

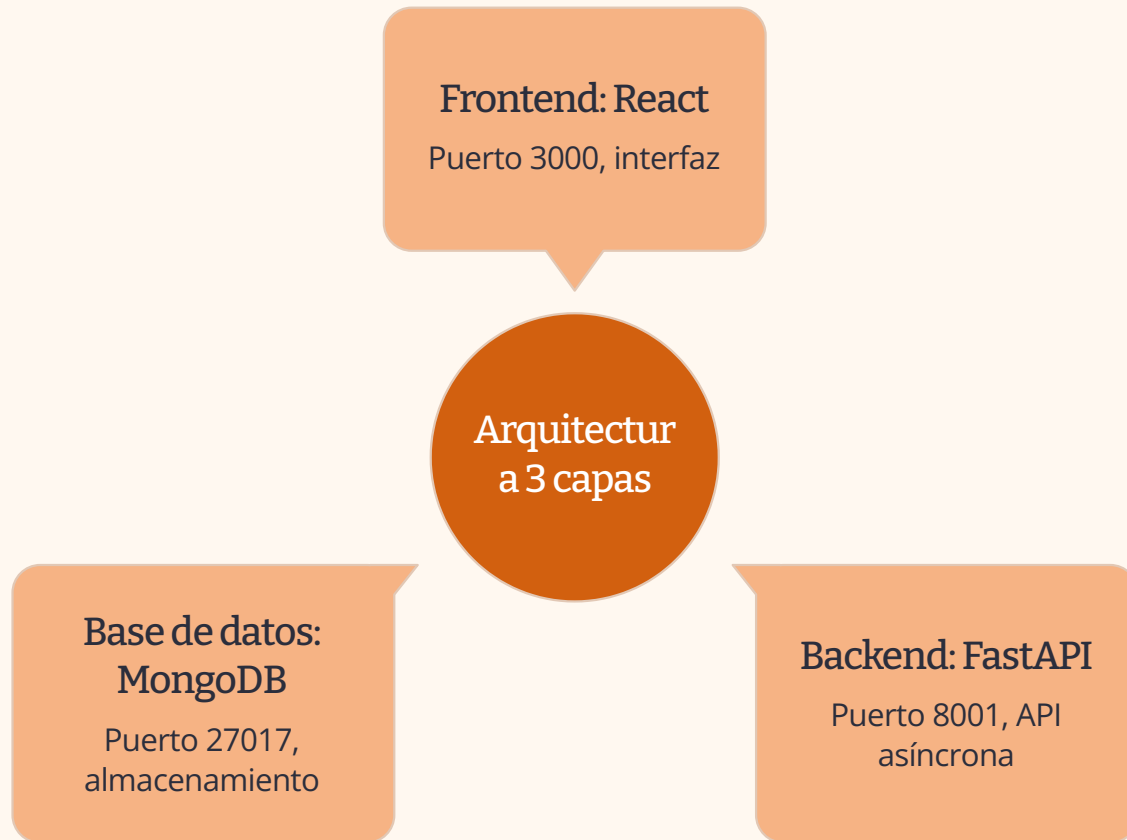
# LABCEL San Antonio

📱 GUÍA TÉCNICA

Documentación completa de arquitectura y funcionamiento de la plataforma web de personalización de fundas para móviles



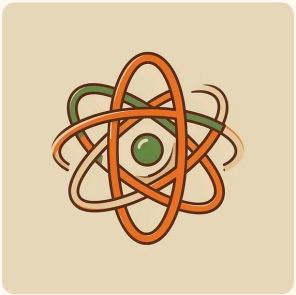
# Arquitectura del Sistema



La aplicación utiliza una arquitectura de tres capas que garantiza escalabilidad y mantenibilidad:

- **Frontend:** React 19 con Tailwind CSS para interfaces responsivas
- **Backend:** FastAPI con Motor para operaciones asíncronas
- **MongoDB para almacenamiento flexible de datos**

# Tecnologías Principales



## Frontend React

React 19 con React Router  
DOM para navegación SPA y  
Axios para comunicación HTTP



## Backend FastAPI

Framework API moderno con  
Pydantic para validación de  
datos y Python-Jose para  
autenticación



## Base de Datos MongoDB

Motor como driver asíncrono  
para operaciones optimizadas  
y esquemas flexibles



## Diseño Tailwind CSS

Framework CSS utilitario con  
componentes Shadcn UI  
basados en Radix para  
accesibilidad

# Estructura de Directorios

## Backend

El servidor API está organizado en un único archivo principal `server.py` que contiene:

- Configuración de FastAPI y MongoDB
- Modelos Pydantic para validación
- Helpers de autenticación
- Definición completa de endpoints

Las dependencias se gestionan mediante `requirements.txt` y las variables de entorno en `.env`

## Frontend

La aplicación React sigue una estructura modular:

- **components/** Navbar, Footer, UI reutilizables
- **pages/** Vistas principales y panel admin
- **context/** Estado global (Auth, Cart)
- **public/** Recursos estáticos

Más de 30 componentes Shadcn UI en `components/ui/` para interfaces consistentes

# Proceso de Compra del Cliente



## Página Principal

El usuario explora ofertas y productos destacados



## Catálogo

Visualiza y filtra fundas disponibles por categoría



## Personalización

Sube imágenes, elige modelo de teléfono y ajusta diseño



## Carrito

Revisa productos y procede al checkout

1

## Checkout

Ingresa datos de envío y selecciona método de pago

2

## Confirmación

Recibe número de pedido y notificación por email

3

## Rastreo

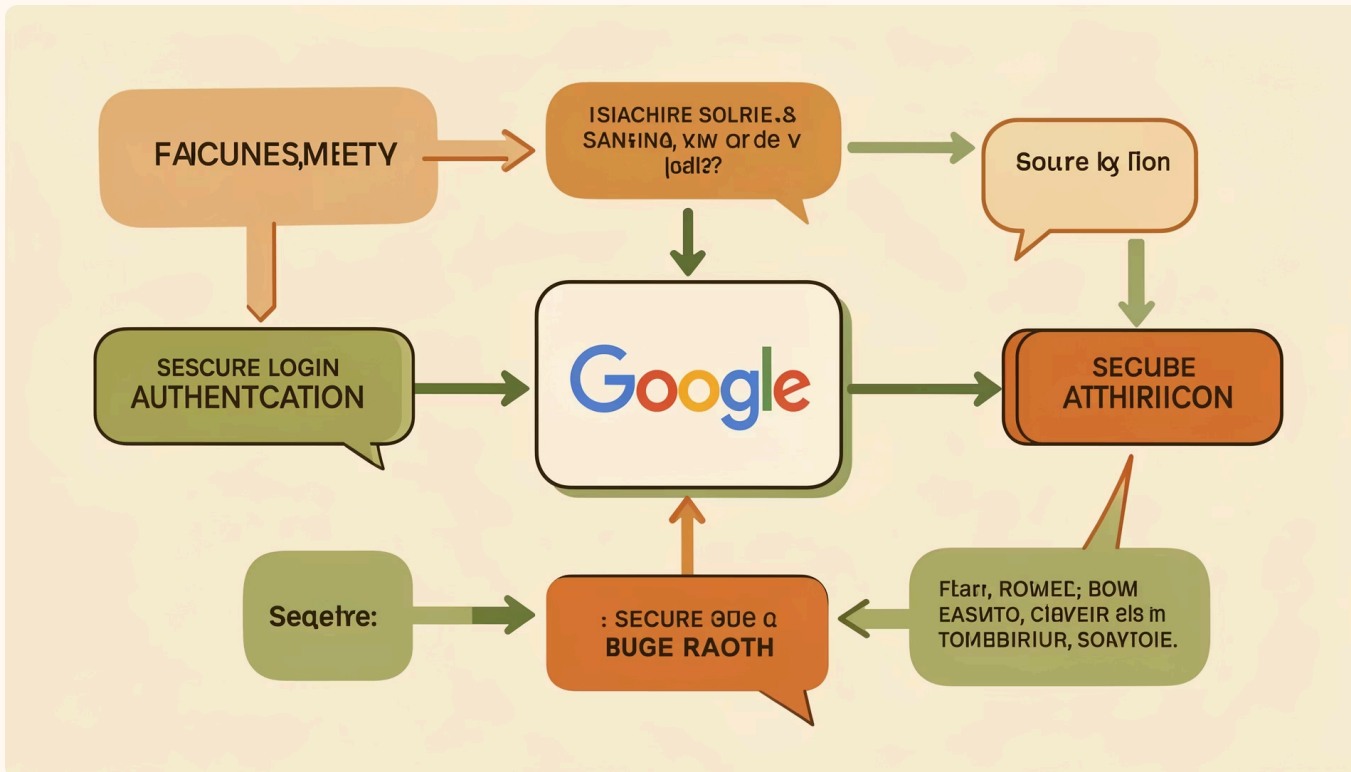
Consulta el estado del pedido en tiempo real

4

## Entrega

Recibe el producto personalizado

# Sistema de Autenticación Google OAuth



La autenticación se gestiona mediante Google OAuth a través del servicio Emergent Auth, proporcionando seguridad y experiencia de usuario optimizada.

01

## Clic en "Iniciar Sesión"

El usuario inicia el proceso desde el frontend

02

## Redirección a Emergent Auth

Se redirige al servicio de autenticación externo

03

Login con Google

El usuario autentica con su cuenta de Google

04

## Intercambio de Tokens

El backend valida el session\_id y crea sesión local

# Base de Datos MongoDB

El sistema utiliza MongoDB con 8 colecciones principales para gestionar todos los aspectos de la aplicación de forma eficiente y escalable.

## users

Datos de usuarios registrados con email, rol y preferencias de contacto

## user\_sessions

Sesiones activas con tokens de autenticación y timestamps

## products

Catálogo completo con precios, descripciones e imágenes base

## orders

Pedidos con items, estados, historial y datos del cliente

## phone\_brands

Marcas de dispositivos compatibles (Apple, Samsung, etc.)

## phone\_models

Modelos específicos con plantillas de cámara configurables

## uploaded\_images

Registro de imágenes personalizadas subidas por usuarios

## notifications

Log histórico de notificaciones enviadas por email/WhatsApp

# Endpoints Principales

## Públicos

- GET /api/products - Listar productos
- GET /api/phone-brands - Marcas disponibles
- GET /api/phone-models - Modelos por marca
- GET /api/orders/track/{id} - Rastrear pedido

## Autenticados

- POST /api/orders - Crear pedido
- GET /api/orders - Mis pedidos
- POST /api/upload/image - Subir imagen

## Autenticación

- POST /api/auth/session - Login OAuth
- GET /api/auth/me - Usuario actual
- POST /api/auth/logout - Cerrar sesión

## Administración

- GET /api/admin/stats - Dashboard admin
- PUT /api/products/{id} - Editar producto
- PUT /api/orders/{id}/status - Cambiar estado
- PUT /api/users/{id}/role - Gestionar roles

# Personalizador de Fundas

## Características Clave

- Vista previa en tiempo real del diseño
- Configuración automática de cámaras por modelo
- Controles de zoom y posicionamiento de imagen
- Upload de imágenes personalizadas
- Integración directa con carrito

El componente `Customizer.js` es el corazón de la experiencia de personalización. Permite a los usuarios:

La configuración de cámaras se adapta automáticamente según el modelo seleccionado (dual, triple, quad) para reflejar con exactitud el dispositivo real.

### 1 Seleccionar marca y modelo

Sistema de selección en cascada que muestra solo modelos compatibles

### 2 Subir imagen personalizada

Upload con validación y preview instantáneo en el dispositivo virtual

### 3 Ajustar diseño con precisión

Controles de escala y posicionamiento con feedback visual inmediato

# Despliegue en Producción



## Requisitos del Servidor

Node.js  $\geq 18.x$ , Python  $\geq 3.11$ , MongoDB  $\geq 6.0$ , mínimo 1GB RAM y 5GB disco



## Build del Frontend

Ejecutar `yarn build` para generar versión optimizada en carpeta build/




## Configuración de Variables

Establecer MONGO\_URL, DB\_NAME, REACT\_APP\_BACKEND\_URL y claves API opcionales



## Servidor Web

Configurar Nginx/Apache para servir frontend estático y proxy reverso al backend FastAPI

 **Tip:** Utilizar dominios separados para frontend (labcel.com) y backend (api.labcel.com) para mejor organización y escalabilidad. El backend se ejecuta con Uvicorn en el puerto 8001.