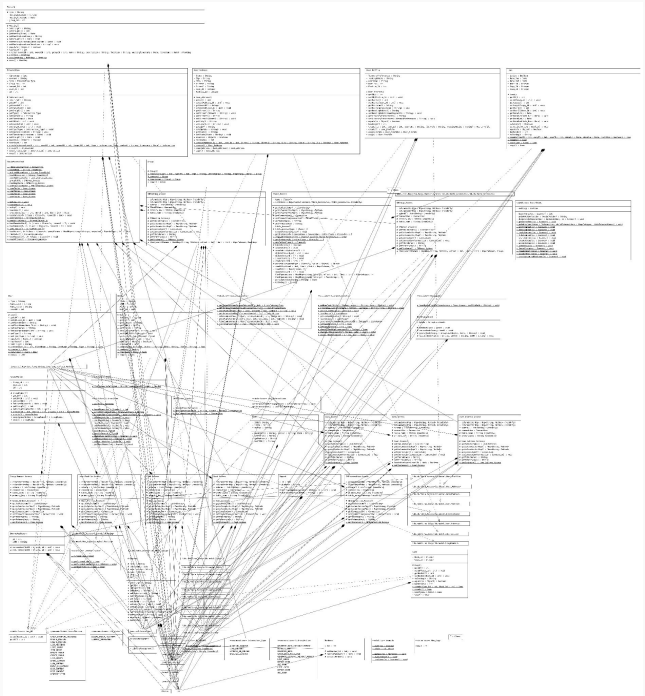
Library/Social Platform

CS401 Summer 2024 Group 2

Jian Xiong Lu
Dan Grace
Seng Heng

Overall System Design



-Model View Controller basic design

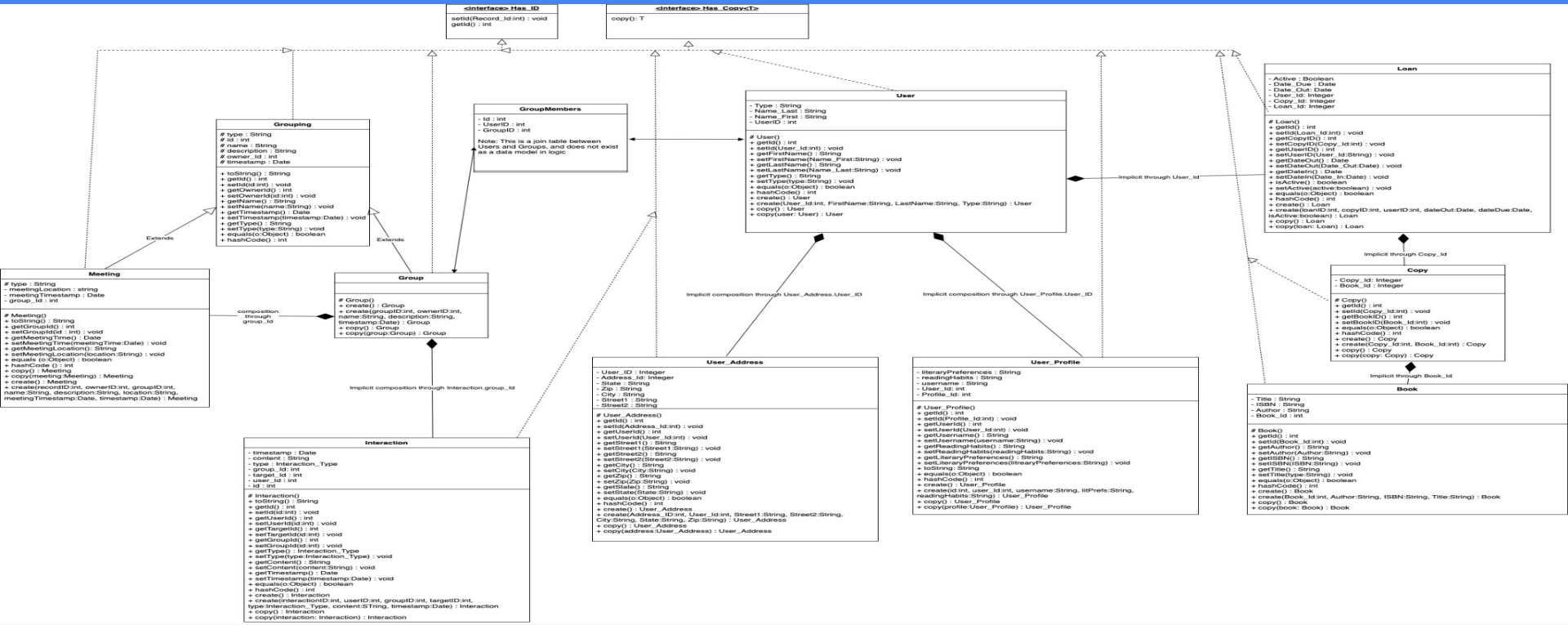
-Database models controlled by Accessors

-Controllers drive business logic

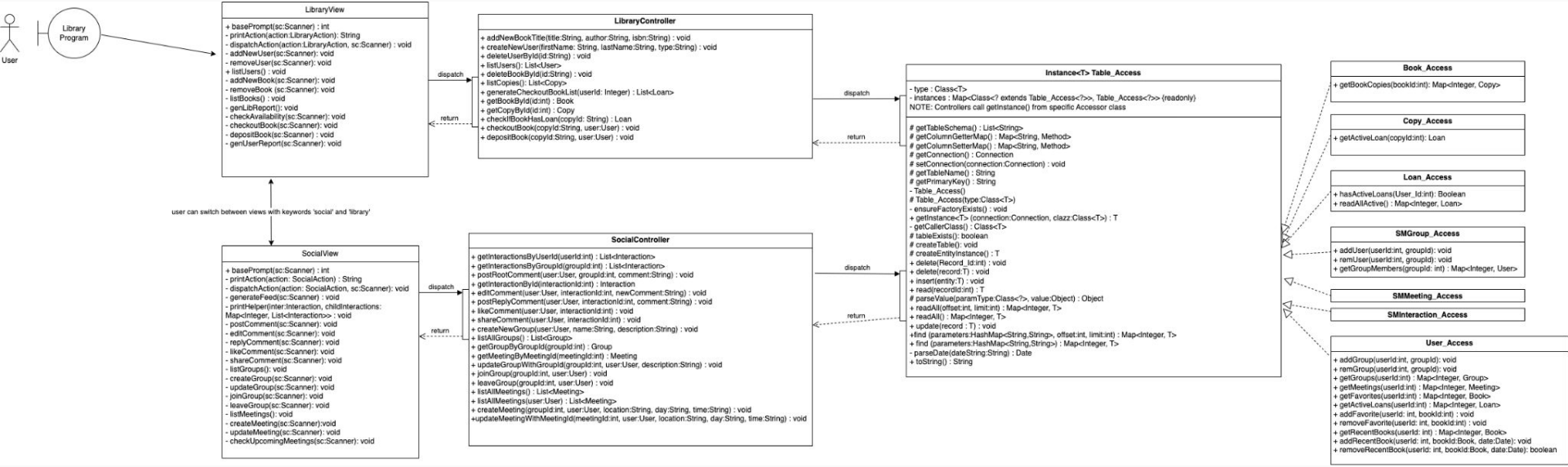
-Views interact solely with controllers

-Bloated UML Diagram

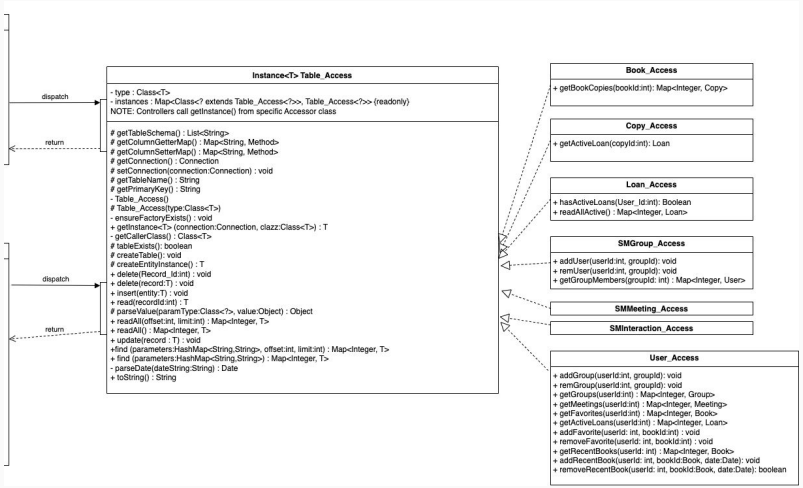
Clean UML Design : Complete Data Model



UML Design: Complete Sequence Flow



Database Accessors

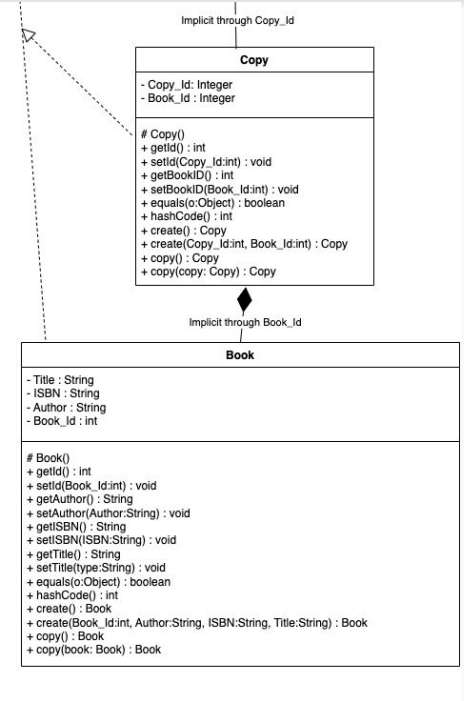


-Uses Singleton Pattern

-Exhibits SRP, OCP and encapsulation by building all DB CRUD operations into abstract super class

-Uses Reflection, generic types, and interfaces (ISP) to enable DB abstraction

Class Object Models



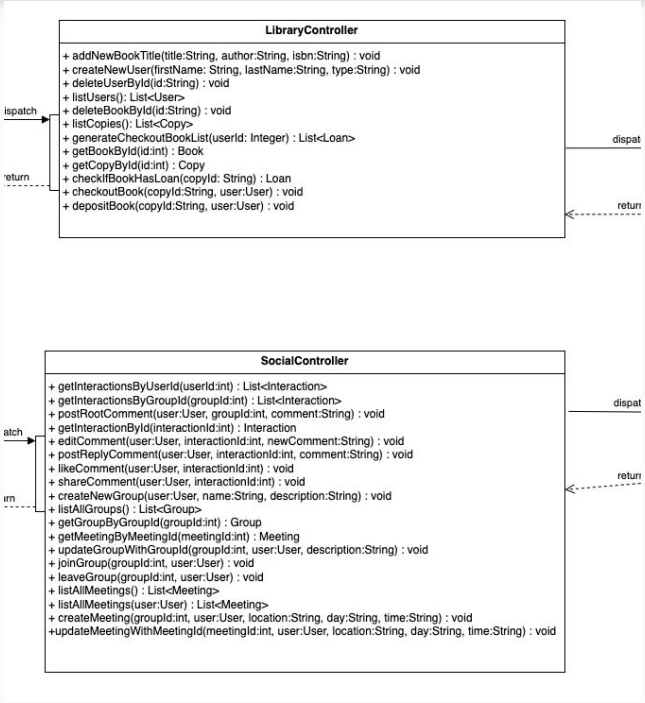
Demonstrates SRP, and in some cases OCP (ie Grouping)

Inline factory pattern for generating new objects by dispatch

Insertion handled by Controller, *NOT* by the object static method

Data objects used by DB Accessors to control columns and DB relations

Controllers



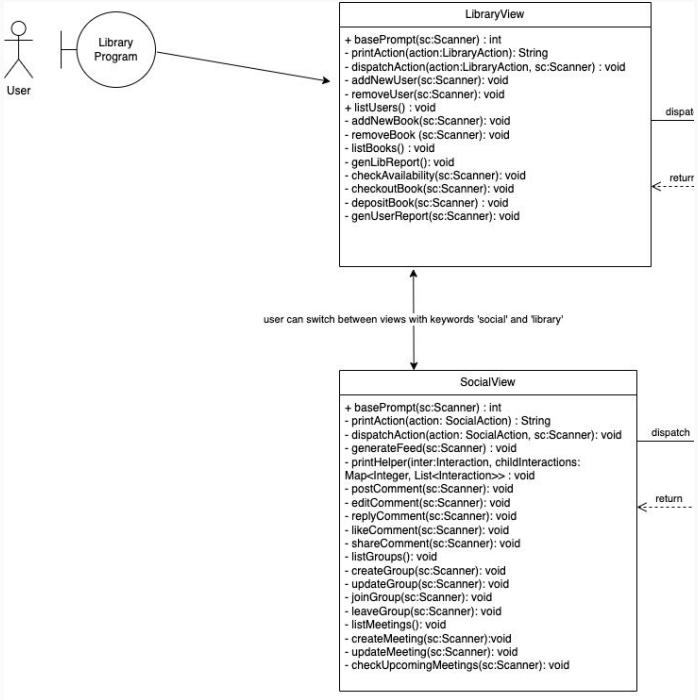
System employs a MVC Pattern for responsibility separation:

Views interact with the User and Controller

Controllers interact with User and DB (Model)

Controllers **responsible for all business logic**

Views

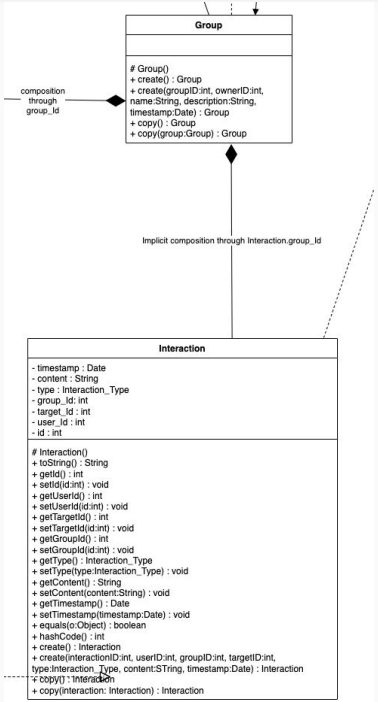


Command line interface to permit user action

Driven by IDs, not great user experience but enough to demonstrate functionality

Views **perform no business logic**, only presentation behavior

Social Platform Functionality Extension



We baked social functionality into our original design, intended to do it from the start

This ensured that we thought about implications of platform mixing early

Ex. event driven function calls directly from LibraryController to SocialController

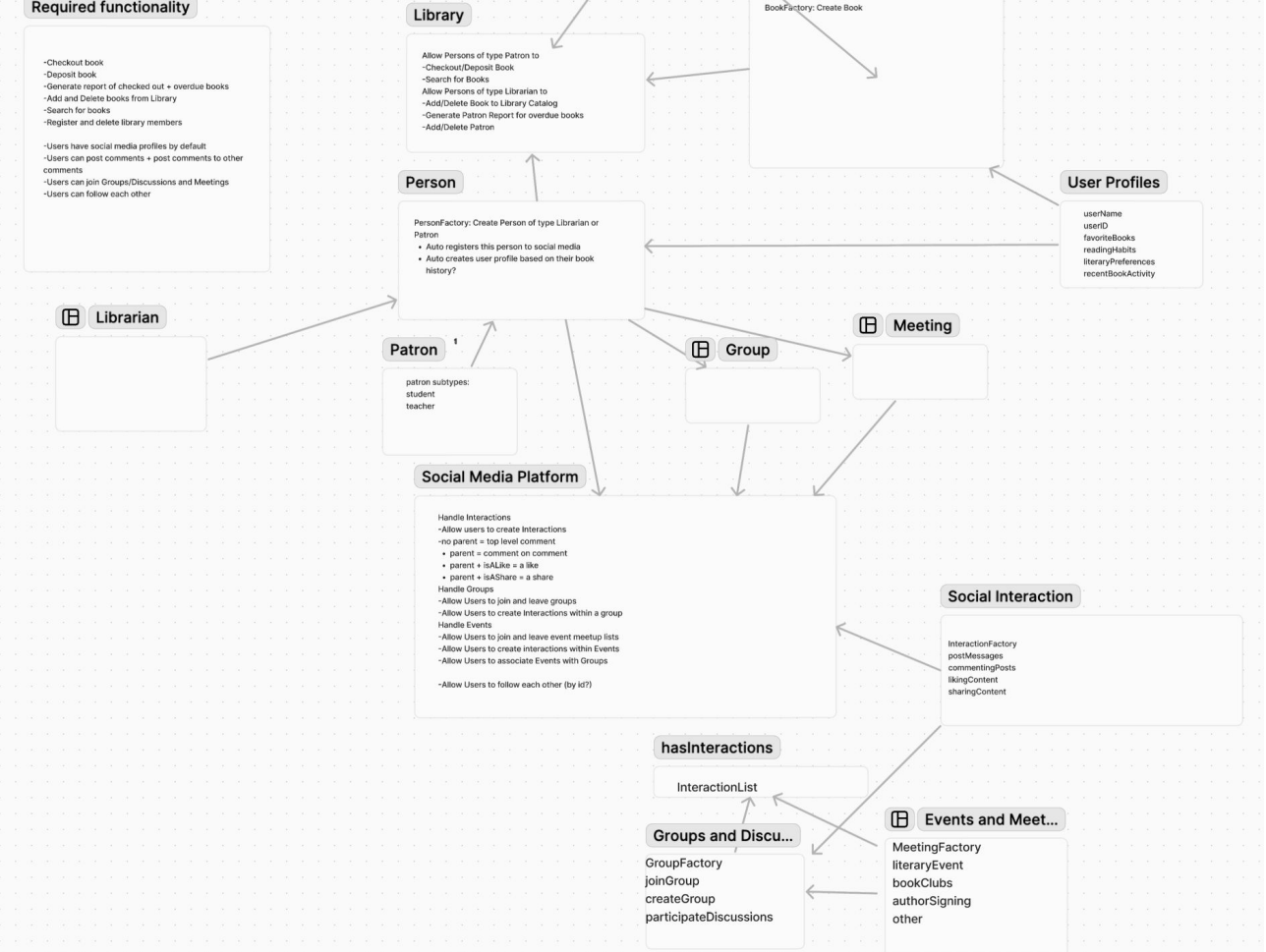
Engineering approach

Started with UML, decided to simplify early to better understand requirements

Went to basic ball-and-spoke sketch design in FigJam

Built off the FigJam, then designed ‘final’ UML sketch

View/Controller/Model/DB/Tests all could be developed in parallel independently



Early collaborative prototyping in FigJam

Engineering Discoveries

Getting the abstract models to play nice with the DB - i.e. list fields derived from implicit join tables

Discussion about more robust frameworks for persistence handling (Spring JPA)

Table design issues with inheritance (Group -> Meeting share nearly all fields)

Reflection enabling greater encapsulation, code reusability.

Applied Engineering Paradigms

MVC

Factory Pattern (instantiating all objects)

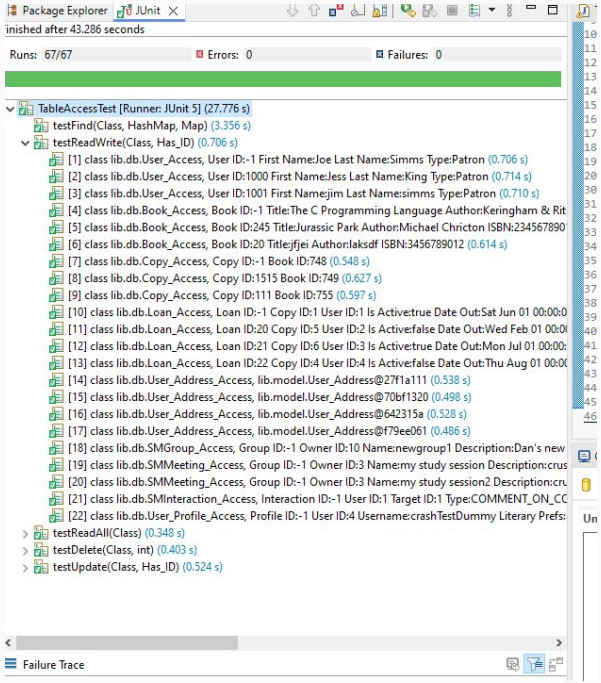
Singleton Pattern (Table Access, Controllers)

Interface Segregation (Has_Id, Has_Copy)

Liskov Substitution Principle (Table Access Controllers)

Dependency injection

Unit Test Cases



LSP in DB classes allows convenient testing as tests can be reused across table objects.

Enabled better parallel development as tests could run before other elements were ready

Test DB enables immutable environment for testing. Reset after every test

Functional test methods allow more granular / rapid debugging

End to End Test Cases

```
Would you like to generate your social feed, or the feed of a specific group?
Enter a group ID, or '0' for your personal feed.
1
-----
Kristina Dickson posted :
First Post!
Posted on Tue Jul 23 20:07:38 PDT 2024
-----
Kristina Dickson posted :

Posted on Tue Jul 23 20:07:11 PDT 2024
-----
Kristina Dickson posted :
First
Posted on Tue Jul 23 20:06:05 PDT 2024
-----
Kristina Dickson posted :
```

Manually tested Library and Social functionality

Library test included scripted user journey to ensure all functionality and edge cases were tested

Interfaced live with DB viewer to debug as needed

Future development could add automated mocked user journey testing.

Learnings after this Project

- Design
  - Our team has learned the application of use case diagram, class diagram, structural diagram from UML. We reflect our raw ideas to these diagrams, then gradually refine them as we're going further on the project development. Finally, we will display these diagrams in the presentation.
- Implementation
  - Our team has learned data structures that interact with the database.
- Project Management
  - Our team has learned to broke whole task in pieces and created a chat for members to report their progress, communicate the following schedule, and post problems encountered in project development and solve these problems with group's efforts.