|  |
| --- |
|  |

ESPECIFICACIÓN DE REQUERIMIENTOS DE SOFTWARE

Aplicación móvil para la optimización de procesos y control en la empresa de servicios de limpieza “Go Cleaning" en la ciudad de Quito.

|  |  |  |
| --- | --- | --- |
|  |  | Junio 2025 |

CONTENIDO

[1. INTRODUCCIÓN 4](#_Toc209542502)

[1.1 Propósito 4](#_Toc209542503)

[1.2 Alcance 4](#_Toc209542504)

[1.3 Personal involucrado 4](#_Toc209542505)

[1.4 Definiciones, acrónimos y abreviaturas 5](#_Toc209542506)

[1.5 Referencias 9](#_Toc209542507)

[1.6 Resumen 9](#_Toc209542508)

[2 DESCRIPCIÓN GENERAL 9](#_Toc209542509)

[2.1 Perspectiva del producto 9](#_Toc209542510)

[2.2 Funcionalidad del producto 10](#_Toc209542511)

[2.3 Características de los usuarios 10](#_Toc209542512)

[2.4 Restricciones 13](#_Toc209542513)

[2.5 Suposiciones y dependencias 14](#_Toc209542514)

[2.6 Evolución previsible del sistema 14](#_Toc209542515)

[3 REQUISITOS ESPECÍFICOS 15](#_Toc209542516)

[3.1 Requisitos comunes de los interfaces 15](#_Toc209542517)

[3.1.1 Interfaces de usuario 15](#_Toc209542518)

[3.1.2 Interfaces de hardware 15](#_Toc209542519)

[3.1.3 Interfaces de software 15](#_Toc209542520)

[3.1.4 Interfaces de comunicación 16](#_Toc209542521)

[3.1 Requisitos funcionales 16](#_Toc209542522)

[3.1.1 Requisito funcional 1: Control de Pedidos 16](#_Toc209542523)

[3.1.2 Requisito funcional 2: Control de Clientes 17](#_Toc209542524)

[3.1.3 Requisito funcional 3: Control de Servicios 17](#_Toc209542525)

[3.1.4 Requisito funcional 4: Asignación de Tareas 17](#_Toc209542526)

[3.1.5 Requisito funcional 5: Control de precios 18](#_Toc209542527)

[3.1.6 Requisito funcional 6: Notificaciones. 18](#_Toc209542528)

[3.1.7 Requisito funcional 7: Control de Usuarios 18](#_Toc209542529)

[3.1.8 Requisito funcional 8: Control de Descuentos 19](#_Toc209542530)

[3.2 Requisitos no funcionales 19](#_Toc209542531)

[3.2.1 Requisitos de rendimiento 19](#_Toc209542532)

[3.2.2 Seguridad 19](#_Toc209542533)

[3.2.3 Fiabilidad 20](#_Toc209542534)

[3.2.4 Disponibilidad 20](#_Toc209542535)

[3.2.5 Mantenibilidad 20](#_Toc209542536)

[3.2.6 Portabilidad 20](#_Toc209542537)

[4 APÉNDICES 20](#_Toc209542538)

[4.1 Diagrama de Casos de Uso 21](#_Toc209542539)

[5 Glosario 22](#_Toc209542540)

# INTRODUCCIÓN

## Propósito

El propósito de este documento es especificar los requisitos para el desarrollo de una aplicación móvil que optimice la operativa de la empresa "GO Cleaning" ubicada en Quito. Este documento tiene la finalidad de proporcionar una guía clara para el desarrollo de una solución eficiente que abarque el control de pedidos, clientes, servicios, precios y asignación de tareas, así como la capacidad de enviar notificaciones automáticas. Está dirigido a los desarrolladores, analistas de negocio, usuarios finales y el equipo de aseguramiento de calidad que participarán en el proyecto.

## Alcance

La aplicación móvil por desarrollar permitirá la administración completa de pedidos de servicios de limpieza, clientes, servicios y precios, además de facilitar la asignación de tareas a los empleados. La solución se integrará con una base de datos MySQL, y se utilizarán PHP 8.3.1 para el backend API, Angular V17 e IONIC para el desarrollo híbrido móvil. La aplicación estará diseñada para funcionar en dispositivos iOS y Android, garantizando su funcionamiento óptimo en ambas plataformas.

## Personal involucrado

* **Desarrolladores**: Encargados de la implementación técnica del sistema móvil.
* **Analistas de negocio**: Encargados de definir los requisitos y validar que el sistema cumpla con las necesidades de la empresa de servicios de limpieza "Go Cleaning".
* **Usuarios finales**: Empleados de "Go Cleaning" que interactuarán con la aplicación móvil.
* **Equipo de QA**: Responsable de las pruebas y validación de la aplicación móvil.

## Definiciones, acrónimos y abreviaturas

* **RF**: Requerimiento Funcional.
* **RNF**: Requerimiento No Funcional.
* **API**: Interfaz de Programación de Aplicaciones. Conjunto de definiciones y protocolos para construir e integrar software de aplicaciones.
* **CRUD**: Crear, Leer, Actualizar, Eliminar. Conjunto de operaciones básicas que se pueden realizar en la gestión de datos.
* **CSS**: Hojas de Estilo en Cascada. Lenguaje de diseño que se utiliza para describir la presentación de un documento escrito en HTML o XML.
* **DBMS**: Sistema de Gestión de Bases de Datos. Software que facilita la creación, administración y manipulación de bases de datos.
* **Frontend**: Parte de la aplicación que interactúa directamente con el usuario. Se refiere a la interfaz y a la experiencia del usuario.
* **Backend**: Parte del software que gestiona la lógica del servidor y la manipulación de datos detrás de escena, no visible para el usuario.
* **HTML**: Lenguaje de Marcado de Hipertexto. Lenguaje estándar para crear y diseñar páginas web.
* **JavaScript**: Lenguaje de programación que permite la creación de contenido dinámico e interactivo en páginas web.
* **PHP**: Hypertext Preprocessor. Lenguaje de programación del lado del servidor utilizado para desarrollar aplicaciones web.
* **Angular**: Framework para el desarrollo del frontend móvil.
* **IONIC**: Framework para el desarrollo de aplicaciones móviles híbridas.
* **SRS**: Documento de Especificación de Requisitos del Software. Documento que describe el comportamiento del sistema de software a desarrollar.
* **SQL**: Lenguaje de Consulta Estructurado. Lenguaje de programación para gestionar y manipular bases de datos relacionales.
* **UI**: Interfaz de Usuario. Diseño y disposición de los elementos de una aplicación que permiten la interacción del usuario con el sistema.
* **UX**: Experiencia del Usuario. Se refiere a la percepción y respuestas de un usuario sobre la interfaz de un producto, especialmente en términos de facilidad de uso.
* **MVC**: Modelo-Vista-Controlador. Patrón de diseño de software que separa una aplicación en tres componentes interconectados para promover la separación de responsabilidades.
* **MySQL**: Sistema de Gestión de Bases de Datos Relacional de código abierto basado en SQL.
* **REST**: Transferencia de Estado Representacional. Estilo de arquitectura de software para sistemas distribuidos como la web.
* **SSL**: Capa de Conexión Segura. Protocolo de seguridad para cifrar la comunicación entre el navegador del usuario y el servidor web.
* **AJAX**: Asynchronous JavaScript and XML. Técnica para crear aplicaciones web interactivas mediante la actualización asíncrona de partes de la página web.
* **Cliente/Servidor**: Arquitectura de red en la que un servidor proporciona servicios o recursos a uno o varios clientes. El cliente solicita servicios o recursos y el servidor los proporciona.
* **Composer**: Herramienta de gestión de dependencias en PHP. Permite a los desarrolladores gestionar y instalar librerías y paquetes necesarios para sus proyectos PHP de manera sencilla y eficiente.
* **TypeScript**: Lenguaje de programación desarrollado por Microsoft que es un superset de JavaScript. Añade tipado estático opcional y otras características avanzadas a JavaScript, facilitando la detección de errores y el desarrollo a gran escala.
* **Material Design V3**: Un sistema de diseño creado por Google que proporciona pautas y componentes predefinidos para la creación de interfaces de usuario coherentes y atractivas.
* **Bootstrap**: Un framework de front-end que facilita el desarrollo de sitios web y aplicaciones móviles con un diseño responsivo. Bootstrap proporciona una colección de herramientas CSS y JavaScript.
* **Node.js**: Entorno de ejecución para JavaScript en el servidor basado en el motor V8 de Google Chrome. Permite la ejecución de código JavaScript fuera del navegador, facilitando el desarrollo de aplicaciones del lado del servidor y de red.
* **JSON**: JavaScript Object Notation. Formato ligero para el intercambio de datos, fácil de leer y escribir para los humanos y fácil de analizar y generar para las máquinas. Se utiliza comúnmente para transmitir datos entre un servidor y una aplicación web.
* **Token**: En el contexto de seguridad, un token es un objeto de datos que se utiliza para autenticar y autorizar solicitudes en un sistema. Puede ser un valor que se envía con cada solicitud para verificar la identidad del usuario.
* **JWT**: JSON Web Token. Estándar abierto para la transmisión segura de información entre partes en formato JSON. Los JWT son compactos, seguros y se utilizan comúnmente para la autenticación y autorización en aplicaciones web.
* **Módulo**: Unidad de código que agrupa funciones, clases o variables relacionadas con una funcionalidad específica. Los módulos permiten una mejor organización y gestión del código, favoreciendo la reutilización y el mantenimiento del software. Cada módulo encapsula un conjunto de responsabilidades y proporciona interfaces para interactuar con otros módulos.
* **VSCode (Visual Studio Code):** Editor de código fuente desarrollado por Microsoft. Ofrece características avanzadas como depuración, control de versiones integrado y una extensa gama de extensiones que facilitan la personalización del entorno de desarrollo.
* **MAMP**: Acrónimo de Mac, Apache, MySQL y PHP. MAMP es un entorno de desarrollo que proporciona una solución integrada para la configuración de un servidor local en sistemas operativos macOS y Windows. Incluye el servidor web Apache, el sistema de gestión de bases de datos MySQL y el lenguaje de programación PHP..
* **Apache**: Servidor web de código abierto ampliamente utilizado para la entrega de contenido web a través del protocolo HTTP. Apache HTTP Server, comúnmente conocido como Apache, es conocido por su robustez y flexibilidad. Permite la configuración detallada a través de archivos .htaccess y soporta una amplia variedad de módulos que extienden su funcionalidad.
* **MSSQL (Microsoft SQL Server):** Un sistema de gestión de bases de datos relacional desarrollado por Microsoft, que se utiliza para almacenar y recuperar datos de manera eficiente.
* **IDE (Integrated Development Environment)**: Entorno de Desarrollo Integrado, un software que combina herramientas de programación como un editor de código, un depurador y un compilador para facilitar el desarrollo de software.
* **Cordova/Capacitor**: Plataformas para crear aplicaciones móviles híbridas.

## Referencias

* IEEE 830-1998: IEEE Recommended Practice for Software Requirements Specifications.
* Documentación de PHP 8.3.1
* Documentación de Angular V17
* Documentación de Bootstrap.
* Google Material Design Guidelines.
* Documentación de TypeScript.

## Resumen

Este documento detalla los requisitos necesarios para el desarrollo de la aplicación móvil destinada a la empresa de servicios de limpieza "Go Cleaning". Incluye aspectos tanto funcionales como no funcionales, define las interfaces del sistema móvil y establece las restricciones aplicables. Sirve como una base integral para el diseño, desarrollo y pruebas del sistema, asegurando que todos los aspectos necesarios para una implementación exitosa estén claramente especificados.

# DESCRIPCIÓN GENERAL

## Perspectiva del producto

Esta aplicación móvil está diseñada para optimizar y automatizar los procesos operativos en la empresa de servicios de limpieza "Go Cleaning", funcionando como una solución híbrida con una interfaz intuitiva y fácil de usar en dispositivos móviles. Su propósito es mejorar la eficiencia en el control de precios, pedidos de servicios de limpieza, clientes y la asignación de tareas a los empleados. La aplicación se integrará de manera fluida con una base de datos MySQL para el almacenamiento y la gestión de datos, utilizando PHP 8.3.1 para el backend API, y Angular V17 junto con IONIC, Material Design, TypeScript, HTML, y CSS para el frontend móvil. La aplicación será desarrollada como una app híbrida compatible con iOS y Android. Este software es propietario, lo que significa que su código fuente y derechos de uso están reservados y no están disponibles para su modificación o distribución externa.

## Funcionalidad del producto

* **Control de Pedidos de Servicios de Limpieza**: Permitir la creación, edición y seguimiento de pedidos de servicios de limpieza.
* **Administración de Clientes**: Control completo de datos y el historial de clientes que solicitan servicios de limpieza.
* **Control de Servicios de Limpieza**: Administración de los servicios de limpieza ofrecidos y sus precios.
* **Asignación de Tareas de Limpieza**: Distribución y seguimiento de tareas de limpieza entre los empleados.
* **Control de Precios**: Actualización y visualización de precios de los servicios de limpieza.
* **Notificaciones Push**: Envío de notificaciones push a clientes sobre el estado de pedidos.
* **Control de Descuentos**: Creación, actualización y visualización de descuentos aplicables a los pedidos de limpieza.
* **Gestión de Usuarios**: Control de acceso y permisos de los usuarios del sistema móvil.

## Características de los usuarios

Los usuarios finales son empleados con variados niveles de familiaridad tecnológica, desde intermedios hasta principiantes. La interfaz debe ser intuitiva para facilitar su uso por todos los niveles de competencia.

* **Perfil de usuarios finales**

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Masculino |
| Edad | 32 |
| Región | Quito, Ecuador |
| Ocupación | Empleado privado |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología intermedio: familiarizado con el uso de aplicaciones básicas y sistemas en línea, pero no necesariamente experto en software avanzado |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Femenino |
| Edad | 31 |
| Región | Quito, Ecuador |
| Ocupación | Empleada privada |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizada con el uso de sistemas en línea. |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Femenino |
| Edad | 35 |
| Región | Quito, Ecuador |
| Ocupación | Empleada privada |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizada con el uso de sistemas en línea. |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Femenino |
| Edad | 45 |
| Región | Quito, Ecuador |
| Ocupación | Empleada privada |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizada con el uso de sistemas en línea. |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Masculino |
| Edad | 38 |
| Región | Quito, Ecuador |
| Ocupación | Emplead privado |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizado con el uso de sistemas en línea. |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Masculino |
| Edad | 43 |
| Región | Quito, Ecuador |
| Ocupación | Empleado privado |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizada con el uso de sistemas en línea. |

|  |  |
| --- | --- |
| **Categoría** | **Descripción** |
| Género | Masculino |
| Edad | 36 |
| Región | Quito, Ecuador |
| Ocupación | Empleado privado |
| Nivel Social | Medio |
| Categoría de usuario | Conocimiento en tecnología bajo: poco familiarizada con el uso de sistemas en línea. |

[Se mantienen perfiles similares para otros empleados de Go Cleaning]

## Restricciones

* **Acceso Exclusivo para Empleados**: El acceso a la aplicación móvil "Go Cleaning" está estrictamente limitado a empleados autorizados de la empresa. Se implementará un sistema de autenticación para garantizar que solo usuarios autorizados puedan acceder.
* **Compatibilidad Móvil**: La aplicación debe ser compatible con dispositivos iOS (versión 12 o superior) y Android (API nivel 21 o superior).
* **Tecnologías**: Uso obligatorio de PHP para backend API, Angular e IONIC para frontend móvil.
* **Modelo Cliente/Servidor**: La aplicación seguirá un modelo cliente/servidor, donde los empleados interactuarán con la interfaz móvil (cliente) para acceder a las funciones proporcionadas por el servidor.
* **Conexión a Internet**: La aplicación requiere conexión a Internet para su funcionamiento completo.
* **Peticiones Restringidas**: Los servicios de la aplicación requieren el envío de un token de seguridad para validar el usuario que realiza las peticiones.

## Suposiciones y dependencias

* La aplicación asumirá una conexión a Internet estable y accesible en dispositivos móviles.
* La funcionalidad dependerá de la compatibilidad con las versiones de iOS y Android especificadas.
* La solución se desarrollará en base a las versiones especificadas de PHP, Angular e IONIC.
* Se asume disponibilidad de servicios push de las plataformas móviles..

## Evolución previsible del sistema

En el futuro, se podrían implementar nuevas funcionalidades móviles, como la integración con sistemas de geolocalización para tracking en tiempo real, la habilitación de pedidos directos por parte de los clientes desde una app cliente, y la optimización del sistema de notificaciones push. Es fundamental que la aplicación móvil sea lo suficientemente flexible para permitir estas extensiones sin comprometer su rendimiento o estabilidad.

# REQUISITOS ESPECÍFICOS

## Requisitos comunes de los interfaces

### Interfaces de usuario

* **Intuitiva y fácil de usar**: La interfaz móvil debe ser diseñada para que los usuarios puedan navegar y operar de manera sencilla en pantallas táctiles, minimizando la curva de aprendizaje.
* **Diseño responsivo**: Debe adaptarse a diferentes tamaños de pantalla móvil (smartphones y tablets).
* **Navegación optimizada para móviles**: Incluir navegación por pestañas o menú hamburguesa optimizado para dispositivos táctiles.
* **Tipos de interfaces según roles**:
* **Administrador**: Acceso completo a todas las funcionalidades del sistema móvil.
* **Empleado**: Interfaz limitada para leer pedidos asignados, ver tareas y modificar estados.

### Interfaces de hardware

* **Compatibilidad general:** No se requieren interfaces específicas de hardware para la operación de la aplicación, lo que permite su uso en una variedad de dispositivos estándar.

### Interfaces de software

* **Base de datos:** La aplicación se integrará con MySQL para el almacenamiento y gestión de datos.
* **Tecnologías de desarrollo:** El proyecto utiliza una variedad de tecnologías para asegurar un desarrollo eficiente y una experiencia de usuario óptima:
  + **Backend**: Se emplea PHP en su versión 8.3.1, junto con Composer para la gestión de dependencias y librerías, lo que facilita la construcción de un backend robusto y escalable.
  + **Frontend**: El desarrollo del frontend se realiza utilizando IONIC con Angular en su versión 17, complementado con Material Design V3 y Bootstrap para asegurar un diseño responsivo y moderno. Además, se utilizan tecnologías como TypeScript, HTML y CSS para crear una interfaz de usuario dinámica y atractiva.
  + **Bases de datos**: Para el manejo de datos, se utiliza Microsoft SQL Server (MYSQL), que ofrece un entorno seguro y eficiente para el almacenamiento y recuperación de la información.
  + **Editor de código**: Visual Studio Code es el editor de código empleado, proporcionando un entorno de desarrollo integrado (IDE) potente y flexible, con soporte para diversas extensiones que optimizan el flujo de trabajo.

### Interfaces de comunicación

* **Notificaciones Push**: Sistema de notificaciones push nativas para informar sobre estados de pedidos y tareas.

## Requisitos funcionales

### Requisito funcional 1: Control de Pedidos

**Descripción:**

* **Creación de pedidos**: El sistema móvil debe permitir la creación de pedidos de servicios de limpieza, donde se pueden agregar múltiples servicios (limpieza residencial, comercial, profunda, etc.), incluyendo la cantidad y especificaciones del servicio.
* **Edición de pedidos**: Modificar servicios de limpieza en un pedido, actualizar direcciones de servicio, cambiar estados del pedido (pendiente, en proceso, completado, cancelado).
* **Seguimiento de pedidos**: Mostrar los cambios de estado del pedido de limpieza.

**Prioridad:** Alta

### Requisito funcional 2: Control de Clientes

**Descripción:**

* **Gestión de clientes:** El sistema debe ofrecer funcionalidades para agregar, modificar, eliminar y consultar datos de clientes. Esto incluye la capacidad de gestionar la información personal del cliente como nombre, dirección, correo electrónico y número de teléfono.

**Prioridad:** Alta

### Requisito funcional 3: Control de Servicios

**Descripción:**

* **Control de servicios:** El sistema debe permitir la creación, edición, eliminación y administración de servicios ofrecidos. Esto incluye detalles específicos como la descripción del servicio, precio, tiempo de entrega y cualquier otra información relevante.
* **Actualización de precios:** El sistema debe permitir la actualización de precios para servicios, así como la gestión de promociones y descuentos específicos aplicados a dichos servicios.

**Prioridad:** Alta

### Requisito funcional 4: Asignación de Tareas

**Descripción:**

* **Asignación de tareas:** El sistema debe permitir a los usuarios asignar tareas a diferentes empleados. La asignación debe considerar la carga laboral de cada empleado y permitir establecer prioridades para cada tarea.
* **Seguimiento de tareas:** Debe ser posible realizar un seguimiento del estado de cada tarea, permitiendo actualizaciones en tiempo real sobre el progreso y finalización de las mismas.

**Prioridad:** Media

### Requisito funcional 5: Control de precios

**Descripción:**

* **Gestión de precios:** El sistema debe permitir la actualización y visualización de precios de servicios en tiempo real, incluyendo la capacidad de gestionar promociones y descuentos.
* **Promociones y descuentos:** Debe haber funcionalidades para la creación y gestión de promociones y descuentos aplicables a diferentes servicios, con la opción de establecer condiciones para su aplicación.

**Prioridad:** Alta

### Requisito funcional 6: Notificaciones.

**Descripción:**

* **Envío de notificaciones:** El sistema debe permitir el envío automático de notificaciones por WhatsApp para informar a los clientes sobre el estado finalizado de sus pedidos.

**Prioridad:** Media

### Requisito funcional 7: Control de Usuarios

**Descripción:**

* **Gestión de usuarios:** El sistema debe permitir la creación, modificación, y eliminación de usuarios. Además, debe permitir la asignación de roles y permisos específicos a cada usuario.
* **Seguridad:** Debe garantizarse la seguridad de la información, permitiendo solo el acceso autorizado a las diferentes funcionalidades del sistema.

**Prioridad:** Alta

### Requisito funcional 8: Control de Descuentos

**Descripción:**

* **Gestión de descuentos:** El sistema debe permitir la creación y gestión de descuentos aplicables a servicios, con la opción de establecer condiciones específicas para su aplicación.
* **Aplicación de descuentos:** Debe ser posible aplicar automáticamente los descuentos a los servicios durante el proceso de creación de pedidos.

**Prioridad:** Media

## Requisitos no funcionales

### Requisitos de rendimiento

* **Descripción**: La aplicación móvil debe responder en menos de 3 segundos en conexiones 3G/4G/5G/WiFi.

### Seguridad

* **Descripción**: La aplicación debe contar con mecanismos de autenticación y autorización robustos para proteger el acceso a los datos sensibles. Esto incluye la implementación de JWT (JSON Web Tokens) para manejar la autenticación de manera segura y eficiente, asegurando que solo los usuarios autenticados puedan acceder a los recursos protegidos. Además, se deben emplear buenas prácticas de seguridad, como el cifrado de datos y la gestión segura de sesiones, para garantizar la integridad y confidencialidad de la información.

### Fiabilidad

* **Descripción**: El sistema debe asegurar un nivel de disponibilidad del 99.9% durante las horas operativas críticas, minimizando el tiempo de inactividad y asegurando que los usuarios puedan acceder a las funcionalidades esenciales sin interrupciones.

### Disponibilidad

* **Descripción**: La aplicación debe estar disponible las 24 horas del día, los 7 días de la semana.

### Mantenibilidad

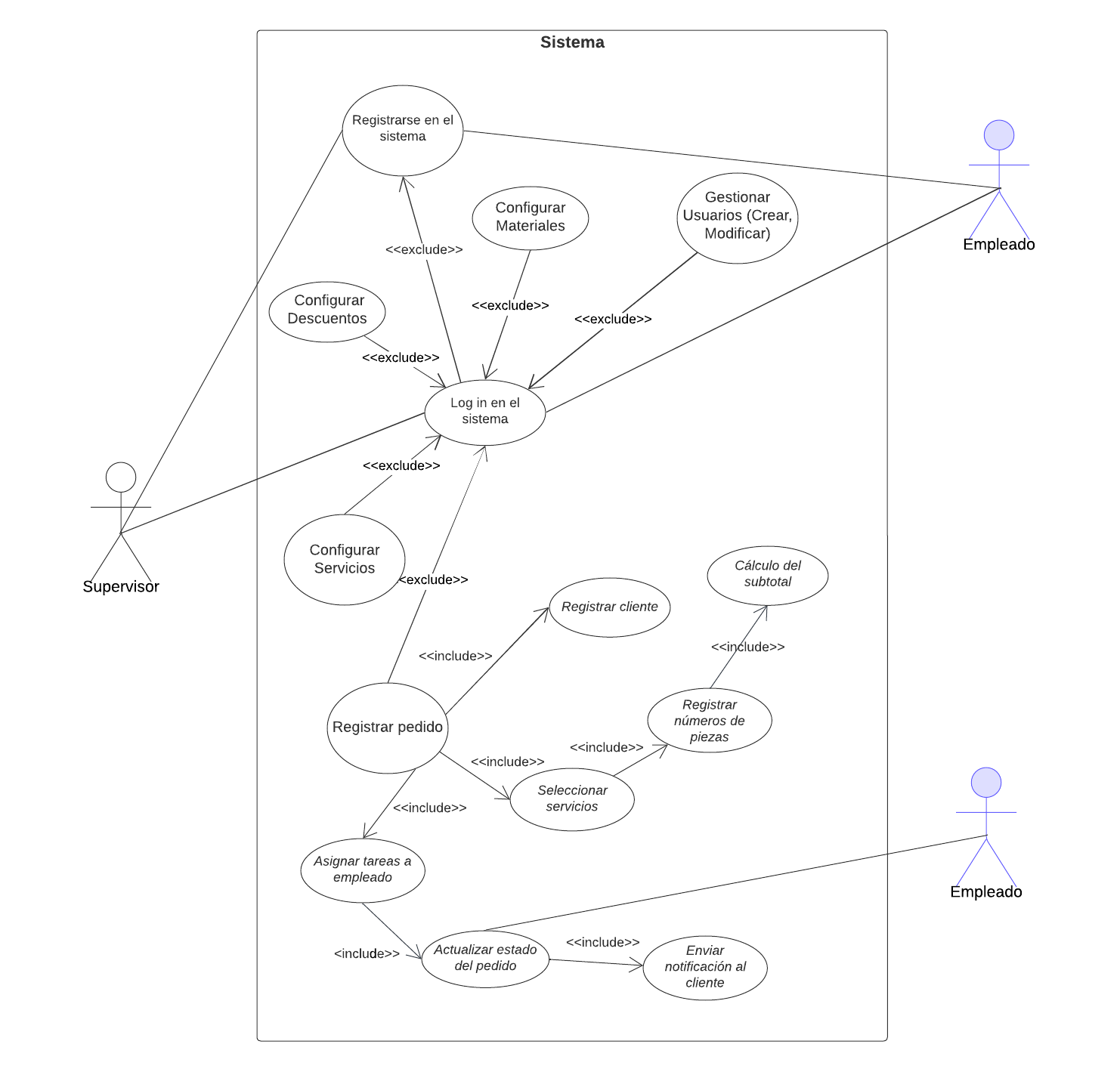
* **Descripción**: El código fuente de la aplicación debe adherirse a estándares de codificación bien definidos y documentados, facilitando su comprensión, modificación y ampliación por parte de los desarrolladores. Además, se deben emplear técnicas de modularidad y reutilización de código para simplificar las tareas de mantenimiento y permitir una fácil incorporación de nuevas funcionalidades en el futuro.

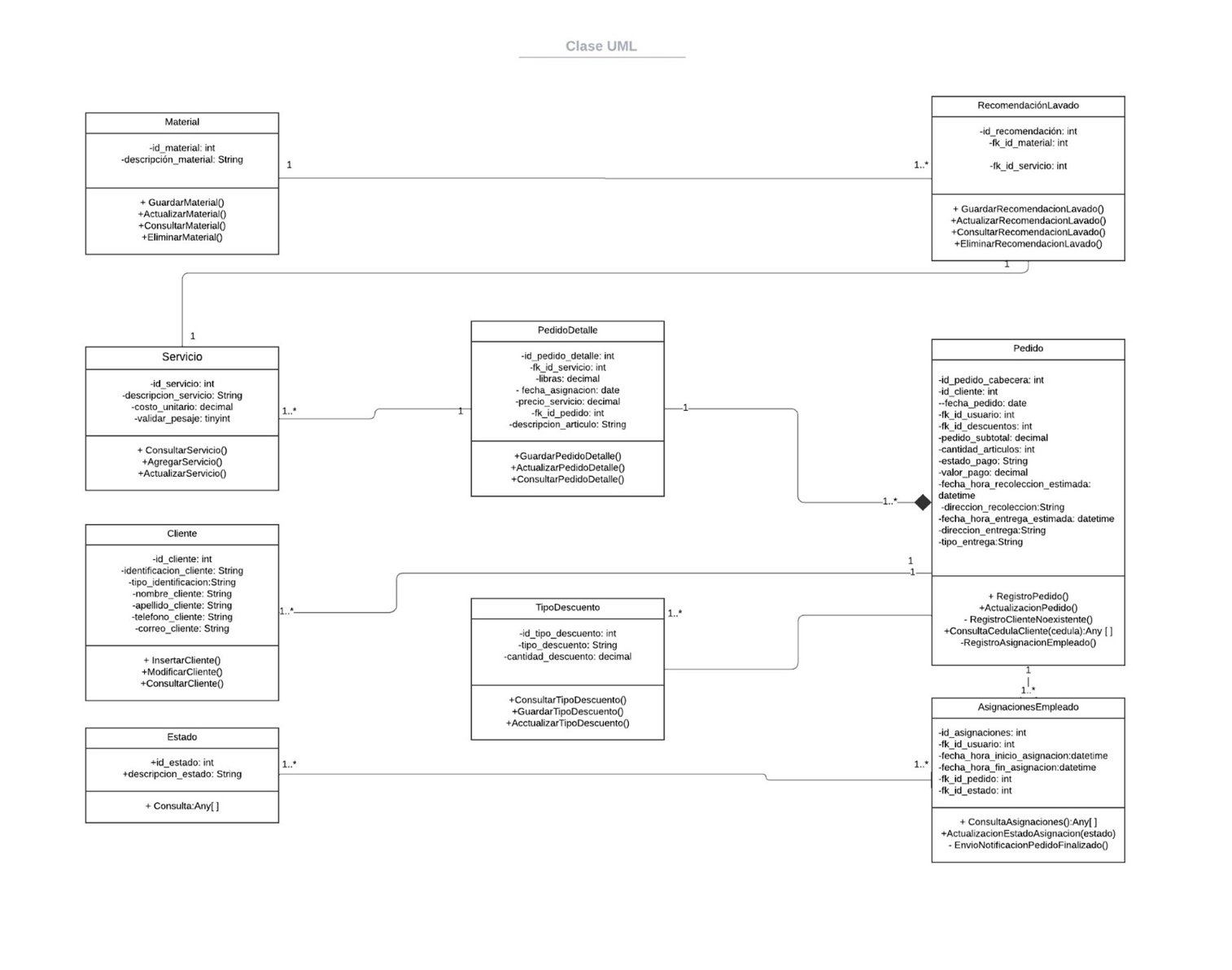
### Portabilidad

* **Descripción**: La aplicación debe ser compatible con una gama de versiones de navegadores basados en Chromium, y con sistemas operativos con iOS 12+ y Android API 21+ (Android 5.0+).

# APÉNDICES

## Diagrama de Casos de Uso

4.2 Diagrama de Clases



# Glosario

* Angular: Framework para el desarrollo del frontend.
* IONIC: Framework para el desarrollo de aplicaciones móviles híbridas.
* Apache: Servidor web de código abierto ampliamente utilizado para la entrega de contenido web a través del protocolo HTTP. Apache HTTP Server, comúnmente conocido como Apache, es conocido por su robustez y flexibilidad. Permite la configuración detallada a través de archivos .htaccess y soporta una amplia variedad de módulos que extienden su funcionalidad.
* API: Interfaz de Programación de Aplicaciones. Conjunto de definiciones y protocolos para construir e integrar software de aplicaciones.
* Backend: Parte del software que gestiona la lógica del servidor y la manipulación de datos detrás de escena, no visible para el usuario.
* **Bootstrap**: Un framework de front-end que facilita el desarrollo de sitios web y aplicaciones móviles con un diseño responsivo. Bootstrap proporciona una colección de herramientas CSS y JavaScript.
* Chromium: Un proyecto de navegador web de código abierto en el cual se basan otros navegadores como Google Chrome y Brave.
* Cliente/Servidor: Arquitectura de red en la que un servidor proporciona servicios o recursos a uno o varios clientes. El cliente solicita servicios o recursos y el servidor los proporciona.
* Composer: Herramienta de gestión de dependencias en PHP. Permite a los desarrolladores gestionar y instalar librerías y paquetes necesarios para sus proyectos PHP de manera sencilla y eficiente.
* CRUD: Crear, Leer, Actualizar, Eliminar. Conjunto de operaciones básicas que se pueden realizar en la gestión de datos.
* CSS: Hojas de Estilo en Cascada. Lenguaje de diseño que se utiliza para describir la presentación de un documento escrito en HTML o XML.
* DBMS: Sistema de Gestión de Bases de Datos. Software que facilita la creación, administración y manipulación de bases de datos.
* Frontend: Parte de la aplicación que interactúa directamente con el usuario. Se refiere a la interfaz y a la experiencia del usuario.
* HTML: Lenguaje de Marcado de Hipertexto. Lenguaje estándar para crear y diseñar páginas web.
* IDE (Integrated Development Environment): Entorno de Desarrollo Integrado, un software que combina herramientas de programación como un editor de código, un depurador y un compilador para facilitar el desarrollo de software.
* JavaScript: Lenguaje de programación que permite la creación de contenido dinámico e interactivo en páginas web.
* JSON: JavaScript Object Notation. Formato ligero para el intercambio de datos, fácil de leer y escribir para los humanos y fácil de analizar y generar para las máquinas. Se utiliza comúnmente para transmitir datos entre un servidor y una aplicación web.
* JWT: JSON Web Token. Estándar abierto para la transmisión segura de información entre partes en formato JSON. Los JWT son compactos, seguros y se utilizan comúnmente para la autenticación y autorización en aplicaciones web.
* MAMP: Acrónimo de Mac, Apache, MySQL y PHP. MAMP es un entorno de desarrollo que proporciona una solución integrada para la configuración de un servidor local en sistemas operativos macOS y Windows. Incluye el servidor web Apache, el sistema de gestión de bases de datos MySQL y el lenguaje de programación PHP..
* **Material Design V3**: Un sistema de diseño creado por Google que proporciona pautas y componentes predefinidos para la creación de interfaces de usuario coherentes y atractivas.
* Módulo: Unidad de código que agrupa funciones, clases o variables relacionadas con una funcionalidad específica. Los módulos permiten una mejor organización y gestión del código, favoreciendo la reutilización y el mantenimiento del software. Cada módulo encapsula un conjunto de responsabilidades y proporciona interfaces para interactuar con otros módulos.
* MSSQL (Microsoft SQL Server): Un sistema de gestión de bases de datos relacional desarrollado por Microsoft, que se utiliza para almacenar y recuperar datos de manera eficiente.
* MVC: Modelo-Vista-Controlador. Patrón de diseño de software que separa una aplicación en tres componentes interconectados para promover la separación de responsabilidades.
* MySQL: Sistema de Gestión de Bases de Datos Relacional de código abierto basado en SQL.
* Node.js: Entorno de ejecución para JavaScript en el servidor basado en el motor V8 de Google Chrome. Permite la ejecución de código JavaScript fuera del navegador, facilitando el desarrollo de aplicaciones del lado del servidor y de red.
* PHP: Hypertext Preprocessor. Lenguaje de programación del lado del servidor utilizado para desarrollar aplicaciones web.
* Responsividad: Capacidad de una aplicación o página web para adaptarse y funcionar correctamente en diferentes dispositivos y tamaños de pantalla.
* REST: Transferencia de Estado Representacional. Estilo de arquitectura de software para sistemas distribuidos como la web.
* RF: Requerimiento Funcional.
* RNF: Requerimiento No Funcional.
* Seguridad Informática: Prácticas y técnicas utilizadas para proteger sistemas de información contra accesos no autorizados y daños.
* Servidor: Computadora o programa que proporciona servicios, recursos, o datos a otros programas o computadoras, conocidos como clientes.
* SQL: Lenguaje de Consulta Estructurado. Lenguaje de programación para gestionar y manipular bases de datos relacionales.
* SRS: Documento de Especificación de Requisitos del Software. Documento que describe el comportamiento del sistema de software a desarrollar.
* SSL: Capa de Conexión Segura. Protocolo de seguridad para cifrar la comunicación entre el navegador del usuario y el servidor web.
* Token: En el contexto de seguridad, un token es un objeto de datos que se utiliza para autenticar y autorizar solicitudes en un sistema. Puede ser un valor que se envía con cada solicitud para verificar la identidad del usuario.
* **TypeScript**: Lenguaje de programación desarrollado por Microsoft que es un superset de JavaScript. Añade tipado estático opcional y otras características avanzadas a JavaScript, facilitando la detección de errores y el desarrollo a gran escala.
* UI: Interfaz de Usuario. Diseño y disposición de los elementos de una aplicación que permiten la interacción del usuario con el sistema.
* Usabilidad: Medida en que una interfaz o sistema es fácil de usar y de aprender para los usuarios finales.
* UX: Experiencia del Usuario. Se refiere a la percepción y respuestas de un usuario sobre la interfaz de un producto, especialmente en términos de facilidad de uso.
* VSCode (Visual Studio Code): Editor de código fuente desarrollado por Microsoft. Ofrece características avanzadas como depuración, control de versiones integrado y una extensa gama de extensiones que facilitan la personalización del entorno de desarrollo.