1. Estructura técnica del sistema

a. Estructura de carpetas del proyecto

i. Frontend

```
frontend/              # Aplicación Ionic/Angular
├── src/
│   ├── app/
│   │   ├── components/ # Componentes reutilizables
│   │   │   ├── button/
│   │   │   ├── header/
│   │   │   └── simple-menu/
│   │   ├── pages/      # Vistas principales
│   │   │   ├── dashboard/
│   │   │   ├── gastos/
│   │   │   ├── home/
│   │   │   ├── informe/
│   │   │   ├── login/
│   │   │   └── register/
│   │   ├── guards/     # Protección de rutas
│   │   └── services/   # Lógica de conexión con backend
│   │       ├── auth.service.ts
│   │       ├── entrada.service.ts
│   │       ├── gastos.service.ts
│   │       ├── saldo-actualizador.service.ts
│   │       └── usuario.service.ts
│   └── assets/         # Imágenes/estilos
```

ii. Backend

```
├── backend/              # API Spring Boot
│   ├── src/main/java/com/econome/miapp/
│   │   ├── config/         # Configuraciones
│   │   │   └── CorsConfig.java
│   │   ├── controller/     # Endpoints API
│   │   │   ├── ABaseController.java
│   │   │   ├── EntradaController.java
│   │   │   ├── GastoController.java
│   │   │   └── UsuarioController.java
│   │   ├── dto/          # Objetos de transferencia
│   │   │   ├── ApiResponseDto.java
│   │   │   ├── UsuarioCreateUpdateDto.java
│   │   │   └── UsuarioResponseDto.java
│   │   ├── entity/       # Modelos de BD
│   │   │   ├── ABaseEntity.java
│   │   │   ├── Entrada.java
│   │   │   ├── Gasto.java
│   │   │   └── Usuario.java
│   │   ├── repository/    # Acceso a datos
│   │   │   ├── IBaseRepository.java
│   │   │   ├── EntradaRepository.java
│   │   │   ├── IGastoRepository.java
│   │   │   └── UsuarioRepository.java
│   │   ├── service/      # Lógica de negocio
│   │   │   ├── impl/     # Implementaciones
│   │   │   │   ├── EntradaServiceImpl.java
│   │   │   │   ├── GastoServiceImpl.java
│   │   │   │   └── UsuarioServiceImpl.java
│   │   │   ├── IBaseService.java
│   │   │   ├── EntradaService.java
│   │   │   ├── GastoService.java
│   │   │   └── UsuarioService.java
│   │   └── EconomicApplication.java  # Clase principal
│   └── src/main/resources/
│       ├── application.properties    # Config BD
│       └── data.sql              # Datos iniciales (opcional)
```

b. Comunicación entre componentes

i. Peticiones HTTP a endpoints

1. POST /api/auth/login (iniciar sesión).
2. GET /api/gastos (listar gastos).

ii. Backend → Base de Datos:

1. Spring Boot usa Hibernate para convertir objetos Java en tablas SQL.

2. Arquitectura del proyecto

a. Capa de presentación (Frontend)

- Tecnologías:
  - Ionic + Angular (TypeScript, HTML, CSS)
- Responsibilidades:
  - Interfaz de usuario (registro, login, formularios de transacciones).
  - Consumo de APIs del backend.

b. Capa de Lógica de negocio (Backend)

- Tecnologías:
  - Java + Spring Boot
  - Base de datos: MySQL
- Responsabilidades:
  - Manejar reglas de negocio
  - Procesar peticiones HTTP

c. Capa de datos (Persistencia)

- Tecnologías:
  - JPA/Hibernate

- Responsabilidades:
    - Guardar y recuperar datos.
    - Guardar y recuperar datos.