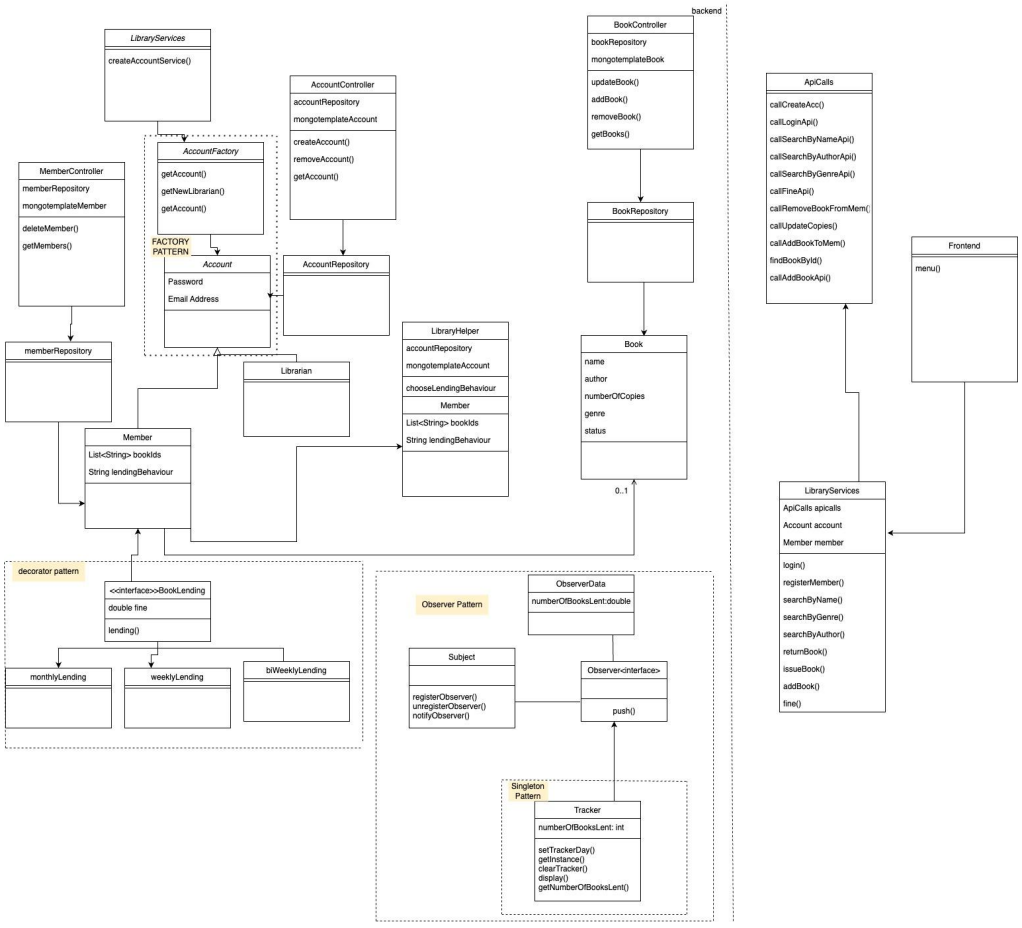
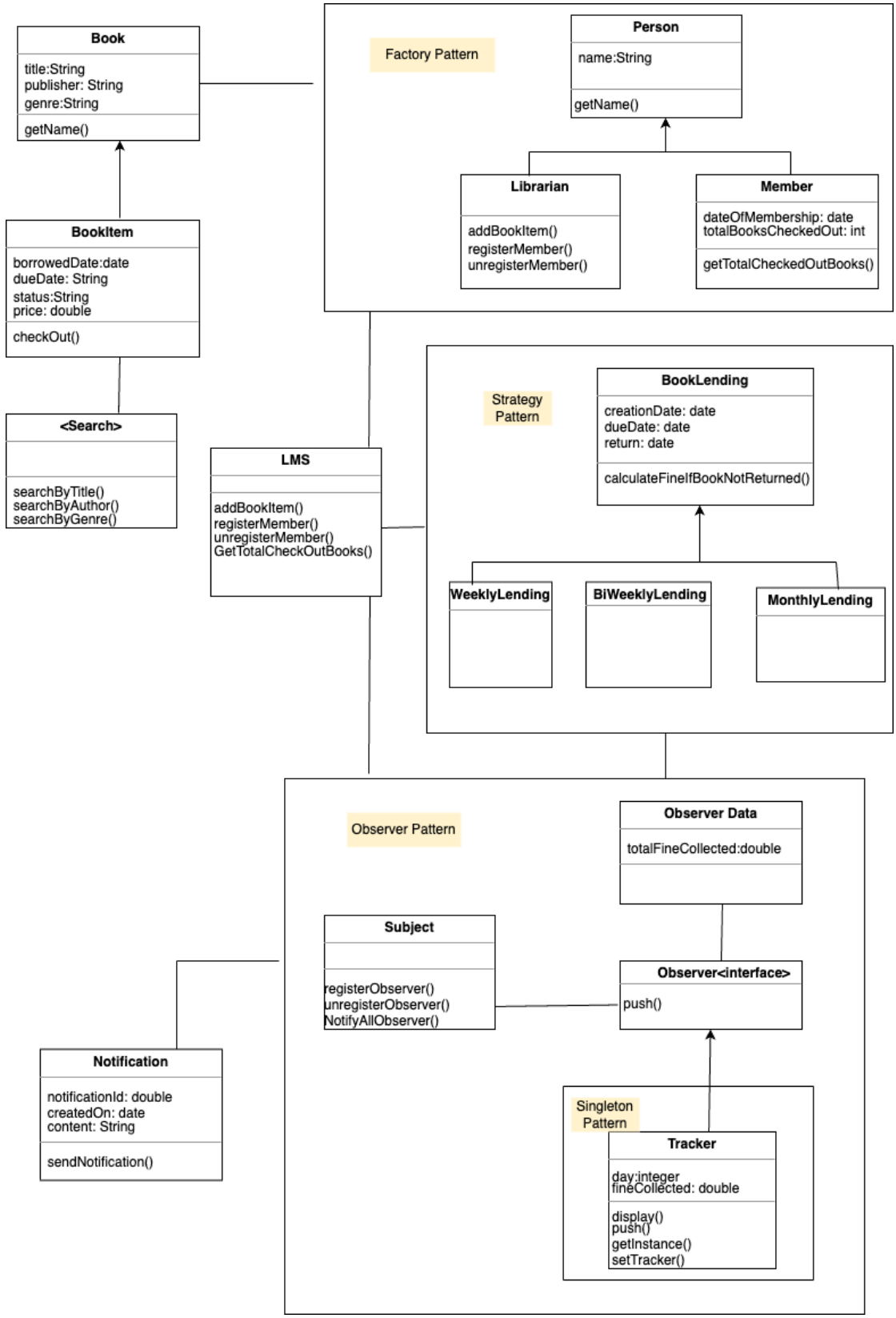
Project 7

1. **Title:-** Library Management System
   Members: Harsha Kalmath , Shashankit Thakur
2. **Final state of the system :-** We implemented the following functionalities for Library Management System:-
   - Login and register new member
   - For librarian , we implemented the add book to database functionality
   - For members , we added search books by name , genre, author , issue a book , return a book and calculated the fine according to lending behavior of the member.
   - We implemented observer and singleton pattern for our backend code files.
   - We implemented factory pattern for creating account of type library or member
   - And we implemented the decorator pattern for choosing lending behavior of a member at run time when the member first registers. The fine is calculated dynamically when the member returns a book.
3. Final UML Class diagram :
   Project 7 Class Diagram



Project 5 Class Diagram:-



What changed since Project 5,6 to 7?
Librarian can only add book and update book copies and status in the database. Librarian cannot remove from database as we have not implemented it. We have not added publication date , issue date and return date of the book as we previously wanted to. The system does not send any notifications to the user , that functionality has not been implemented and the book has only two statuses , available or out-of-stock.

4. **Code from third party** :
   To setup the spring boot framework , we referred to the following website - https://www.bezkoder.com/spring-boot-mongodb-crud/
   To write code for making http api calls for the front end , we referred - https://kodejava.org/how-do-i-send-post-request-with-a-json-body-using-the-httpclient/ , https://www.tutorialspoint.com/apache_httpclient/apache_httpclient_http_get_request.htm
5. **Design process elements:-**

   Requirement, Use Cases and Scope:-
   This really helped us in understanding what needs to be built for our application and also where to restrict ourselves when committing our final deliverable. The use cases we thought of really helped us in realizing what we had in our mind to be built in our application. However, we overestimated the number of functionalities we would be able to build in the given time frame.

   Class Diagram:-
   Class Diagram played an important role. It helped us in knowing all the building blocks of the applications such as classes and design patterns.It really helped us when we started implementation work for our project because we already had all the elements which we had to make use of to build the application.

   API Contract:-
   We built the apis but we did not build the API contract. Had we built it we would have had some ease while consuming and calling the APIs I built. The API contract contains information on how the API should be used. Since most of the APIs were built by one of us and the other one was responsible for consuming them we faced a difficulty since we didn't have the contract. The member had to go back and look into the code to understand parameters and calls. This is some we would definitely do first next time when we work on the project