Microfrontends con Angular - Ejemplo de la vida real

■ Escenario

Imagina que trabajas en un portal bancario con varios módulos:
- Módulo de cuentas: consulta de saldos, movimientos, transferencias.
- Módulo de inversiones: compra de acciones, fondos, reportes.
- Módulo de préstamos: solicitudes, pagos, historial.
Cada módulo es desarrollado por equipos distintos y debe verse como una sola app.

■ Arquitectura con Microfrontends

1. Shell App (Angular principal): Layout, navegación, autenticación.
2. Microfrontend: Cuentas (Angular) -> dominio: https://cuentas.banco.com/
3. Microfrontend: Inversiones (Angular) -> dominio: https://inversiones.banco.com/
4. Microfrontend: Préstamos (Angular) -> dominio: https://prestamos.banco.com/

■ Configuración básica con Module Federation

```
// webpack.config.js
new ModuleFederationPlugin({
  remotes: {
    cuentas: 'cuentas@https://cuentas.banco.com/remoteEntry.js',
    inversiones: 'inversiones@https://inversiones.banco.com/remoteEntry.js',
    prestamos: 'prestamos@https://prestamos.banco.com/remoteEntry.js'
  }
})


// app-routing.module.ts
const routes: Routes = [
  { path: 'cuentas', loadChildren: () => import('cuentas/CuentasModule').then(m => m.CuentasModule) },
  { path: 'inversiones', loadChildren: () => import('inversiones/InversionesModule').then(m => m.Inversi
  { path: 'prestamos', loadChildren: () => import('prestamos/PrestamosModule').then(m => m.PrestamosModu
];
```

■ Ventajas reales

- Equipos despliegan módulos sin esperar al resto.
- Un fallo en un módulo no tumba toda la app.
- Escala mejor en equipos grandes.
- Incluso pueden usar versiones distintas de Angular.