_heading=h.a7r3npxuzaqm)

[DA Visualizar Alerta 18](#_heading=h.nnusipt9vz8z)

[Diagramas de Actividad desglosado - Administrar Productos 19](#_heading=h.wp80onp6hmt7)

[DA Ingresar Producto 19](#_heading=h.y7ouvtjbglhv)

[DA Modificar Producto 20](#_heading=h.24xrvf31m70s)

[DA Eliminar Producto 21](#_heading=h.jwx4z01qz2rr)

[DA Visualizar Producto 22](#_heading=h.pbaafqc61rru)

[**5 Vista Lógica 23**](#_heading=h.2jxsxqh)

[**Parte Dinámica (Diagrama de Secuencias) 23**](#_heading=h.dta07471jatf)

[Autenticar Usuario 23](#_heading=h.quaj4fwqq1oi)

[Administrar Usuario 24](#_heading=h.hhvz8thg9soo)

[Visualizar Historial Movimientos 25](#_heading=h.e769fffgii8g)

[Auditar Acciones y Actividades 25](#_heading=h.z1gqi22bu8a0)

[Generar Reporte 26](#_heading=h.58yrr8ekzokn)

[Visualizar Dashboard 26](#_heading=h.oq3iqlynmlxr)

[Administrar Productos 27](#_heading=h.yidh7oak7q)

[**6 Vista de Componentes 28**](#_heading=h.2xcytpi)

[**7 Vista Física 28**](#_heading=h.1ci93xb)

[**8 Decisiones de Diseño y Selección de Alternativas 29**](#_heading=h.3whwml4)

[**Análisis de Reutilización 30**](#_heading=h.2bn6wsx)

# Introducción

## Contexto del Problema

La empresa Alimentos Frescos del Sur, dedicada a la distribución de productos alimenticios, enfrenta importantes desafíos en la gestión de su inventario, especialmente con productos perecibles y congelados. La gestión manual actual no solo genera errores y retrasos en la actualización del stock, sino que también provoca problemas en el control adecuado de los tiempos de almacenamiento, lo que incrementa el riesgo de pérdida de productos perecibles. Esto ha derivado en ineficiencias operativas, aumento de costos y una disminución en la satisfacción del cliente, por lo que se busca implementar StockWise, un sistema automatizado que optimice el manejo del inventario y garantice el control adecuado de productos sensibles a las condiciones de almacenamiento.

## Propósito

Desarrollar un sistema informático en la plataforma web que permita gestionar, automatizar y optimizar los procesos de control de inventarios dentro de Alimentos Frescos del Sur, con un enfoque especial en los productos perecibles y congelados. El sistema, denominado StockWise, mejorará la eficiencia operativa, reducirá el riesgo de pérdida de productos, y permitirá la integración con los sistemas actuales y futuros de la empresa. Además, contará con una interfaz intuitiva y funcional para cada actor involucrado en la cadena de suministro, asegurando una gestión ágil y eficiente.

## Ámbito

Se desarrollará una solución web, que contempla la gestión administrativa y operativa de inventarios en Alimentos Frescos del Sur, desde el control de stock hasta la predicción de demandas de productos perecibles. El sistema será utilizado por el personal encargado de la cadena de suministro, con el fin de optimizar tanto el almacenamiento como la distribución de productos congelados y frescos, asegurando una adecuada trazabilidad y reducción de desperdicios.

## Definiciones, acrónimos y abreviaciones

| **ACRÓNIMO** | **DESCRIPCIÓN** |
| --- | --- |
| *Modelo de Vistas 4+1* | Perspectiva organizada de los elementos de un proyecto y sus relaciones. A través de la integración de diferentes vistas de la arquitectura, se abarcan diversos aspectos del diseño del sistema, resultando en una arquitectura de solución integral y comprensible para todos los interesados. |
| *CU* | Casos de Uso |
| *DA* | Diagrama de Actividades |
| *DC* | Diagrama de Clases |
| *DCOM* | Diagrama de Componentes |
| *DD* | Diagrama de despliegue |
| *DS* | Diagrama de secuencia |

## Resumen ejecutivo

El proyecto StockWise tiene como objetivo proporcionar una solución integral para la gestión de inventarios en empresas de productos perecibles, especialmente para Alimentos Frescos del Sur. El sistema permite optimizar la administración de inventarios a través de una aplicación web, mejorando la eficiencia en la gestión de productos, trazabilidad de movimientos y alertas sobre productos en riesgo de caducidad.

Las principales funcionalidades del sistema incluyen la autenticación de usuarios, administración de perfiles y roles, auditoría de acciones, generación de reportes, visualización de dashboards para el análisis de datos, gestión de inventarios, alertas automáticas para productos perecibles, y administración eficiente de productos.

## Representación

La arquitectura del sistema StockWise está representada siguiendo el enfoque del framework 4+1 y las recomendaciones del proceso unificado. Las vistas incluidas en esta versión del documento son:

* **Vista de Escenarios**:  
    
  La vista de escenarios describe los principales casos de uso del sistema StockWise, enfocados en la gestión de inventarios para PYMES. Estos casos de uso incluyen:
  + Autenticar usuario: Proceso que permite a los usuarios registrados ingresar al sistema.
  + Gestión de inventario: Permite al operador agregar, actualizar y visualizar el estado de productos en tiempo real.
  + Generación de reportes: Funcionalidad que permite a los analistas crear reportes detallados para la toma de decisiones.
  + Alertas automáticas: El sistema envía notificaciones de stock mínimo o productos cercanos a caducar.

Actores:

* + Administrador: Responsable de la configuración del sistema y la gestión de usuarios.
  + Operador: Encargado de la actualización y control de inventarios.
  + Analista: Genera reportes y análisis de inventarios.

Escenarios de Calidad Relevantes:

* + Rendimiento del sistema: El sistema debe responder en menos de 2 segundos bajo condiciones normales.
  + Seguridad: Los accesos al sistema y a los datos deben estar protegidos mediante autenticación robusta y encriptación de datos.
  + Disponibilidad: El sistema debe garantizar un tiempo de actividad del 99.9%.
* **Vista de Procesos**:   
    
  Esta vista describe los procesos clave que ocurren dentro de StockWise, desde el ingreso de un usuario hasta la generación de alertas y reportes.  
  + Proceso de autenticación: Los usuarios ingresan sus credenciales que se validan contra la base de datos de Firebase.
  + Sincronización del inventario en Tiempo Real: Cada vez que el operador actualiza un producto, el sistema sincroniza el inventario en tiempo real en la base de datos.
  + Generación de alertas**:** Las alertas de stock bajo o productos cercanos a caducar son generadas automáticamente y enviadas al operador.
  + Reporte y análisis predictivo: El sistema ejecuta procesos de análisis de datos basados en la demanda histórica, generando reportes para los analistas.

Relaciones de Comunicación y Sincronización:

* + Comunicación entre módulos:Cada vez que un usuario interactúa con el sistema, los procesos se ejecutan a través de llamadas API, que se comunican con el backend (Node.js) y la base de datos (Firebase).
* **Vista Lógica**:   
    
  El sistema StockWise está compuesto por módulos lógicos bien definidos que permiten un diseño modular:
  + Módulo de autenticación: Maneja el ingreso seguro de los usuarios y la gestión de permisos.
  + Módulo de gestión de Inventario: Controla las acciones de agregar, eliminar y modificar productos.
  + Módulo de reportes: Genera informes sobre el estado del inventario, movimientos y predicciones.
  + Módulo de notificaciones: Envía alertas a los usuarios sobre el estado del inventario y productos cercanos a caducar.

Responsabilidades y Dependencias:

* + Frontend (Angular/Ionic): Interfaz que interactúa con los usuarios.
  + Backend (Node.js): Procesa las solicitudes y gestiona la lógica del sistema.
  + Base de datos (Firebase): Almacena toda la información del sistema en tiempo real.
* **Vista de Desarrollo o Despliegue**:   
    
  El despliegue de StockWise se realizará en la nube, asegurando escalabilidad y redundancia:  
  + Servidores Replicados: El sistema utilizará servidores replicados para asegurar la alta disponibilidad y la redundancia.
  + Base de Datos en Firebase: Se utiliza Firebase como base de datos en tiempo real, lo que asegura un manejo eficiente de grandes volúmenes de datos.
  + Componentes del Despliegue:
    - Frontend: Aplicación web con Angular e Ionic.
    - Backend: Node.js gestionando la lógica de negocio.
    - Base de Datos: Firebase.
* **Vista Física**:   
    
  Esta vista aborda las restricciones tecnológicas y normativas que afectan las decisiones de la arquitectura:  
  + Restricciones Tecnológicas:
    - El sistema deberá ser compatible con navegadores web modernos (Chrome, Firefox, Edge) y navegadores de dispositivos móviles.
    - Se cuenta con servidores replicados y una base de datos en la nube para garantizar el rendimiento.
  + Normativas y Estándares:
    - El sistema deberá cumplir con normativas locales sobre protección de datos personales (Ley 19.628 de Chile) y seguir buenas prácticas de seguridad para evitar vulnerabilidades.
  + Infraestructura: La infraestructura está basada en tecnología de código abierto para minimizar costos y facilitar el mantenimiento futuro.

# **Metas y Restricciones de la Arquitectura**

A continuación, se revisan las metas y restricciones de la arquitectura.

## Metas de la arquitectura

De acuerdo a las reuniones y al análisis de los requerimientos, se listan los principales conductores iniciales de la arquitectura los cuales corresponden a las metas arquitectónicas iniciales ( atributos de calidad)

* **Desempeño**: El sistema debe garantizar una respuesta rápida (<2 segundos) bajo condiciones normales, incluso con varios usuarios concurrentes.
* **Tolerancia** a fallos: El uso de servidores replicados y la base de datos en la nube proporcionan redundancia y garantizan la alta disponibilidad del sistema.
* **Seguridad**: Los datos sensibles de los usuarios y los inventarios serán protegidos mediante encriptación y autenticación robusta.
* **Modificabilidad/Reuso**: La arquitectura modular permite la adición de nuevas funcionalidades sin afectar la estructura existente del sistema.
* **Operatividad**: El sistema será fácil de usar, con una interfaz intuitiva y accesible desde cualquier dispositivo.
* **Adaptabilidad**: StockWise está diseñado para adaptarse a las necesidades futuras de las PYMES, permitiendo la integración de nuevas funcionalidades como análisis de ventas o gestión logística.

## Restricciones de la Arquitectura

Existen restricciones que han sido levantadas con los stakeholders, las cuales se presentan a continuación:

* **Tiempo de construcción**: El proyecto debe ser completado en 4 meses, dividido en fases, asegurando que el sistema esté completamente funcional para la empresa Alimentos Frescos del Sur dentro de los plazos establecidos.
* **Infraestructura**: Se utilizarán servidores de aplicaciones replicados y la base de datos en Firebase, que ofrece un servicio en tiempo real y escalabilidad.
* **Otros componentes de software**: No se contempla la adquisición de licencias adicionales de software, ya que se utilizarán tecnologías de código abierto como Angular, Node.js, y Firebase.

## 

## Otros antecedentes y consideraciones

La empresa desarrolladora cuenta con un framework que considera los siguientes componentes que permiten satisfacer los requerimientos arquitectónicos:

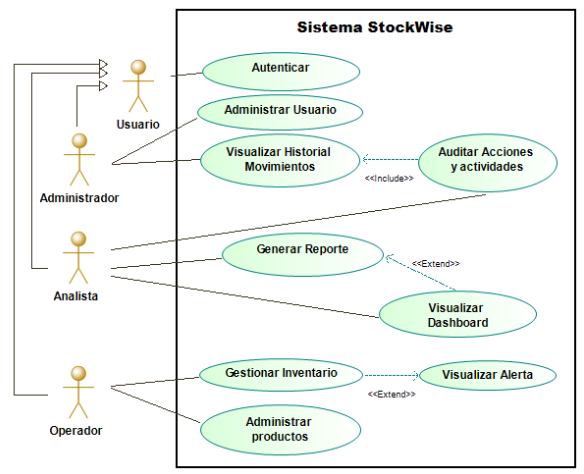
* **Framework de Desarrollo**:
  + Angular/Ionic: Para el desarrollo del frontend, se utilizará Angular e Ionic, que permiten la creación de aplicaciones web con una sola base de código.
  + Node.js: El backend será gestionado con Node.js, proporcionando una estructura escalable y eficiente para procesar las solicitudes de los usuarios.
  + Firebase: Base de datos en tiempo real para el almacenamiento y la gestión de inventarios, que asegura alta disponibilidad y rendimiento.
* **Framework de Seguridad**:
  + Se implementará un framework de seguridad para gestionar la autenticación de los usuarios y proteger los datos sensibles mediante encriptación y protocolos de seguridad.

# **Vista de Escenarios**

Esta sección describe en detalle el conjunto de escenarios funcionales y no funcionales que obtuvieron la mayor prioridad en el análisis. Para esto se presenta y describe el diagrama de casos de uso y los casos de uso prioritarios, así como los escenarios en que uno o más atributos de calidad se ven involucrados de manera significativa.

## Modelo de Casos de Uso

**Ilustración 1: Diagrama de Caso Uso General del Sistema**

****

## Casos de Usos Extendidos

Los casos de uso considerados son los más relevantes para el desarrollo de la arquitectura. Se adjunta el documento o planilla caso uso.

[**Documento Casos de Usos Extendidos Link.**](https://docs.google.com/document/d/1qxUS2PCI6wHg52KP529E9ThJno07BmKX/edit?usp=drive_link&ouid=111378954683525623569&rtpof=true&sd=true)

A continuación, se listan los casos de uso relevantes, los cuales pueden ser encontrados con su especificación detallada en el documento “Casos de Uso Extendido”.

| **Código** | **Nombre** | **Actores** | **Prioridad** |
| --- | --- | --- | --- |
| CU-001 | Autentificar Usuario | Todos los usuarios | Alta |
| CU-002 | Administrar Usuario | Administrador | Alta |
| CU-003 | Visualizar Historial Movimientos | Administrador | Baja |
| CU-004 | Auditar Acciones y Actividades | Administrador, Analista | Baja |
| CU-005 | Generar Reporte | Analista | Alta |
| CU-006 | Visualizar Dashboard | Analista | Alta |
| CU-007 | Gestionar Inventario | Operador | Alta |
| CU-008 | Visualizar Alerta | Operador | Media |
| CU-009 | Administrar Productos | Operador | Media |

## Especificación de los Escenarios de Calidad Relevantes

Después de un análisis en conjunto con los stakeholders, los escenarios de calidad se expresan a continuación:

| Identificador: EC-001 | | |
| --- | --- | --- |
| Escenario(s): | | Aplicación WEB |
| Atributos de Calidad relevantes: | | Portabilidad - Adaptabilidad |
| Componentes del Escenario | Estímulos: | Acceso a la aplicación web |
| Fuente del estímulo | Usuario del sistema |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Aplicación |
| Respuesta: | Debe visualizarse todo el contenido de manera correcta desde distintos dispositivos. |
| Medida de Respuesta | Probado en al menos 3 navegadores (Chrome, Firefox, Safari) y sistemas operativos Android e iOS. |

| Identificador: EC-002 | | |
| --- | --- | --- |
| Escenario(s): | | Facilidad de uso y aprendizaje |
| Atributos de Calidad relevantes: | | Usabilidad – Facilidad de aprendizaje |
| Componentes del Escenario | Estímulos: | Usar el sistema para gestionar inventarios |
| Fuente del estímulo | Usuario (Operador y Administrador) |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Aplicación |
| Respuesta: | La aplicación debe ser intuitiva, permitiendo una rápida adaptación de los usuarios. |
| Medida de Respuesta | Los usuarios deben ser capaces de utilizar las funciones básicas en 2 días de entrenamiento. |

| Identificador: EC-003 | | |
| --- | --- | --- |
| Escenario(s): | | Aplicación responsiva |
| Atributos de Calidad relevantes: | | Portabilidad - Adaptabilidad |
| Componentes del Escenario | Estímulos: | Acceso a la aplicación desde diferentes dispositivos |
| Fuente del estímulo | Usuario (Administradores y Operadores) |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Aplicación |
| Respuesta: | Se debe adaptar el contenido a distintos tamaños de pantalla de dispositivos móviles y de escritorio. |
| Medida de Respuesta | Probado y adaptado para dispositivos móviles y pantallas de computadoras de escritorio. |

| Identificador: EC-004 | | |
| --- | --- | --- |
| Escenario(s): | | Disponibilidad |
| Atributos de Calidad relevantes: | | Fiabilidad - Disponibilidad |
| Componentes del Escenario | Estímulos: | Acceso a la aplicación para gestionar inventarios |
| Fuente del estímulo | Usuario |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Aplicación |
| Respuesta: | El sistema debe estar disponible para los usuarios en horarios operativos de la empresa. |
| Medida de Respuesta | Disponibilidad mínima del 99.9% durante los horarios de trabajo. |

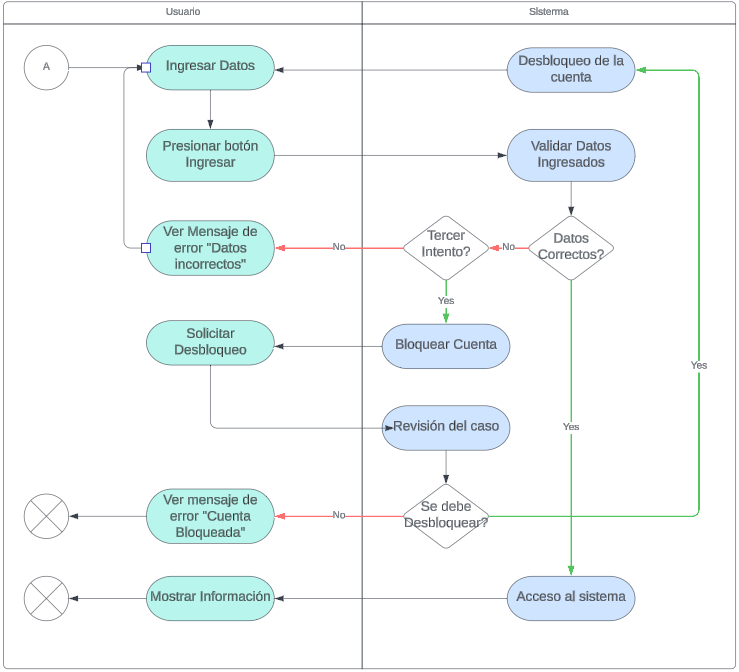
| Identificador: EC-005 | | |
| --- | --- | --- |
| Escenario(s): | | Validación de Roles |
| Atributos de Calidad relevantes: | | Funcionalidad - Seguridad |
| Componentes del Escenario | Estímulos: | Acceso a diferentes módulos según rol |
| Fuente del estímulo | Usuario (Administrador, Operador) |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Aplicación |
| Respuesta: | El acceso a las funciones estará limitado por el rol asignado a cada usuario. |
| Medida de Respuesta | Validación de roles en menos de 1 segundo, mostrando opciones según el rol del usuario. |

| Identificador: EC-006 | | |
| --- | --- | --- |
| Escenario(s): | | Eficiencia en Respuesta |
| Atributos de Calidad relevantes: | | Eficiencia - Comportamiento en el tiempo |
| Componentes del Escenario | Estímulos: | Realizar consultas y actualizaciones de stock |
| Fuente del estímulo | Operador |
| Ambiente: | Aplicación finalizada y operativa |
| Artefacto: | Base de Datos |
| Respuesta: | La respuesta del sistema debe ser rápida y efectiva, actualizando el stock en tiempo real. |
| Medida de Respuesta | Las consultas y actualizaciones de stock deben completarse en menos de 2 segundos. |

| Identificador: EC-007 | | |
| --- | --- | --- |
| Escenario(s): | | Tolerancia a Fallos |
| Atributos de Calidad relevantes: | | Fiabilidad - Tolerancia a Fallos |
| Componentes del Escenario | Estímulos: | Desconexión de la red o caídas temporales del sistema |
| Fuente del estímulo | Usuario (Operador, Administrador) |
| Ambiente: | Aplicación operativa |
| Artefacto: | Aplicación |
| Respuesta: | El sistema debe continuar operando y recuperar datos al reconectar. |
| Medida de Respuesta | La aplicación debe almacenar datos localmente y sincronizarlos en menos de 5 minutos tras la reconexión. |

# Vista de Procesos

#### DA Autenticar

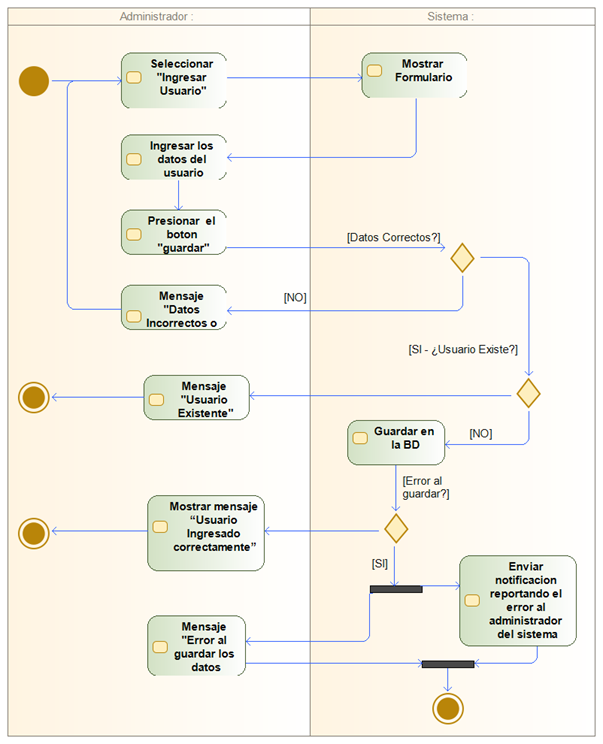


#### 

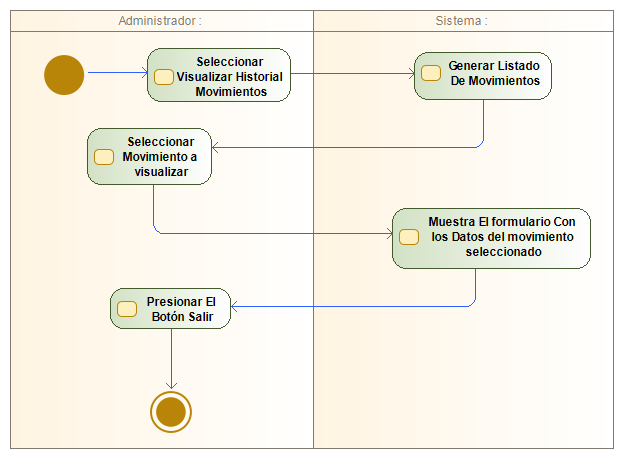
#### 

#### 

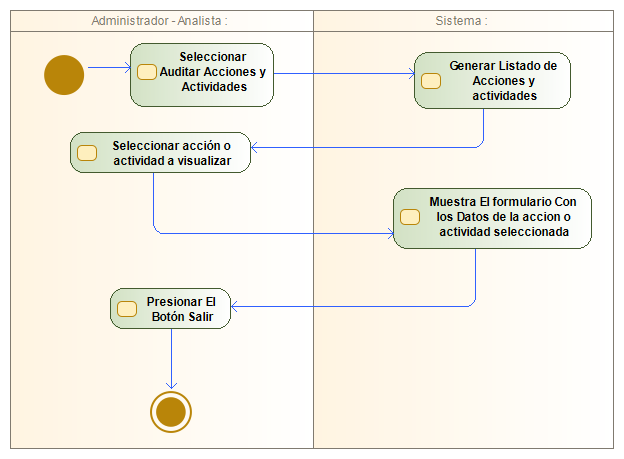
#### DA Administrar Usuario



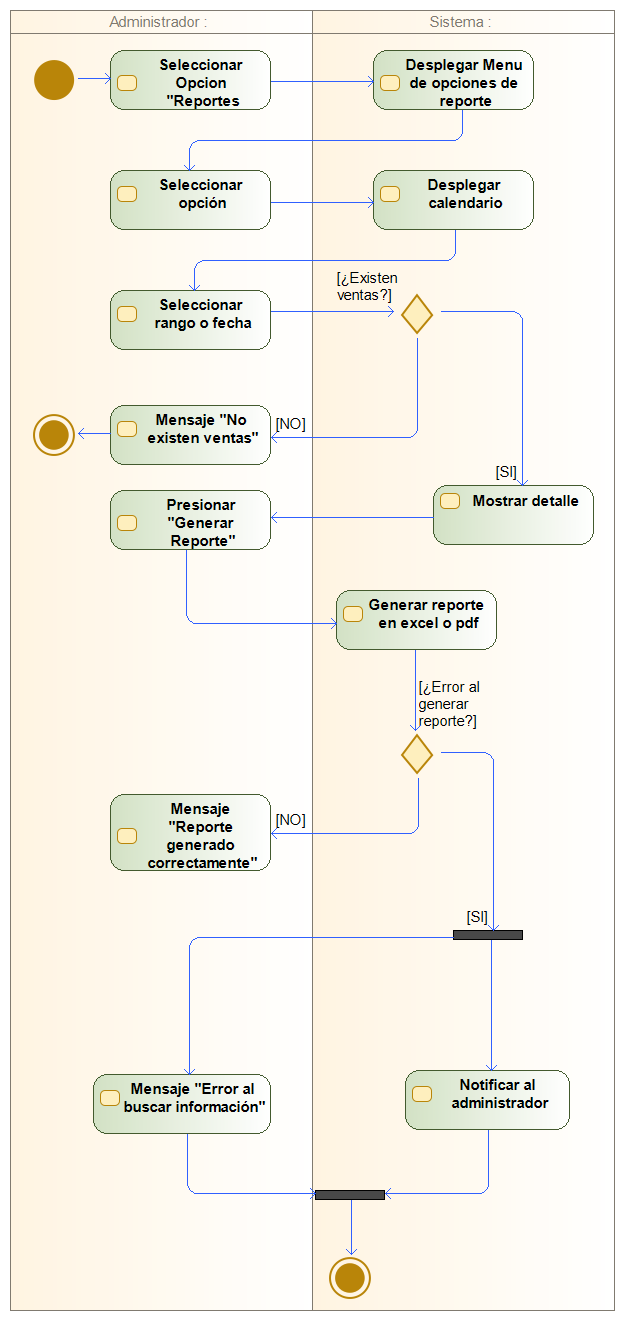
#### DA Visualizar Historial Movimientos

****

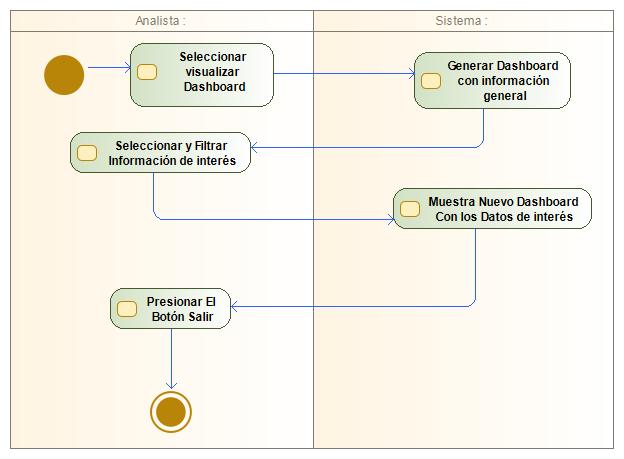
#### DA Auditar Acciones y Actividades

****

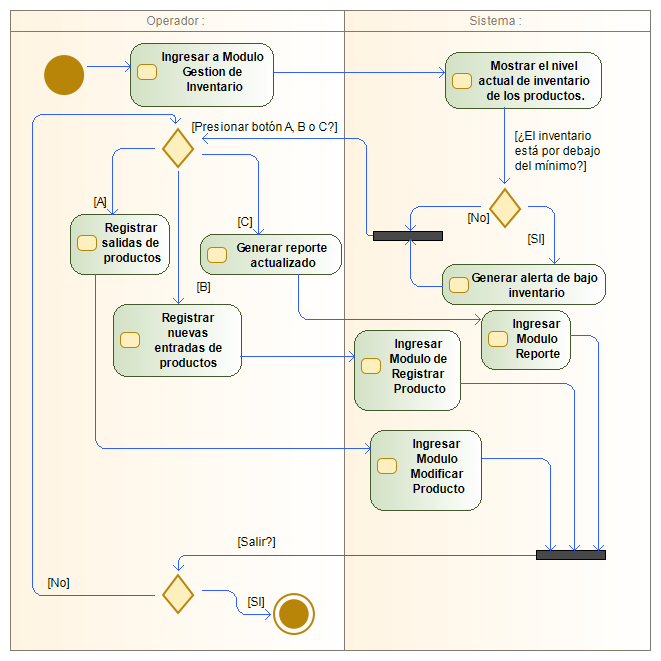
#### DA Generar Reporte



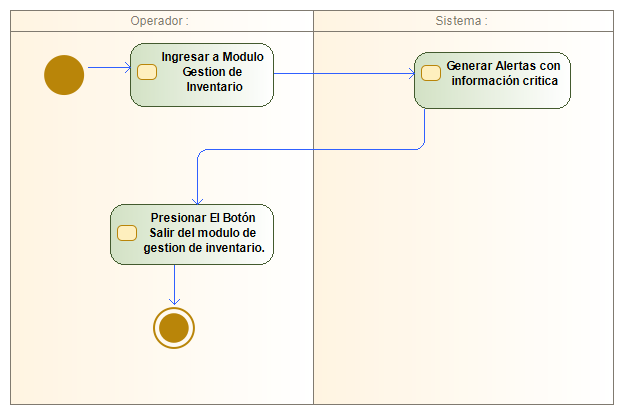
#### DA Visualizar Dashboard

****

#### DA Gestionar Inventario

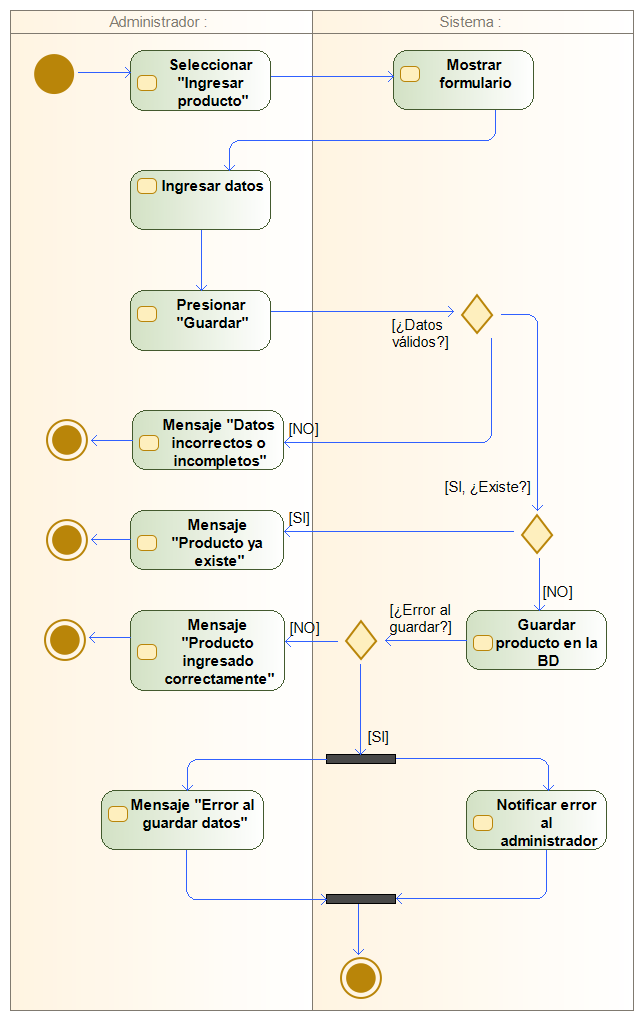
****

#### DA Visualizar Alerta

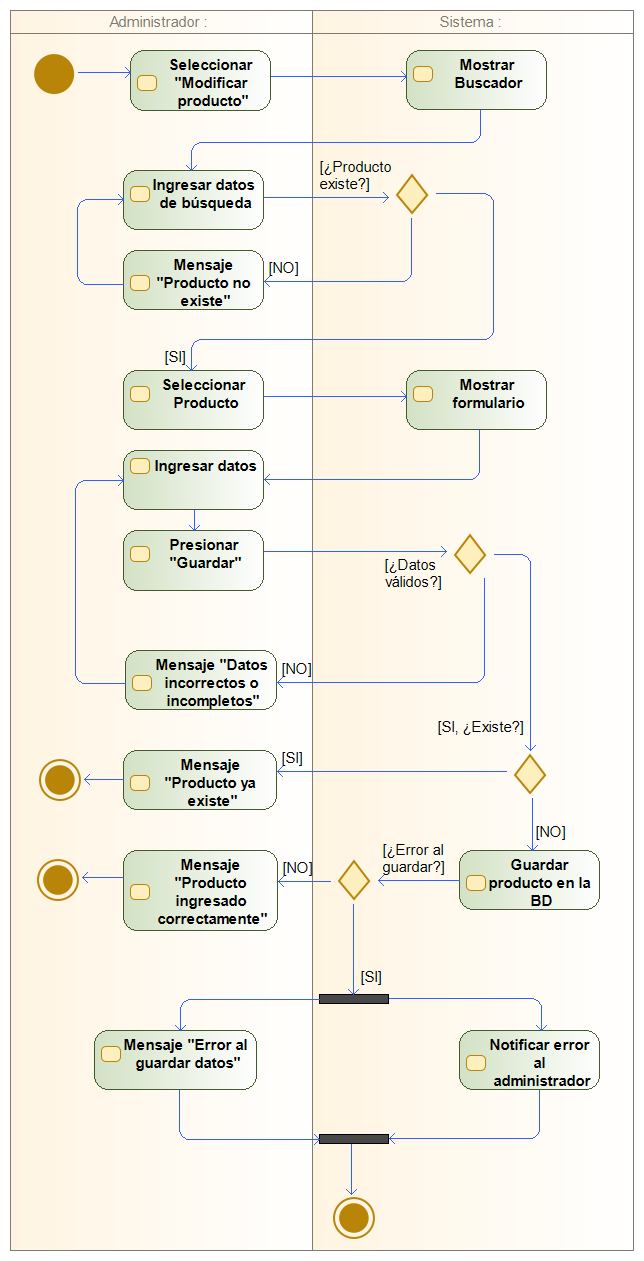


### Diagramas de Actividad desglosado - Administrar Productos

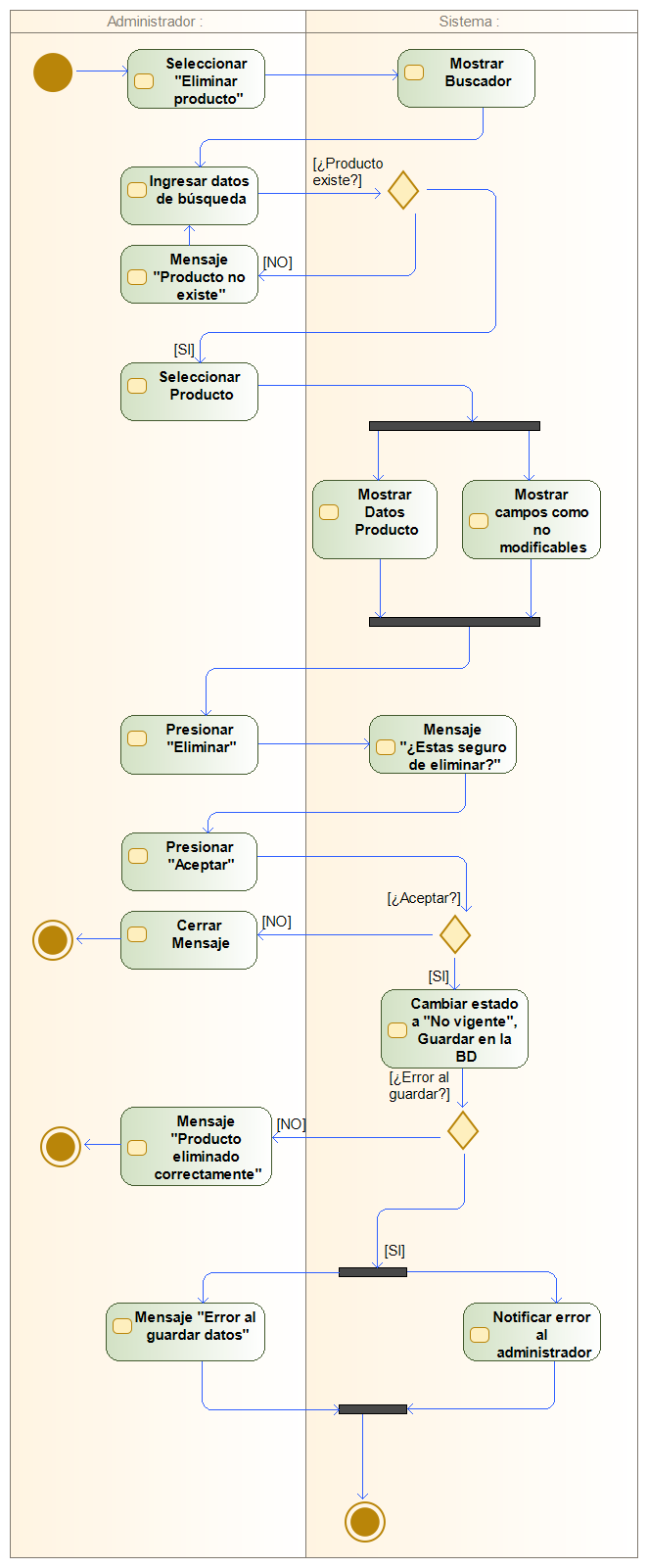
#### DA Ingresar Producto

****

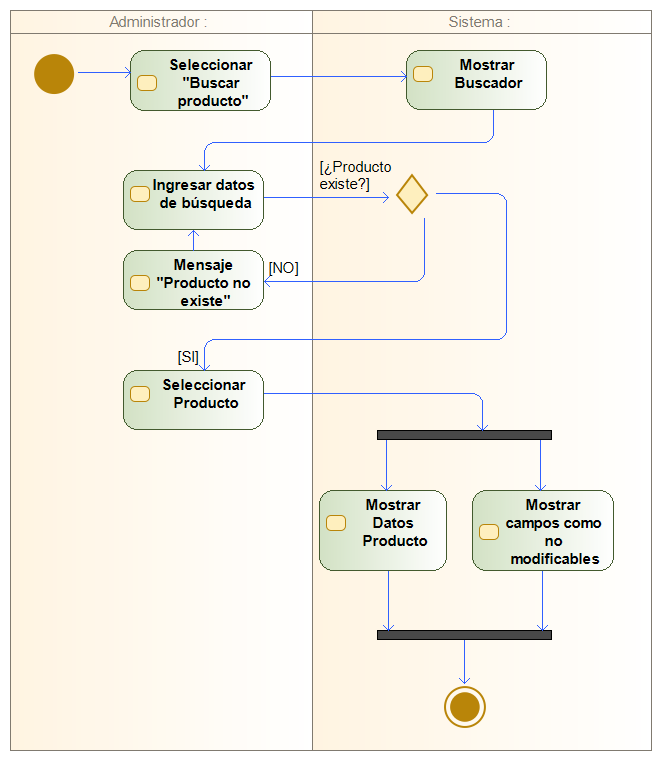
#### DA Modificar Producto

****

#### DA Eliminar Producto

****

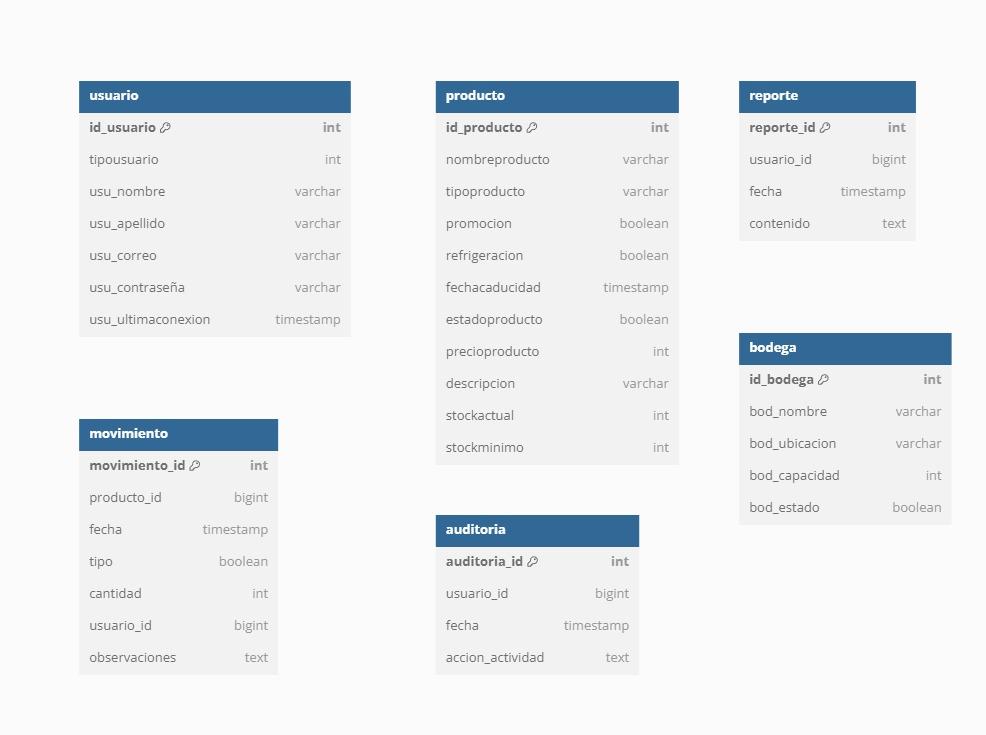
#### DA Visualizar Producto

****

# Vista Lógica

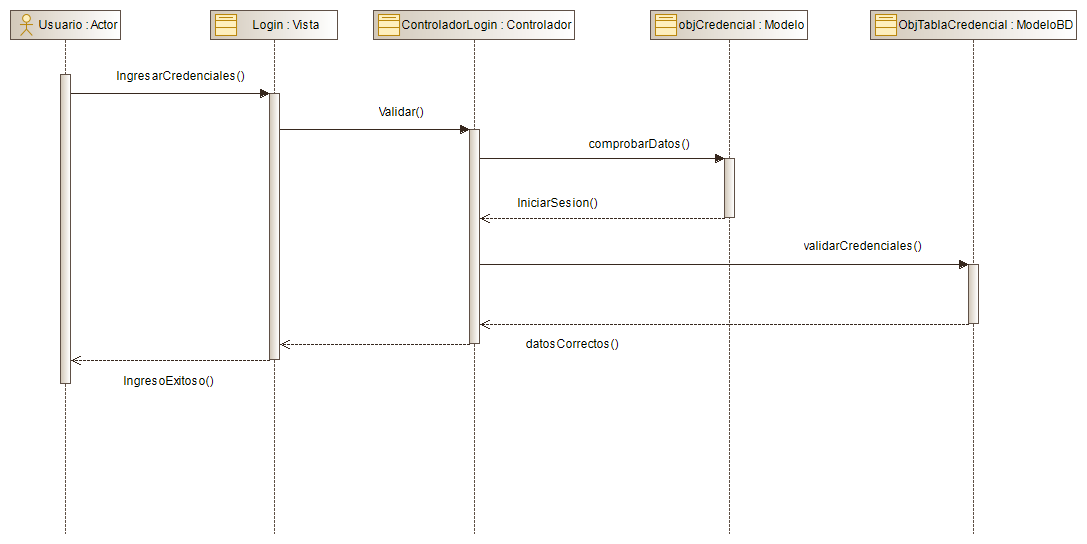
Se adjunta Link del modelo de las tablas, las cuales corresponden a una base de datos no Relacional en Firebase.

[Documento Tablas Diccionario Datos Link](https://docs.google.com/document/d/13k-nVLPqpFkzI9CrsZS-sNYkA6IT4zOO0n7w2PKIa5Q/edit?usp=sharing)

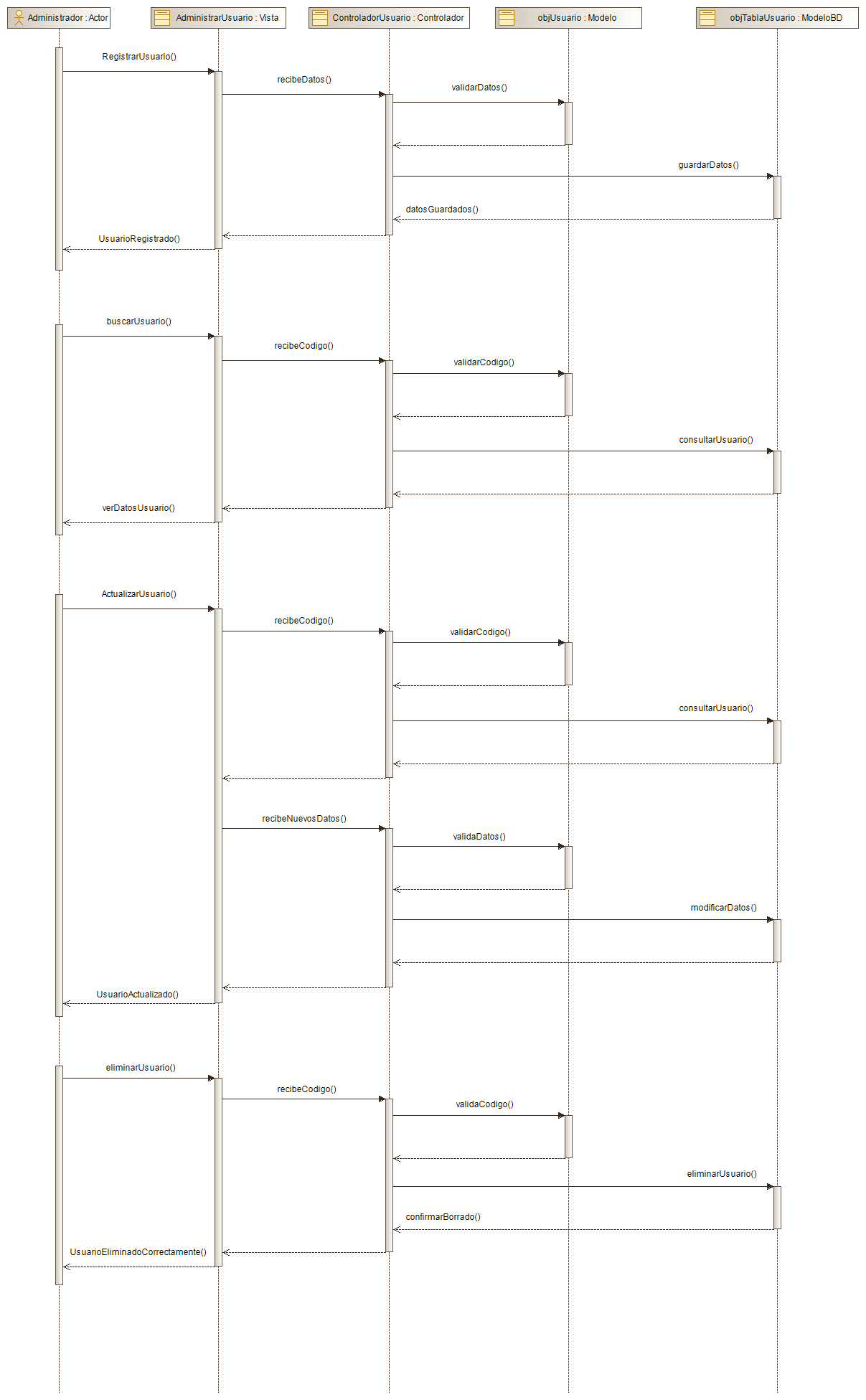


# Parte Dinámica (Diagrama de Secuencias)

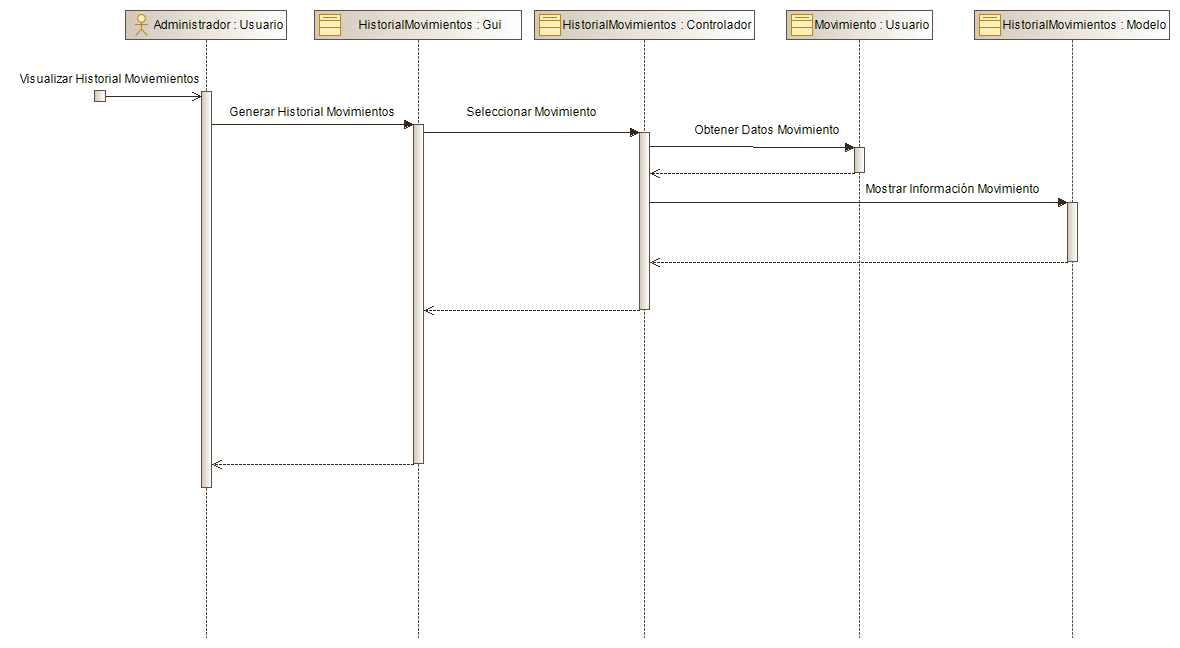
## Autenticar Usuario



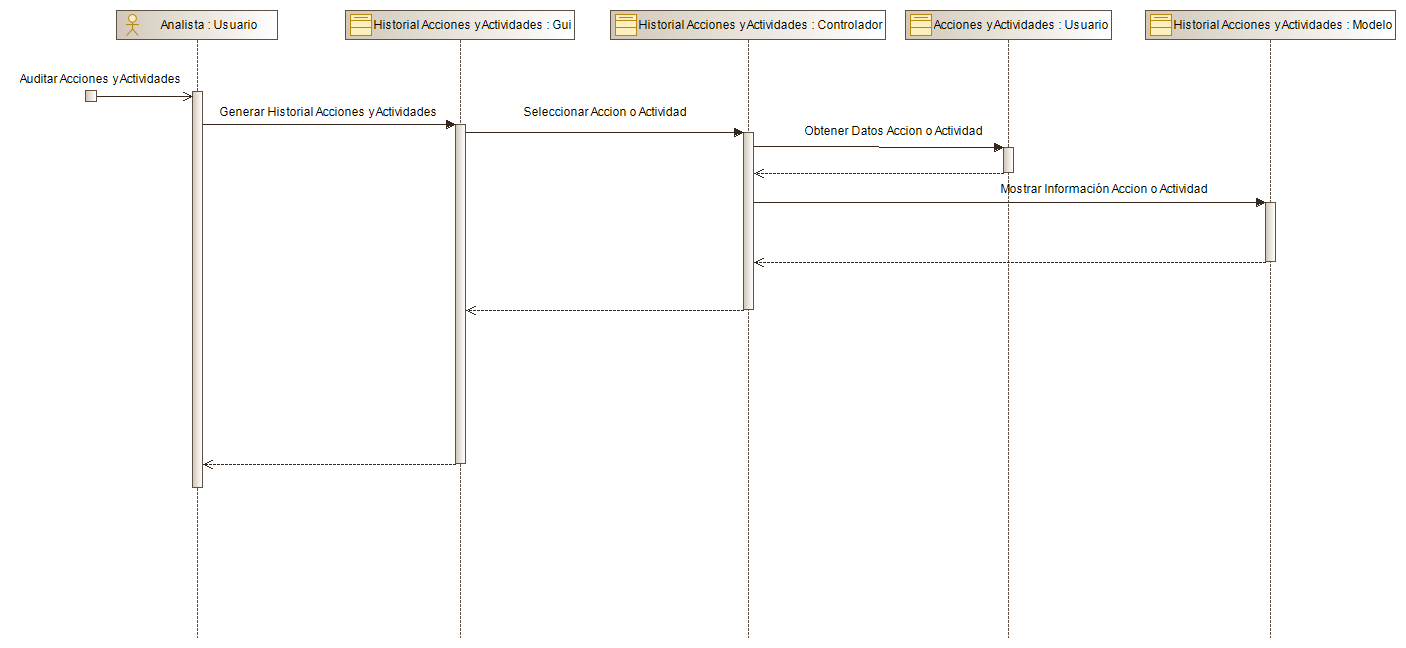
## Administrar Usuario



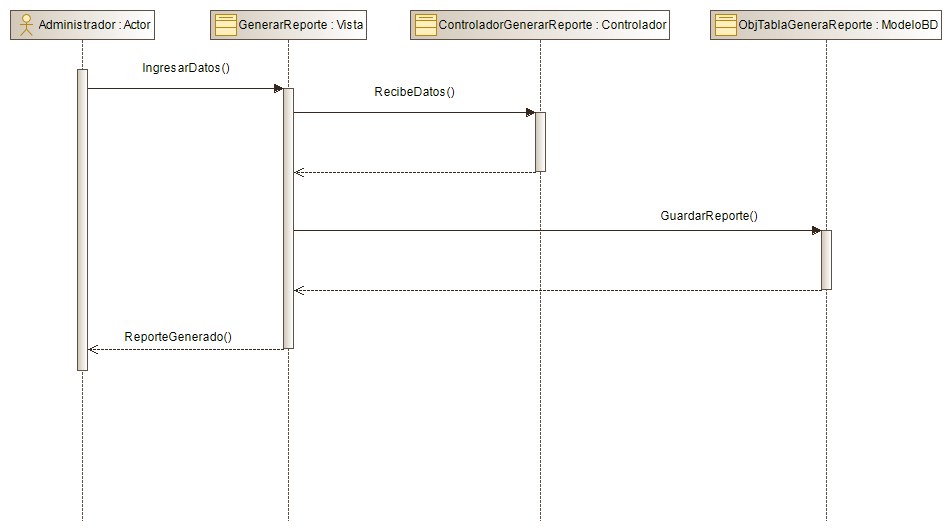
## Visualizar Historial Movimientos

****

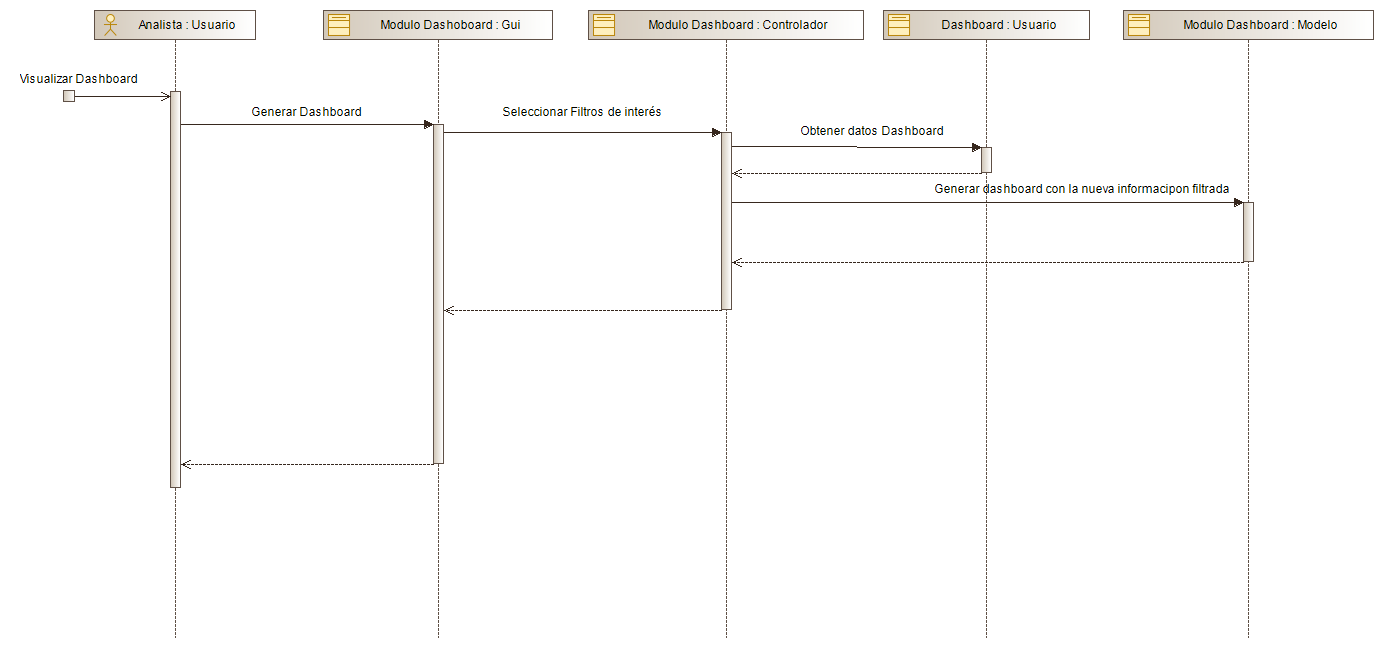
## Auditar Acciones y Actividades

****

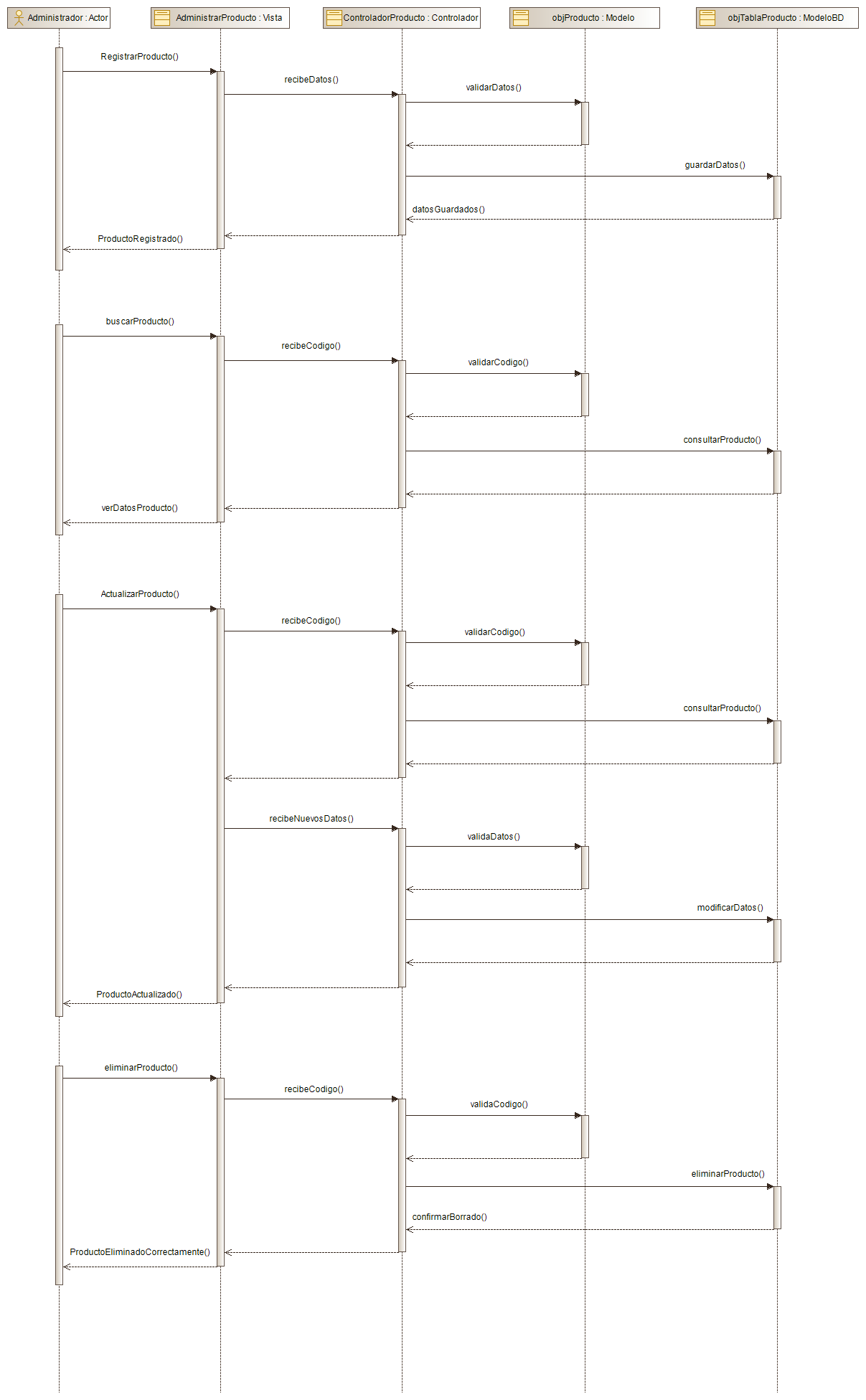
## Generar Reporte



## Visualizar Dashboard

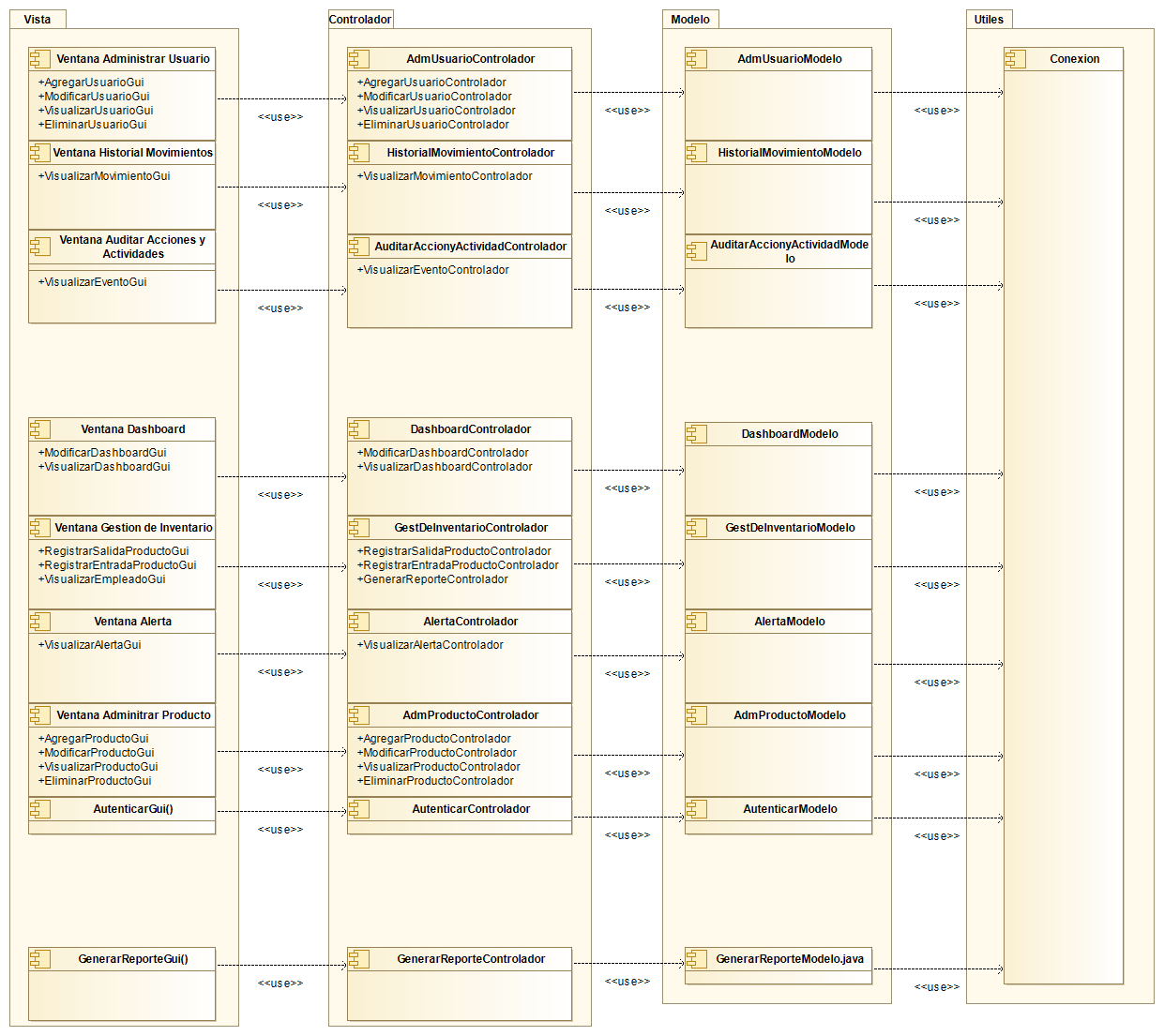
****

## Administrar Productos



# Vista de Componentes

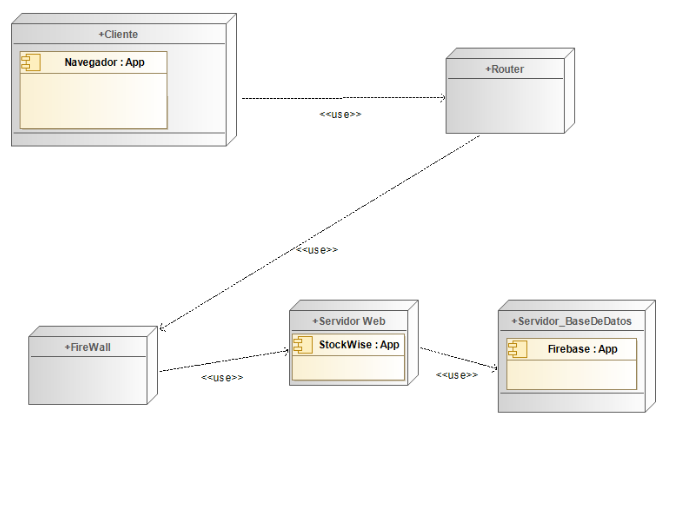
**Ilustración : Diagrama de componentes**



# **Vista Física**

En esta vista se despliegan los nodos que participan con el sistema.

**Ilustración 7: Diagrama de Despliegue**

****

# **Decisiones de Diseño y Selección de Alternativas**

Las decisiones arquitectónicas clave se tomaron considerando la restricción del tiempo de construcción. Dado que el proyecto debe completarse en un plazo ajustado de 5 meses, sin margen de holgura, se optó por una arquitectura MVC debido a su bajo riesgo de implementación.

Además, la arquitectura fue diseñada de forma modular, permitiendo que cada módulo sea testeable de manera unitaria, lo que asegura que cada componente tenga una baja tasa de fallas.

Un segundo aspecto considerado en la arquitectura fue la restricción de infraestructura que debe cumplir el sistema. Combinado con el requisito de calidad de tolerancia a fallos, esto nos llevó a desarrollar el sistema en módulos distribuidos en una aplicación web.

Finalmente, el requisito de calidad relacionado con la mantenibilidad influyó en un modelado orientado a la separación de componentes y al uso del patrón cliente-servidor. Esto permite que el sistema delegue sus requerimientos de información a sistemas externos, mediante piezas de software desacopladas, facilitando así su extensibilidad futura.

# **Análisis de Reutilización**

Nuestro sistema está desarrollado en Ionic Angular, un framework diseñado para fomentar el acoplamiento débil y la estricta separación entre las distintas partes de una aplicación y los diferentes sistemas operativos. Al seguir esta filosofía, es posible realizar cambios en áreas específicas de la aplicación sin afectar otras. En las funciones de vista, por ejemplo, destacamos la importancia de separar la lógica de negocios de la lógica de presentación, utilizando un sistema de plantillas. Aplicamos este mismo enfoque a la capa de base de datos, para un acceso lógico y estructurado a los datos.

Estas tres partes — la lógica de acceso a la base de datos, la lógica de negocios y la lógica de presentación — se integran bajo el patrón de arquitectura de software Modelo-Vista-Controlador (MVC).

En este patrón, el "Modelo" corresponde al acceso a la capa de datos, la "Vista" se encarga de la selección y presentación de la información, y el "Controlador" determina qué vista utilizar según la entrada del usuario, accediendo al modelo cuando es necesario.