Library Management System

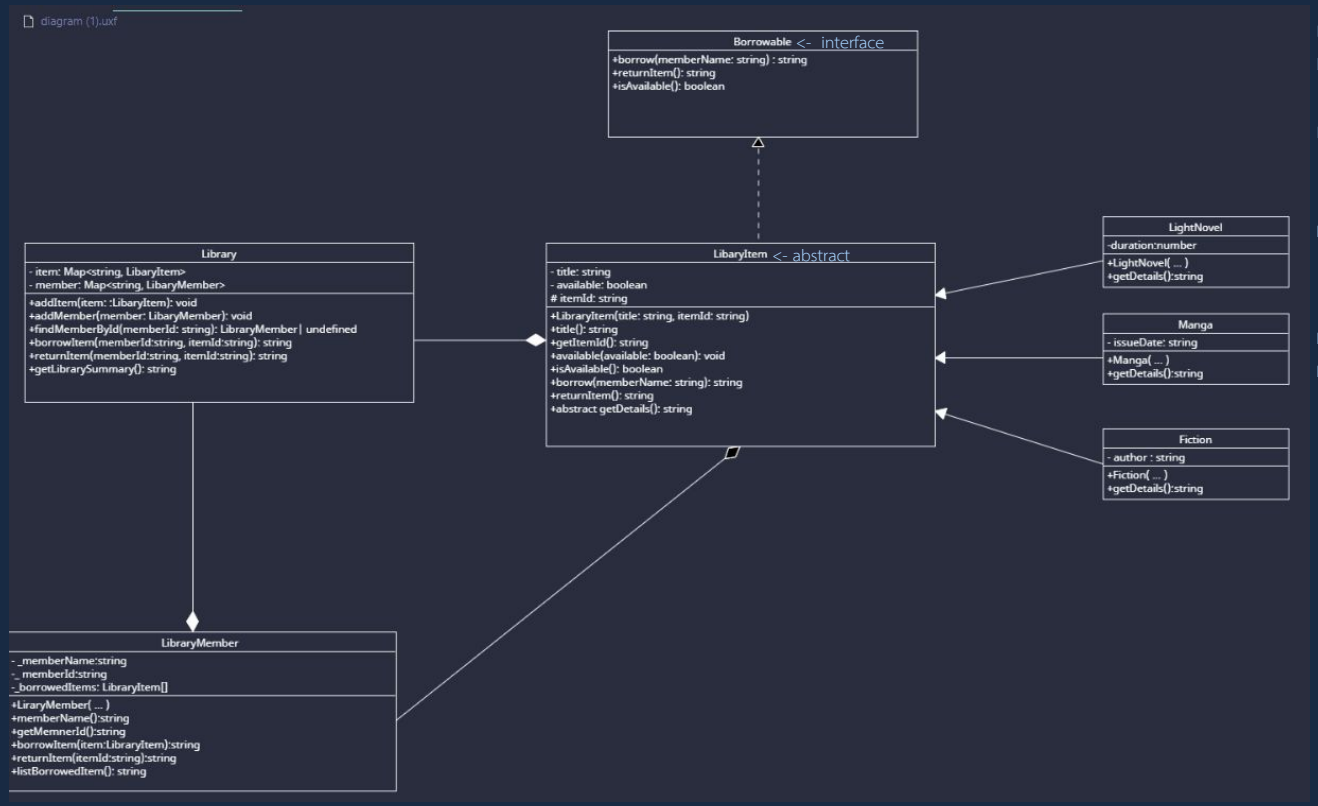
Feature

Borrowable : เป็นฟังก์ชันที่จะมีข้อความแสดงหนังสือที่เรายืมไปว่าชื่ออะไร ยืมโดยใครและแสดงประเภทของหนังสือ ชื่อผู้แต่ง และ ไอดีของหนังสือ โดยการกรอกข้อมูลในโค้ดถ้าไม่พบ

Returnitem : จะแสดงข้อมูลว่าหนังสือที่เรายืมไปนั้น ชื่อว่าอะไรและจะแสดงข้อความบอกอีกว่ายังมีหนังสือที่ต้องยืมอีกไหม

isAvailable : จะเป็นฟังก์ชันคอยตรวจสอบว่าหนังสือที่จะยืมนั้น สามารถยืมได้หรือไม่

Class Diagram



Borrowable คือ interface ที่มี attribute เป็น borrow,returnitem,isAvailable

LibraryItem เป็น abstract class Implement มาจาก Borrowable

LightNovel, Manga, Fiction สืบทอดมาจาก LibraryItem

Library มี relationship เป็น Aggregation กับ Library และ LibraryMember

โค้ดนี้มีการใช้ **OOP Concepts** อะไรบ้าง 👇

```
class LightNovel extends LibraryItem {
    private author: string;

    constructor(title: string, itemId: string, author: string) {
        super(title, itemId);
        this.author = author;
    }

    public getDetails(): string {
        return `LightNovel: ${this.title()} by ${this.author} (ID: ${this.getItemId()})`;
    }
}
```

```
interface Borrowable {  Show usages
    borrow(memberName: string): string;
    returnItem(): string;
    isAvailable(): boolean;
}
abstract class LibraryItem implements Borrowable {...}
```

1.Inheritance (การสืบทอดคลาส)

LightNovel, Manga, Fiction ที่เป็น subclass ⟶ extends จาก LibraryItem ทำให้ subclass ทั้งหมดสามารถใช้ method ของ LibraryItem ได้ทันที (borrow, returnItem, isAvailable) และยัง override method getDetails() ได้ตามความต้องการ

2. Interface (การกำหนดสัญญา)

คือการที่กำหนด interface ต้องการคุณสมบัติการ ยืม,คืน,ตรวจเช็คสถานะ

โค้ดนี้มีการใช้ **OOP Concepts** อะไรบ้าง 👇(ต่อ)

```
class LibraryMember {  Show usages
    private _memberName: string;  Field can be readonly.
    private _memberId: string;  Field can be readonly.
    private _borrowedItems: LibraryItem[] = [];

    constructor(memberName: string, memberId: string) {  Show usages
        this._memberName = memberName;
        this._memberId = memberId;
    }
```

3.Encapsulation (การห่อหุ้มข้อมูล)

ใช้ private และ protected เพื่อซ่อนข้อมูลภายใน _title, _available ใน LibraryItem ⟶ ไม่สามารถแก้ไขจากภายนอกโดยตรง

LibraryMember ซ่อน _borrowedItems เอาไว้ (เข้าถึงได้แค่ผ่าน method เช่น listBorrowedItems()) ช่วยป้องกันการเข้าถึงข้อมูลโดยไม่ตั้งใจ

```
class Library {  Show usages
    private items: Map<string, LibraryItem> = new Map();
    private members: Map<string, LibraryMember> = new Map();

    public addItem(item: LibraryItem): void {  Show usages
        this.items.set(item.getItemId(), item);
    }
```

4. Aggregation (การรวมกันแบบอิสระ)

ใช้ private และ protected เพื่อซ่อนข้อมูลภายใน _title, _available ใน LibraryItem ⟶ ไม่สามารถแก้ไขจากภายนอกโดยตรง

LibraryMember ซ่อน _borrowedItems เอาไว้ (เข้าถึงได้แค่ผ่าน method เช่น listBorrowedItems()) ช่วยป้องกันการเข้าถึงข้อมูลโดยไม่ตั้งใจ

จุดที่ทำให้ระบบขยายได้ 🌿

Interface (Borrowable)

ตอนนี้ทุก LibraryItem ที่ยืมได้ต้อง implement interface Borrowable ในอนาคตสามารถเพิ่ม DVD, E-Book, หรือ Game ได้ง่าย ๆ โดยแค่ทำ class DVD extends LibraryItem { ... } โดยไม่ต้องแก้โค้ดเก่า

Interface (การกำหนดสัญญา)

interface Borrowable กำหนด method ที่ต้องมี (borrow, returnItem, isAvailable)

abstract class LibraryItem ⟶ implements Borrowable

ทำให้มั่นใจว่า ทุก item ที่สืบทอดมาจาก LibraryItem จะมีคุณสมบัติที่สามารถยืม-คืนและตรวจสอบได้ว่าหนังสือเล่มนั้นสามารถยืมได้หรือไม่ โดยไม่ต้องแก้โค้ดเก่า

Encapsulation (การห่อหุ้มข้อมูล)

ใช้ private และ protected เพื่อซ่อนข้อมูลภายใน ตัวอย่าง:_title, _available ใน LibraryItem ⟶ ไม่สามารถแก้ไขจากภายนอกโดยตรง

LibraryMember ซ่อน _borrowedItems เอาไว้ (เข้าถึงได้แค่ผ่าน method เช่น listBorrowedItems())

ช่วยป้องกันการเผลอแก้ไขข้อมูลโดยไม่ตั้งใจทำให้แก้ไขและขยายได้ง่าย

Logic และ Error Handling⚠️

```
public borrow(memberName: string): string {
    if (!this._available) {
        return `❌ Item "${this._title}" is already borrowed.`;
    }
    this._available = false;
    return `✅ Item "${this._title}" borrowed by ${memberName}`;
}
```

borrow()

ถ้า _available = false ⟶ คืนข้อความ error ว่า item ถูกยืมแล้ว

ถ้า _available = true ⟶ เปลี่ยน _available = false และคืนข้อความยืม

Thank you (*^-^*)