```
                                    <<enumera
                                       PersonT

                             Author

                             BookAuthor

                             Contributor

                             Editor

                             SeriesEditor

                             Translator



                                    Perso

                 + personType: Person

                 + firstName: String { r

                 + lastName: String { r


                 + Person(
                     personType: Per
                     firstName: String
                     lastName: String
                   )
```
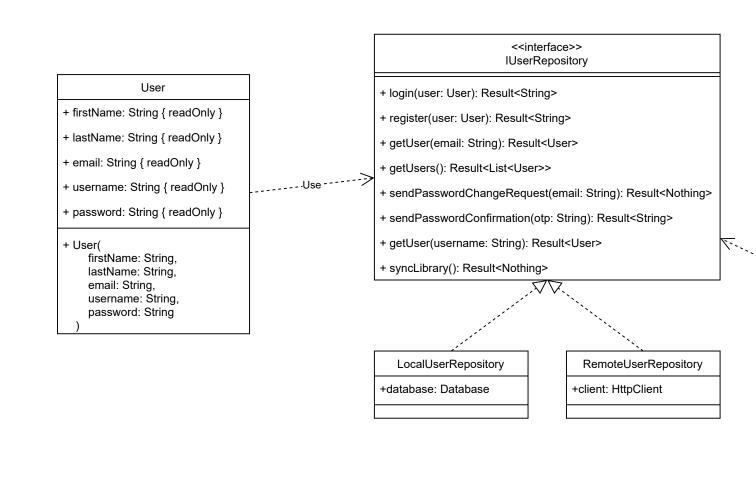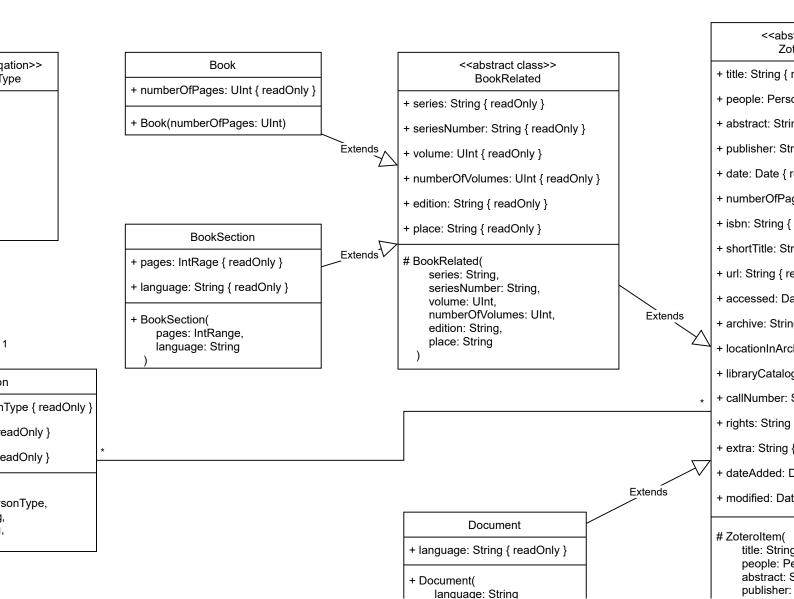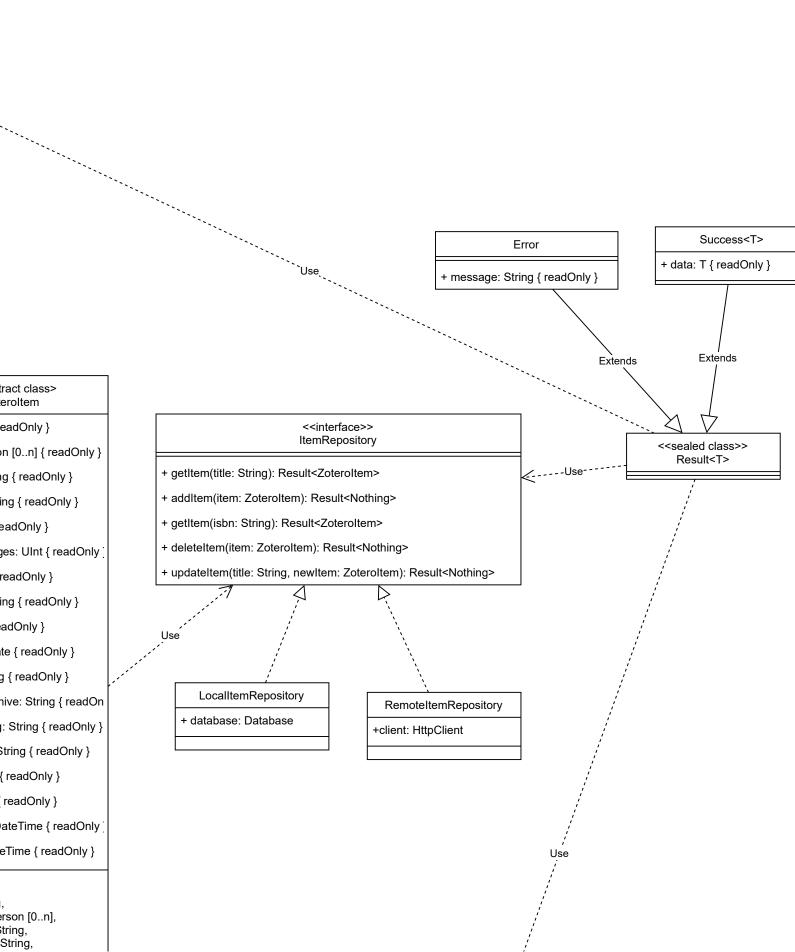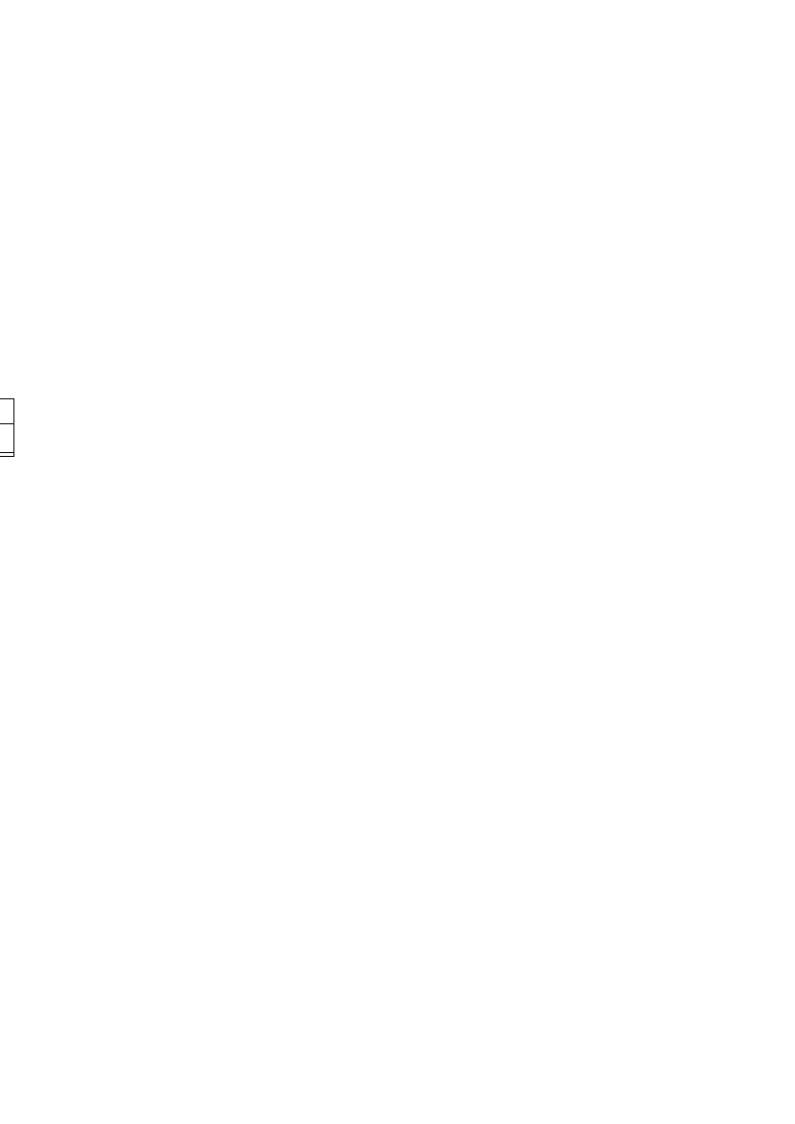
User

+ firstName: String { readOnly }

+ lastName: String { readOnly }

+ email: String { readOnly }

+ username: String { readOnly }

+ password: String { readOnly }

+ User(
    firstName: String,
    lastName: String,
    email: String,
    username: String,
    password: String
)

-- Use -->

<<interface>>
IUserRepository

+ login(user: User): Result<String>

+ register(user: User): Result<String>

+ getUser(email: String): Result<User>

+ getUsers(): Result<List<User>>

+ sendPasswordChangeRequest(email: String): Result<Nothing>

+ sendPasswordConfirmation(otp: String): Result<String>

+ getUser(username: String): Result<User>

+ syncLibrary(): Result<Nothing>

LocalUserRepository

+database: Database

RemoteUserRepository

+client: HttpClient

<<...gation>>
...Type

Book

+ numberOfPages: UInt { readOnly }

+ Book(numberOfPages: UInt)

<>
BookRelated

+ series: String { readOnly }

+ seriesNumber: String { readOnly }

+ volume: UInt { readOnly }

+ numberOfVolumes: UInt { readOnly }

+ edition: String { readOnly }

+ place: String { readOnly }

BookRelated(
    series: String,
    seriesNumber: String,
    volume: UInt,
    numberOfVolumes: UInt,
    edition: String,
    place: String
)

Extends

BookSection

+ pages: IntRage { readOnly }

+ language: String { readOnly }

+ BookSection(
    pages: IntRange,
    language: String
)

Extends

<<abst...
Zot...

+ title: String { r...

+ people: Pers...

+ abstract: Strin...

+ publisher: Stri...

+ date: Date { re...

+ numberOfPag...

+ isbn: String {...

+ shortTitle: Str...

+ url: String { re...

+ accessed: Da...

+ archive: String...

+ locationInArch...

+ libraryCatalog...

+ callNumber: S...

+ rights: String...

+ extra: String {...

+ dateAdded: D...

+ modified: Dat...

ZoteroItem(
    title: String...
    people: Pe...
    abstract: S...
    publisher:...

...n

...nType { readOnly }

...readOnly }

...eadOnly }

...rsonType,
...,
...,

Document

+ language: String { readOnly }

+ Document(
    language: String

Extends

1

*

*

Error

+ message: String { readOnly }

Success<T>

+ data: T { readOnly }

Extends

Extends

<<interface>>
ItemRepository

+ getItem(title: String): Result<ZoteroItem>

+ addItem(item: ZoteroItem): Result<Nothing>

+ getItem(isbn: String): Result<ZoteroItem>

+ deleteItem(item: ZoteroItem): Result<Nothing>

+ updateItem(title: String, newItem: ZoteroItem): Result<Nothing>

<<sealed class>>
Result<T>

Use

Use

Use

Use

<tract class>
eroItem

eadOnly }

on [0..n] { readOnly }

g { readOnly }

ing { readOnly }

eadOnly }

ges: UInt { readOnly }

eadOnly }

ing { readOnly }

eadOnly }

te { readOnly }

g { readOnly }

hive: String { readOn

y: String { readOnly }

String { readOnly }

{ readOnly }

readOnly }

ateTime { readOnly }

eTime { readOnly }

LocalItemRepository

+ database: Database

RemoteItemRepository

+client: HttpClient

,
erson [0..n],
tring,
String,

date: Date
numberOf
isbn: String
shortTitle:
url: String.
accessed:
archive: St
locationInA
libraryCata
callNumbe
rights: Strir
extra: Strir
dateAddec
modifier: D
)

*

ZoteroColle

+ title: String { readOnly

+ items: Item [1..n] { rea

+ ZoteroCollection(
    title: String,
    items: Item[1..n]
)

Pages: UInt,
g,
String,

Date,
tring,
Archive: String,
alog: String,
r: String,
ng,
ng,
d: DateTime,
DateTime

*

ection

y }

adOnly }

- - - - - Use - - - ->

LocalCollectionRepository

+ database: Database

RemoteCollectionRepository

+client: HttpClient

<<interface>>
ICollectionRepository

+ getCollection(title: String): Result<ZoteroCollection>

+ addCollection(item: ZoteroCollection): Result<Nothing>

+ deleteCollection(item: ZoteroCollection): Result<Nothing>

+ updateCollection(title: String, newCollection: ZoteroCollection): Result<Nothing>