UCM CDs

SRS

SUBRAYADO EN ROJO: SE PRETENDE ELIMINAR
SUBRAYADO EN AMARILLO: TERMINADO
SUBRAYADO EN VERDE: se ha modificado de cines a cds

Pablo Camacho Carrasco
Gonzalo Casado Valcárcel
Iván Cepeda Álvarez
Antonio Cuenca Bravo
Daniel Fernández Ortiz
Diego Flores Simón
Cristian García Moruno
Ignacio Garro San Millán
Iulian-Nicolae Muntean

TABLA DE CONTENIDO

https://drive.google.com/file/d/1_HesF6jrgc94GqsQiBimDkIHTpT32cSp/view?usp=share_link

tabla de control de cambios? después de hacer cambios al documento se registra y se sube a github

Versión	**Fecha**	**Autores**	**Descripción**
1	28/01/2023	Daniel Fernández Ortiz Cristian García Moruno Ignacio Garro Iulian-Nicolae Muntean	Corrección de la mayoría de las notas proporcionadas por el profesor
2	30/01/2023	Daniel Fernández Ortiz Ignacio Garro	Cambio de proyecto de UCM Cines a UCM CDs
3	01/02/2023	Daniel Fernández Ortiz Diego Flores Simón	Cambio de entidad Alquiler por la de Proveedores y cambio del modelo del dominio
4	05/02/2023	Daniel Fernández Ortiz	Corrección de pequeños detalles (3.2 Funciones) Me he apoyado de las transparencias 67 y 68 del tema 6

1. Introducción

1.1 Propósito

El documento de especificación de requisitos de software (SRS) tiene como propósito establecer la base para un acuerdo entre clientes y desarrolladores sobre lo que debe hacer el software. A través de este documento se consigue establecer la base para un acuerdo entre clientes y desarrolladores sobre qué debe hacer el software, consiguiendo así reducir el esfuerzo de desarrollo, proporcionando una base para la planificación y la verificación.

La SRS tiene como objetivo especificar el "qué" de lo que va a hacer el sistema, no el "cómo". Es una forma de justificar las acciones para llevar a cabo el desarrollo del proyecto.

1.2 Alcance

Identificaremos al proyecto de software por el nombre de "UCM CDs".

El objetivo principal de nuestro proyecto es la creación de una aplicación que gestione el negocio "UCM CDs". Esta aplicación se encargará de gestionar las actividades económicas que realiza esta empresa, como es la gestión de clientes, gestión de ventas y la gestión de productos disponibles. También habrá que facilitar al usuario los productos disponibles que vaya a querer comprar en la aplicación.

A través de esta práctica, podremos proporcionar a "UCM CDs" una aplicación de software de calidad que permitirá controlar y manejar sus actividades económicas de una forma sencilla y eficaz, mejorando la productividad del personal y las ventas del negocio.

1.3 Definiciones, acrónimos y abreviaturas

Software: Instrucciones que forman programas informáticos a través de lenguajes de programación, que serán las instrucciones que dirán a los ordenadores las tareas que deben realizar.

Hardware: Los componentes físicos que constituyen un ordenador o sistema informático.

Interfaces: Es la conexión funcional entre dos sistemas informáticos.

GUI: Es la interfaz gráfica que un usuario experimenta al utilizar una aplicación digital, normalmente representada por texto, imágenes, video y diseño gráfico, con la finalidad de crear un conexión entre la información del sistema informático y el usuario.

Base de Datos: Es el conjunto de datos e información que está organizado y estructurado de tal modo específico, para que su contenido pueda ser tratado y analizado de manera eficiente, sencilla y económica con un programa informático como es SQL.

SQL: Significa Lenguaje de Consulta Estructurada, y es un lenguaje de programación que se especializa en realizar operaciones sobre una base de datos.

JAVA: Es un lenguaje de programación de propósito general que utiliza la programación orientada a objetos.

JDBC: (Java Data Base Connectivity) es una API para llevar a cabo operaciones sobre bases de datos.

API: *Se trata de un conjunto de definiciones y protocolos que se utiliza para desarrollar e integrar el software de las aplicaciones, permitiendo la comunicación entre dos aplicaciones

Entidades:

- Películas: productos que se proyectan en el cine.
- Salas: lugares donde se proyectan las películas.
- Clientes: personas que visualizan las películas.

- Ventas: registro de la ventas de entradas.
- Administradores: gestionan las películas y las salas.
- Empresas cinematográficas: organizaciones que proporcionan las películas.

1.4 Referencias

Para llevar a cabo esta SRS, aparte de la teoría que se nos ha aportado en el campus virtual y de haber seguido el estándar IEEE Std. 830-1998, hemos acudido a las siguientes páginas para consultar la información adicional que necesitábamos.

https://definicion.de/base-de-datos/

https://economipedia.com/definiciones/politicas-regulatorias.html

https://www.redeszone.net/tutoriales/internet/que-es-protocolo-tls-handshake/

https://blogcandidatos.springspain.com/desarrollo-profesional/supuestos-del-proyecto-que-son-y-como-gestionarlos/#:~:text=Las%20suposiciones%20son%20creencias%20basadas,no%20llegar%20a%20buen%20t%C3%A9rmino.
https://www.xataka.com/basics/api-que-sirve

1.5 Resumen

En esta SRS describiremos tanto la descripción general del producto, como el conjunto de requisitos que serán necesarios en la implementación de la aplicación “UCM CDs” para poder cumplir con las necesidades de nuestro cliente. Así podremos crear una aplicación con todas las funcionalidades necesarias para los usuarios de la aplicación, proporcionando una experiencia completa.

Este SRS está organizado usando la estructura proporcionada por nuestros profesores de la UCM.

2. Descripción general

Esta aplicación se va a llevar a cabo, para poder gestionar la tienda de CD 's de la UCM, con el objetivo de agilizar tanto la gestión de información de la empresa para los clientes, como la facilitación de la compra de entradas para los usuarios.

Con ella los administradores serán capaces de acceder a toda la información de su negocio, para poder ser más productivos y tomar mejores decisiones gracias a la nueva herramienta proporcionada. Dispondrán de una lista ordenada de todas las salas con sus respectivos horarios y además la información de contacto de los clientes.

Por otra parte, habrá una interfaz para los usuarios de la aplicación que les aportará toda la información necesaria para poder realizar la compra de productos desde la aplicación web.

2.1 Perspectiva del producto

En cuanto al contexto de la implantación de la aplicación, se llevará a cabo el negocio ficticio “UCM CD’s” para poder facilitar la gestión de la empresa, tanto por el lado del personal del cine, como por el lado de los clientes que facilitará la obtención de información y compra de productos.

- **Interfaces del sistema:**
    - El sistema tendrá una interfaz para los clientes y otra diferente para los administradores.
- **Interfaces de usuario:**
    - La interfaz está diseñada para poder realizar las necesidades de nuestros usuarios, conectándose a nuestra base de datos de los usuarios y pudiendo realizar las funciones que deseen, siempre y cuando tengan el permiso para hacerlo.
- **Interfaces de administrador**:

- Esta interfaz está diseñada para disponer de toda la información “privada” del cine, además de las mismas funcionalidades que tienen los usuarios.

- **Interfaces hardware:**
  - La interfaz hardware está dividida en el ordenador de escritorio que tendrá el negocio para poder administrar sus datos, y en otro ordenador desde donde podrán acceder los clientes.
- **Interfaces software:**
  - Para llevar a cabo esta aplicación haremos uso de JAVA y de MYSQL.
- **Interfaces de comunicación:**
  - Usaremos JDBC, un API que nos permitirá conectarnos a la base de datos usando JAVA.
- **Memoria:**
  - Todavía no se han tomado en consideración las características o límites de la memoria, debido a nuestra falta de experiencia en tener en cuenta estos parámetros a la hora de crear una aplicación.
- **Operaciones:**
  - Hay dos modos de operación para los usuarios, por un lado está el de los clientes y por otro el del personal. Éste tendrá acceso a más información que le ayudará a gestionar el negocio.
- **Requisitos de adaptación:**
  - En el local se instalarán dos ordenadores con nuestra aplicación, uno para los clientes y otro para el personal. En un futuro se podría adaptar a más dispositivos.

2.2 Funciones del producto

DEBE ESTAR ACORDE AL PUNTO 3.4

Nuestra aplicación tiene como objetivos principales:

- Facilitar al usuario la cartelera de películas.
- Facilitar al usuario la disponibilidad en tiempo real de las salas de cine.
- Facilitar al usuario la venta de entradas a los clientes con los puntos anteriores.

Además, la aplicación se divide en las siguientes entidades/ los siguientes módulos:

- **Productos**
  - **Añadir producto**: añade un producto.
  - **Eliminar producto**: deshabilita un producto de la lista.
  - **Actualizar producto**: actualiza la información de un producto.
  - **Buscar producto**: muestra el informe de un producto.
  - **Listar productos**: muestra el informe de todos los productos.
- **Clientes**
  - **Alta cliente**: añade un cliente nuevo a la base de datos.
  - **Baja cliente**: deshabilita a un cliente.
  - **Actualizar cliente**: actualiza el informe del cliente.
  - **Buscar cliente**: muestra el informe de un cliente.
  - **Listar clientes**: muestra el informe de todos los clientes.
- **Proveedores**
  - **Añadir proveedor**: añade un proveedor nuevo a la base de datos.
  - **Eliminar proveedor**: deshabilita a un proveedor.
  - **Actualizar proveedor**: actualiza los datos de un proveedor.
  - **Buscar proveedor**: muestra el informe de un proveedor.
  - **Listar proveedores**: muestra un informe de todos los proveedores.
- **Ventas**
  - **Crear venta**: asocia una venta a un cliente.
  - **Cerrar venta**: guarda los datos y crea la factura.
  - **Devolución venta**: calcula el importe de los productos devueltos.
  - **Buscar venta**: muestra el informe de una venta.
  - **Listar ventas**: muestra un informe de todas las ventas.
- **Marcas**
  - **Añadir marca**: añade una nueva marca.
  - **Eliminar marca**: deshabilita una marca.
  - **Actualizar marca**: actualiza los datos de una marca.

    - **Buscar marca**: muestra el informe de una marca.
    - **Listar marcas**: muestra el informe de todas las marcas.
- **Empleados**
    - **Añadir empleado**: añade un nuevo empleado.
    - **Eliminar empleado**: deshabilita a un empleado.
    - **Actualizar empleado**: actualiza un empleado.
    - **Buscar empleado**: muestra el informe de un empleado.
    - **Listar empleados**: muestra el informe de todos los empleados.

2.3 Características del usuario

Habrá dos tipos de usuarios a los que va dirigida la aplicación, por una parte están los usuarios interesados en comprar un producto de la tienda y por otro están los administradores de la aplicación, que será el personal de la tienda que se encargue de gestionar todo lo relacionado con la actividad de la tienda.
Cada usuario tendrá necesidades distintas, y por consiguiente sus requisitos serán distintos. Además los administradores necesitarán gestionar de una forma eficaz y rápida su negocio, así que debemos tener esto en cuenta.
Este personal tendrá experiencia con el negocio y capacidad de adaptarse a las tareas de la aplicación, ya que esta no requiere de un gran nivel educativo o capacidad intelectual. Por otro lado los clientes que quieren comprar una entrada, estarán solo interesados en la información de los productos disponibles y de poder realizar una compra rápida del producto, sin requerir educación del servicio o experiencia anterior. Por lo tanto el proceso de venta de entradas como la gerencia de la empresa, deben ser intuitivos.

2.4 Restricciones

- Políticas de regulación:

- La aplicación se desarrollará utilizando Java IDE, y será una aplicación de escritorio, de software libre, por lo que no será necesario pagar por su uso.

- Limitaciones hardware:

- Para nuestra aplicación no se necesitará de ningún hardware específico para que funcione correctamente, tan solo un PC para poder correr la aplicación y conectarlo una Base de Datos lo suficientemente potente para poder almacenar todos los datos y correr eficientemente.

- Interfaces con otras aplicaciones:

- Nuestra aplicación estará conectada a una base de datos para poder estar actualizado con todos los cambios de información.

- Operaciones en paralelo:

- La aplicación estará pensada para solo operar con un administrador a la vez. Pero en el caso de los usuarios que quieran comprar tickets, se podrá manejar un gran número de usuarios a la vez.

- Funciones de auditoría:

- No se considerarán las auditorías para nuestra aplicación de software.

- Funciones de control:

- El control de la aplicación se llevará a cabo en un PC, donde se podrá registrar el administrador, o lo podrá acceder un cliente para poder utilizarlo.

- Requisitos de lenguaje de alto nivel:

- La aplicación se desarrollará utilizando el programa JAVA.

- Protocolos de signal handshake:

- No se considerarán la seguridad de los datos de nuestros usuarios.

- Requisitos de fiabilidad:

- La aplicación debe de tener una alta fiabilidad, teniendo un porcentaje bajo de errores, y ser capaz de seguir funcionando cuando aparece un error y no bloquearse.

- Criticidad de la aplicación:
    - Nuestra aplicación no tendrá un uso real en un negocio por lo que no es muy importante. Pero en el caso en el que sí lo fuera, sería crucial para el negocio de la empresa, ya que un software con errores podría causar grandes pérdidas al establecimiento.

- Consideraciones de robustez y seguridad:

    - No se considera la seguridad de nuestra aplicación contra posibles ciber criminales.

2.5 Supuestos y dependencias

El programa sólo estará optimizado para el sistema operativo Windows, así que dispositivos como Mac, quedarán inutilizados.
En caso de que se produzcan múltiples entradas de usuarios o clientes en el sistema, éste será capaz de responder de manera eficaz y no colapsar, pero en el caso de que ocurra cualquier fallo, nuestro equipo será capaz de resolverlo y mantener todo bajo control.

2.6 Requisitos futuros (ADAPTAR A TIENDA)

La meta principal en un futuro es implementar la versión para dispositivos móviles o tablets en la que la interfaz será ligeramente diferente a la Web para Desktop (escritorio). Esto permitirá a los clientes comprar entradas desde cualquier lugar con conexión a internet además de disponer de la cartelera en todo momento.

La meta principal en un futuro es poder implementar

3. Requisitos específicos

3.1 Interfaces externos

PONER ALGUNOS PANTALLAZOS DE LA APLICACIÓN (habrá que esperarse a terminarla)

- La aplicación contendrá una interfaz que estará formada por elementos de texto, imágenes, márgenes, links, listas, menús y botones, todo con el propósito de poder realizar su función de una forma óptima y correcta.

3.2 Funciones

DEBE ESTAR ACORDE AL PUNTO 2.2 Y NO DEBERÍAN HABER DETALLES DE LA INTERFAZ DE USUARIO EN LOS DIAGRAMAS DE ACTIVIDAD
DAR COMENTARIOS MÁS ESPECÍFICOS
¡FALTA TERMINAR DE CORREGIR!

1. **Productos**

1.1. Añadir Producto

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** añade un producto a la base de datos de productos
- **Actor:** administrador
- **Entrada:** nombre del producto
- **Salida:** identificador del producto
- **Origen:** interfaz (añadir) producto y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de productos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y no hay otro producto con el mismo nombre. En caso positivo, se da de alta el producto y se devuelve su identificador correspondiente
- **Precondición:** sin repeticiones de identificadores ni de nombres de productos en la base de datos
- **Postcondición:** sin repeticiones de identificadores ni de nombres de productos en la base de datos; si hay éxito, se ha añadido el nuevo producto

- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

1.2. Eliminar Producto

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** deshabilita un producto de la tienda
- **Actor:** administrador
- **Entrada:** nombre o identificador del producto
- **Salida:** informe del producto
- **Origen:** interfaz (eliminar) producto y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de productos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el identificador existe en la base de datos. En caso positivo, se deshabilita el producto y se muestra su informe correspondiente
- **Precondición:** el nombre introducido debe existir en la base de datos y que esté habilitada en la tienda
- **Postcondición:** se deshabilita el producto de la tienda si existe en la base de datos de productos y está habilitado
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

1.3. Actualizar Producto

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** actualiza la información de un producto de la tienda
- **Actor:** administrador
- **Entrada:** nombre o identificador del producto
- **Salida:** informe del producto
- **Origen:** interfaz producto y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de productos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el identificador existe en la base de

datos. En caso positivo, se pueden actualizar los datos del producto y se muestra su informe correspondiente
- **Precondición:** el nombre introducido debe existir en la base de datos y que el producto esté habilitado en la tienda
- **Postcondición:** los datos del producto quedan actualizados si existe en la base de datos de productos
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

1.4. Buscar Producto
- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra el informe de un producto
- **Actor:** empleado/administrador
- **Entrada:** nombre o identificador del producto
- **Salida:** informe del producto
- **Origen:** interfaz (buscar) producto y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de productos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el producto existe en la base de datos de productos. En caso positivo, se muestra el informe del producto
- **Precondición:** debe haber al menos un producto en la base de datos
- **Postcondición:** en caso de que exista, se muestra el informe del producto
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

1.5. Listar Productos
- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra el informe de todos los productos
- **Actor:** empleado/administrador
- **Entrada:** ninguna
- **Salida:** informe de todos los productos

- **Origen:** interfaz menú y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de productos
- **Acciones:** se muestran todos los productos que existen en la base de datos
- **Precondición:** debe haber al menos un producto en la base de datos
- **Postcondición:** si existe al menos un producto, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

2. Clientes

2.1. Alta cliente

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** añade un cliente a la base de datos de clientes
- **Actor:** empleado/administrador
- **Entrada:** DNI, nombre completo, identificador de tarjeta y teléfono
- **Salida:** identificador del cliente (DNI)
- **Origen:** interfaz (alta) cliente y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de clientes
- **Acciones:** comprobar que los datos son sintácticamente correctos y que no hay otro cliente con el mismo DNI. En caso positivo, se da de alta al cliente y se devuelve su identificador
- **Precondición:** sin repeticiones de identificadores de clientes
- **Postcondición:** sin repeticiones de identificadores de clientes. Si hay éxito es porque el cliente ha sido añadido a la base de datos
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

2.2. Baja cliente

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** deshabilita un cliente y muestra su informe
- **Actor:** empleado/administrador

- **Entrada:** identificador del cliente
- **Salida:** informe del cliente
- **Origen:** interfaz (baja) cliente y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de clientes
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que existe el cliente. En caso positivo, se deshabilita el cliente
- **Precondición:** el cliente existe en la base de datos y está habilitado
- **Postcondición:** si el cliente existe, queda deshabilitado
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

2.3. Actualizar cliente

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** actualizar los datos de un cliente determinado y muestra su informe
- **Actor:** empleado/administrador
- **Entrada:** identificador del cliente
- **Salida:** informe del cliente
- **Origen:** interfaz cliente y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de clientes
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el cliente existe. En caso positivo, se actualizan los datos del cliente y se muestra su informe
- **Precondición:** el cliente existe en la base de datos
- **Postcondición:** si el cliente existe, sus datos quedan actualizados
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

2.4. Buscar cliente

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** busca a un cliente determinado y muestra su informe
- **Actor:** empleado/administrador

- **Entrada:** identificador del cliente
- **Salida:** muestra el informe del cliente
- **Origen:** interfaz cliente y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de clientes
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el cliente existe en la base de datos. En caso positivo, se muestra el informe del cliente
- **Precondición:** el cliente existe en la base de datos
- **Postcondición:** si existe, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

2.5. Listar clientes

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra un informe de todos los clientes
- **Actor:** empleado/administrador
- **Entrada:** ninguna
- **Salida:** el informe de todos los clientes
- **Origen:** interfaz cliente y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de clientes
- **Acciones:** se muestran todos los clientes que existen en la base de datos
- **Precondición:** debe haber al menos un cliente en la base de datos
- **Postcondición:** si hay al menos un cliente, se muestra su informe.
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3. Proveedores

3.1. Añadir proveedor

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** añade un proveedor a la base de datos de proveedores
- **Actor:** administrador
- **Entrada:** DNI, nombre completo y teléfono

- **Salida:** identificador del proveedor (DNI)
- **Origen:** interfaz (añadir) proveedor y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de proveedores
- **Acciones:** comprobar que los datos son sintácticamente correctos y que no hay otro proveedor con el mismo DNI. En caso positivo, se da de alta al proveedor y se devuelve su identificador
- **Precondición:** no hay repeticiones de identificadores ni de nombres de proveedores
- **Postcondición:** no hay repeticiones de identificadores ni de nombres de proveedores. Si hay éxito, el proveedor queda añadido a la base de datos
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3.2. Eliminar proveedor

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** deshabilita a un proveedor
- **Actor:** administrador
- **Entrada:** identificador del proveedor
- **Salida:** se muestran el informe del proveedor
- **Origen:** interfaz proveedor y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de proveedores
- **Acciones:** comprobar que los datos son sintácticamente correctos y que el proveedor existe en la base de datos. Si existe, el proveedor queda deshabilitado
- **Precondición:** que el proveedor exista en la base de datos y que esté habilitado
- **Postcondición:** si el proveedor existe, queda deshabilitado
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3.3. Actualizar proveedor

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta

- **Descripción:** actualiza los datos de un proveedor
- **Actor:** administrador
- **Entrada:** identificador del proveedor
- **Salida:** informe del proveedor
- **Origen:** interfaz proveedor y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de proveedores
- **Acciones:** comprobar que los datos son sintácticamente correctos y que el proveedor existe en la base de datos. En caso positivo, se actualizan sus datos y se muestra su informe
- **Precondición:** el proveedor existe en la base de datos
- **Postcondición:** si existe, se actualizan sus datos y se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3.4. Buscar proveedor

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** buscar un proveedor determinado
- **Actor:** administrador
- **Entrada:** identificador del proveedor
- **Salida:** muestra el informe del proveedor
- **Origen:** interfaz proveedor y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de proveedores
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el proveedor existe en la base de datos. En caso positivo, se muestra su informe
- **Precondición:** el proveedor debe existir en la base de datos
- **Postcondición:** si existe, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3.5 Listar proveedores

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta

- **Descripción:** mostrar un informe de todos los proveedores
- **Actor:** administrador
- **Entrada:** ninguna
- **Salida:** el informe de todos los proveedores
- **Origen:** interfaz proveedor y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de proveedores
- **Acciones:** se listan los informes de todos los proveedores
- **Precondición:** debe haber al menos un proveedor en la base de datos
- **Postcondición:** si existe al menos un proveedor, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

4. Ventas

4.1. Crear venta

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** se guardan los datos de la venta
- **Actor:** empleado/administrador
- **Entrada:** identificador del cliente
- **Salida:** identificador de la venta
- **Origen:** interfaz (crear) venta y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de ventas
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el cliente existe en la base de datos. En caso positivo, se crea la venta, se asocia con el cliente y se devuelve el identificador de la venta
- **Precondición:** no hay repeticiones de identificadores de ventas
- **Postcondición:** no hay repeticiones de identificadores de ventas. Si hay éxito, se ha creado la venta
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

4.2. Cerrar venta

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** se guardan los datos de la venta y se crea la factura
- **Actor:** empleado/administrador
- **Entrada:** identificador de la venta
- **Salida:** se muestra el informe de la venta
- **Origen:** interfaz venta y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de ventas
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y la venta existe. En caso afirmativo, se cierra la venta y se muestra su informe
- **Precondición:** la venta existe en la base de datos y está abierta
- **Postcondición:** si existe, se cierra la venta y se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

4.3. Devolución de venta

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** medio
- **Estabilidad:** alta
- **Descripción:** se calcula el importe de los productos vendidos
- **Actor:** empleado/administrador
- **Entrada:** identificador de la venta
- **Salida:** se muestra el informe de la venta y el importe a devolver
- **Origen:** interfaz venta y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de ventas
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos, que la venta existe. En caso afirmativo, se devuelve el importe y los productos a sus correspondientes lugares
- **Precondición:** la venta existe y está cerrada
- **Postcondición:** si existe, se devuelven el importe y los productos
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

4.4. Buscar ventas

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra un informe de una venta buscada
- **Actor:** empleado/administrador
- **Entrada:** identificador de la venta
- **Salida:** el informe de la venta
- **Origen:** interfaz venta y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de ventas
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que la venta existe y está cerrada. En caso positivo, se muestra el informe de la venta
- **Precondición:** la venta existe en la base de datos y está cerrada
- **Postcondición:** si la venta existe y está cerrada, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

4.5 Listar ventas

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra el informe de todas las ventas
- **Entrada:** ninguna
- **Salida:** el informe de todas las ventas
- **Origen:** interfaz ventas y empleado/administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos de ventas
- **Acciones:** se muestra el informe de todas la ventas
- **Precondición:** debe haber al menos una venta en la base de datos
- **Postcondición:** si existe al menos una venta, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

5. Marcas

5.1. Añadir Marca

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta

- **Estabilidad:** alta
- **Descripción:** añade una marca a la base de datos
- **Actor:** administrador
- **Entrada:** nombre de la marca
- **Salida:** identificador de la marca
- **Origen:** interfaz (añadir) marca y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que no hay otra marca con el mismo nombre. En caso afirmativo, se da de alta la marca y se devuelve su identificador
- **Precondición:** no hay repeticiones de identificadores ni de nombres de marcas
- **Postcondición:** no hay repeticiones de identificadores ni de nombres de marcas. Si hay éxito, la marca queda añadida a la base de datos
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

5.2. Eliminar Marca

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** media
- **Estabilidad:** alta
- **Descripción:** deshabilita una marca
- **Actor:** administrador
- **Entrada:** identificador de la marca
- **Salida:** se muestra un informe de la marca deshabilitada
- **Origen:** interfaz marca y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que la marca existe y está habilitada en la base de datos. En caso afirmativo, se deshabilita la marca y se muestra su informe
- **Precondición:** la marca debe existir en la base de datos y estar habilitada
- **Postcondición:** si existe y está habilitada, se deshabilita y se muestra su informe

- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

5.3. Actualizar Marca

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** medio
- **Estabilidad:** alta
- **Descripción:** actualizar los datos de una marca determinada
- **Actor:** administrador
- **Entrada:** identificador de la marca
- **Salida:** el informe de la marca
- **Origen:** interfaz marca y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que la marca existe en la base de datos. En caso afirmativo, se actualizan los datos de la marca y se muestra su informe
- **Precondición:** la marca debe existir en la base de datos
- **Postcondición:** si existe, se actualizan sus datos y se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

5.4. Buscar Marca

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra el informe de una marca determinada
- **Actor:** administrador
- **Entrada:** identificador de la marca
- **Salida:** el informe de la marca
- **Origen:** interfaz marca y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que existe en la base de datos. En caso afirmativo, se muestra el informe de la marca
- **Precondición:** la marca debe existir en la base de datos

- **Postcondición:** si existe, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

5.5. Listar Marca

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra un informe de todos las marcas
- **Actor:** administrador
- **Entrada:** ninguna
- **Salida:** la lista de las marca
- **Origen:** interfaz marca y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** se muestra la lista de todas las marcas
- **Precondición:** debe haber al menos una marca en la base de datos
- **Postcondición:** si existe al menos una marca, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

6. Empleados

6.1. Añadir Empleado

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** añade un empleado a la base de datos
- **Actor:** administrador
- **Entrada:** DNI, nombre completo, teléfono y número de cuenta
- **Salida:** identificador del empleado
- **Origen:** interfaz (añadir) empleado y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos son sintácticamente correctos y que no hay otro cliente con el mismo identificador. En caso positivo, se da de alta al empleado y se devuelve su identificador
- **Precondición:** sin repeticiones de DNIs de empleados
- **Postcondición:** sin repeticiones de DNIs de empleados. Si hay éxito, el cliente queda añadido

- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

6.2. Eliminar Empleado

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** medio
- **Estabilidad:** alta
- **Descripción:** deshabilita un empleado
- **Actor:** administrador
- **Entrada:** identificador del empleado
- **Salida:** informe del empleado
- **Origen:** interfaz empleado y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos son sintácticamente correctos y que el empleado existe en la base de datos y está habilitado. En caso positivo, se deshabilita al empleado y se muestra su informe
- **Precondición:** el empleado debe existir en la base de datos y estar habilitado
- **Postcondición:** si existe y está habilitado, se deshabilita y se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

6.3. Actualizar Empleado

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** medio
- **Estabilidad:** alta
- **Descripción:** actualiza los datos de un empleado determinado
- **Actor:** administrador
- **Entrada:** identificador del empleado
- **Salida:** informe del empleado
- **Origen:** interfaz empleado y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos son sintácticamente correctos y que el empleado existe en la base de datos
- **Precondición:** el empleado debe existir en la base de datos

- **Postcondición:** si existe, se actualizan sus datos y se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

6.4. Buscar Empleado

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra el informe de un empleado determinado
- **Actor:** administrador
- **Entrada:** identificador del empleado
- **Salida:** se muestra el informe del empleado
- **Origen:** interfaz empleado y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** comprobar que los datos introducidos son sintácticamente correctos y que el empleado existe en la base de datos. En caso positivo, se muestra su informe
- **Precondición:** el empleado debe existir en la base de datos
- **Postcondición:** si existe, se muestra su informe
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

6.5. Listar Empleados

- **Descrita por:** Daniel e Ignacio
- **Prioridad:** alta
- **Estabilidad:** alta
- **Descripción:** muestra un informe de todos los empleados
- **Actor:** administrador
- **Entrada:** ninguna
- **Salida:** se muestra un listado de los empleado
- **Origen:** interfaz empleado y administrador invocador
- **Destino:** el sistema software
- **Necesita:** base de datos
- **Acciones:** se muestra el nombre de todos los empleados
- **Precondición:** debe existir al menos un empleado en la base de datos
- **Postcondición:** si existe, se muestra su nombre

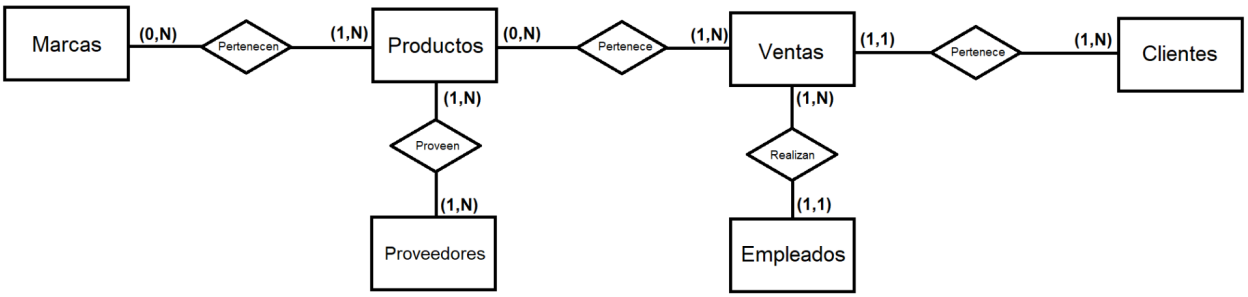
- **Efectos laterales:** si no se puede realizar la operación se muestra un mensaje de error explicando el motivo

3.3 Requisitos de rendimiento

El rendimiento de la aplicación debe de permitirse ser capaz de llevar a cabo sus funciones sin tardar más de 1 segundo por transacción y podrá soportar a un usuario a la vez. El usuario se podrá conectar solamente desde una terminal y cuando pasen más de 10 minutos de inactividad, se desconectará de la aplicación.

3.4 Modelo del dominio (Logical Database Requirements)

SE SUPONE QUE DEBEMOS SACARLO DEL PROYECTO



ESTO DE ABAJO CREO QUE SOBRA

ENTIDADES	Películas	Salas	Clientes	Ventas	Taquillas	Empresas
Películas	X					
Salas		X				
Clientes			X			
Ventas				X		
Taquillas					X	

Empresas						X

	Menú registro	Registro administradores	registro usuario	menú administrador	menú usuario	películas	Gestión de cartelera	Administrador	Compra
Menú registro	X							Ninguno/ varios	
Registro administradores		X						Ninguno/ varios	
registro usuario	Uno		X		Uno				
menú administrador		Uno		X		Ninguno /Varios	Una	Ninguno/ varios	
menú usuario					X	ninguna /varias			Uno
películas					Ninguno /Varios	X			
Gestión de cartelera						Ninguno /Varios	X	Ninguno/ Varios	
Administrador		Uno		Uno		Ninguno /Varios	Uno	X	Ninguna
Compra					Uno	Ninguna /varias			X

- **Menú registro**: interfaz que se encarga de diferenciar los dos tipos de registros.
- **Registro administradores:** interfaz con dos opciones, una para nuevo administrador y otra para uno ya existente.
- **Registro usuario**: interfaz con dos opciones, una para nuevo usuario y otra para uno ya existente.
- **Menú administrador**: interfaz que muestra las diferentes opciones de administrador.
- **Menú usuario**: interfaz que muestra las diferentes opciones de un usuario.
- **Películas**: se encarga de mostrar las películas, sus horarios, salas y asientos ocupados.
- **Gestión de cartelera:** se encarga de gestionar, modificar o actualizar películas de la cartelera.
- **Administrador:** se encarga de mostrar la información diaria de cada sala.

- **Compra**: se encarga de la gestión de ventas de entradas a los clientes.

3.5 Restricciones de diseño

Restricciones de formato:
El proyecto se ajusta al estándar IEEE Std. 830-1998.

Restricciones de tiempo:
El proyecto tendrá dos entregas:

- 1ª entrega: será el día 16/12/2022 y consistirá en la aportación de los siguientes documentos: Plan del proyecto, Especificación de Requisitos (SRS), el archivo sobre la planificación temporal con el formato de MS Project, los archivos correspondientes a los diagramas de casos de uso y de actividades UML con el formato IBM Rational Software Architect 9.6, además de un informe del trabajo desarrollado por cada integrante del equipo en el que se califica al resto de integrantes de manera individual.
- 2ª entrega: será el día 5/5/2023 y consistirá en la entrega del producto final.

Restricciones de software:
La aplicación se desarrollará a través del lenguaje de programación Java.

3.6 Atributos del sistema software

Fiabilidad:

- La aplicación ha de tener una tasa de fallos muy baja cuando interactúa con sus usuarios y Base de Datos.

Disponibilidad:

- La aplicación debe de siempre estar disponible para su uso, aunque al depender de servidores para la Base de Datos, se apunta por una disponibilidad del 99%.

Seguridad:

- No se considerará la seguridad de nuestra aplicación contra ciberataques.

Mantenibilidad:

- Gracias a que la aplicación está programada en JAVA, será fácil de expandir y añadir nuevas funcionalidades a la aplicación y también será fácil de expandirlo a nuevos sistemas operativos.

Portabilidad:

- La aplicación puede ser multiplataforma al estar desarrollada en JAVA, haciendo el código reutilizable para nuevos sistemas operativos.