# IONIC

Tema 3 – Base de datos

# IONIC

- Instalación (Ionic con Angular):
  - npm install @ionic/storage-angular
  - npm install @ionic/storage

# IONIC

- Configuración app.module.ts: (con módulos)

```
import { IonicStorageModule } from '@ionic/storage-angular';

@NgModule({
  declarations: [AppComponent],
  imports: [BrowserModule, IonicModule.forRoot(), AppRoutingModule,
HttpClientModule, IonicStorageModule.forRoot() ],
  providers: [{ provide: RouteReuseStrategy, useClass:
IonicRouteStrategy }, HttpdataService ],
  bootstrap: [AppComponent],
})
```

# IONIC

- Configuración main.ts: (con componentes standalone)

```
import { IonicStorageModule } from '@ionic/storage-angular’;
import { importProvidersFrom } from ‘@angular/core’;

bootstrapApplication(AppComponent, {
providers: [
    { provide…},
    provideIonicAngular(),
    provideRouter(…),
    importProvidersFrom(IonicStorageModule.forRoot()),
],
});
```

# IONIC

- Servicio:

```
import { Injectable } from '@angular/core';
import { Pelicula } from '../interfaces/pelicula';
import { Storage } from '@ionic/storage-angular';
import { Observable } from 'rxjs';

const NODO_RAIZ = "peliculas";

@Injectable({
  providedIn: 'root'
})
export class LocaldatabaseService {

  constructor(private storage: Storage) {
    this.init();
  }

  async init() {
    this.storage = await this.storage.create();
  }
```

# IONIC

- Servicio:

```
guardarPelicula(pelicula: Pelicula) {
    this.storage.get(NODO_RAIZ).
      then((data) => {
        if (data == null) {
          let peliculas = new Array();
          peliculas.push(pelicula);
          this.storage.set(NODO_RAIZ, peliculas);
        } else {
          let peliculas = data;
          peliculas.push(pelicula);
          this.storage.set(NODO_RAIZ, peliculas);
        }
      }).
      catch((error) => {
        console.error("Error:" + error);
      }).
      finally(() => {
        console.log("Fin del proceso de almacenamiento");
      });
  }
```

# IONIC

- Servicio:

```
  getPelicula(titulo: string) : Observable<Pelicula>{
    return new Observable(subscriber=>{
      this.storage.get(NODO_RAIZ).then((peliculas) => {
        let pelicula = peliculas.find((p: Pelicula) => p.Title ==
titulo);
        subscriber.next(pelicula);
        subscriber.complete();
      })});
  }

  getAllPeliculas() {
    return this.storage.get(NODO_RAIZ);
  }
```

# IONIC

- Componente:

```
  guardarPelicula() {
    this.ldbs.guardarPelicula(this.pelicula);
  }

  buscarPeliculaExistente() {
    this.ldbs.getPelicula(this.titulo).subscrib
e(pelicula => {
      this.pelicula = pelicula;
    })
  }
```