

## Desarrollo del Frontend

Este documento describe las fases seguidas para la implementación del frontend de la aplicación de onboarding de compra con pago por tarjeta utilizando Wompi.

# X Stack Tecnológico

• Framework: React.js

State Management: Redux Toolkit

Routing: React Router DOM

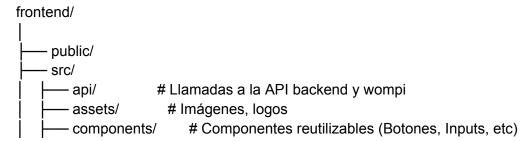
• Estilo: TailwindCSS + Material UI

Validación: React Hook Form + Yup

Persistencia local: localStorage

• HTTP Client: Axios

# Estructura de Carpetas



```
# Vistas: Home, Checkout, Summary, Result
pages/
redux/
              # Store y slices
            # Definiciones de tipos TS
types/
            # Funciones auxiliares y helpers
- utils/
             # Configuración de rutas
App.tsx
               # Punto de entrada principal
main.tsx
```

# 🚀 Fases del Desarrollo

### 1. Configuración inicial

- Proyecto creado con Vite + React + TypeScript
- Instalación de TailwindCSS, Material UI, React Router, Redux Toolkit, React Hook Form, Yup y Axios

## 2. Definición del flujo de navegación

- Página de producto (/)
- Formulario de pago (/checkout)
- Resumen de transacción (/summary)
- Resultado del pago (/result)

## 3. Pantalla de producto

- Se obtiene producto desde el backend
- Muestra nombre, descripción, unidades disponibles y precio
- Botón "Pagar con tarjeta de crédito"

## 4. Formulario de pago

- Inputs para:
  - Nombre del titular

- Número de tarjeta (con logos VISA/MC detectados)
- Fecha de expiración (MM/YY)
- o CVC
- o Nombre, teléfono, dirección de envío
- Validaciones con Yup
- Almacena datos en Redux y localStorage

#### 5. Pantalla de resumen

- Se calcula el total incluyendo:
  - o Precio del producto
  - Tarifa base
  - o Tarifa de envío
- Botón de "Confirmar pago"
- Llama a backend para crear transacción con estado PENDING
- Llama API de Wompi para procesar pago

## 6. Resultado del pago

- Se muestra mensaje de éxito o error según respuesta de Wompi
- Se actualiza el stock del producto
- Redirección a la página inicial tras unos segundos

### 7. Persistencia del progreso

- Los datos del formulario y estado del pago se guardan en Redux y en localStorage
- Se cargan desde el storage al reiniciar o refrescar la app

#### 8. Pruebas unitarias

- Pruebas con Jest + Testing Library
- Cobertura >80% en componentes clave



# **Ejecución local**

cd frontend npm install npm run dev



## Build para producción

npm run build



## Despliegue

La app fue desplegada en AWS (CloudFront + S3) y se conecta con el backend Nest.js desplegado en AWS.

## 🔽 Estado Final

### La SPA permite:

- Visualizar producto
- Completar y validar datos de pago y entrega
- Mostrar resumen del pago
- Procesar pago en Wompi
- Mostrar resultado y actualizar stock

Con una UI amigable, validaciones robustas y almacenamiento del progreso, se cumple el flujo completo de onboarding de compra solicitado.