

WORK TO DO…..

Book[1:1]     M:1            [M:1]Publisher: Inject Publisher in Book class: @ManyToOne  
Reader[1:M]    M:M            [M:1]Book :

Since we want to add many columns in relationship class, we will create a separate entity for this.   
            ReaderBook : Inject Reader and Book in ReaderBook class and extra properties

Author[1:M]    M:M            [1:M]Book  
- Since we do not want any extra columns in relationship table, we can go for @ManyToMany

API:  
----  
Publisher: POST, GETALL, GetById, PUT, DELETE  
Author: POST, GETALL, GetById, PUT, DELETE  
Book: POST, GETALL, GetById, PUT, DELETE  
ReaderBook: POST, GETALL, GetById, PUT, DELETE

Business API  
============  
GetBooksByPublisherId  
getBooksByAuthor  
getBooksByReaderID  
getReadersByBookId  
getReadersByAuthorId  
getTotalRentByBookId  
getTOtalRentByReaderId  
getTotalRentByPublisherId  
getTotalRentByAuthorId

**Entity to made…..>>>**

Publisher: Id , name, bookid(@ManyToOne)   
Author: id, name, bookid, book reference(@ManyToMany)  
Book: id, name, price ,publisher reference (@ManyToOne)

Reader: id, name  
ReaderBook: id, reference\_of\_book (@ManyToOne), reference\_of\_reader (@ManyToOne)

**Controler**

Bookcontroler

publisherControler

AutherControler

ReaderControler

ReaderBookControler

**Service**

BookService

PublisherService

AutherService

ReaderService

ReqaderBookService

**WORK DISTRIBUTION**

**DILIP AND AMBUJ>>>>>>>>**

Publisher: POST, GETALL, GetById, PUT, DELETE

getTOtalRentByReaderId  
getTotalRentByPublisherId  
getTotalRentByAuthorId

**SAI AND BISWAJIT>>>>>>>>>**

Book: POST, GETALL, GetById, PUT, DELETE  
ReaderBook: POST, GETALL, GetById, PUT, DELETE

Reader:POST,GETALL,GetById,PUT,DELETE

getReadersByBookId  
getReadersByAuthorId  
getTotalRentByBookId

**PRANIL AND VISWANATH>>>>>>>>>**

Author: POST, GETALL, GetById, PUT, DELETE

GetBooksByPublisherId  
getBooksByAuthor  
getBooksByReaderID