# MARCKET UCT

**Elías caranza Jaramillo, Diego Leiva Caballero, Nicolás Martínez Cáceres,
Daniel Prado Marambio, Rodrigo Pedraza Valencia,
Martin Sanhueza fernandez y Esban vejar**

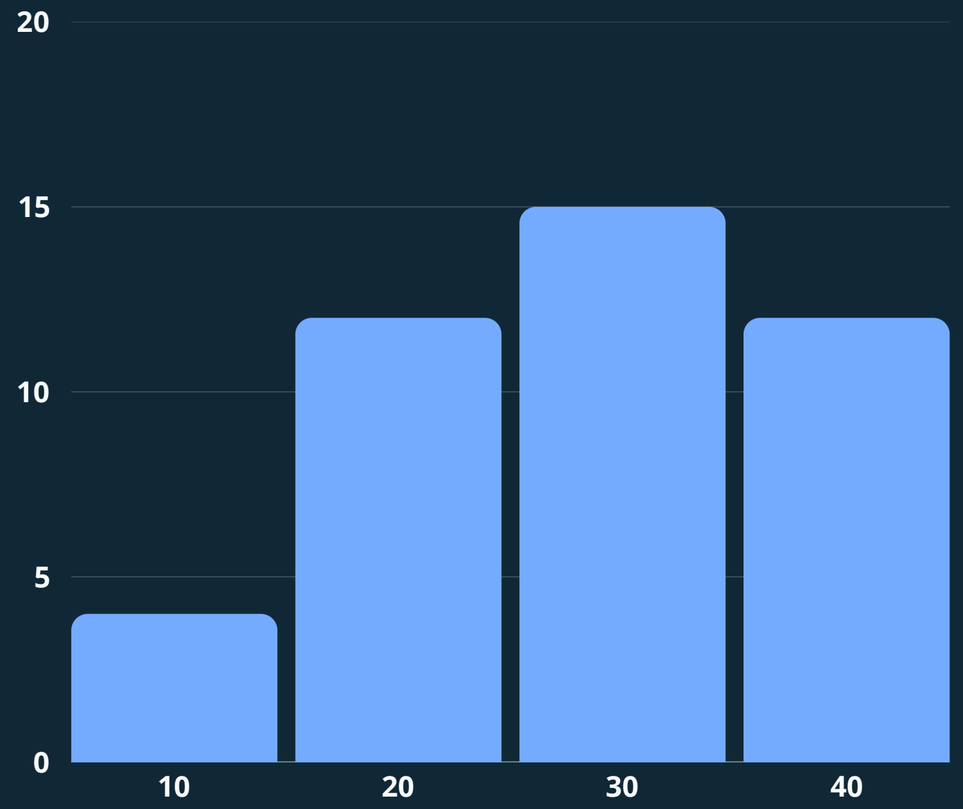
# Índice de CONTENIDOS

# contexto inicial

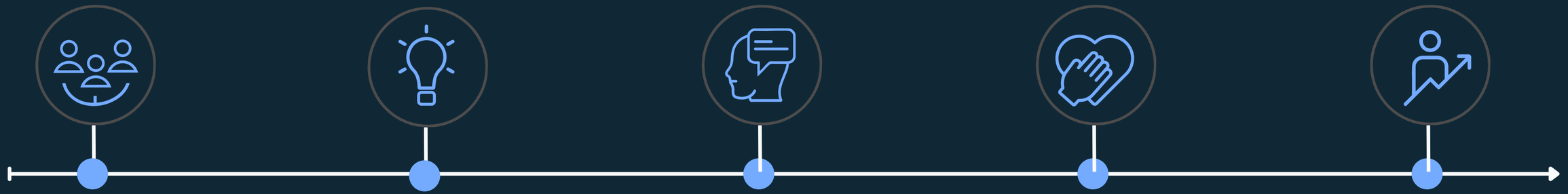
**Desarrollo de un micro-marketplace local exclusivo para la comunidad de la Universidad Católica de Temuco (UCT), diseñado para facilitar la compra, venta e intercambio de productos y servicios entre sus estudiantes y profesores.**

# objetivo general

Implementar y conectar las funcionalidades clave de experiencia de usuario y gestión (paneles de control, ubicaciones y notificaciones), integrando los hooks del frontend con los endpoints reales del backend para operar la plataforma con datos reales y eliminar los datos simulados.



# Objetivos especificos



Permitir que los usuarios definan su campus principal y lugares preferidos de entrega, visibles en sus publicaciones.

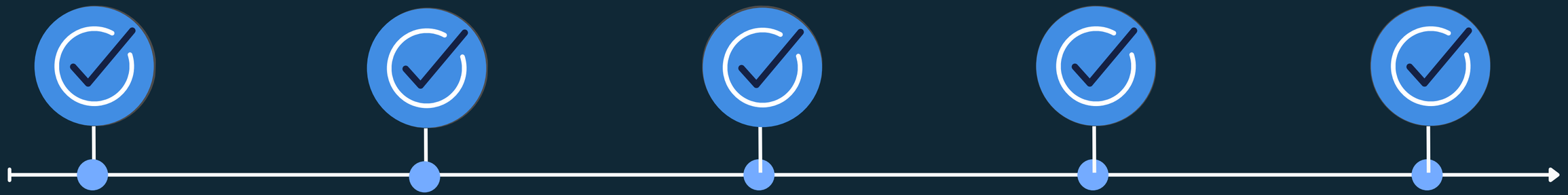
Crear dashboards para usuarios y administradores con métricas, publicaciones, transacciones y reputación.

Alertar a los usuarios sobre mensajes, actualizaciones y confirmaciones relevantes en tiempo real.

Aplicar principios de diseño limpio y arquitectura modular para una navegación más intuitiva

Asegurar la estabilidad, eficiencia y satisfacción del usuario antes del siguiente sprint.

# Objetivos cumplidos



Permitir que los usuarios definan su campus principal y lugares preferidos de entrega, visibles en sus publicaciones.

Crear dashboards para usuarios y administradores con métricas, publicaciones, transacciones y reputación.

Alertar a los usuarios sobre mensajes, actualizaciones y confirmaciones relevantes en tiempo real.

Aplicar principios de diseño limpio y arquitectura modular para una navegación más intuitiva

Asegurar la estabilidad, eficiencia y satisfacción del usuario antes del siguiente sprint.

# TECNOLOGIAS USADAS

### FRONTEND

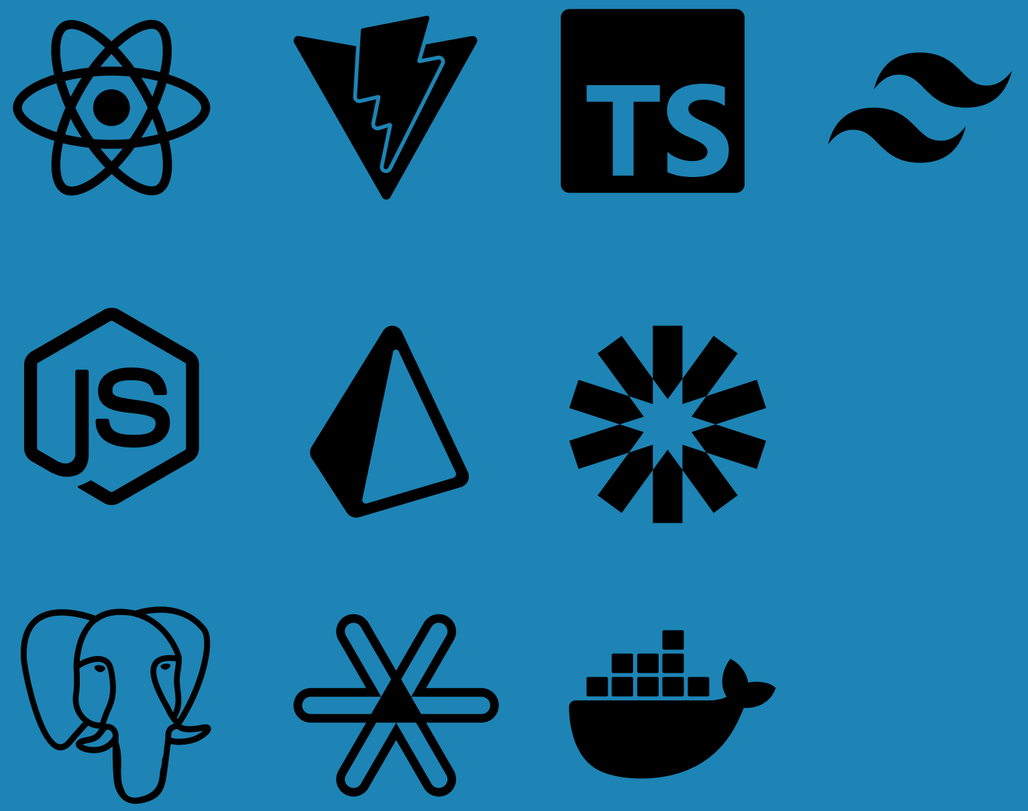
- React
- Vite
- TypeScript
- Tailwind CSS
- Zustand

### BACKEND

- Node.js (Express)
- Prisma
- JWT
- Bcrypt

### PLATAFORMA Y BASE DE DATOS

- PostgreSQL
- Google OAuth 2.0
- Docker



# TABLA DE TECNOLOGIAS USADAS

| Categoría | Tecnología Elegida (Nuestro Stack) | Alternativas Populares | Justificación de la Elección |
|---|---|---|---|
| Framework Frontend | React (con Vite) | Angular, Vue.js, Svelte | React tiene la comunidad más grande y un ecosistema maduro. Vite nos da un entorno de desarrollo ultra-rápido (HMR) comparado con el antiguo Webpack/CRA. |
| Base de Datos | PostgreSQL | MySQL, MongoDB (NoSQL) | PostgreSQL es más robusto para consultas complejas y escalabilidad que MySQL. Elegimos un modelo relacional (SQL) sobre NoSQL porque la estructura de un marketplace (Usuarios, Productos, Ventas, Mensajes) es inherentemente relacional y requiere integridad de datos |
| Backend | Node.js (Express) | Django (Python), Laravel (PHP) | Coherencia del Lenguaje: Usamos TypeScript en el frontend y en el backend. Esto nos permite reutilizar lógica y mantener un solo estándar de código en todo el proyecto. |
| Conexión a BD (ORM) | Prisma | Sequelize, TypeORM | Prisma es un ORM "Next-Gen" que garantiza la seguridad de tipos (Type-Safety) de principio a fin. Su schema.prisma es mucho más simple y menos propenso a errores que otros ORMs |
| Manejo de Estado (App) | Zustand | Redux, React Context | Zustand es una librería moderna, ligera y potente. Ofrece manejo de estado global con casi cero "boilerplate" (código de configuración), a diferencia de la complejidad de Redux. |

# Problemas

## PROBLEMA 01

**Falta de coordinación efectiva entre los integrantes del equipo.**

## PROBLEMA 02

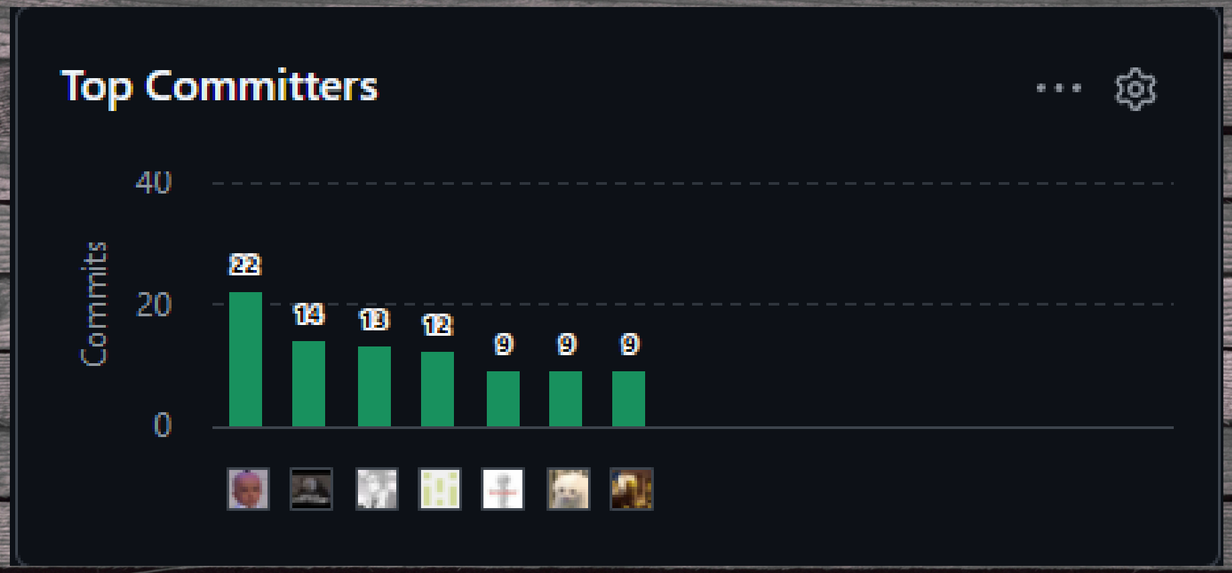
# Soluciones

## SOLUCION 01

**Implementar reuniones diarias de seguimiento para que cada miembro informe sobre sus avances, obstáculos y próximas tareas.**

## MÉTODO 02

# 06 BackLog




# CONSTRIBUCION DEL GRUPO

# 08

# Funcionamiento de la aplicacion

# Conclusiones

no se si vamos a decir esto

Muchas

# GRACIAS