database

index.ts

```ts
export * as db from './db';

export * from './seed-data';

export * as dbEntries  from './dbEntries';
```

db.ts

```ts
import { log } from "console";
import mongoose from "mongoose";

//  0 = disconnected
//  1 = connected
//  2 = connecting
//  3 = disconnecting

const mongoConnection = {
    isConnected: 0
}

export const connect = async() => {

    if (mongoConnection.isConnected) {
        console.log("Ya estábamos conectados");
        return;
    }

    if (mongoose.connections.length > 0) {
        mongoConnection.isConnected = mongoose.connections[0].readyState;

        if (mongoConnection.isConnected === 1) {
            console.log('Usando conexión anterior');
            return;
        }

        await mongoose.disconnect();
    }

    await mongoose.connect(process.env.MONGO_URL || '');
    mongoConnection.isConnected = 1;
    console.log('conectado a MongoDB: ', process.env.MONGO_URL);

}

export const disconnect = async() => {
    if ( process.env.NODE_ENV === 'development'   ) return;

    if (mongoConnection.isConnected === 0   ) return;

    await mongoose.disconnect();

    mongoConnection.isConnected = 0;
```

```
    console.log('Desconectado de MonoDB ');

}
```

dbEntries.ts

```
import { isValidObjectId } from "mongoose"
import { db } from ".";
import { Entry, IEntry } from "../models";

 export const getEntryById = async (id: string): Promise<IEntry | null> => {

    if (!isValidObjectId(id)) return null;

    await db.connect();

    const entry = await Entry.findById(id).lean();
    await db.disconnect();

    return JSON.parse( JSON.stringify(entry));


 }
```

seed-data.ts

```
interface SeedData {
    entries: SeedEntry[];
}

interface SeedEntry {
    description: string;
    status: string;
    createdAt: number;
}

export const seedData: SeedData = {
     entries: [
        {
            description: 'Pendiente: Primera línea de prueba',
            status: 'pending',
            createdAt: Date.now(),
        },
        {
            description: 'En-Progreso: Segunda línea de prueba',
            status: 'in-progress',
            createdAt: Date.now() - 1000000,
        },
        {
            description: 'Terminadas: Tercera línea de prueba',
            status: 'finished',
            createdAt: Date.now() - 100000,
        },
     ]
}
```