

# Especificación de Requisitos Software

## **Componentes del grupo:**

*Juan Alberto Camino Sáez*  
*Samuel Antonio Eugercios Nevado*  
*Víctor Fernández Duque*  
*Daniel García Molero*  
*Manuel Gómez Lagóstena*  
*Arturo Marino Quintana*

# Índice de páginas:

## **1. Introducción**

- 1.1. Propósito.
- 1.2. Alcance.
- 1.3. Definiciones, acrónimos y abreviaturas.
- 1.4. Referencias.
- 1.5. Resumen.

## **2. Descripción general**

- 2.1. Perspectiva del producto.
- 2.2. Funciones del producto.
- 2.3. Características del usuario.
- 2.4. Restricciones.
- 2.5. Supuestos y dependencias.
- 2.6. Requisitos futuros.

## **3. Requisitos específicos**

- 3.1. Interfaces externos.
- 3.2. Funciones
- 3.3. Requisitos de rendimiento.
- 3.4. Requisitos lógicos de la base de datos.
- 3.5. Restricciones de diseño.
- 3.6. Atributos del sistema software

# 1.- Introducción

## 1.1.- Propósito:

El propósito de la SRS es establecer una descripción precisa y detallada de los requisitos necesarios para conseguir una base sólida para un contrato entre el desarrollador y el cliente.

En cuanto a la audiencia de nuestra SRS:

- Cliente: El profesor de Modelado de Software.
- Desarrolladores: El grupo de alumnos de Modelado de Software.
- Usuario: El profesor de Modelado de Software.

## 1.2.- Alcance:

Nuestro producto recibe el nombre de *BiciSoft*, va a ser un Software de gestión de tiendas de bicicletas.

Nuestro producto gestionará por un lado todo lo referente a la organización interna de una tienda (ventas, modelos de bicicletas etc...), mientras que por el otro, gestionará la parte empresarial de la compañía (empleados, departamentos, *tiendas*...).

## 1.3.- Definiciones, acrónimos y abreviaturas:

Añadir aquí las definiciones y demás de lenguaje específico usado en la documentación:

- Definiciones: *No procede.*
- Acrónimos:
  - SRS: *Software Requirements Specification* (Especificación de Requisitos de Software).
  - JRE: Java Runtime Environment.
  - GCS: Gestión de la Configuración Software.
  - IBM RSA v9.0: *IBM Rational Software Architect, versión 9.0.*
- Abreviaturas: *No procede.*

## 1.4.- Referencias:

A la hora del desarrollo, nos basaremos y observaremos el funcionamiento de aplicaciones similares, incluyendo la aplicación desarrollada por tres de los componentes del grupo el pasado curso 2015-2016 en la asignatura *Ingeniería del Software*, de nombre "*Aerotype*".

### ***1.5.- Resumen:***

Gracias a la SRS vamos a ahorrarnos un tiempo valioso a la hora de ponernos de acuerdo con el cliente, así como utilizarla como contrato para evitar posteriores conflictos y utilizarla como guía a la hora de diseñar el programa.

En esta SRS, tendremos en cuenta muchos factores, como por ejemplo la mantenibilidad del producto, valorando los requisitos futuros o probables, ya que es posible que la SRS sea actualizada a medida que avance el proyecto (GCS).

Este documento por tanto, tiene el objetivo de reducir el esfuerzo a la hora de diseñar la aplicación, así como encargarse de eliminar ambigüedades y malentendidos con el cliente.

La presente SRS está desarrollada por los estudiantes de Modelado de Software, los cuales colaborarán con su redacción y avalarán y seguirán a la hora del diseño los contenidos recogidos en la misma.

## 2.- Descripción general:

En esta sección se describen los factores concernientes al producto y a sus requisitos.

### *2.1.- Perspectiva del producto:*

El Software *Bicisoft* es un programa de gestión de una tienda de bicicletas que permite al usuario conocer toda la información relativa a la tienda tanto a nivel local como a nivel empresa

#### 2.1.1.- Interfaces con el sistema:

Debido a que el programa es totalmente nuevo sin ninguna versión anterior o aplicación similar o relacionada, no requiere interactuar con sistemas externos, ni requiere interactuar con procesos de migración u adaptación con sistemas ya existentes.

#### 2.1.2.- Interfaces con el usuario:

Aquí se presentan las interfaces con las que el usuario interactuará con el sistema. Los usuarios podrán introducir y recibir datos gracias a los periféricos de entrada y de salida de su ordenador respectivamente. Los primeros permitirán a los usuarios, introducir información necesaria para el funcionamiento y usar las diferentes opciones y posibilidades de la aplicación. Los segundos permitirán a los usuarios visualizar la respuesta de la aplicación, tanto información como mensajes de información, menús, etc.

#### 2.1.3.- Interfaces con el Hardware:

La aplicación es una aplicación Java de una complejidad bastante reducida, por lo que, salvo excepciones, cualquier computador actual, sean cuales sean sus componentes, en principio, debería ser capaz de ejecutar la aplicación con unos tiempos de ejecución razonables.

#### 2.1.4.- Interfaces con el Software:

Nuestra aplicación no interactúa con ningún tipo de Software externo, sin embargo, es posible que se deba actualizar el JRE (*Java Runtime Environment*) en la máquina en la que se desee ejecutar el programa a una versión actual, en caso de no estarlo. Dicho Software contiene material básico y necesario para la ejecución de la mayoría de aplicaciones Java.

#### 2.1.5.- Interfaces de Comunicación:

No usaremos ningún tipo de infraestructura de comunicación.

#### 2.1.6.- Operaciones:

La aplicación podrá ser usada por los empleados de la tienda para realizar todas las operaciones pertinentes a la hora de gestionar el comercio solicitadas por el cliente.

#### 2.1.7.- Requisitos de adaptación:

En principio, no será necesaria ninguna instalación para la ejecución del programa.

### **2.2.- Funciones del producto:**

El producto es una aplicación informática dedicada a la gestión de una tienda de bicicletas.

Se encarga de gestionar las bicicletas en stock, las ventas y los clientes, así como la gestión de los empleados, las tiendas y los presupuestos de los mismos.

Los módulos se van a agrupar en dos grandes grupos. El primero se encargará de la gestión de la parte de tienda, mientras que el segundo se encargará de la empresa a nivel oficinas.

El primer grupo estará compuesto por los siguientes módulos:

- I. **Bicicleta:** Módulo encargado de gestionar todos los modelos de bicicletas con los que opera la tienda.
- II. **Compra:** Módulo encargado de gestionar las ventas de bicicletas a los clientes, así como el carrito de la compra de la presente venta.
- III. **Cliente:** Módulo encargado de gestionar los usuarios del comercio, posee todos los datos necesarios de cada cliente.

En cuanto a las funciones de estos módulos:

#### **1) Bicicleta:**

1. *Alta bicicleta montaña:* Da de alta un modelo de bicicleta de montaña en el sistema.
2. *Alta bicicleta carretera:* Da de alta un modelo de bicicleta de carretera en el sistema.
3. *Baja bicicleta:* Da de baja un modelo de bicicleta en el sistema.
4. *Modificar bicicleta:* Modifica los atributos de una bicicleta en concreto.
5. *Mostrar bicicleta:* Muestra los detalles y características de un modelo de bicicleta en concreto.
6. *Listar bicicletas:* Muestra una lista de todos los modelos de bicicletas disponibles.

#### **2) Compra:**

1. *Iniciar compra:* Solicita el ID del cliente, que está asociado en la base de datos a un nombre, DNI, número de tarjeta...

2. *Añadir artículo*: Añade un artículo (bicicleta) al carrito (línea de compra), junto con una cantidad.
3. *Eliminar artículo*: Elimina todas las unidades de un artículo del carrito.
4. *Finalizar compra*: Tramitar la venta, guardando los artículos comprados en la base de datos.
5. *Devolución compra*: Devuelve las unidades solicitadas de un producto de una compra.
6. *Mostrar compra*: Muestra todos los artículos comprados en una factura concreta.
7. *Cantidad bicicletas*: Muestra la cantidad de bicicletas que ha comprado un cliente de todas sus compras.
8. *Número clientes*: Muestra el número de clientes que han comprado un modelo de bicicleta dado.

### 3) Cliente:

1. *Alta cliente*: Da de alta un cliente en la base de datos.
2. *Baja cliente*: Da de baja un cliente en la base de datos.
3. *Modificar cliente*: Modifica los datos de un cliente.
4. *Mostrar cliente*: Muestra todos los campos de un cliente en concreto.
5. *Listar clientes*: Muestra una lista de todos los clientes en la base de datos.

El segundo grupo estará compuesto por los restantes módulos:

- IV. **Empleado**: Módulo encargado de gestionar todos los empleados de la compañía.
- V. **Tienda**: Módulo encargado de gestionar todas las tiendas que son propiedad de la compañía.
- VI. **Departamento**: Módulo encargado de gestionar los departamentos de la compañía.

En cuanto a las funciones de estos módulos:

### 4) Empleado:

1. *Alta empleado*: Da de alta un empleado en el sistema.
2. *Baja empleado*: Da de baja un empleado en el sistema.
3. *Modificar empleado*: Modifica los atributos de un empleado en concreto.
4. *Mostrar empleado*: Muestra los detalles y características de un empleado en concreto.
5. *Listar empleados*: Muestra una lista de todos los empleados de la base de datos.

### **5) Tienda:**

1. *Alta tienda:* Añade una tienda a la base de datos.
2. *Baja tienda:* Da de baja un tienda de la base de datos.
3. *Modificar tienda:* Modifica los atributos de una tienda.
4. *Mostrar tienda:* Muestra los detalles de una tienda en concreto.
5. *Listar tiendas:* Muestra una lista de todas las tiendas.
6. *Añadir departamento:* Añade un nuevo departamento a una tienda.
7. *Eliminar departamento:* Elimina un departamento de una tienda.
8. *Asignar presupuesto:* Asigna un presupuesto a un departamento de una tienda en concreto.

### **6) Departamento:**

1. *Alta departamento:* Da de alta un departamento en la base de datos.
2. *Baja departamento:* Da de baja un departamento en la base de datos.
3. *Modificar departamento:* Modifica los datos de un departamento.
4. *Mostrar departamento:* Muestra todos los campos de un departamento en concreto.
5. *Listar departamentos:* Muestra una lista de todos los departamentos en la base de datos.
6. *Calcular nómina:* Calcula el salario de todos los trabajadores de un departamento, ya sea dependiente o directivo.

### **2.3.- Características del usuario:**

Esta aplicación está dirigida a un público inexperto en el campo de la informática, por lo que tendrá una interfaz amigable y sencilla para que cualquier usuario con un mínimo de conocimientos pueda manejarla y realizar las operaciones posibles gracias al programa.

Adicionalmente, la aplicación no tendrá distinciones entre usuarios (del tipo administrador/trabajador con distintos privilegios u opciones dentro del mismo).



## *2.4.- Restricciones:*

A la hora de desarrollar la presente aplicación, no existe ninguna restricción explícita de recursos (tiempo de ejecución o memoria en uso), sin embargo, la aplicación será lo más eficiente posible y estará diseñada acorde a principios de Estructura de Datos y Algoritmos para minimizar el uso de recursos para así maximizar la eficiencia de la misma. Además, el lenguaje a usar debe ser Java.

## *2.5.- Supuestos y dependencias:*

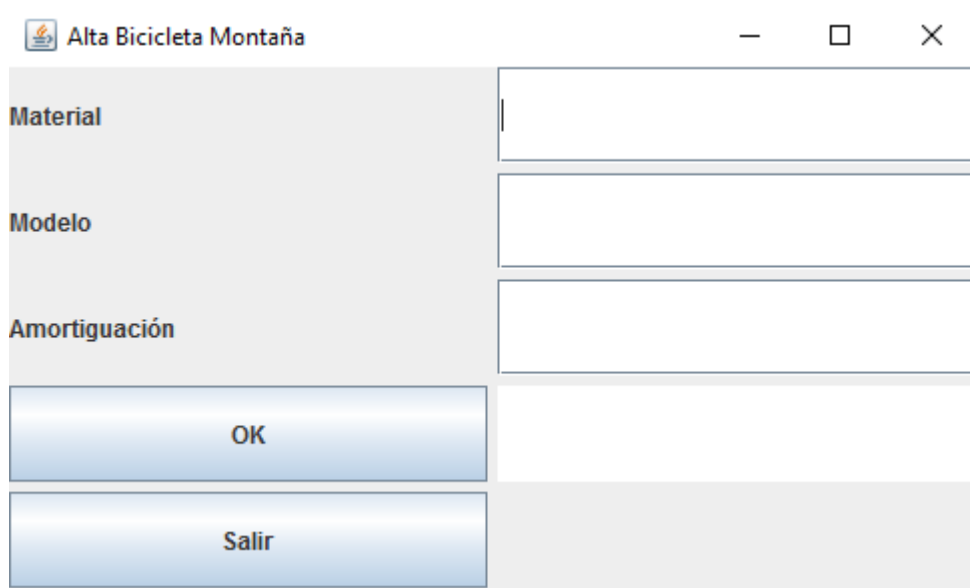
Ya que el desarrollo será en Java, todo el código estará hecho con orientación a objetos, estructurando el programa de manera pertinente y siempre acorde a este principio. Por el mismo motivo, será necesario tener instalado el JRE en una versión actualizada para poder ejecutar la aplicación, como se cita en el punto 2.1.4.

## *2.6.- Requisitos futuros:*

En principio no se van a realizar versiones futuras de nuestra aplicación, no obstante, la documentación, el modelo y el código estarán organizados para facilitar al máximo la implementación de nuevas funcionalidades (o modificación de las existentes) aplicando el uso de patrones y demás principios de Modelado de Software.

## 3.- Requisitos específicos:

### 3.1.- Interfaces externos:



Alta Bicicleta Montaña

Material

Modelo

Amortiguación

OK

Salir

### 3.2.- Funciones:

#### 1) Bicicleta

Función 1.1	Alta bicicleta montaña
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta un modelo de bicicleta de montaña
Entrada	Material, marca, precio y amortiguación
Salida	Resultado del proceso
Necesita	Acceso a la base de datos de bicicletas
Acciones	Dar de alta bicicleta de montaña
Precondición	No debe existir el modelo de bicicleta o si existe estar desactivado
Efectos laterales	Aumenta el número de modelos de bicicletas a la venta

Función 1.2	Alta bicicleta carretera
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta un modelo de bicicleta de carretera
Entrada	Material, marca, precio y velocidad máxima
Salida	Resultado del proceso
Necesita	Acceso a la base de datos de bicicletas
Acciones	Dar de alta bicicleta de carretera
Precondición	No debe existir el modelo de bicicleta o si existe estar desactivado

<b>Efectos laterales</b>	Aumenta el número de modelos de bicicletas a la venta
--------------------------	---

<b>Función 1.3</b>	<b>Baja bicicleta</b>
<b>Prioridad</b>	Media
<b>Estabilidad</b>	Media
<b>Descripción</b>	Da de baja un modelo de bicicleta
<b>Entrada</b>	Id de la bicicleta
<b>Salida</b>	Resultado del proceso
<b>Necesita</b>	Acceso a la base de datos de bicicletas
<b>Acciones</b>	Dar de baja bicicleta de carretera
<b>Precondición</b>	Debe existir el modelo de bicicleta y estar activo
<b>Efectos laterales</b>	Disminuye el número de modelos de bicicletas a la venta

<b>Función 1.4</b>	<b>Modificar bicicleta</b>
<b>Prioridad</b>	Baja
<b>Estabilidad</b>	Media
<b>Descripción</b>	Modificar los atributos de una bicicleta
<b>Entrada</b>	Id de la bicicleta y atributos a modificar
<b>Salida</b>	Resultado del proceso
<b>Necesita</b>	Acceso a la base de datos de bicicletas
<b>Acciones</b>	Modifica atributos de una bicicleta
<b>Precondición</b>	Debe existir el modelo de bicicleta y estar activo
<b>Efectos laterales</b>	No tiene

<b>Función 1.5</b>	<b>Mostrar bicicleta</b>
<b>Prioridad</b>	Alta
<b>Estabilidad</b>	Media
<b>Descripción</b>	Muestra las características de un modelo de bicicleta
<b>Entrada</b>	Id de la bicicleta
<b>Salida</b>	Atributos de la bicicleta
<b>Necesita</b>	Acceso a la base de datos de bicicletas
<b>Acciones</b>	Obtiene los atributos de una bicicleta
<b>Precondición</b>	Debe existir el modelo de bicicleta
<b>Efectos laterales</b>	No tiene

<b>Función 1.6</b>	<b>Listar bicicletas</b>
<b>Prioridad</b>	Media
<b>Estabilidad</b>	Media
<b>Descripción</b>	Muestra las características de las bicicletas
<b>Entrada</b>	No tiene
<b>Salida</b>	Atributos de las bicicletas
<b>Necesita</b>	Acceso a la base de datos de bicicletas
<b>Acciones</b>	Obtiene los atributos las bicicletas
<b>Precondición</b>	No tiene
<b>Efectos laterales</b>	No tiene

## 2) Compra

Función 2.1	Iniciar compra
Prioridad	Alta
Estabilidad	Media
Descripción	Solicita id del cliente para comenzar la compra
Entrada	Id de cliente
Salida	Resultado del proceso
Necesita	Acceso a la base de datos de clientes
Acciones	Inicia la compra solicitando el id de cliente, creando un carrito vacío.
Precondición	Debe existir el cliente
Efectos laterales	No tiene

Función 2.2	Añadir artículo
Prioridad	Alta
Estabilidad	Media
Descripción	Añade un artículo al carrito de la compra
Entrada	Id bicicleta y cantidad
Salida	Resultado del proceso
Necesita	Acceso a la base de datos de bicicletas
Acciones	Añade bicicletas al carrito de la compra
Precondición	Debe existir el modelo de bicicleta
Efectos laterales	No tiene

Función 2.3	Eliminar artículo
Prioridad	Alta
Estabilidad	Media
Descripción	Elimina un artículo del carrito de la compra
Entrada	Id bicicleta
Salida	Resultado del proceso
Necesita	Carrito de la compra
Acciones	Elimina bicicletas del carrito de la compra
Precondición	Debe existir la bicicleta en el carrito de la compra
Efectos laterales	No tiene

Función 2.4	Finalizar compra
Prioridad	Alta
Estabilidad	Media
Descripción	Finaliza la compra y la guarda en BD
Entrada	Lista de artículos comprados
Salida	Resultado del proceso
Necesita	Base de datos línea de compra y compra
Acciones	Guarda el carrito de la compra en la base de datos
Precondición	El carrito de la compra no debe estar vacío
Efectos laterales	No tiene

Función 2.5	Devolución compra
Prioridad	Media
Estabilidad	Media
Descripción	Devuelve las unidades solicitadas de un producto
Entrada	Id compra, id modelo producto y cantidad
Salida	Resultado del proceso
Necesita	Base de datos línea de compra y compra
Acciones	Modifica la cantidad de una línea de compra. Si llega a 0, da de baja la línea de compra.
Precondición	Debe existir la compra y el producto en la compra.
Efectos laterales	No tiene

Función 2.6	Mostrar compra
Prioridad	Baja
Estabilidad	Media
Descripción	Muestra los artículos de una compra en una factura concreta
Entrada	Id de la compra
Salida	Datos de la compra.
Necesita	Base de datos línea de compra
Acciones	Muestra los artículos de una compra
Precondición	Tiene que existir la compra
Efectos laterales	No tiene

Función 2.7	Cantidad de bicicletas
Prioridad	Baja
Estabilidad	Media
Descripción	Cantidad de bicicletas que ha comprado un cliente
Entrada	Id de Cliente
Salida	Resultado del proceso
Necesita	Base de datos línea de compra y compra
Acciones	El número de bicicletas que ha comprado un cliente
Precondición	Debe existir el cliente
Efectos laterales	No tiene

Función 2.8	Número de clientes
Prioridad	Baja
Estabilidad	Media
Descripción	Número de clientes que han comprado un modelo de bicicleta
Entrada	Id de modelo de bicicleta.
Salida	Resultado del proceso
Necesita	La base de datos línea de compra y de compra
Acciones	Muestra el número de clientes que han comprado un modelo de bicicleta

Precondición	Debe existir el id del modelo de bicicleta
Efectos laterales	No tiene

### 3) Cliente

<b>Función 3.1</b>	<b>Alta cliente</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta un cliente
Entrada	Nombre, tarjeta de crédito y DNI
Salida	Resultado del proceso
Necesita	Base de datos cliente
Acciones	Da de alta un cliente o lo pone activo si estaba desactivado
Precondición	No debe existir el cliente y si existe estar desactivado
Efectos laterales	No tiene

<b>Función 3.2</b>	<b>Baja cliente</b>
Prioridad	Media
Estabilidad	Media
Descripción	Elimina un cliente de la base de datos
Entrada	Id del cliente
Salida	Resultado del proceso
Necesita	Base de datos de cliente
Acciones	Desactiva un cliente de la base de datos
Precondición	Debe existir el cliente y estar activo
Efectos laterales	No tiene

<b>Función 3.3</b>	<b>Modificar cliente</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Modifica los atributos de un cliente
Entrada	Id del cliente, nombre, DNI y tarjeta de crédito
Salida	Resultado del proceso
Necesita	Base de datos de cliente
Acciones	Modifica los atributos de un cliente
Precondición	Debe existir el cliente y estar activo
Efectos laterales	No tiene

<b>Función 3.4</b>	<b>Mostrar cliente</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra los atributos de un cliente
Entrada	Id del cliente
Salida	Datos del cliente.
Necesita	Base de datos de cliente
Acciones	Muestra todos los atributos de un cliente
Precondición	Debe existir el cliente
Efectos laterales	No tiene

<b>Función 3.5</b>	<b>Listar clientes</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra una lista con todos los clientes
Entrada	No tiene
Salida	Lista de clientes
Necesita	Base de datos de cliente
Acciones	Muestra lista con los atributos de los clientes
Precondición	No tiene
Efectos laterales	No tiene

#### 4) Empleado

<b>Función 4.1</b>	<b>Alta empleado</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta un empleado
Entrada	Nombre, DNI y teléfono
Salida	Resultado del proceso
Necesita	Base de datos de empleado
Acciones	Da de alta un empleado o lo pone activo si estaba desactivado
Precondición	No debe existir el empleado y si existe estar desactivado
Efectos laterales	No tiene

<b>Función 4.2</b>	<b>Baja empleado</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Elimina un empleado de la base de datos
Entrada	Id del empleado
Salida	Resultado del proceso
Necesita	Base de datos de empleado
Acciones	Desactiva un empleado de la base de datos
Precondición	Debe existir el empleado y estar activo
Efectos laterales	No tiene

<b>Función 4.3</b>	<b>Modificar empleado</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Modifica los atributos de un empleado
Entrada	Id del empleado, nombre, DNI y teléfono
Salida	Resultado del proceso
Necesita	Base de datos de empleado
Acciones	Modifica los atributos de un empleado
Precondición	Debe existir el empleado y estar activo

Efectos laterales	No tiene
-------------------	----------

<b>Función 4.4</b>	<b>Mostrar empleado</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra los atributos de un empleado
Entrada	Id del empleado
Salida	Datos del empleado
Necesita	Base de datos de empleado
Acciones	Muestra todos los atributos de un empleado
Precondición	Debe existir el empleado
Efectos laterales	No tiene

<b>Función 4.5</b>	<b>Listar empleados</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra una lista con todos los empleados
Entrada	No tiene
Salida	Lista de empleados.
Necesita	Base de datos de empleado
Acciones	Muestra lista con los atributos de los empleados
Precondición	No tiene
Efectos laterales	No tiene

## 5) Tienda

<b>Función 5.1</b>	<b>Alta tienda</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta una tienda
Entrada	Nombre y dirección
Salida	Resultado del proceso
Necesita	Base de datos tienda
Acciones	Da de alta una tienda o la pone activo si estaba desactivado
Precondición	No debe existir la tienda y si existe estar desactivada
Efectos laterales	No tiene

<b>Función 5.2</b>	<b>Baja tienda</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Elimina una tienda de la base de datos
Entrada	Id de la tienda
Salida	Resultado del proceso
Necesita	Base de datos tienda
Acciones	Desactiva una tienda de la base de datos
Precondición	Debe existir la tienda y estar activa
Efectos laterales	No tiene



<b>Función 5.3</b>	<b>Modificar tienda</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Modifica los atributos de una tienda
Entrada	Id de tienda, nombre y dirección
Salida	Resultado del proceso
Necesita	Base de datos de tienda
Acciones	Modifica los atributos de una tienda
Precondición	Debe existir la tienda y estar activa
Efectos laterales	No tiene

<b>Función 5.4</b>	<b>Mostrar tienda</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra los atributos de una tienda
Entrada	Id de tienda
Salida	Datos de la tienda
Necesita	Base de datos de tienda
Acciones	Muestra todos los atributos de una tienda
Precondición	Debe existir la tienda
Efectos laterales	No tiene

<b>Función 5.5</b>	<b>Listar tiendas</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra una lista con todas las tiendas
Entrada	No tiene
Salida	Lista de tiendas
Necesita	Base de datos de tienda
Acciones	Muestra lista con los atributos de las tiendas
Precondición	No tiene
Efectos laterales	No tiene

<b>Función 5.6</b>	<b>Añadir departamento</b>
Prioridad	Media
Estabilidad	Media
Descripción	Asigna departamento a tienda
Entrada	Tienda y departamento
Salida	Resultado del proceso
Necesita	Base de datos de presupuesto
Acciones	Añade departamento a tienda
Precondición	Debe existir el departamento y la tienda
Efectos laterales	No tiene

<b>Función 5.7</b>	<b>Eliminar departamento</b>
Prioridad	Media
Estabilidad	Media
Descripción	Elimina un departamento de una tienda
Entrada	Id de la tienda y el departamento
Salida	Resultado del proceso
Necesita	Base de datos de presupuesto
Acciones	Elimina el departamento de la tienda
Precondición	La tienda debe tener asignado ese departamento
Efectos laterales	No tiene

<b>Función 5.8</b>	<b>Asignar presupuesto</b>
Prioridad	Media
Estabilidad	Media
Descripción	Asigna un presupuesto a un departamento de una tienda
Entrada	Presupuesto, id de tienda e id de departamento
Salida	Resultado del proceso
Necesita	Base de datos de presupuesto
Acciones	Modifica el presupuesto de un departamento de una tienda
Precondición	Debe existir la asignación departamento tienda, el presupuesto debe ser mayor que 0 y la tienda debe existir
Efectos laterales	No tiene

## 6) Departamento

<b>Función 6.1</b>	<b>Alta departamento</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Da de alta un departamento
Entrada	Nombre y descripción
Salida	Resultado del proceso
Necesita	Base de datos departamento
Acciones	Da de alta un departamento o la pone activo si estaba desactivado
Precondición	No debe existir el departamento y si existe estar desactivado
Efectos laterales	No tiene

<b>Función 6.2</b>	<b>Baja departamento</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Elimina un departamento de la base de datos
Entrada	Id del departamento
Salida	Resultado del proceso
Necesita	Base de datos departamento
Acciones	Desactiva un departamento de la base de datos

Precondición	Debe existir el departamento y estar activo
Efectos laterales	No tiene

<b>Función 6.3</b>	<b>Modificar departamento</b>
Prioridad	Alta
Estabilidad	Media
Descripción	Modifica los atributos de un departamento
Entrada	Id de tienda, nombre y descripción
Salida	Resultado del proceso
Necesita	Base de datos de departamento
Acciones	Modifica los atributos de un departamento
Precondición	Debe existir el departamento y estar activo
Efectos laterales	No tiene

<b>Función 6.4</b>	<b>Mostrar departamento</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra los atributos de una departamento
Entrada	Id de departamento
Salida	Datos del departamento
Necesita	Base de datos de departamento
Acciones	Muestra todos los atributos de un departamento
Precondición	Debe existir el departamento
Efectos laterales	No tiene

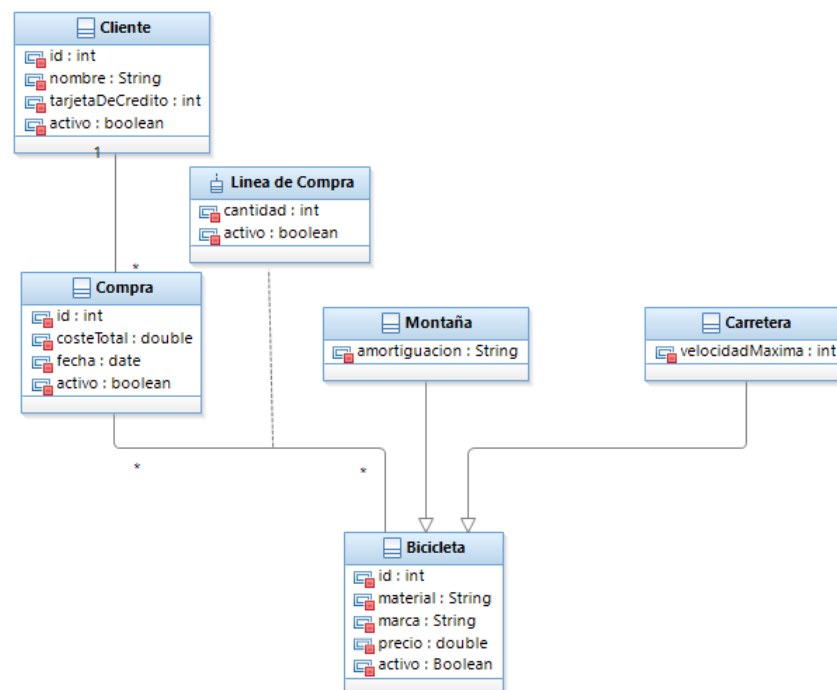
<b>Función 6.5</b>	<b>Listar departamentos</b>
Prioridad	Media
Estabilidad	Media
Descripción	Muestra una lista con todos los departamentos
Entrada	No tiene
Salida	Lista de departamentos
Necesita	Base de datos de departamento
Acciones	Muestra lista con los atributos de los departamentos
Precondición	No tiene.
Efectos laterales	No tiene

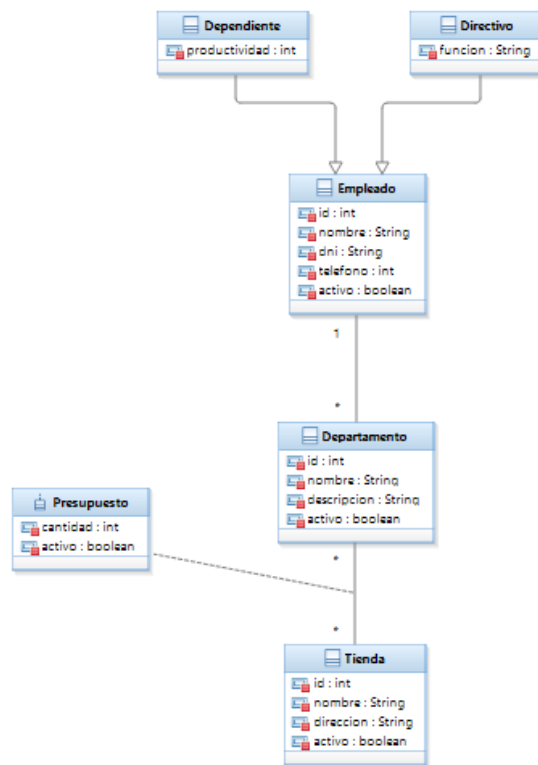
<b>Función 6.6</b>	<b>Calcular nómina</b>
Prioridad	Media
Estabilidad	Media
Descripción	Calcula el salario de todos los empleados del departamento
Entrada	No tiene
Salida	Salario de los empleados
Necesita	Base de datos de departamento y empleados
Acciones	Muestra el salario de todos los empleados de un departamento
Precondición	Debe existir el departamento
Efectos laterales	No tiene

### 3.3.- Requisitos de rendimiento:

- *Requisitos del producto:*
  - Tiempo de reacción mínimo o instantáneo.
  - Minimización de errores, idealmente, sin que exista ninguno en la versión final.
- *Requisitos organizativos:*
  - La aplicación deberá ser escritorio o web.
  - El lenguaje de implementación debe ser Java.
  - La persistencia de datos se deberá realizar de manera relacional usando bases de datos MySQL.

### 3.4.- Requisitos lógicos de la base de datos:





### 3.5.- Restricciones de diseño:

Como ha sido citado anteriormente, el lenguaje de programación debe ser Java.

Adicionalmente, para el desarrollo del programa se deberá utilizar el programa IBM RSA v9.0.

### 3.6.- Atributos del sistema Software:

**3.6.1.- Fiabilidad:** El sistema debería ser infalible y conservar y garantizar la persistencia de los datos y la efectividad de las operaciones realizadas.

**3.6.2.- Disponibilidad:** El sistema estará disponible siempre, con la única restricción de acceso a reservas o cancelaciones cuando se estén realizando labores de mantenimiento en dicho sistema.

**3.6.3.- Seguridad:** La base de datos deberá hallarse en una ubicación segura y el acceso a la aplicación se deberá restringir únicamente a ordenadores del confianza, ya que la aplicación no dispone de codificación y cifrado de los datos de por sí, ni de ningún tipo de mecanismo de login.

**3.6.4.- Mantenibilidad:** El sistema deberá disponer de documentación y modelos necesarios, así como el uso de variables descriptivas y código claro

y limpio para posibilitar y facilitar las futuras actualizaciones para satisfacer cualquier modificación en las necesidades de negocio del cliente.

3.6.5.- Portabilidad: El sistema será implementado para cualquier sistema operativo que soporte la ejecución de una aplicación Java.