Sistema de pedidos abasto Aldebarán

Documento de Arquitectura de Software

Versión <2.0>

Historia de Revisiones

|  |  |  |  |
| --- | --- | --- | --- |
| **Fecha** | **Versión** | **Descripción** | **Autor** |
| *<31/10/2024>* | *<1.0>* | *<Se añadieron las restricciones de diseño de la aplicación y el modelo de dominio>* | Santiago Chirinos  *Eros Dos Ramos*  *Mariangel Goncalves*  *Jesús Obando* |
| <01/12/2024> | <2.0> | <Añadidas las restricciones arquitectónicas, estructura de componentes, vista de datos, diagrama de clases, lista de archivos por componentes y las interfaces de usuario correspondientes al sprint 1> | Santiago Chirinos  *Eros Dos Ramos*  *Mariangel Goncalves*  *Jesús Obando* |
|  |  |  |  |
|  |  |  |  |

Tabla de Contenido

1. Parte I 4

1.1 Restricciones de Diseño. 5

1.2 Modelo de Dominio 4

1.3 Características de la Arquitectura 4

1.4 Decisiones Arquitectónicas 4

1.5 Estructura de Componentes 4

1.5.1 Diagrama de Contexto 5

1.5.2 Diagrama de Contenedores 5

1.5.3 Diagrama de Componentes 5

2. Parte II 5

2.1 Sprint 1 5

2.1.1 Vista de Datos 5

2.1.2 Diagrama de clases 5

2.1.3 Lista de Archivos por componentes 5

2.1.4 Interfaces de Usuario 5

2.2 Sprint 2… 5

Documento de Arquitectura de Software

Este documento proporciona una descripción arquitectónica del sistema de pedidos abasto Aldebarán, consta de dos partes, la realizada por el arquitecto de software y la realizada por el desarrollador. Esta última será en función de los Sprint.

# Parte I

## Restricciones de Diseño.

Para el backend de la aplicación se va a usar para programar el framework Spring mvc que utiliza el lenguaje de programación java, a su vez se utilizarán repositorios JPA para manejar la base de datos de la aplicación y así implementar la persistencia.

Para el frontend se uilizará Jquery, se programará en los lenguajes de HTML, CSS y javascript para que la aplicación pueda ser ejecutada en un navegador web. Para el frontend se utilizará el editor de código Visual Studio Code.

Para el desarrollo de la aplicación se utilizará git para llevar un historial de versiones de la aplicación y el repositorio online github para permitir el trabajo remoto de los miembros del equipo y llevar un control del código de la aplicación.

## Blank diagram (1)Modelo de Dominio

## Características de la Arquitectura

* Autorización: Ciertas funciones deben estar restringidas a los usuarios que no tengan el estatus necesario para acceder a ellas.
* Autenticación: para acceder a un perfil de cualquier tipo, el usuario deberá proveer la contraseña correspondiente a dicho perfil, de otro modo, no podrá acceder.
* Disponibilidad: El sistema debe estar operativo las 24 horas del día de al menos el 95% de los días del año para cualquier usuario que tenga una conexión a internet disponible.
* Rendimiento: Una vez recibida una solicitud, el sistema debe estar listo para enviar una respuesta en máximo 2 segundos.

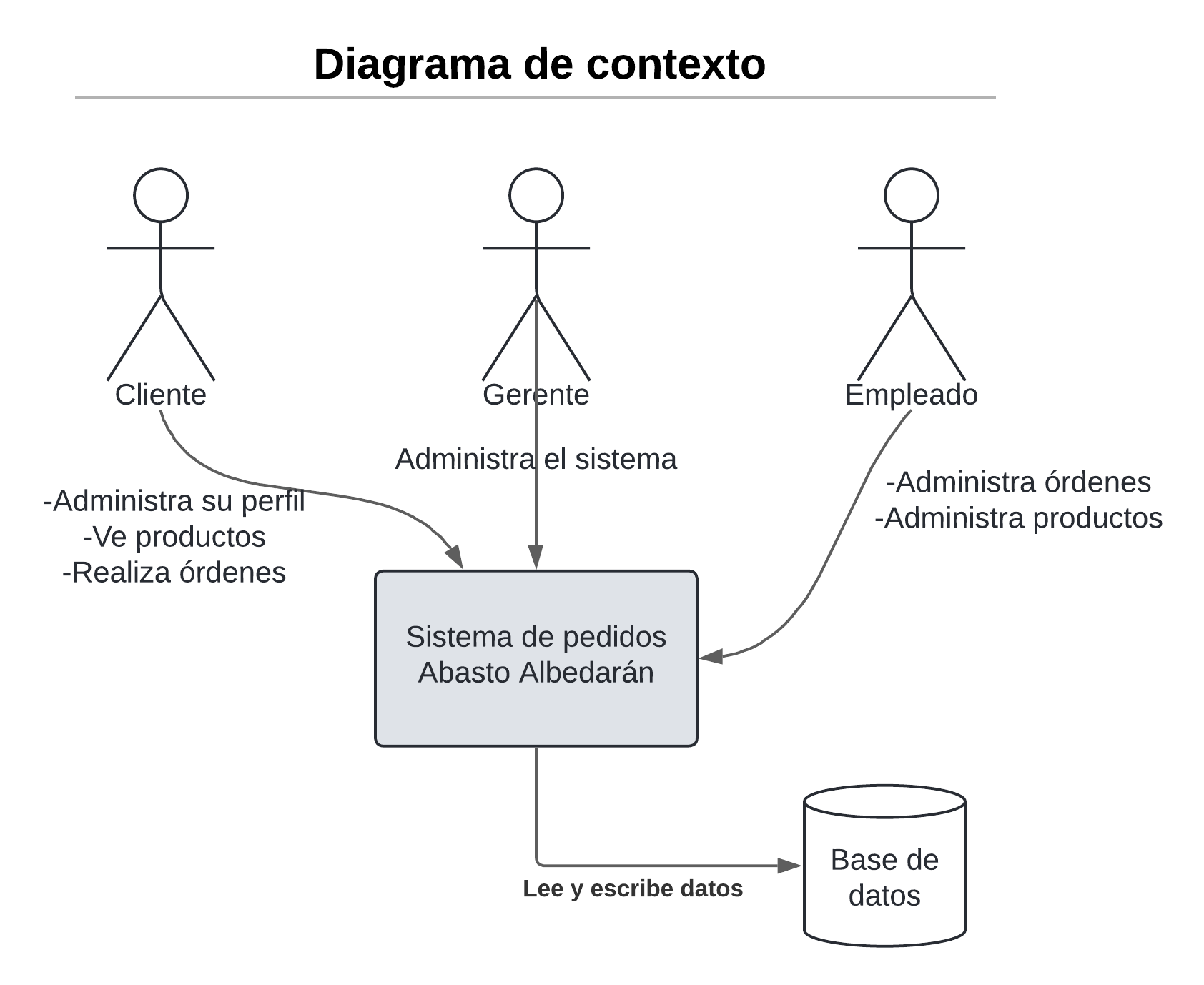
## Decisiones Arquitectónicas

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Fecha** | **Descripción** | **Justificación** |
| *<1>* | *<06/11/24>* | *Se decidió que el programa utilizará una estructura por capas divididas por funcionalidad inspiradas por el modelo MVC (model, view, controller).* | *Se tomó esta decisión porque se consideró que la estructura por capas es la más adecuada para el software que se va a diseñar y el modelo MVC es adecuado para una aplicación web* |
| *<2>* | *<06/11/24>* | *Se decidió que las conexiones a la base de datos se realizarán mediante clases dedicadas a conectarse a la base de datos, habrá una de estas clases por cada clase model* | *Se toma esta decisión para mejorar la modularidad del programa y además.*  *Con esto se espera que se pueda llevar un mejor seguimiento de los métodos de acceso a la base de datos y poder trabajar de manera más ordenada.* |
| <3> | <12/11/24> | Se decidió crear tres containers: modelos que va a contener el model del programa, controladores que contendrá los controladores y repositorios que contendrá los accesos a la base de datos | *Se tomó esta decisión para seguir la decisión arquitectónica número 1 y para poder agrupar lógicamente los archivos que se van creando en el programa* |
| <4> | <12/11/24> | Se a crear un supercontainer llamado ApirRestSB que va a contener tres containers: modelos, controladoresr y repositorios | Se toma esta decisión para garantizar la modularidad del programa, además, a la hora de desplegar el programa todo lo que se encuentre en este supercontainer será lo que se ejecutará por el lado del servidor. |
| <5> | <12/11/24> | Se decidió crear 4 containers: view que contendrá los archivos html en los que se basará la parte visual del programa, img que contendrá las imágenes usadas, css que contendrá archivos css que modificarán estéticamente los archivos en view y scripts que contendrá archivos javascript que le dará funcionalidad a los archivos en view | Se toma esta decisión para agrupar de forma lógica la gran cantidad de archivos que se va a distribuir en estos containers y para seguir la decisión arquitectonica número 1 |
| <6> | <12/11/24> | Se va a crear un supercontainer llamado resources que va a contener 4 containers: css, view, scripts y img | Se toma esta decisión para garantizar la modularidad del programa, además, a la hora de desplegar el programa todo lo que se encuentre en este supercontainer se va a ejecutar en el lado del cliente. |

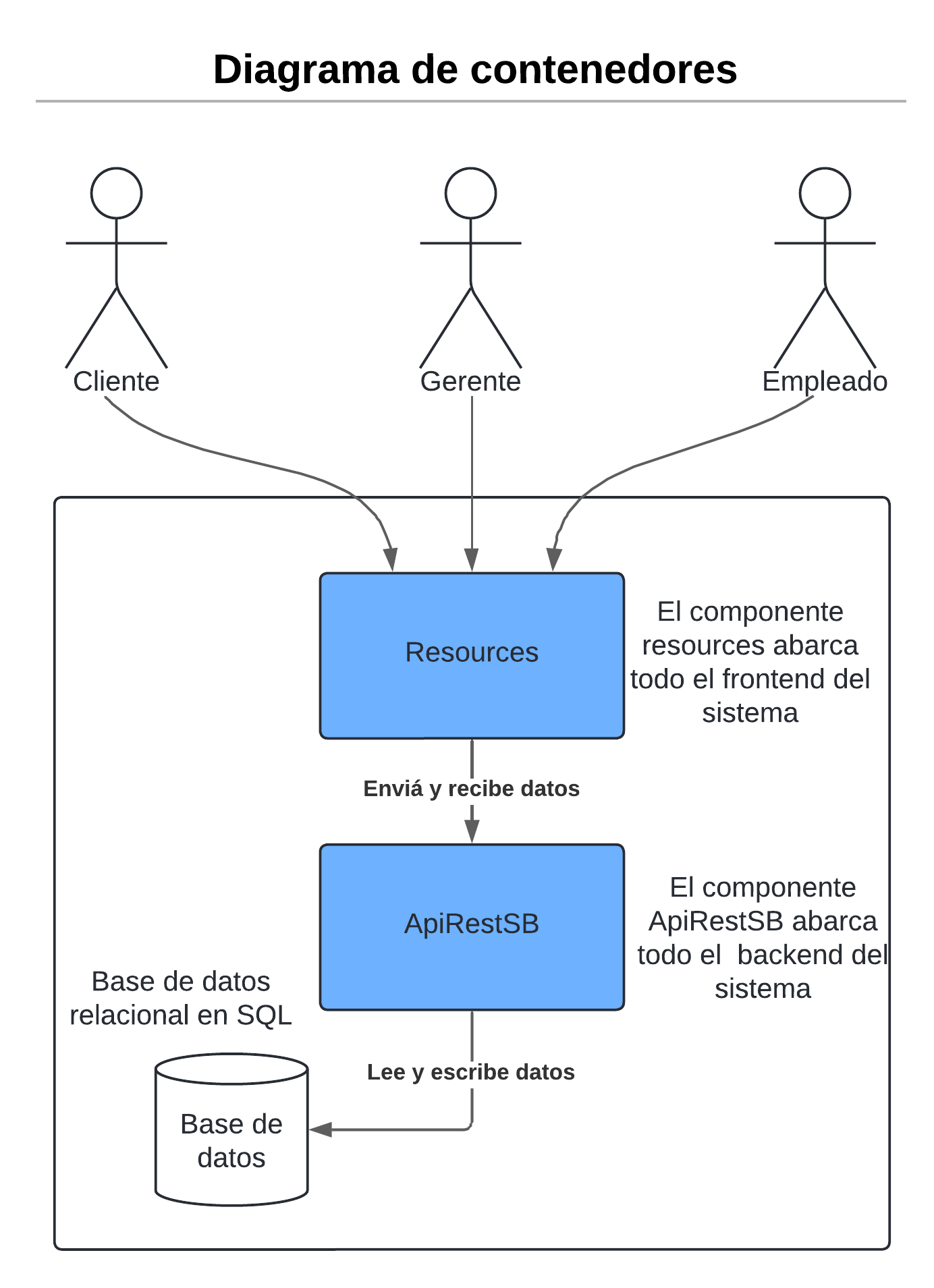
## Estructura de Componentes

Los diagramas desarrollados en esta entrega se pueden ver a través del siguiente enlace: <https://drive.google.com/drive/folders/1Wd4UYP4IWK5uLLJIyeYubAXzZNsZO92H?usp=sharing>

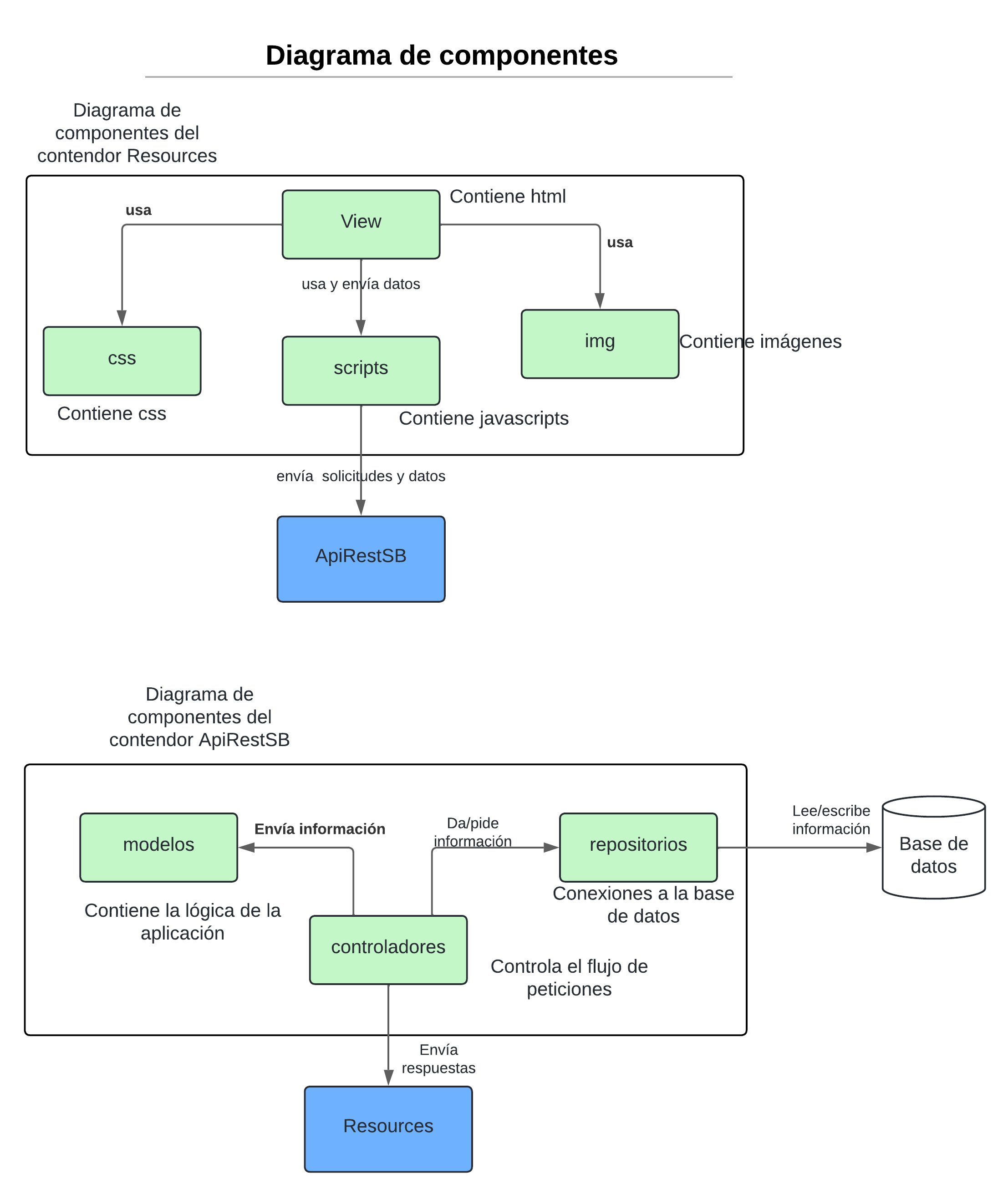
### Diagrama de Contexto

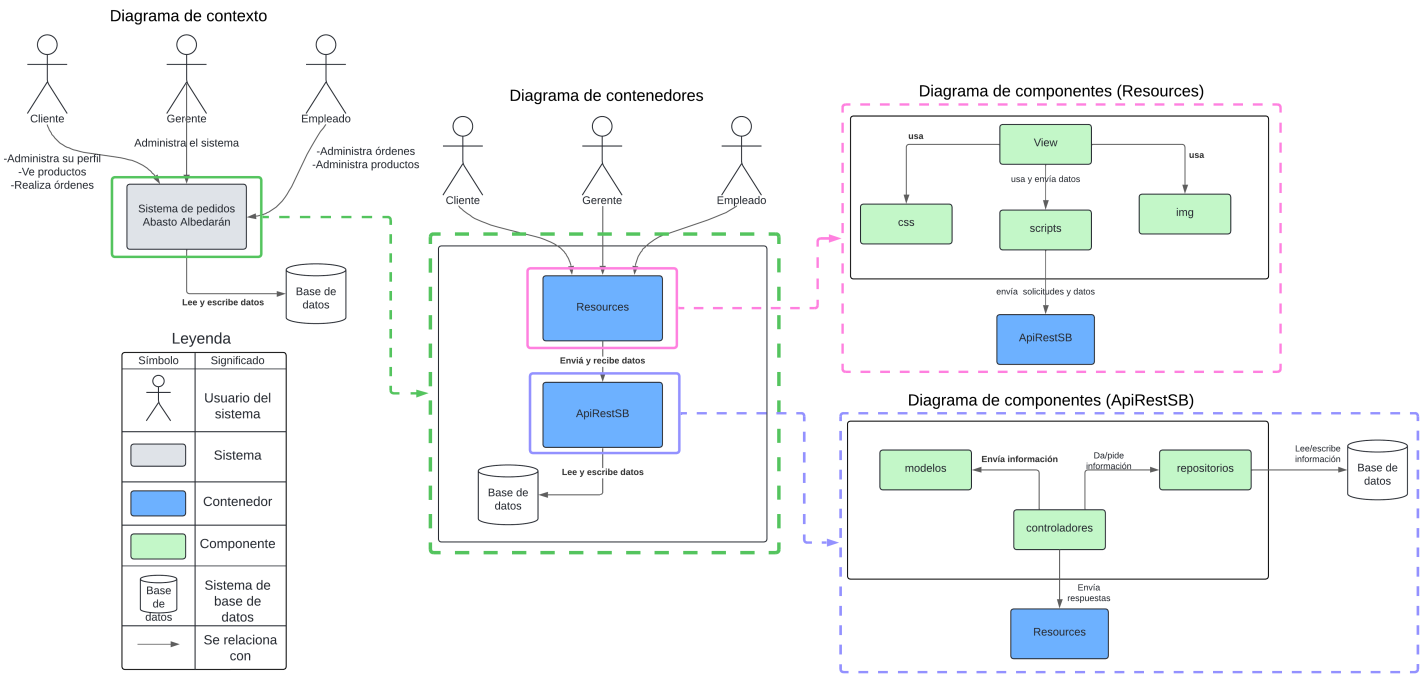


### Diagrama de Contenedores



### Diagrama de Componentes





# Parte II

[La elabora el Desarrollador y se ira modificando por cada Sprint que se desarrolle.]

## Sprint 1

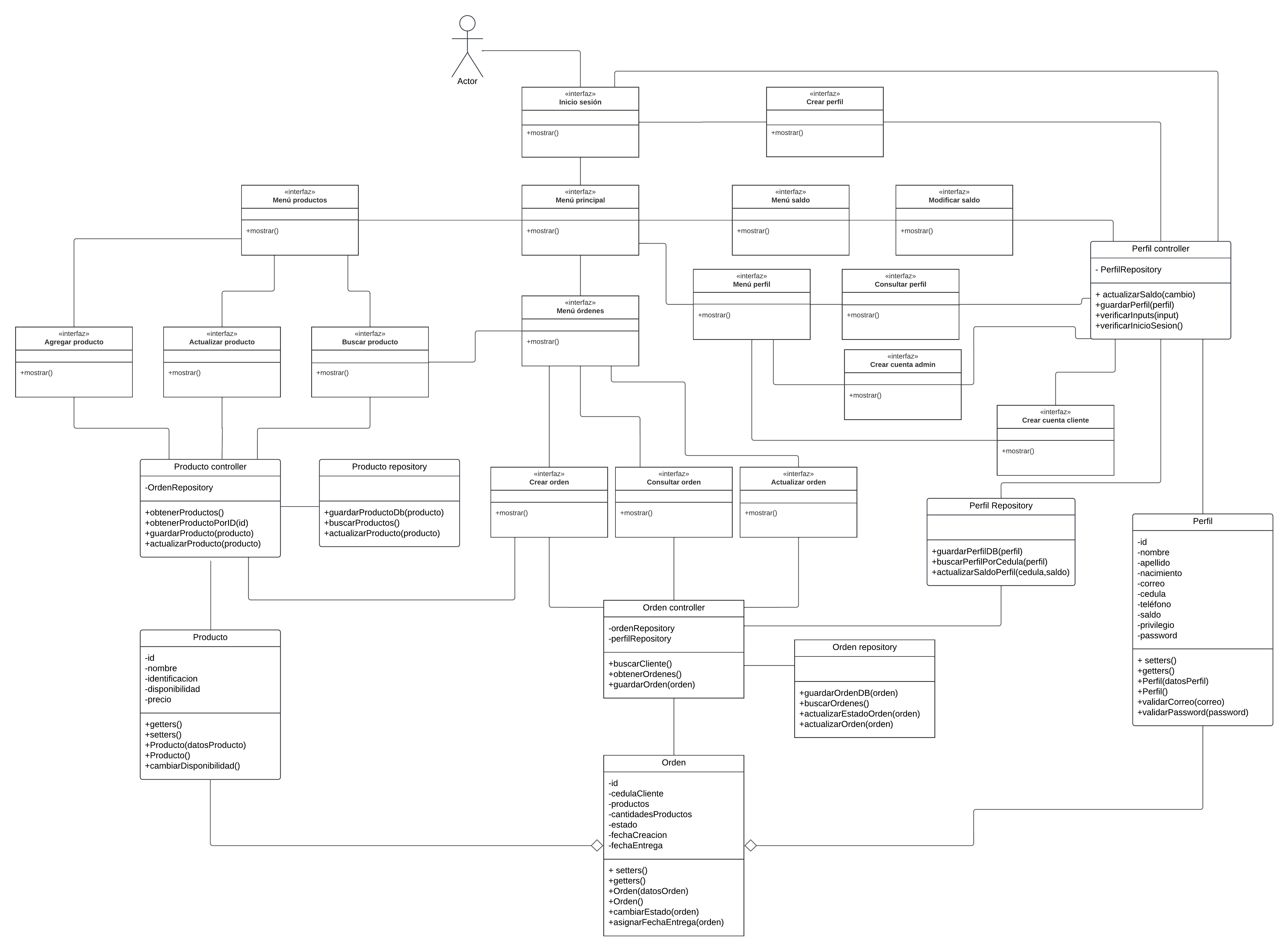
Historias desarrolladas en el sprint 1:

* Crear perfil empleado.
* Crear perfil cliente.
* Consultar perfil.
* Crear orden.
* Buscar producto.
* Consultar órdenes.
* Modificar estado orden.
* Modificar Saldo.
* Crear producto.
* Consultar producto.

### Vista de Datos

[no se va a realizar en este curso]

### Diagrama de clases



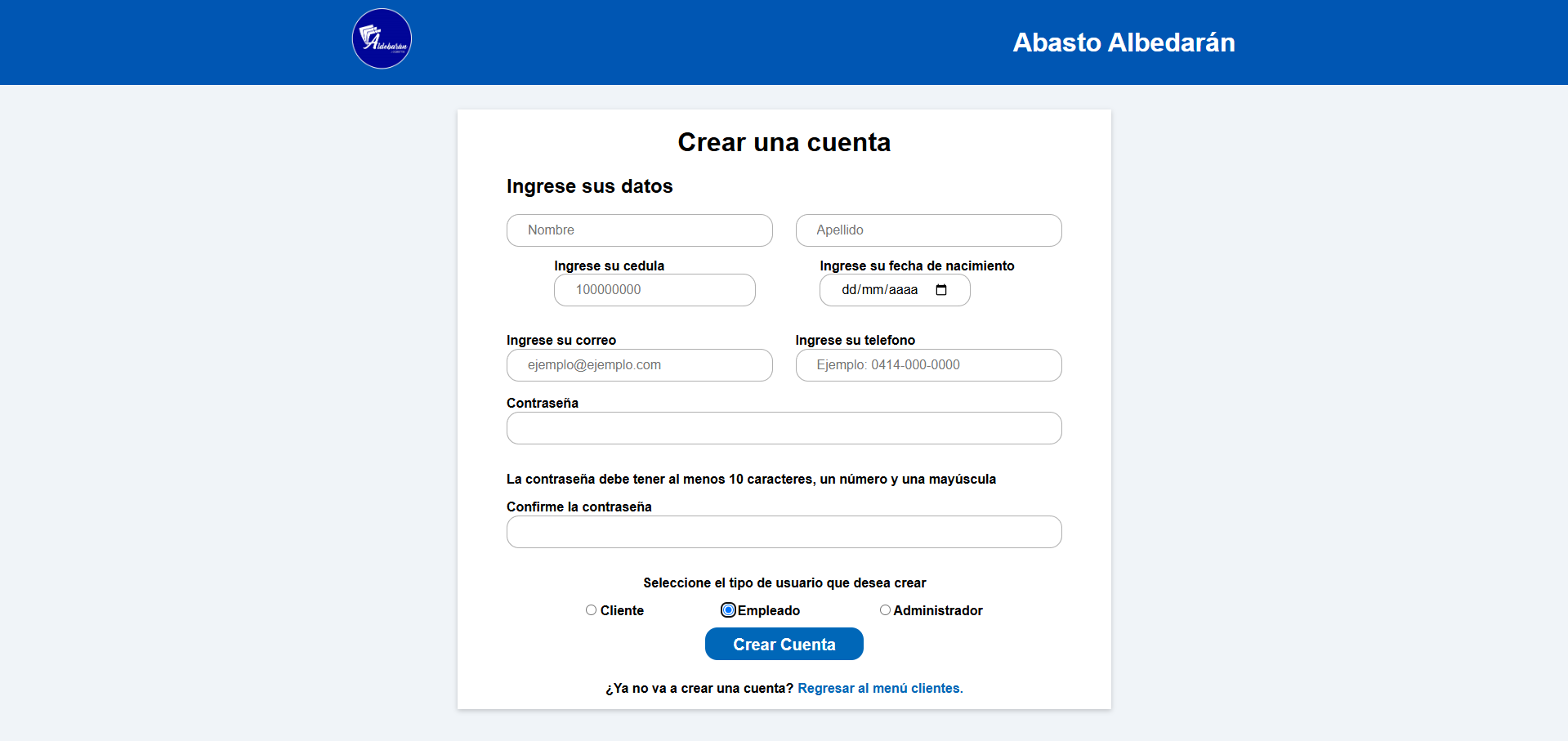
### Lista de Archivos por componentes

|  |  |
| --- | --- |
| **Nombre del Componente** | **Lista de archivos** |
| Controladores | * OrdenController.java * PerfilController.java * ProductoController.java |
| Modelos | * Orden.java * Perfil.java * Producto.java |
| Repositorios | * OrdenRepository.java * PerfilRepository.java * ProductoRepository.java |
| CSS | * ConsultarPerfil.css * ManejoSaldo.css * MenuClientes.css * MenuPrincipal.css * css.css * normalize.css * styleIS.css * styleNewaccount.css * styleRegistrarProducto.css * table.css |
| Img | * 1.png * LogoAldebaran.png * Placeholder.webp * Publicidad1.jpg * Publicidad2.jpg * Publicidad3.jpg * Publicidad4.png.jpg * Publicidad5.png.jpg * Publicidad7.png.jpg * aerial-beautiful-shot-seashore-with-hills-background-sunset.jpg * lone-tree.jpg * misurina-sunset.jpg |
| Scripts | * Consultarproducto.js * MenuPrincipal.js * consultarPerfil.js * modificarSaldo.js * scriptActualizarEstadoOrden.js * scriptActualizarProducto.js * scriptAgregarProducto.js * scriptConsultarOrden.js * scriptCrearCuentaAdmin.js * scriptCrearCuentaCliente.js * scriptInicioSesion.js * scriptRegistrarOrden.js * scriptTablaProductos.js * table.js |
| View | * ActualizarEstadoOrden.html * ActualizarProducto.html * AgregarProducto.html * ConsultarOrden.html * ConsultarPerfil.html * ConsultarProducto.html * CrearOrden.html * InicioSesion.html * ManejoSaldo.html * MenuClientes.html * MenuOrdenes.html * MenuProductos.html * MenuSaldo.html * NuevaCuentaAdmin.html * NuevaCuentaCliente.html * NuevaCuentaClienteMenu.html * menuPrincipal.html * table.html |

### Interfaces de Usuario

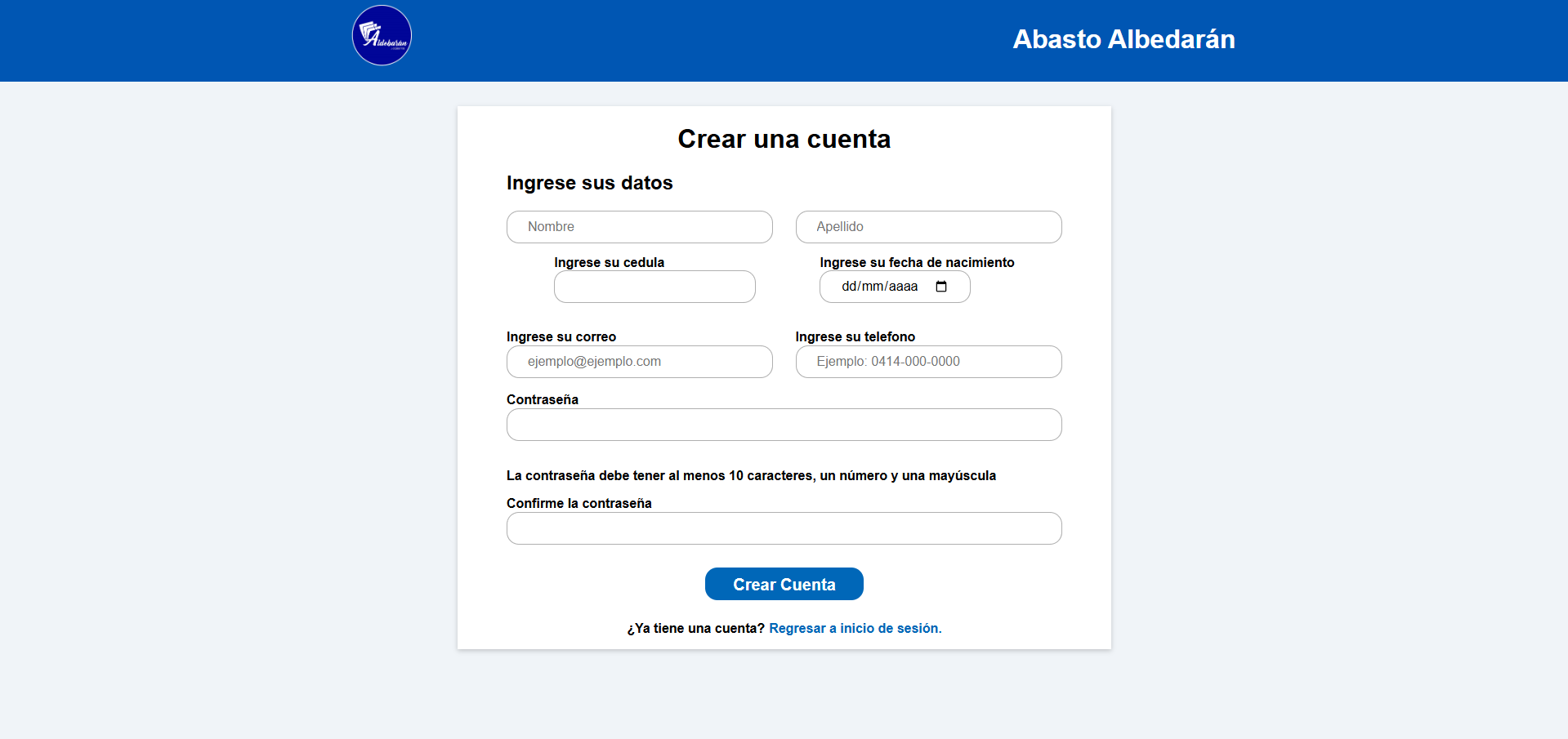
NuevaCuentaAdmin

Pertenece a la historia Crear perfil empleado



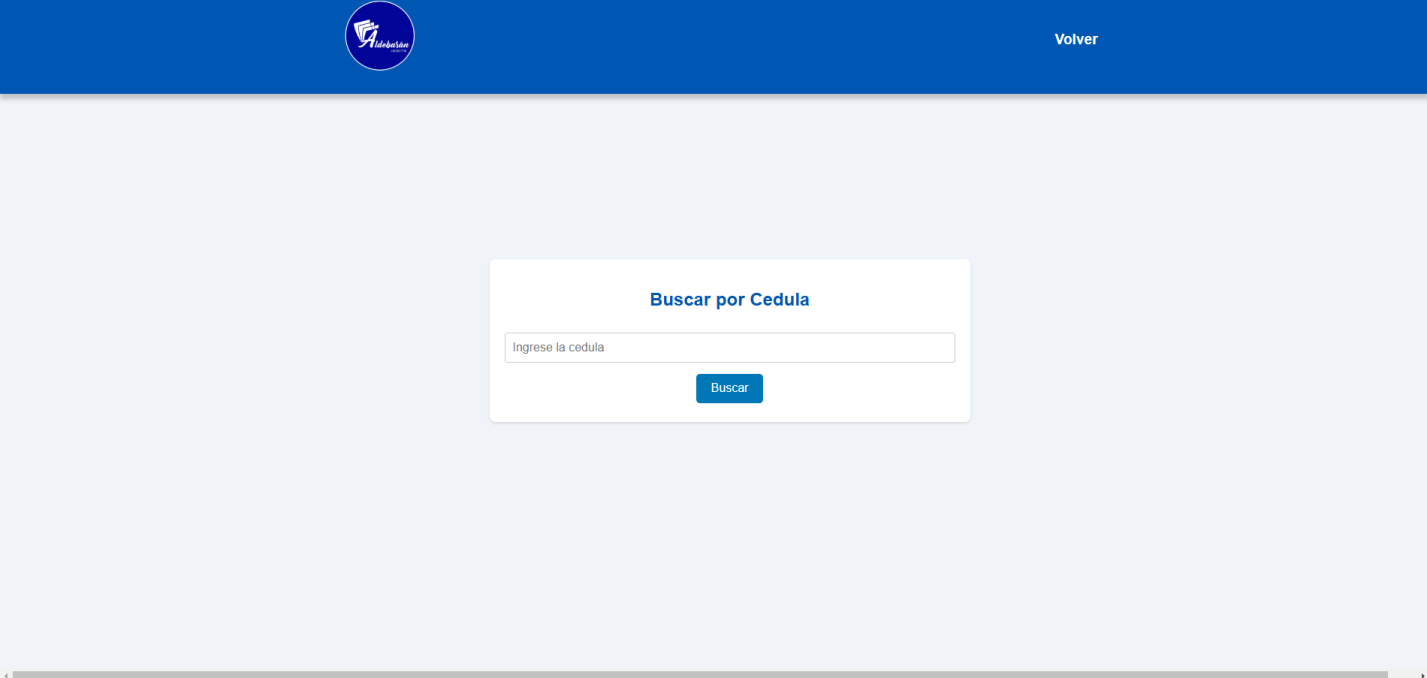
NuevaCuentaCliente

Pertenece a la historia Crear Perfil Cliente del entregable Perfil



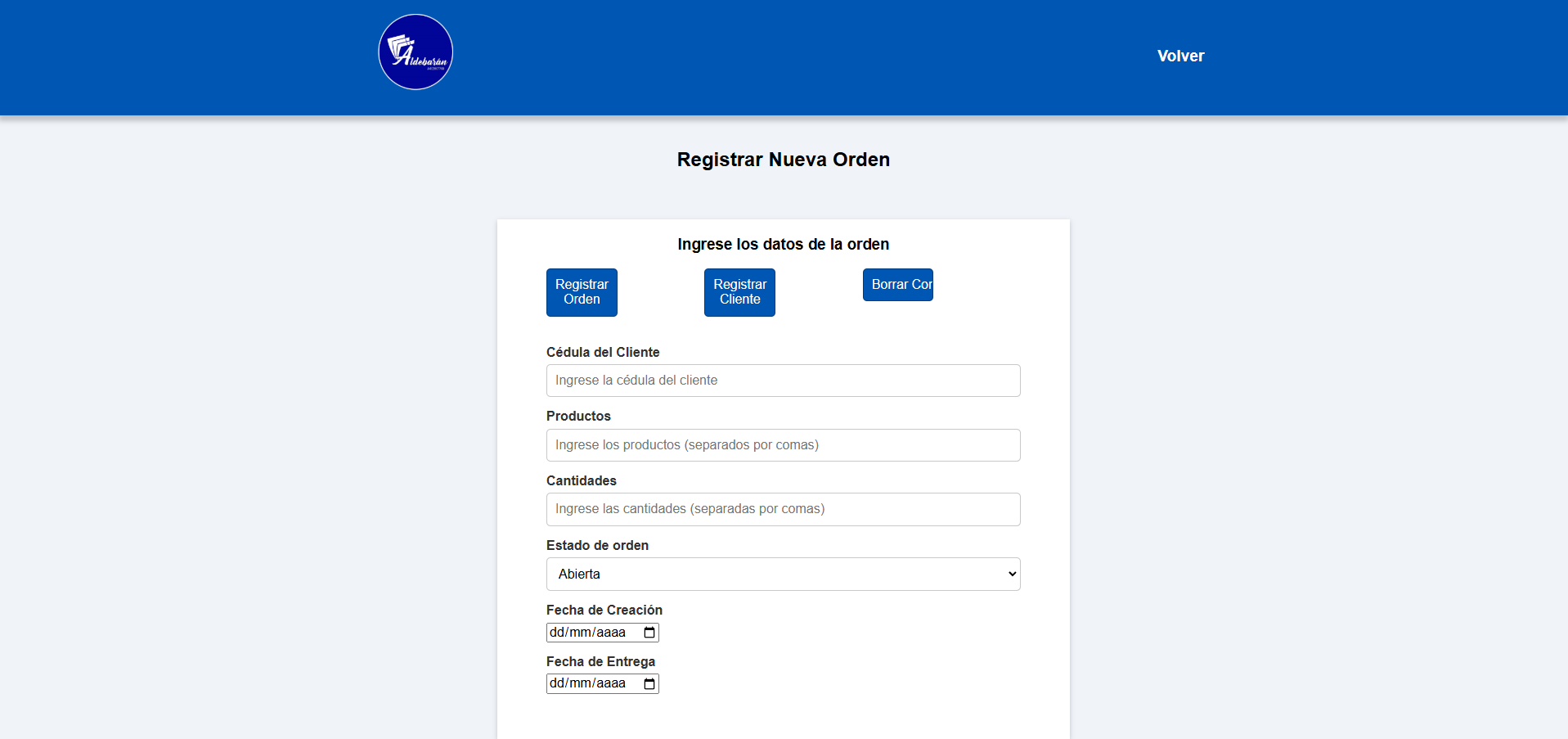
ConsultarPerfil

Pertenece a la historia Consultar Perfil del entregable Perfil



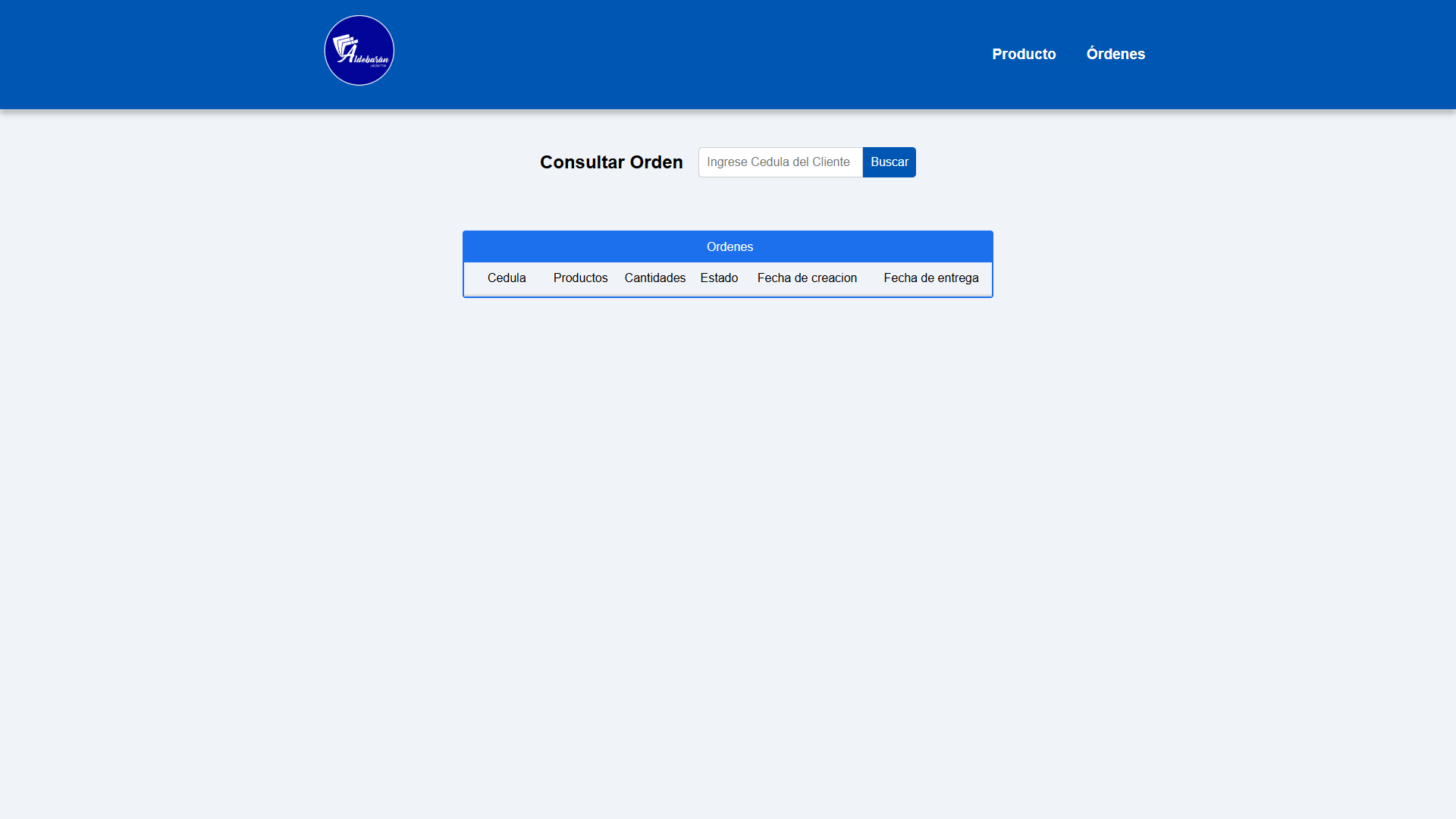
CrearOrden

Pertenece a la historia Crear Orden del entregable Orden



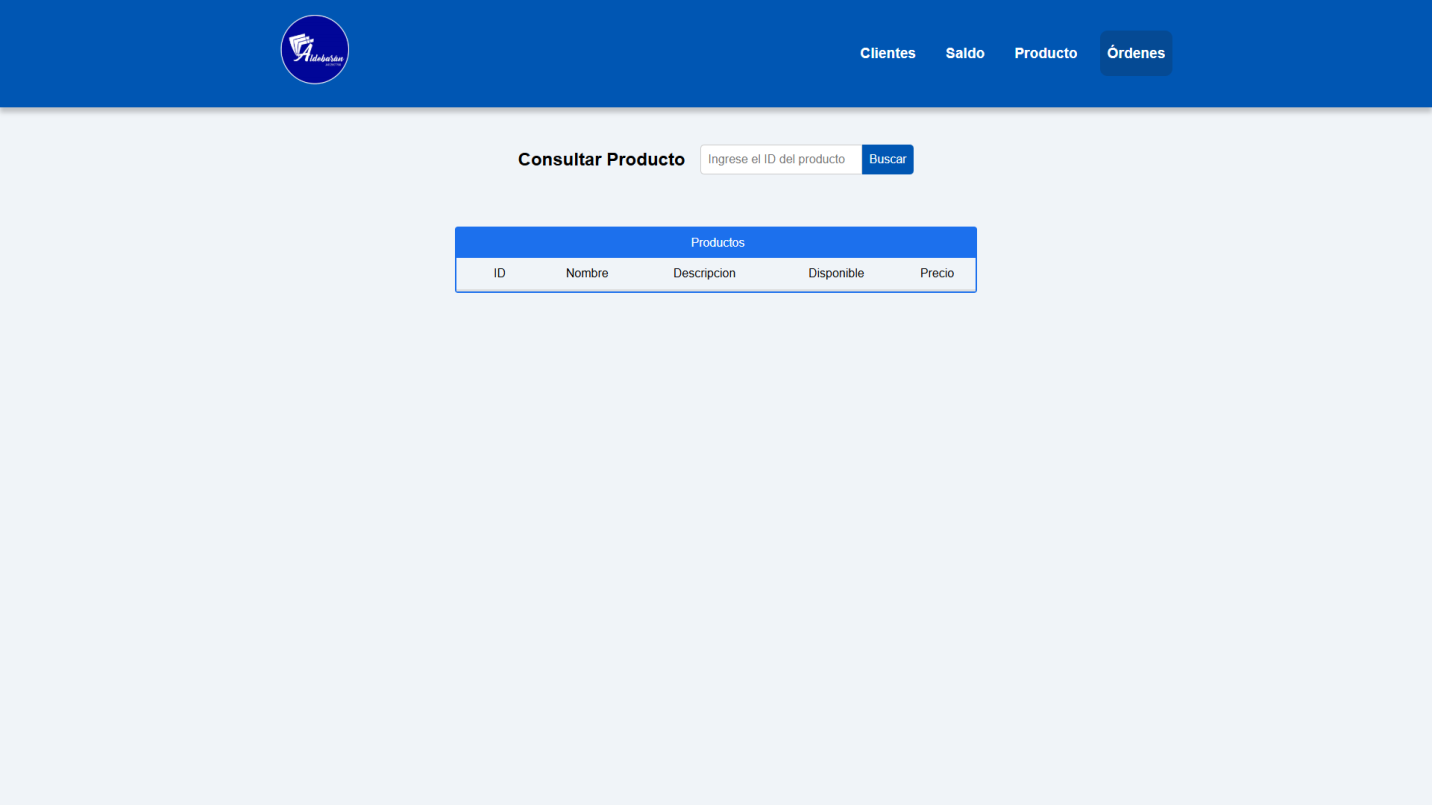
ConsultarOrden

Pertenece a la historia Consultar Orden del entregable Orden



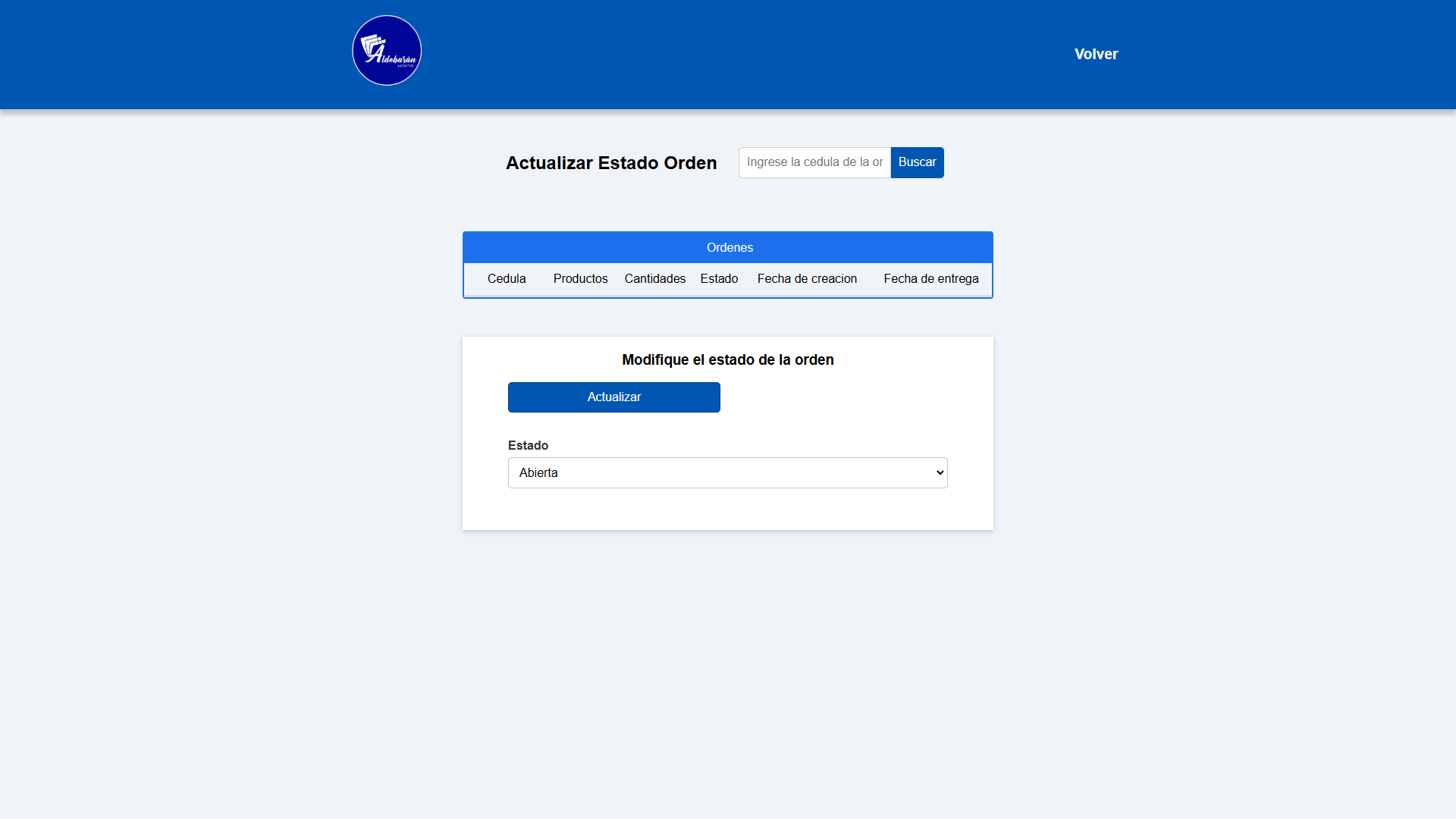
BuscarProducto

Pertenece a la historia Buscar Producto del entregable Orden y a Consultar Producto del entregable Producto



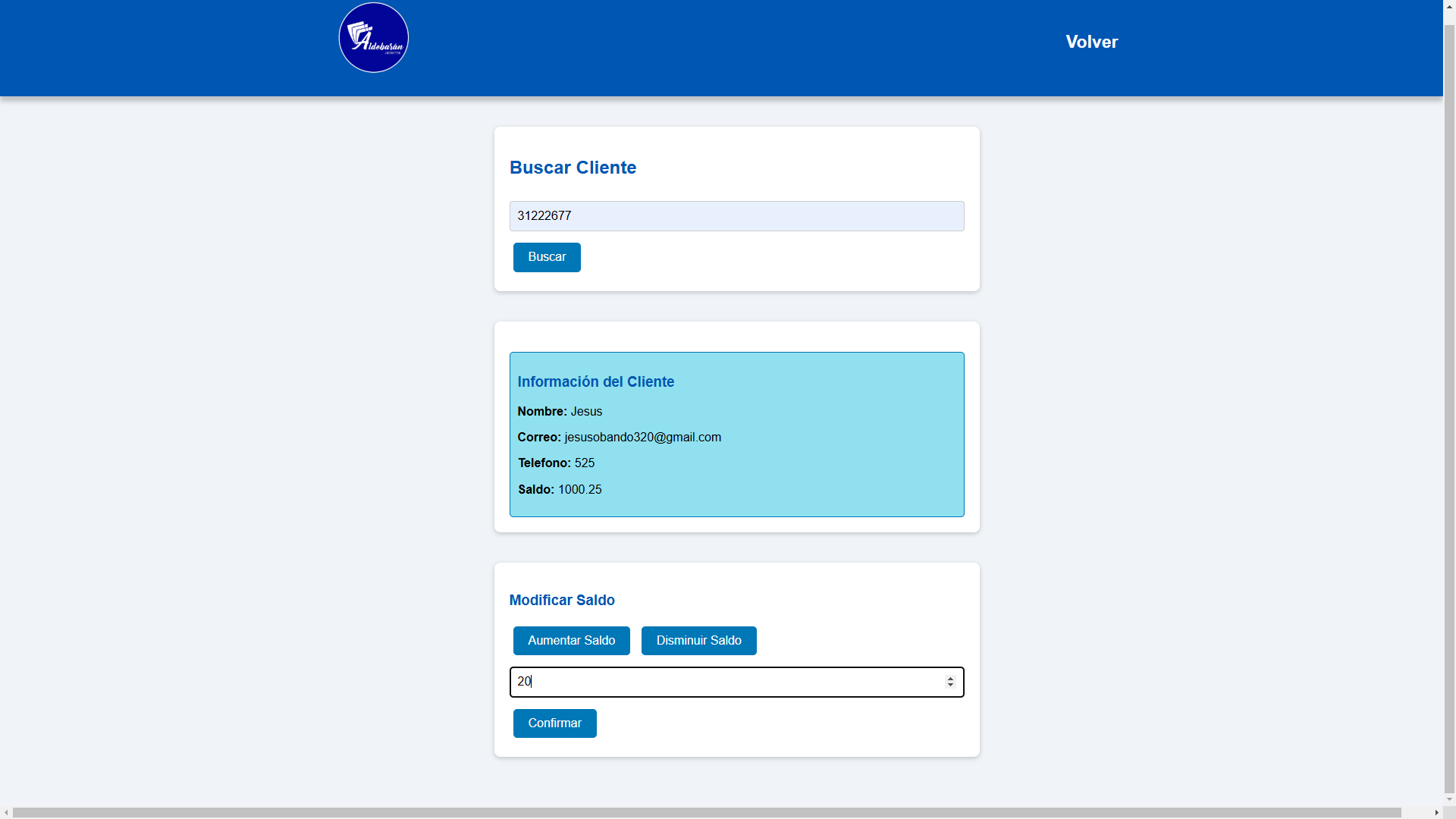
ActualizarEstadoOrden

Pertenece a la historia Modificar Estado Orden del entregable Orden



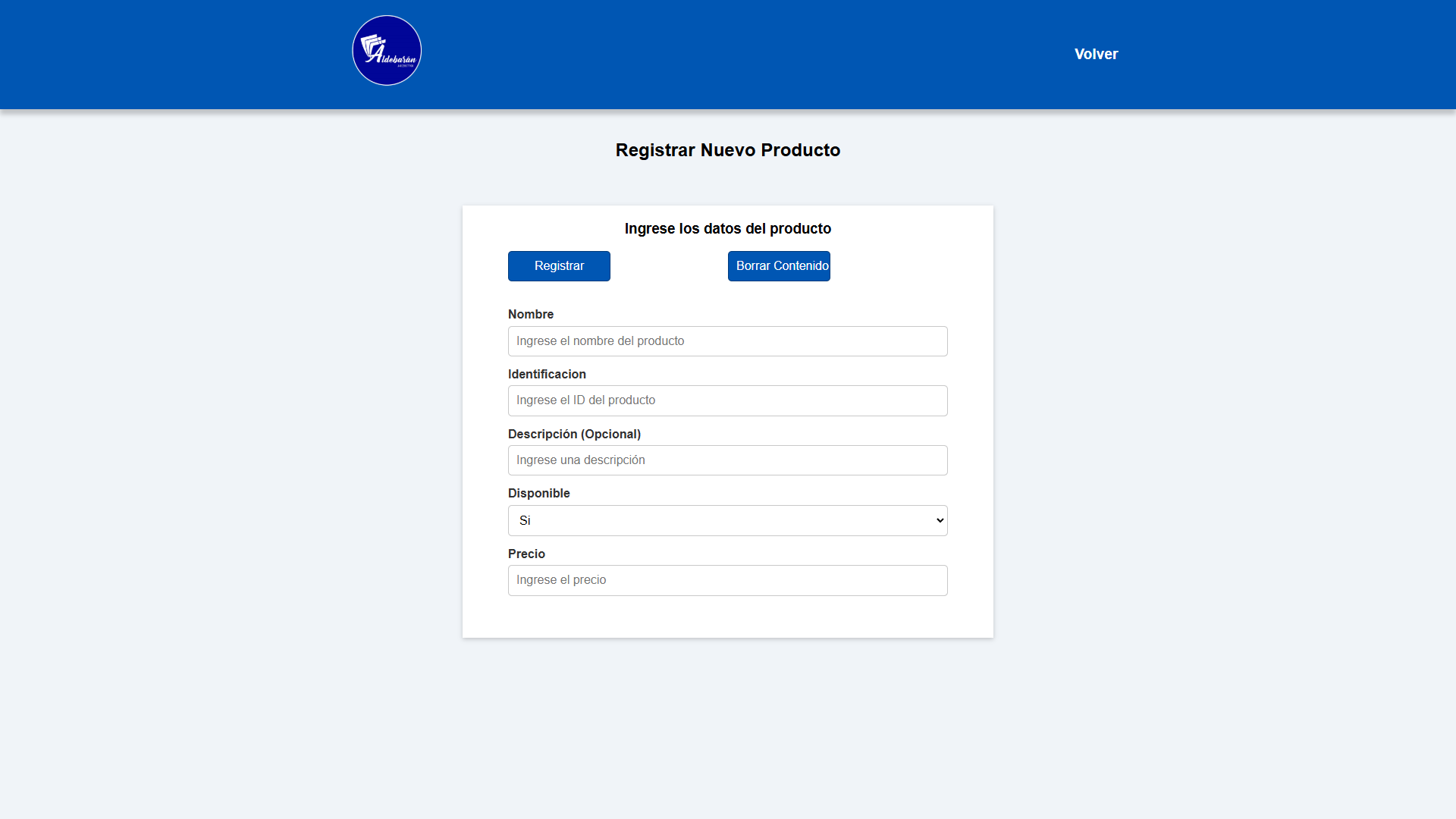
ManejoSaldo

Pertenece a la historia Manejar Saldo del entregable Saldo



AgregarProducto

Pertenece a la historia Crear Producto del entregable Producto



## Sprint 2…