UTN.BA
UNIVERSIDAD TECNOLÓGICA NACIONAL
FACULTAD REGIONAL BUENOS AIRES
CÁTEDRA DESARROLLO DE SOFTWARE

Tienda Sol - Plataforma de Comercio Electrónico

Trabajo Práctico
-2C 2025-

Contexto general

Durante los últimos años, el crecimiento del comercio electrónico ha sido exponencial. La posibilidad de comprar desde cualquier lugar, comparar productos, acceder a múltiples métodos de pago y recibir los artículos en casa en pocos días ha transformado radicalmente la forma de consumir bienes y servicios.

En ese marco, pequeñas y medianas empresas buscan cada vez más canales digitales para ofrecer sus productos, sin necesidad de depender de marketplaces centralizados. A su vez, muchas personas emprendedoras necesitan soluciones simples pero escalables para administrar su tienda digital sin tener conocimientos técnicos avanzados.

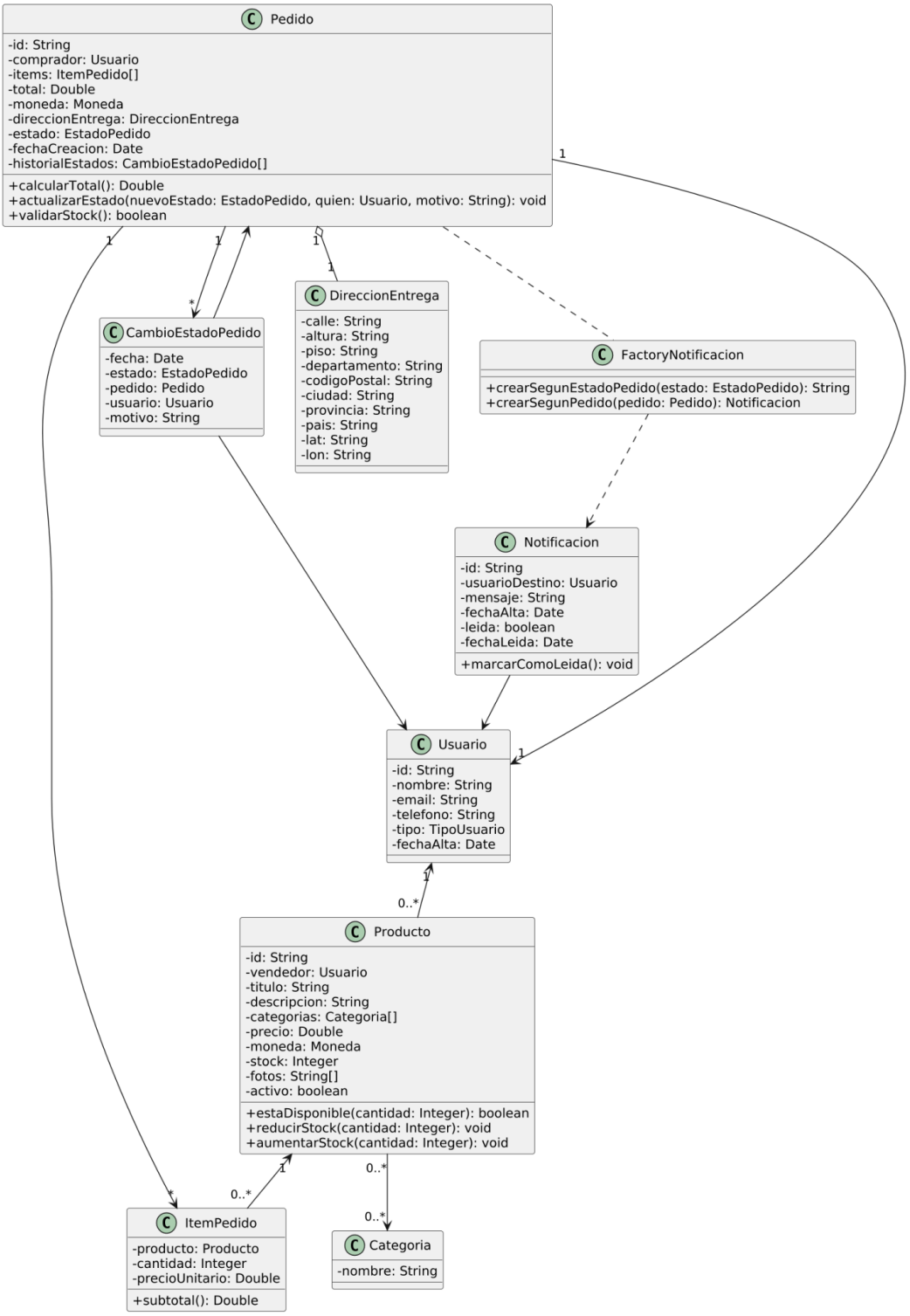
En respuesta a esta necesidad, una empresa emergente del sector tecnológico ha solicitado el desarrollo de una plataforma web. Esta plataforma permitirá a vendedores publicar productos, gestionar su stock y recibir pedidos, mientras que los compradores podrán explorar catálogos, buscar productos por categorías o palabras clave, ver detalles, agregar ítems al carrito y realizar compras. También se espera que la plataforma cuente con un sistema de notificaciones para mantener informados tanto a compradores como vendedores sobre los eventos importantes relacionados con sus productos o pedidos.

Para lograr este objetivo, diseñaremos e implementaremos ***Tienda Sol***, una solución web moderna con experiencia fluida para que tanto los vendedores como compradores se sientan a gusto comprando y vendiendo productos y servicios. El desarrollo del Sistema se organizará en cuatro iteraciones, cada una enfocada en una parte clave de la implementación.

Entrega 1: Implementación primeras funcionalidades + Configuración del Proyecto

Contexto

En esta primera iteración nos enfocaremos en implementar una primera iteración de la App Backend. Dejamos a disposición un diagrama de clases que podría ser de utilidad.



Respecto a las notificaciones, nuestro Tech Leader nos ha mencionado que:

- Cada vez que se realice un pedido, es necesario enviarle una notificación al Vendedor, donde se le indique quién realizó el pedido, qué productos incluye, el total del mismo y dirección de entrega.
- Cada vez que un vendedor marque un pedido como enviado, es necesario notificar al comprador.
- Si un comprador decide cancelar un pedido, es necesario notificar al vendedor.

Entregables

Para esta primera entrega las funcionalidades a implementar, no requieren tener una persistencia real contra Mongo. Para solucionar esto se puede usar cualquier estructura que les deje almacenar valores en memoria (Array, lista, mapa… etc), para usar de mock de DB.

1. **Implementación** del endpoint “Health Check”
    a. Path que responda 200 en /health-check
2. **Funcionalidades**
    a. Creación de un pedido, validando el stock disponible de cada producto
    b. Consulta de un pedido ya hecho
    c. Cancelación de un pedido antes de que haya sido enviado
    d. Consulta del historial de pedidos de un usuario
    e. Marcado de un pedido como enviado por parte del vendedor
3. ***Explicación*** del Git flow definido para utilizar a lo largo del proyecto.

Detalles a tener en cuenta

Implementar los endpoints necesarios utilizando la Arquitectura REST y diseñando la App internamente con Arquitectura en capas.

Tecnologías a utilizar

- Backend
    - Lenguaje de programación base: JavaScript
    - Entorno de ejecución: Node.js
    - Framework: Express

- Git como Sistema de Versionado de Código, con repositorios en Github.

Entrega 2: Completitud de APIs + Persistencia

Contexto

En esta segunda iteración debemos completar lo realizado en la 1ra entrega con los siguientes endpoints necesarios para garantizar el cumplimiento de los requerimientos listados a continuación, teniendo en cuenta que todos los datos que se manipulen deben ser persistentes. Además, nuestro Tech Leader nos ha solicitado que la API se base en el enfoque REST.

Búsqueda y Visualización de Productos

Los usuarios deben poder realizar búsquedas de productos en función de diferentes criterios para encontrar el lugar que mejor se adapte a sus necesidades. Se espera:

- Un endpoint para listar los productos de un vendedor en particular, con la posibilidad de aplicar filtros como:
  - Término de búsqueda (nombre, categoría, descripción).
  - Rango de precios.
- Implementación de paginación para mejorar la eficiencia de las consultas.
- Ordenamiento por precio ascendente/descendente, y por “más vendido”.

Visualización de notificaciones

Los usuarios deben recibir información relevante a través de notificaciones dentro de la plataforma. Las notificaciones estarán relacionadas con:

- Confirmación de pedido.
- Aviso de pedido enviado.
- Cancelación de pedido.

Se espera:

- Endpoint para obtener la lista de notificaciones sin leer de un usuario.
- Endpoint para obtener la lista de notificaciones leídas de un usuario.
- Endpoint para marcar una notificación como leída.

Por otro lado, también es necesario implementar los tests unitarios sobre la capa de Servicios del Sistema.

Entregables

1. ***Implementación*** de la API REST del Sistema, que incluya todos los endpoints necesarios para dar solución a los requerimientos listados.
2. ***Implementación*** de la persistencia de las entidades de dominio en una base de datos NoSQL Documental.
3. ***Implementación*** de los Test Unitarios de la capa de servicios y de la capa de dominio.
4. ***Documentación*** de la API REST con Swagger.

Tecnologías a utilizar

- Todas las mencionadas en la primera iteración.
- MongoDB como Base de Datos Documental NoSQL.
- Jest como framework de Testing Unitario.
- Swagger como Herramienta de Documentación de APIs. Se permite la utilización de alguna dependencia que genere el contenido de Swagger.

Entrega 3: UI + Integración con Backend

Contexto

En esta tercera iteración nos enfocaremos en el desarrollo inicial del Frontend de nuestro aplicativo. En particular, trabajaremos en diseñar, maquetar e implementar las pantallas, con la correspondiente navegabilidad entre ellas, pero solamente integrando una de las funcionalidades (por ahora) con nuestro Backend.

Nuestro Tech Leader nos ha dicho que tenemos la libertad de generar las interfaces de usuario según nuestro gusto y criterio, pero teniendo en cuenta que es importante respetar los siguientes requerimientos no funcionales:

Interfaz Intuitiva

- El diseño de la aplicación debe ser claro y fácil de entender para usuarios sin experiencia previa en plataformas similares.
- Se debe utilizar una estructura de navegación coherente, asegurando que los usuarios puedan acceder a las funcionalidades principales en pocos clics.

Aprendizaje Rápido

- La interfaz debe seguir patrones de diseño comunes en aplicaciones de ecommerce para que los usuarios reconozcan fácilmente las funcionalidades.
- Se deben incluir textos descriptivos y elementos visuales que guíen a los usuarios en cada paso del proceso de compra.

Feedback Visual y Notificaciones

- Deben incluirse mensajes de error y confirmación en cada interacción relevante, como la adición de productos al carrito o la confirmación del pedido.
- Indicadores visuales (loaders, skeletons, etc.) deben mostrar el estado de carga de las solicitudes al Backend para evitar que el usuario piense que la aplicación está inactiva.

Diseño Responsivo

- El aplicativo debe ser completamente funcional en distintos dispositivos (desktop, tablets y móviles).
- Se debe asegurar que todos los elementos sean accesibles en pantallas de distintos tamaños sin afectar la usabilidad.

Accesibilidad

- Se deben seguir buenas prácticas de accesibilidad, como el uso de contrastes adecuados, etiquetas ARIA y soporte para navegación mediante teclado.
- Los textos y botones deben ser lo suficientemente grandes para facilitar su uso en dispositivos móviles.

Consistencia en la UI

- Los estilos, colores y tipografías deben ser homogéneos en toda la aplicación para mejorar la experiencia del usuario.
- Los componentes reutilizables deben seguir un diseño coherente para evitar confusión.

Por otro lado, tal como fue mencionado, tendremos que implementar de forma completa la funcionalidad de "*Búsqueda y Visualización de Productos*", realizando las correspondientes llamadas a nuestro Backend.

Además, se deberá implementar un módulo de manipulación del carrito de compras, que permita a los usuarios:

- Agregar productos desde la vista de producto o catálogo.
- Ver un resumen del carrito actualizado con totales.
- Eliminar productos del carrito.

El carrito de compras será gestionado exclusivamente del lado del cliente, sin persistencia en base de datos.

Entregables

1. ***Maquetado*** de todas las pantallas que el equipo considere necesarias para dar cumplimiento a todos los requerimientos funcionales de la iteración 2, con la correspondiente navegabilidad entre ellas y ajuste al cumplimiento de los requerimientos no funcionales listados en esta iteración.
2. ***Implementación*** de funcionalidad completa de "*Búsqueda y Visualización de Productos*, incluyendo su integración con el backend.
3. ***Implementación*** del carrito de compras del lado cliente.
4. ***Justificación*** acerca del cumplimiento de los requerimientos no funcionales.

Tecnologías a utilizar

- Todas las mencionadas en las anteriores entregas.
- Next.js como Framework para el desarrollo de nuestro Frontend.
- React como biblioteca de generación de componentes para nuestro Frontend.
- Axios como Cliente HTTP para integrar nuestro Frontend con nuestro Backend.

Entrega 4: Entrega final

Contexto

En esta última iteración nos enfocaremos en terminar de implementar todas las funcionalidades de nuestro aplicativo, integrando de forma completa el Frontend con el Backend. Además, también tendremos que ocuparnos de realizar Tests de Integración en la capa de controladores de nuestro Backend, y Test E2E en nuestro Frontend.

Por último, nuestro Tech Leader nos ha indicado que llegó el momento de poner en producción la primera versión del aplicativo para que esté disponible en la Web para todos los usuarios, motivo por el cual tendremos que ocuparnos de desplegar la solución en nube.

Entregables

1. ***Implementación*** de todas las funcionalidades completas, con la correspondiente integración con el Backend.
2. ***Despliegue*** del aplicativo en la nube (Backend + Frontend).
3. ***Documentación*** acerca del despliegue del aplicativo que contenga el detalle de los pasos a seguir para desplegar el aplicativo por primera vez y por cada vez que se quiere subir a producción una nueva release.
4. ***Tests de integración*** en la Capa de Controladores del Backend: solamente es necesario realizar 1 test, cuyo endpoint a testear queda a criterio del equipo.
5. ***Tests E2E*** en el Frontend: solamente es necesario realizar 1 test, cuya funcionalidad a testear queda a criterio del equipo.

Tecnologías a utilizar

- Todas las mencionadas en las anteriores entregas.
- Render como Cloud Application Platform para el despliegue de nuestro Backend (se pueden utilizar algunas otras alternativas, a criterio del equipo y en común acuerdo con el equipo docente).
- Netlify como plataforma para despliegue de nuestro Frontend (se pueden utilizar algunas otras alternativas, a criterio del equipo y en común acuerdo con el equipo docente).
- Cypress como herramienta de Testing E2E. Está permitido buscar y utilizar alguna otra herramienta alternativa (como Jest, por ejemplo).
- Jest como framework para Testing de Integración