

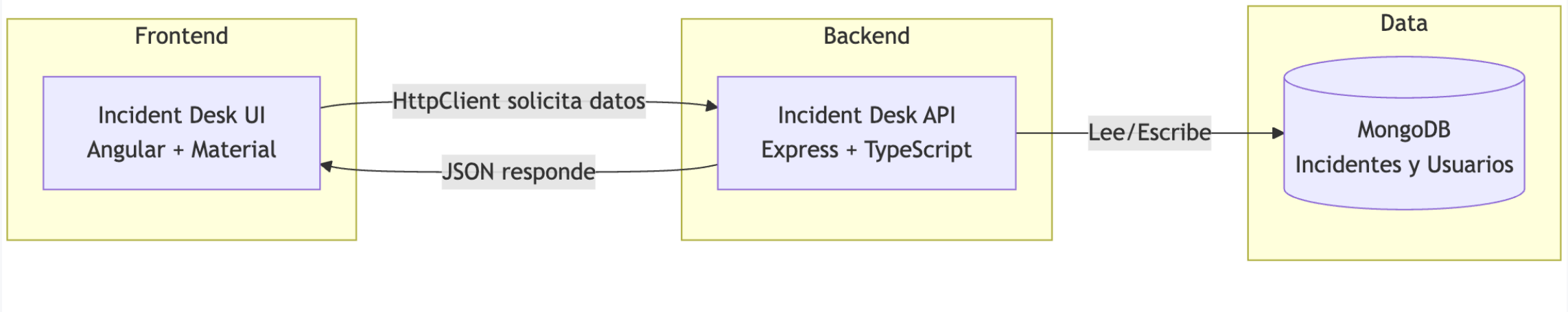
Clase 3

Gestión de autenticación con JWT y Angular

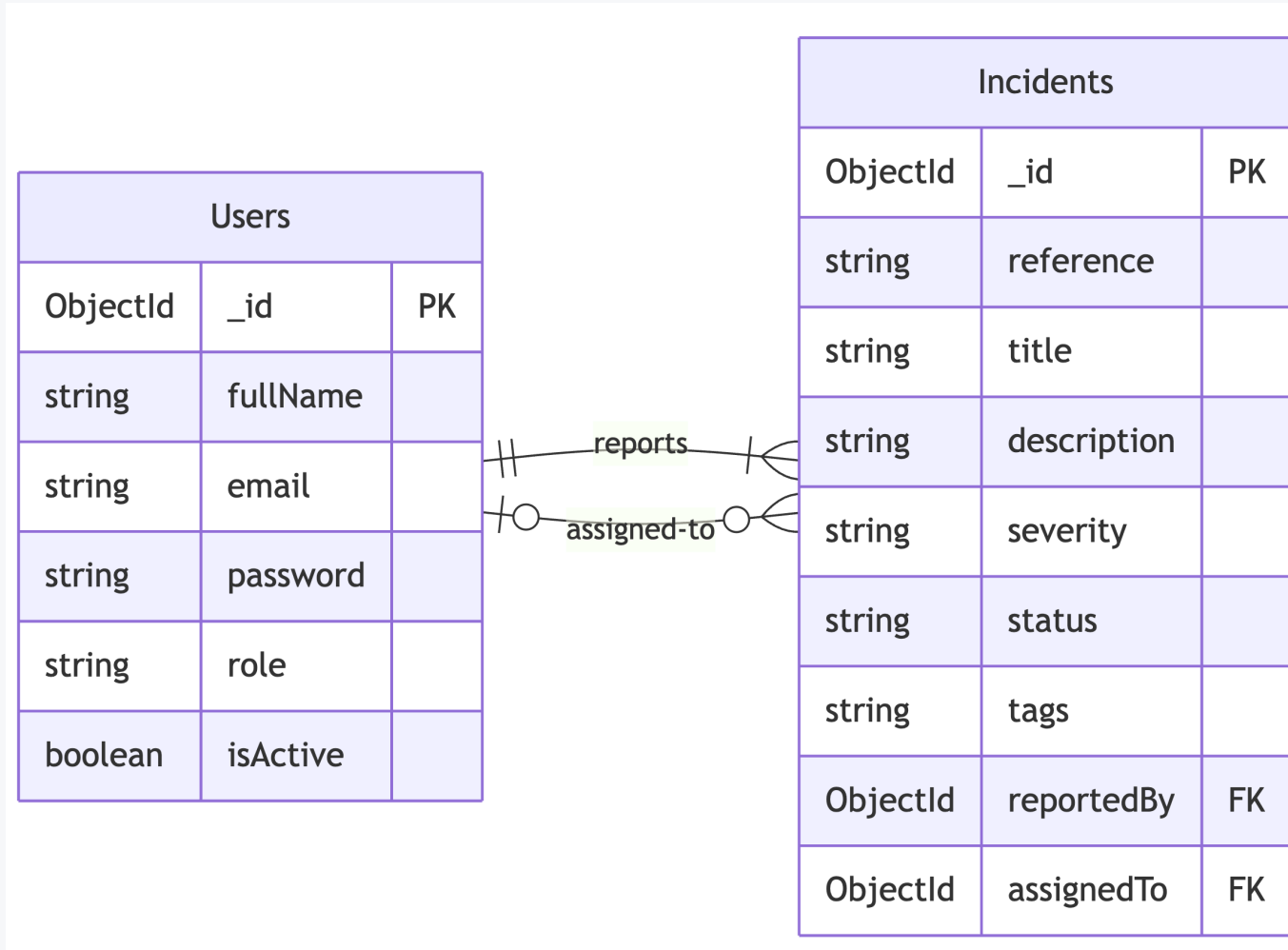
¿Qué haremos hoy?

- Entender qué es un token JWT.
- Proteger la API para que sólo procese peticiones autorizadas.
- Guardar la sesión en Angular con un `AuthService`.
- Dejar que un interceptor agregue el token por nosotros.
- Construir un `LoginForm` standalone.
- Controlar la navegación con `AuthLayout` + guards.





Arquitectura de la aplicación



Modelo de entidades



Token en pocas palabras (JWT)

-  **JWT** = un JSON firmado que dice quién eres.
-  La API lo firma con un secreto (`JWT_ACCESS_SECRET`).
-  Tiene fecha de caducidad (`15m` por defecto).
-  Si el token es falso o vencido → la petición se rechaza.

```
Header(xxxxx).Payload(yyyyy).Signature(zzzzz)  
xxxxx.yyyyy.zzzzz
```

Partes de un JWT

- **Header (xxxxx)**

- Algoritmo (`alg`, ejemplo: HS256)
- Tipo (`typ`, siempre `JWT`)





- **Payload (yyyyy)**

- Datos firmados: `sub`, `email`, `role`, `exp`, etc.
- No debe incluir información sensible (passwords)

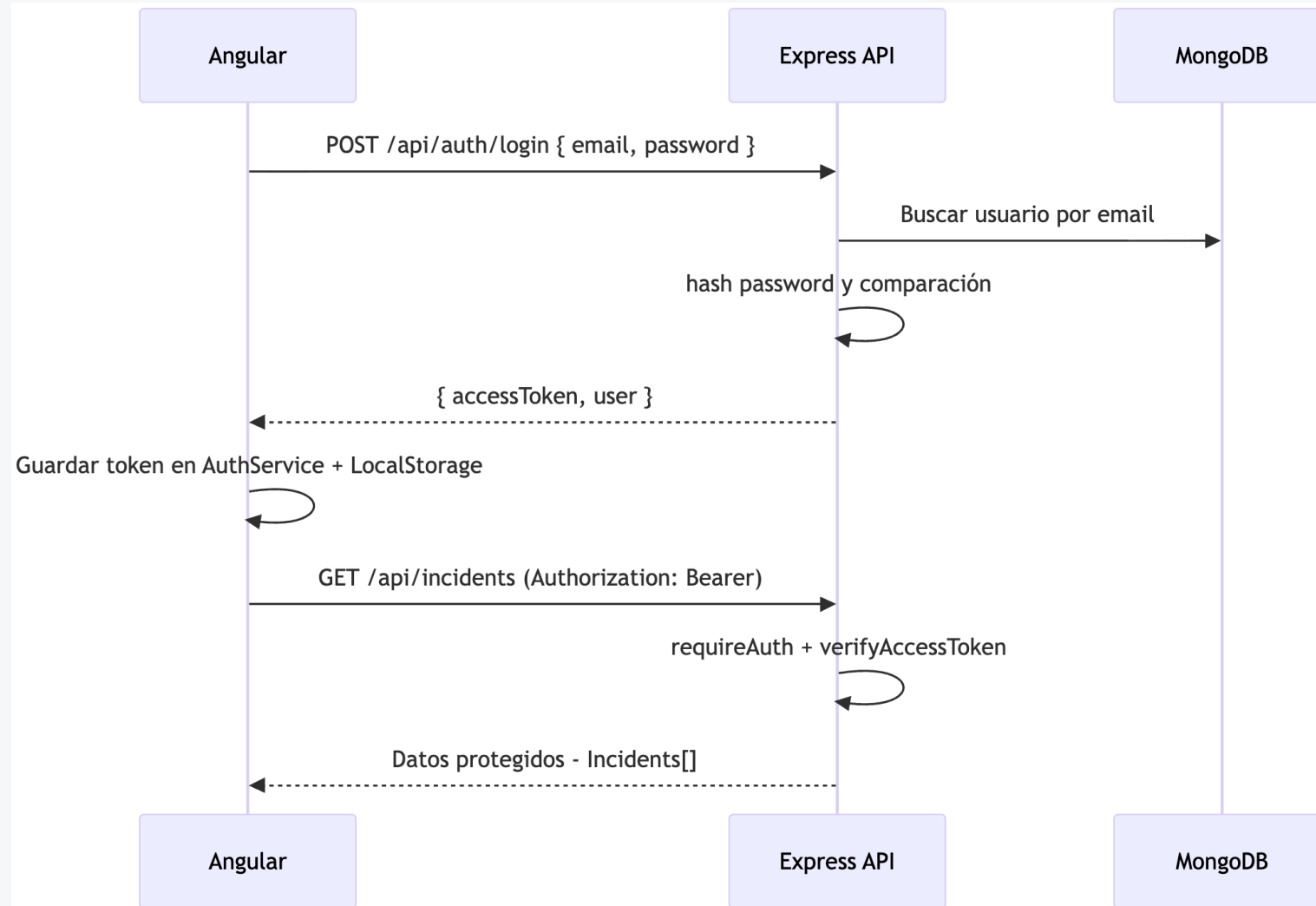
- **Signature (zzzzz)**

- `HMACSHA256(base64(header) + "." + base64(payload), JWT_ACCESS_SECRET)`
- Cambia si alguien modifica el header o payload

Viaje del token

1.  Front envía email + password a `/api/auth/login`.
2.  API busca el usuario y compara el password con `bcrypt`.
3.  API crea el token (`accessToken`) y lo devuelve con el usuario.
4.  Front guarda `{ accessToken, user }` en `AuthService` + `localStorage`.

Flujo completo



Flujo Angular

