PRUEBA_TECNICA_CANDIDATOS

Prueba Técnica - Senior Mobile Developer

Posición: Senior Mobile Developer
Duración: 3 días (72 horas desde recepción)
Empresa: Teamcore

Bienvenida

Gracias por tu interés en unirte a nuestro equipo. Esta prueba técnica está diseñada para evaluar tus habilidades en desarrollo móvil y tu capacidad de adaptación a nuevas tecnologías.

Buscamos desarrolladores que puedan trabajar tanto en aplicaciones móviles como contribuir al backend de nuestros servicios, el cual está desarrollado en **Go**.

El Proyecto: TaskFlow

Construirás una aplicación de **gestión de tareas colaborativas** con su respectivo backend. La aplicación permite a usuarios crear, gestionar y dar seguimiento a sus tareas.

Parte 1: Backend (Go)

Desarrolla una API REST en **Go** que soporte la aplicación móvil.

Endpoints Requeridos

Autenticación

```
POST   /api/v1/auth/register    Registro (email, password, nam
e)
POST   /api/v1/auth/login       Login, retorna JWT
POST   /api/v1/auth/refresh     Refresh token
```

Tareas (autenticación requerida)

```
GET    /api/v1/tasks            Listar tareas (paginado)
POST   /api/v1/tasks            Crear tarea
```

```
GET    /api/v1/tasks/{id}       Obtener tarea
PUT    /api/v1/tasks/{id}       Actualizar tarea
DELETE /api/v1/tasks/{id}       Eliminar tarea
PATCH  /api/v1/tasks/{id}/status   Cambiar estado
POST   /api/v1/tasks/{id}/assign   Asignar a usuario
```

Modelo de Tarea

```
{
  "id": "uuid",
  "title": "string (max 100)",
  "description": "string (max 500)",
  "status": "pending | in_progress | completed | cancelled",
  "priority": "low | medium | high | urgent",
  "due_date": "2026-01-20T15:00:00Z",
  "created_by": "user_id",
  "assigned_to": "user_id o null",
  "created_at": "2026-01-15T10:00:00Z",
  "updated_at": "2026-01-15T10:00:00Z"
}
```

Especificaciones Técnicas

- Go 1.21 o superior
- Framework: Gin, Echo, Chi, o Fiber (a tu elección)
- Base de datos: PostgreSQL o SQLite
- Autenticación: JWT
- Incluir Dockerfile funcional

Opcional (valorado)

- Tests unitarios
- WebSocket para notificaciones
- Documentación Swagger/OpenAPI

Parte 2: Aplicación Móvil

Elige la plataforma de tu preferencia:

Opción	Tecnología
A	Android (Kotlin)
B	iOS (Swift/SwiftUI)
C	Flutter
D	Kotlin Multiplatform
E	React

Pantallas

1. Splash Screen

- Logo de la aplicación
- Verificar si existe sesión activa

2. Autenticación

- Login con email y contraseña
- Registro de nuevo usuario
- Validación de campos
- Mostrar errores de forma clara

3. Lista de Tareas (Home)

- Mostrar tareas del usuario
- Filtrar por estado (pending, in_progress, completed)
- Pull-to-refresh
- Indicador visual de prioridad
- Botón para crear nueva tarea

4. Detalle de Tarea

- Información completa de la tarea
- Editar título, descripción, prioridad
- Cambiar estado
- Seleccionar fecha límite
- Eliminar tarea (con confirmación)

5. Perfil

- Datos del usuario
- Estadísticas: tareas completadas, pendientes
- Cerrar sesión

Especificaciones Técnicas

- Arquitectura: MVVM, MVI, o Clean Architecture
- Inyección de dependencias
- Persistencia local para modo offline
- Manejo de estados: loading, error, éxito
- Funcionar sin conexión mostrando datos en cache

Opcional (valorado)

- Modo oscuro
- Animaciones y transiciones
- Tests unitarios o de UI
- Autenticación biométrica

Parte 3: Documentación

Tu repositorio debe incluir un **README.md** con:

1. **Descripción** breve del proyecto
2. **Cómo ejecutar** el backend y la app móvil
3. **Arquitectura** - Diagrama simple y explicación
4. **Decisiones técnicas** - Por qué elegiste cada tecnología/framework
5. **Mejoras futuras** - Qué harías con más tiempo

Entregables

Repositorio Git

Estructura esperada:

```
taskflow/
├── backend/           # Código Go
```

```
├── mobile/             # Código móvil
├── README.md           # Documentación
└── docker-compose.yml # (opcional)
```

Video Demo

- Máximo 5 minutos
- Mostrar el backend funcionando
- Demostrar todas las pantallas de la app
- Mostrar el modo offline

Aplicación

- APK (Android) o instrucciones de build (iOS/Flutter)

Recursos para Go

Si no tienes experiencia previa con Go, estos recursos te ayudarán:

Recurso	URL
Tour of Go	https://go.dev/tour/
Go by Example	https://gobyexample.com/
Gin Framework	https://gin-gonic.com/docs/
GORM (ORM)	https://gorm.io/docs/
JWT en Go	https://github.com/golang-jwt/jwt

Distribución Sugerida del Tiempo

Día	Enfoque
1	Configuración + Backend (auth y CRUD básico)
2	Completar backend + Estructura de app móvil + UI
3	Integración completa + Offline mode + Documentación

Reglas

1. **Originalidad**: El código debe ser tuyo. Puedes usar librerías y consultar documentación, pero no copiar soluciones completas.

2. **Herramientas de IA**: Puedes usar asistentes de código, pero debes entender y poder explicar todo el código que entregues.
3. **Comunicación**: Si tienes algún impedimento para entregar a tiempo, comunícalo antes de que venza el plazo.
4. **Dudas**: Puedes escribir a [EMAIL_CONTACTO] para dudas sobre los requisitos. No responderemos preguntas sobre implementación.

Entrega

- **Fecha límite:** [FECHA_LIMITE]
- **Enviar a:** [EMAIL_ENTREGA]
- **Asunto:** Prueba Técnica - [TU_NOMBRE]
- **Contenido:** Link al repositorio + Link al video demo

Evaluación

Valoramos:

- **Funcionalidad**: Que la aplicación funcione end-to-end
- **Calidad de código**: Limpio, organizado, idiomático
- **Arquitectura**: Separación de responsabilidades clara
- **UX**: Interfaz usable y manejo de errores amigable
- **Adaptabilidad**: Tu capacidad de aprender Go

No esperamos perfección. Preferimos una solución completa y funcional sobre una parcialmente perfecta.

¿Preguntas? Escríbenos a [EMAIL_CONTACTO]

¡Éxito!