

Trabajo Final Integrador

TTPS Opción Ruby

En este Trabajo Final Integrador (TFI), buscaremos que apliquen los conocimientos adquiridos en la materia para resolver un problema real, basado en un caso hipotético. El objetivo de la implementación que deberán hacer es concebir un sistema que permita resolver las necesidades expuestas en este documento, como si se los hubiera pedido un potencial cliente o persona interesada en los servicios de desarrollo de software que ustedes proveen.

Contexto

Están a cargo del desarrollo de una aplicación de gestión de inventario de una disquería que vende discos (CD's y vinilos) tanto nuevos como usados. La aplicación deberá permitir al personal del negocio (empleados/as) administrar el stock de productos, asentar ventas y tener una página pública que permita al público ver los discos que la tienda tiene para ofrecer. En este documento se describirán los puntos principales y la funcionalidad mínima que la aplicación debe proveer, a la cual podrán agregar soporte para más acciones o interacciones, siempre respetando lo especificado aquí como base.

Requisitos técnicos

El desarrollo debe realizarse utilizando el siguiente stack tecnológico:

- La versión estable más reciente del *framework* Ruby on Rails (8.1.0 al momento de escribir este documento).
 - Una versión reciente de Ruby (3.4.0 o superior), acorde a lo requerido por el framework.
 - Una base de datos para dar soporte de persistencia. SQLite es ideal y suficiente.
- Adicionalmente, se les pide que no reinventen la rueda y cuando sea posible utilicen gemas bien establecidas que proveen funcionalidad habitual en una aplicación *web*, como puede ser el caso de la autenticación de usuarios.

Alcance

Deberán desarrollar una aplicación *web* para gestionar el inventario de un negocio que comercializa discos de música (una disquería), que brindará la siguiente funcionalidad:

- **Storefront:** un sitio web público que permita a las personas visitantes ver los productos disponibles. No es requerido que puedan realizarse compras online, pero sí que se pueda ver el stock disponible. Ver apartado Storefront.
- **Administración de productos:** una interfaz de administración que permita a las personas empleadas de la disquería gestionar los productos, su stock y su precio. Ver apartado Administración de productos.
- **Administración de ventas:** una interfaz de administración que permita a las personas empleadas registrar las ventas realizadas y los productos vendidos. Ver apartado Administración de ventas.
- **Gestión de usuarios:** un sistema de autenticación y autorización que permita solo a las personas empleadas de la disquería acceder a la interfaz de administración. Ver apartado Gestión de usuarios.

Storefront

El Storefront (la parte pública) es la cara visible del negocio.

- Cualquier persona que entre al sitio, sin necesidad de registrarse ni nada, puede ver los productos disponibles.
- Para cada producto, el cliente quiere que se muestre:
 - Una o más imágenes, (en caso de ser varias imágenes debe definir cuál es la imagen de portada)
 - El nombre del disco
 - Una descripción breve
 - El precio
 - El stock disponible
- Que se pueda navegar y por categorías y tipos (vinilos, CDs, nuevos, usados).
- Que se pueda buscar (filtrar) productos por:
 - Título del album
 - Artista
 - Año
 - Tipo de producto (vinilo o CD)
 - Estado (nuevo o usado)
 - Género (pop, rock, jazz, metal, etc)
- Además, opcionalmente, para los discos usados que se puede asociar un pequeño audio de muestra, para que los clientes tengan una idea de cómo se escucha.
- Que se muestre una sección de 'Discos relacionados' o "Tal vez también te interese..." según el producto que se esté viendo.
- Importante: en el storefront no se van a poder hacer compras online. Solo mostrar la información de los productos.

Backstore

En esta parte se gestiona todo el negocio.

Administracion de usuarios

Roles de usuario

Existirán distintos **roles de usuario**, cada uno con diferentes permisos:

- **Administrador:** acceso a todas las funcionalidades de la aplicación. Puede incluso crear/editar/eliminar otros usuarios administradores.
- **Gerente:** puede administrar productos y ventas, y gestionar usuarios, pero no puede crear ni modificar usuarios con rol de administrador.
- **Empleado:** puede administrar productos y ventas, pero no puede gestionar usuarios.

Todos los roles deben poder modificar los datos de su propia cuenta, **menos su rol**.

Administración de discos

La aplicación debe permitir gestionar los productos, su stock y su precio. Para eso:

- **Listar productos:** ver una lista con todos los productos existentes.
- **Agregar producto:** poder crear un nuevo producto.
- **Modificar producto:** editar un producto existente.
- **Eliminar producto:** hacer un borrado lógico. Es decir, no se borra de la base, sino que se pone la **fecha de baja** y el **stock en 0**.
- **Cambiar stock:** modificar el stock disponible de un producto.

Cada producto debe tener como mínimo estos atributos:

- Nombre
- Descripción
- Autor
- Precio unitario
- Stock disponible
- Imágenes (al menos una, podría haber más)
- Categoría (género musical)
- Tipo (vinilo o CD)
- Estado (Nuevo o usado)

- Audio (opcional, solo para los usados, de corta longitud)
- Fecha de ingreso al inventario
- Fecha de última modificación
- Fecha de baja (si fue eliminado)

Gestión de ventas

Aunque el storefront no habilite compras online, se necesita que los empleados/as registren en el sistema las ventas presenciales que hacen en la disquería.

- **Listar ventas:** ver todas las ventas realizadas.
- **Asentar venta:** crear una nueva venta.
 - Cada venta debe tener una fecha y hora de realización.
 - Debe permitir agregar productos vendidos, con su cantidad y precio de venta.
- **Cancelar venta:** permitir cancelar una venta.
 - Al cancelar, el stock de los productos debe volver automáticamente.
 - Las ventas canceladas no se eliminan de la base, si no que se hace una baja lógica.
- **Generar factura:** genera un pdf de la factura de venta y permite descargarla.

Importante:

- Las ventas deben **afectar el stock de los productos**.
- No se puede vender más stock del disponible.
- La operación de registrar una venta debe ser **transaccional**: si no alcanza el stock, no se debe registrar nada.

Datos mínimos de una venta:

- Fecha y hora de realización
- Productos vendidos (con cantidad y precio)
- Total de la venta
- Empleado/a que realizó la venta
- Datos del/la comprador/a

Bonus track: módulo de reportes de ventas. (Si les sobra tiempo de desarrollo y quieren pensar en un Modulo de reportes, bienvenido sea)

Consideraciones generales

- El diseño del modelo de datos que armen tiene que considerar no solo los modelos explícitamente mencionados en este enunciado, sino también cualquier otro que sea necesario para cumplir con los requerimientos. Esa decisión queda a su criterio.
- Se espera que todos los modelos tengan las validaciones adecuadas para asegurar la integridad de los datos.
- Se deberá incluir un conjunto de datos de prueba para poder evaluar el sistema. Estos datos deberán generarse mediante el script `db/seeds.rb` del proyecto Ruby on Rails.
- Se espera que la interfaz de usuario sea cómoda, entendible, y lo más intuitiva posible, pero no se solicitará para la entrega ningún tipo de librería ni interfaz en particular. Debería ser un diseño responsive, que se visualice bien en distintos dispositivos (desktop y mobile).
- Se deberá documentar en el archivo `README.md` del raíz del proyecto cualquier decisión de diseño importante que se haya tomado, así como también se deberán dejar escritos los requisitos técnicos y los pasos (comandos) para hacer funcionar la aplicación de manera local.
- La funcionalidad solicitada en este enunciado es la mínima que debe implementarse de base para poder aprobar el trabajo. En caso de desearlo, podrán agregar más funcionalidad a la aplicación, siempre y cuando respeten lo solicitado en este documento. No se permitirá el intercambio de funcionalidades solicitadas por otras que no lo estén.

Modalidad de la entrega

El trabajo deberá realizarse de manera grupal. Los trabajos entregados se compararán entre grupos para encontrar posibles similitudes en su estructura que pudieran provenir de copias entre distintas entregas. Esta entrega es de carácter obligatorio, y su aprobación (en cualquiera de sus instancias) es condición necesaria para aprobar la cursada de la materia. El trabajo tiene que estar versionado en un repositorio Git de GitHub **de todos** los participantes del grupo, cuya URL deberán entregarnos mediante una tarea dedicada que publicaremos en el curso de la plataforma de Cátedras (Moodle). Quienes sean parte del grupo deben contar con commits y participación en el repositorio.

En caso que el repositorio sea privado, vamos a necesitar que nos brinden acceso al mismo para poder descargarlo y evaluarlo. Si esta es su decisión, por favor consultá antes de la fecha límite de la entrega con los docentes para saber a qué usuarios de GitHub deberán brindarles acceso a sus repositorios. Las fechas límite de entrega del trabajo fueron publicadas en el curso de la plataforma de Cátedras. No se admitirán entregas fuera de término.