THE hand
Innovation *for* a better life

WELLIB
BOOK FOR LIFE

[Version 1.6]

SOFTWARE ARCHITECTURE DOCUMENT

for Wellib

Presented by :
Group 9 - The Hand (21CLC10)

Project Name:
Wellib

Group 9 - The Hand

Wellib
Software Architecture Document

Version 1.6

Revision History

Date	Version	Description	Author
28/Jul/23	1.0	Finish: Introduction, Architectural Goals and Constraints.	Nguyễn Tuấn Tuấn
01/Aug/23	1.1	Finish: Logical View	Nguyễn Tuấn Tuấn
03/Aug/23	1.2	Finish and update: Use-Case Model	Trần Hiếu Tâm
04/Aug/23	1.3	Update: Logical View	Trần Hiếu Tâm
16/Aug/23	1.4	Finish main components	Nguyễn Tuấn Tuấn
18/Aug/23	1.5	Finish Deployment & Implementation View	Nguyễn Tuấn Tuấn
18/Aug/23	1.6	Update: Use-Case Model	Trần Hiếu Tâm

Table of Contents

1. Introduction **5**

2. Architectural Goals and Constraints **5**

3. Use-Case Model **6**

4. Logical View **13**

MAIN COMPONENTS **14**

4.1 GUI User: 14

4.2 GUI Guest: 16

4.3 GUI Librarian: 17

4.4 GUI Administrator: 19

PACKAGE COMPONENTS **22**

4.5 Component: Security 22

4.6 Component: User Interface 23

4.7 Component: Search/Filter tool 24

4.8 Component: Feedback tool 25

4.9 Component: Membership 26

4.10 Component: Transactions 27

4.11 Component: Database update 28

4.12 Component: Account update 29

4.13 Component: Models Mangament 30

4.14 Component: Accounts Management 31

5. Deployment **32**

6. Implementation View **33**

Software Architecture Document

1. Introduction

This Software Architecture Document provides an overview of the entire document, which includes many of its categories, such as purpose, scope, definitions, acronyms, abbreviations, references, and total document overview. And the categories will display information about each segment of the document with great details.

2. Architectural Goals and Constraints

This segment describes the software requirements and objectives that have some significant impact on the architecture such as safety, security, privacy, use of an off-the-shelf product, portability, distribution, and reuse. It also captures the special constraints that may apply: design and implementation strategy, development tools, team structure, schedule, legacy code, etc.

Architectural Goals:

- **Security:** Provides a secure and protected system, to make sure users experience is a safe and error-free one, with authorized access and regulated control.
- **Maintainability:** The system design is to be clear and easy-to-understand, so further maintenance and updates might be possible in the future and poses almost to none difficulty.
- **Performance:** Provides good services that have no performance issues, and be able to handle a peak load of 1000 users simultaneously and up to 100000 transactions per day.
- **Usability:** The system should be easy to use, provides effective tools for intended purposes, is efficient and engaging for the users to acquaintance themselves with and continue to utilize our services.
- **Legal and Copyrights:** Must abide by all copy rights laws and such to avoid lawsuits and legal conflicts.
- **Support-ability:** The website for the library system must be able to run in the following browsers:
    + Google Chrome 54 or later.
    + Microsoft Internet Explorer 11 or later.
    + Mozilla Firefox ESR 45 or version 48 or later.

- **Accessibility:** Upon launch, the system shall be modified to become most accessible to all users, including those with disabilities including visual impairment, audio impairments, mobility impairments,…
- **Extendability:** To be able to add new features to the website at any point in the future by creating a system that allow for models to be easily implemented.
- **Online manual:** Provides a manual with instructions and guides for new readers to use and navigate the website.

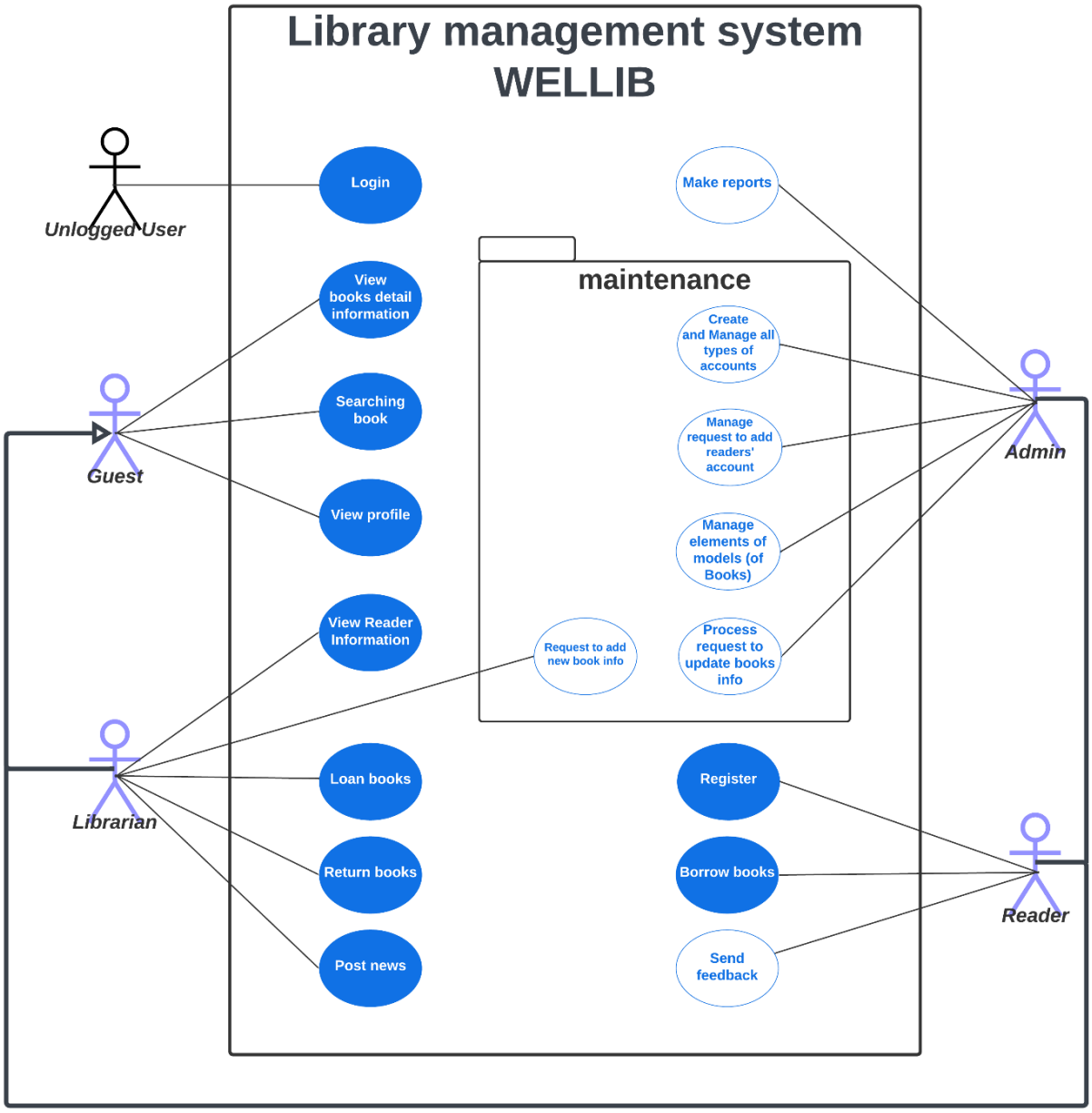
Architectural Constraints:

- **System Design:** Since the system has many moving parts and components to manage, trying to achieve efficiency while implementing the system is a running problem of the development process, therefore system design must be well-thought-out, accommodate for all requirements and must be adaptive.
- **Tools:** While Django serves well at providing the main framework for the project, there are still many requirements lacking which prompts for further research for other tools to address the problems at hand and is a cause for time loss in development.
- **Team collaboration:** While the process of diving work and managing team members for the project is going smoothly, further improvement is still warranted to keep up with project demands.
- **Time:** Since the project have many stages, deadlines must be implemented into the process and must be followed. However, unexpected errors and other outside matters and interference could still affect the overall workflow of the group, making it harder to meet the proposed deadline as intended.

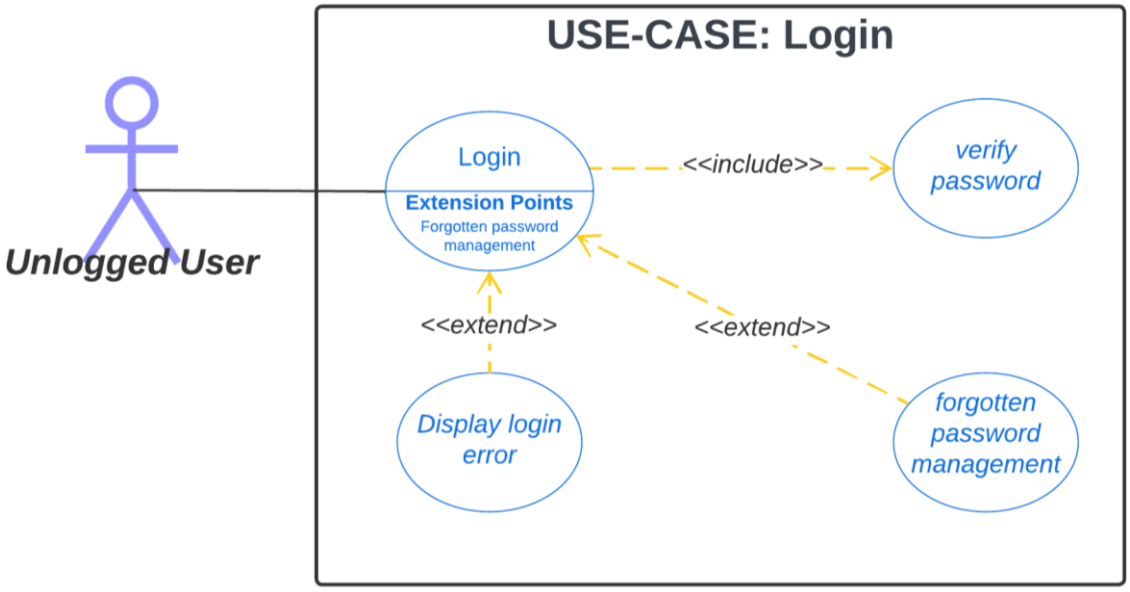
3. Use-Case Model

This segment includes a use-case diagram to demonstrate the inner workings of the system as intended.

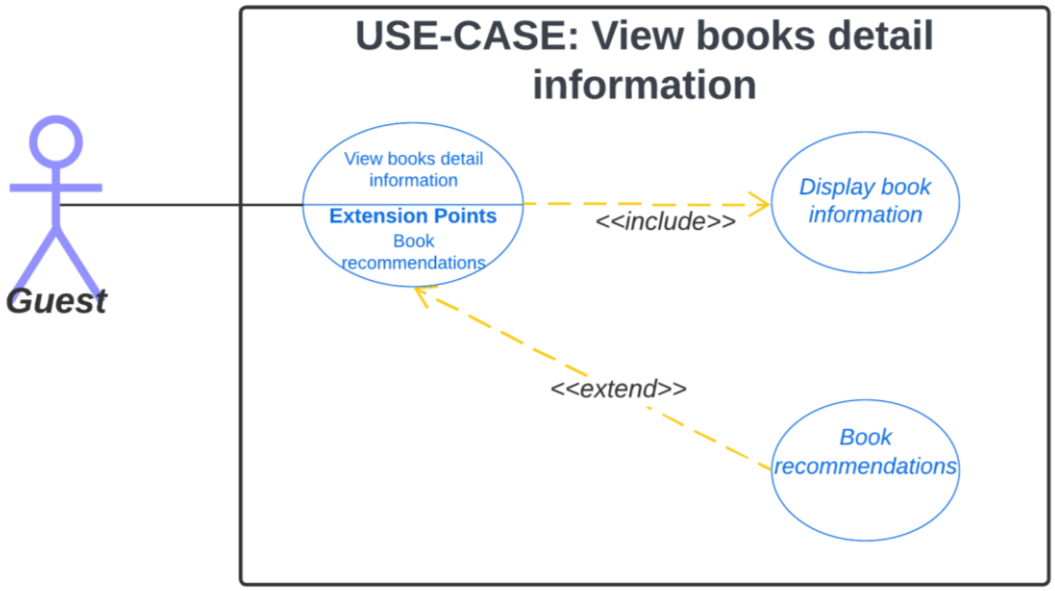
3.1 Overview Use-Case Model



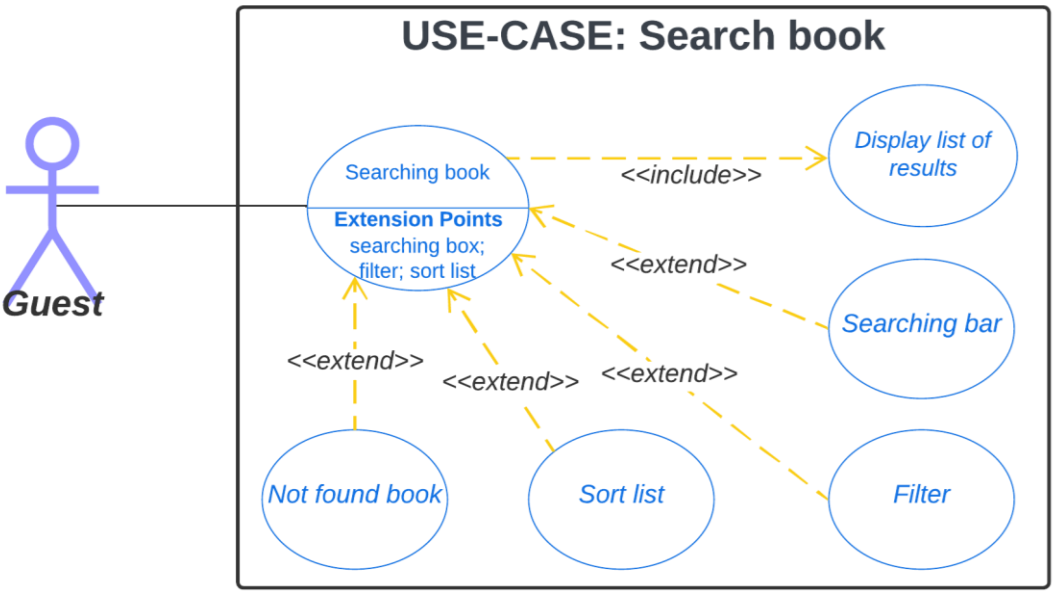
3.2 USE-CASE: Login



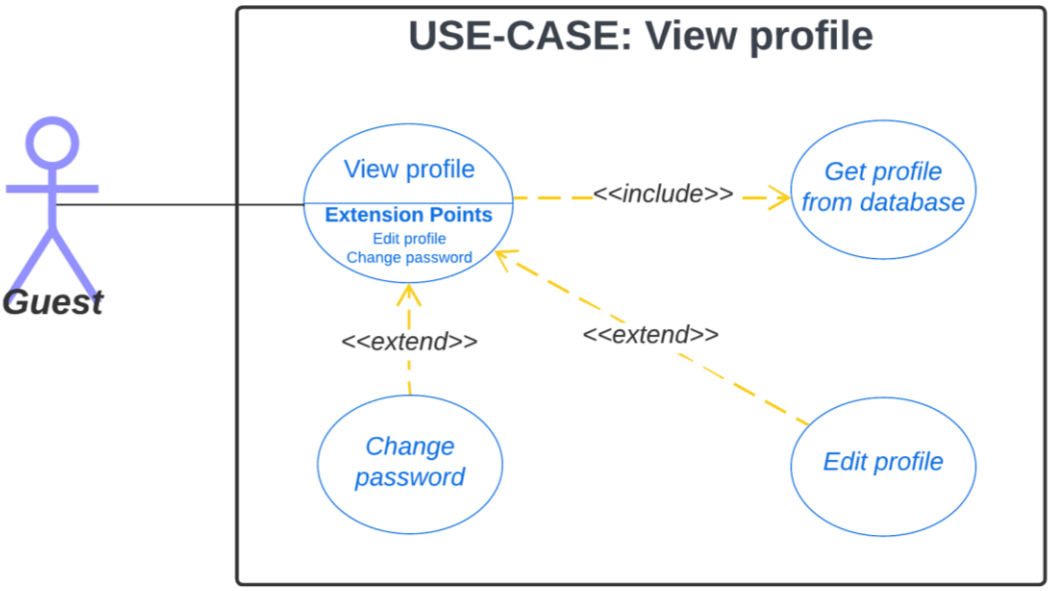
3.3 USE-CASE: View books detail information



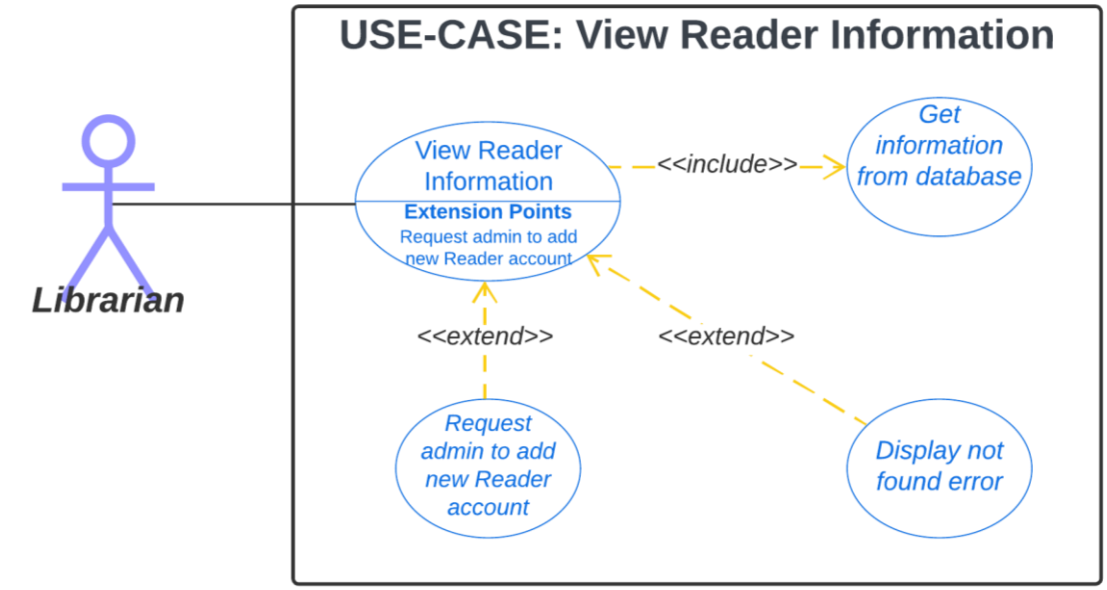
3.4 USE-CASE: Search book



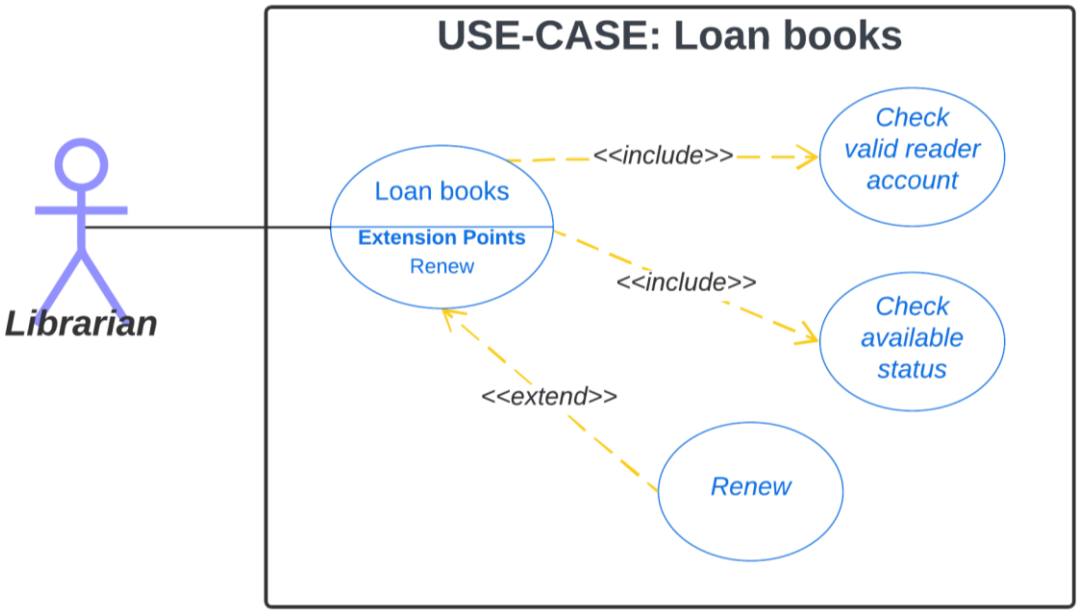
3.5 USE-CASE: View profie



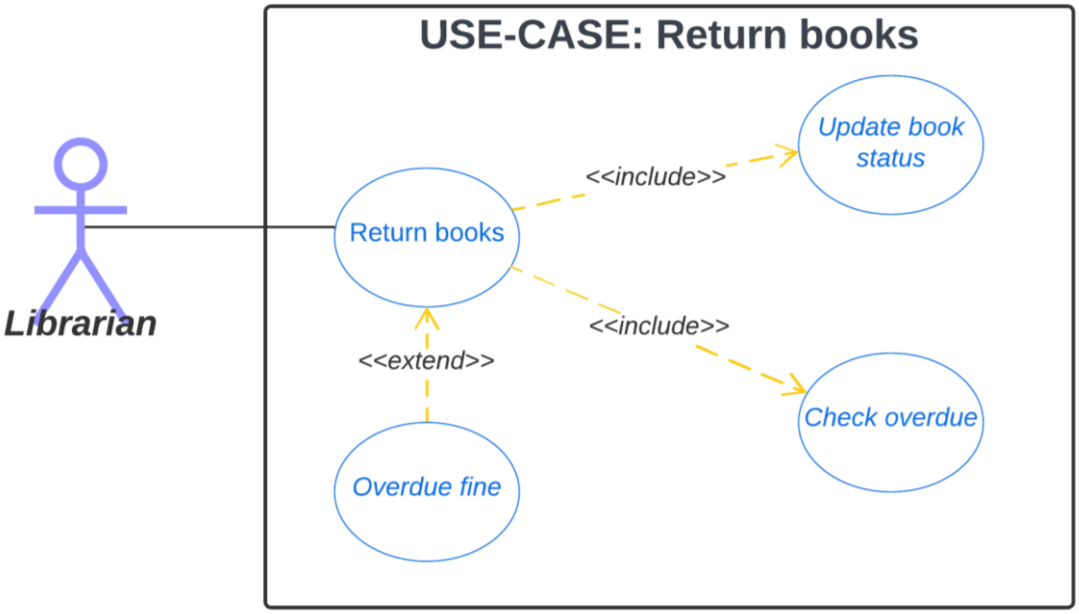
3.6 USE-CASE: View Reader Information



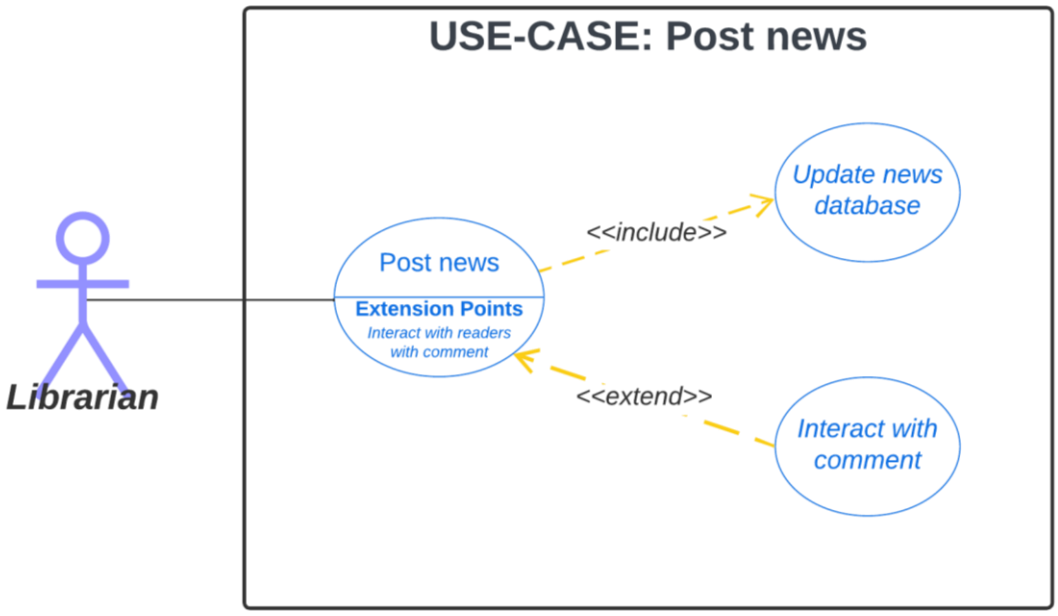
3.7 USE-CASE: Loan books



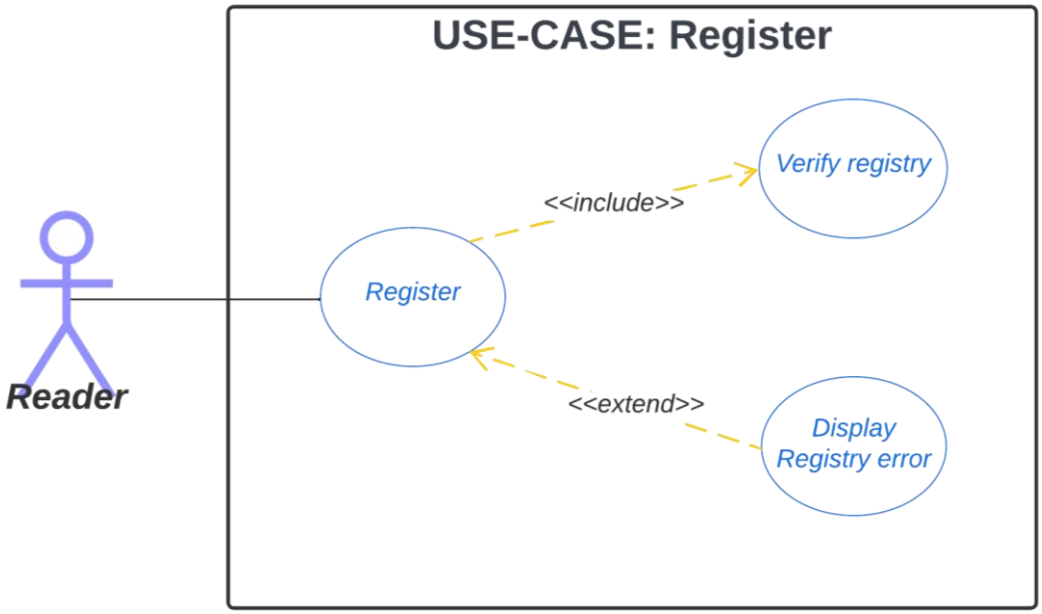
3.8 USE-CASE: Return books



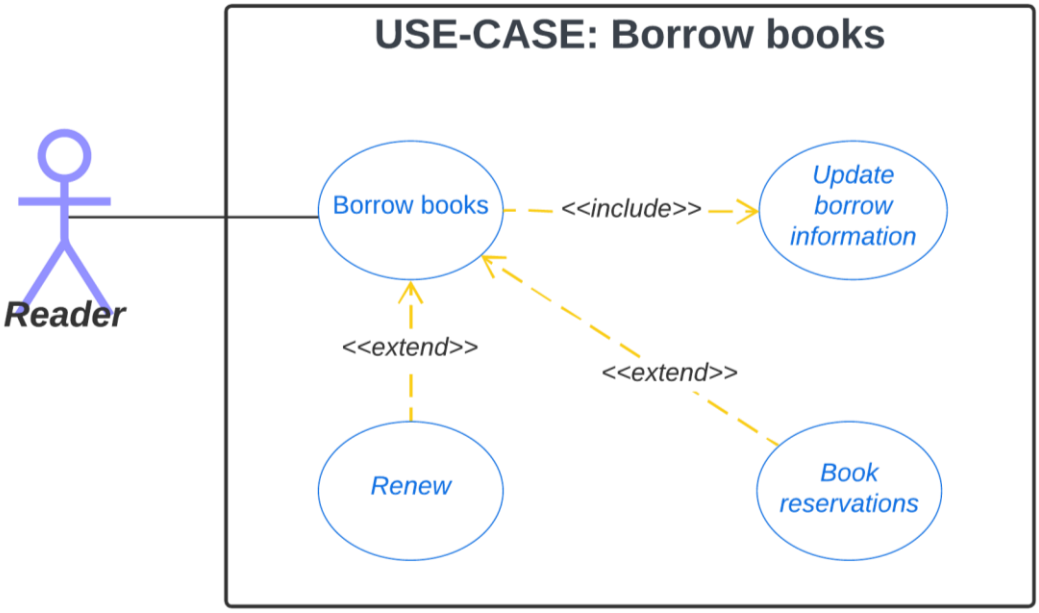
3.9 USE-CASE: Post news



3.10 USE-CASE: REGISTER



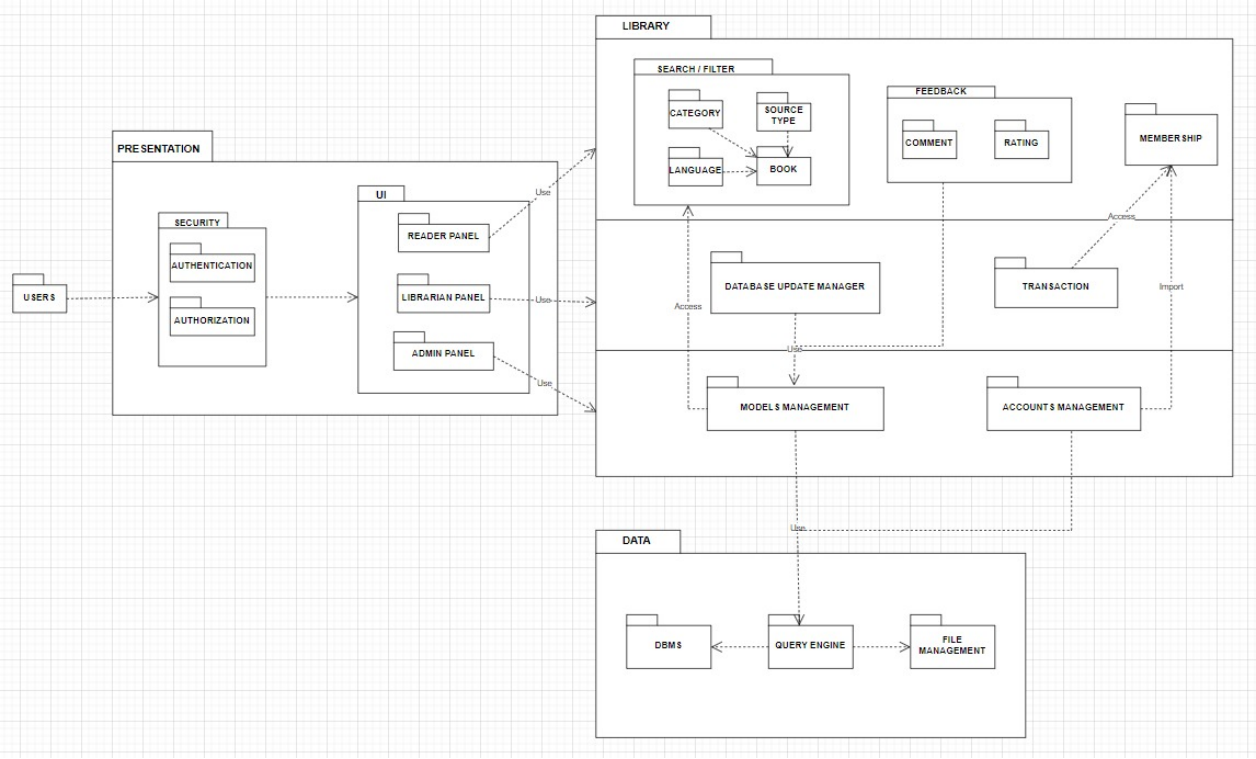
3.11 USE-CASE: Borrow books



4. Logical View

This section the architecture of the system along with it's main components and relationships among them. This will includes the main flow of the workings between components and demonstrate how the links and relationships inside the system operates.

The following diagram provides a high-level view of the main packages composing the system. The diagram demonstrates how the users of the website will interact with the UI layers as well as the main functionalities of the website and it's main components stored within the database. This is an overview representing different application layers and give an overview of the organization of the project's code.



The system comprises of three main layers. With presentation, the users connects to the web page via the Internet, they will perform a login that will be validated and sorted into the right interface for that particular user, these separate panels will have distinct functionalities provided by the system based on their role in the library, a Search / Filter tool, a Feedback tool and Membership are all functionalities available to the reader, librarian can perform Transactions of books in the library as well as track and update the database notice and the admin will manage the page's abilities and users accounts.

The system is follow the main model design of MVC represented by the main components of the system which is divided into many GUI, controllers and services.

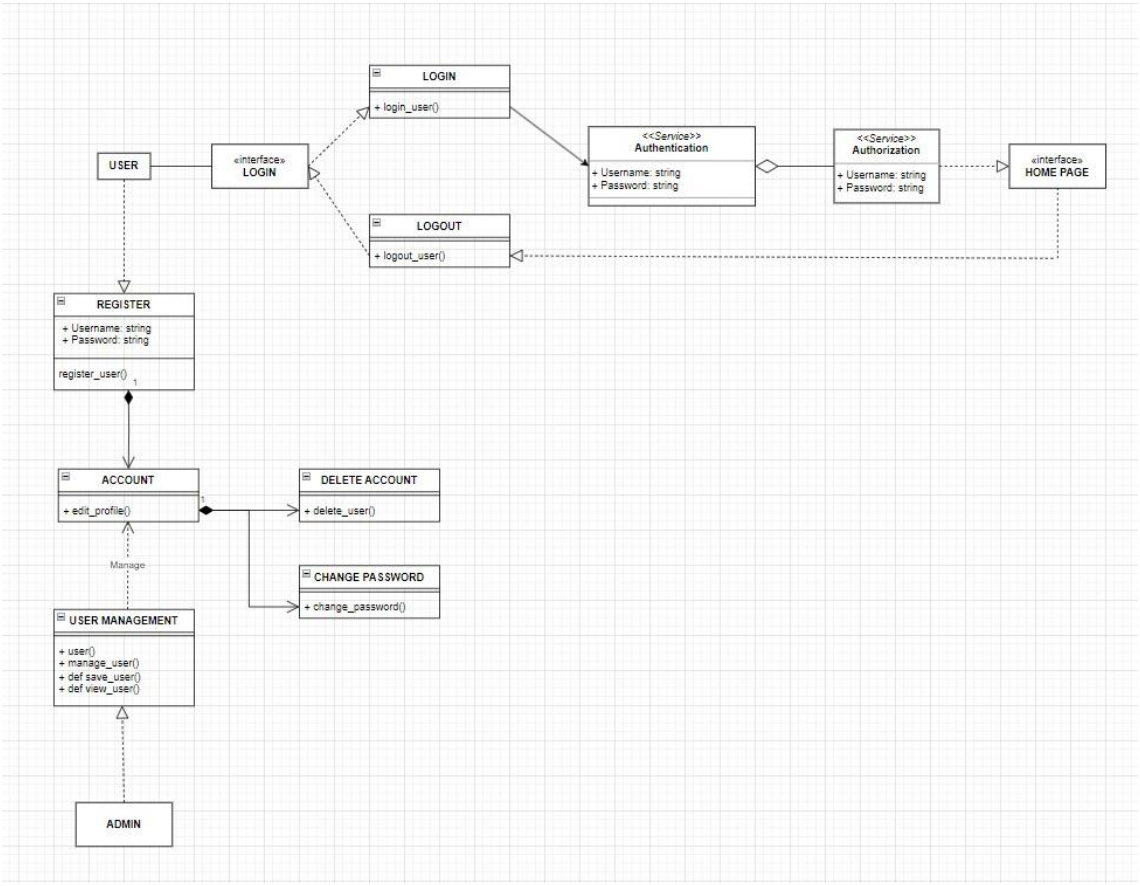
MAIN COMPONENTS

4.1 GUI User:

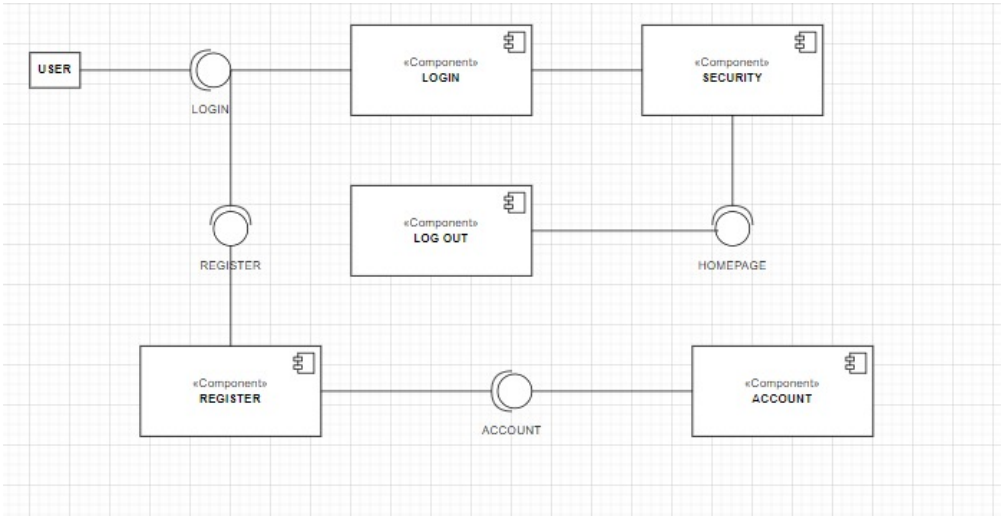
The standard interface the will provides the option to login and register, inputting ID and password will redirect user to the other interfaces according to their individual purposes. As well as creating an account for further use.

- **Login**: Users can log in to their account with ID and Password.
- **Log out:** Users can log out of their account.
- **Register:** Sign up an account that is managed by Administrators.
- **Edit Profile:** Changing information of user's account.
- **Delete User:** Delete the account if it is no longer needed.
- **Change Password:** User can change password of their account after registering.
- **Authentication:** A Service that checks username and Password before redirecting
- **Authorization:** Checks and redirect user to corresponding UI based on their role in the system

Class diagram:



Component diagram:

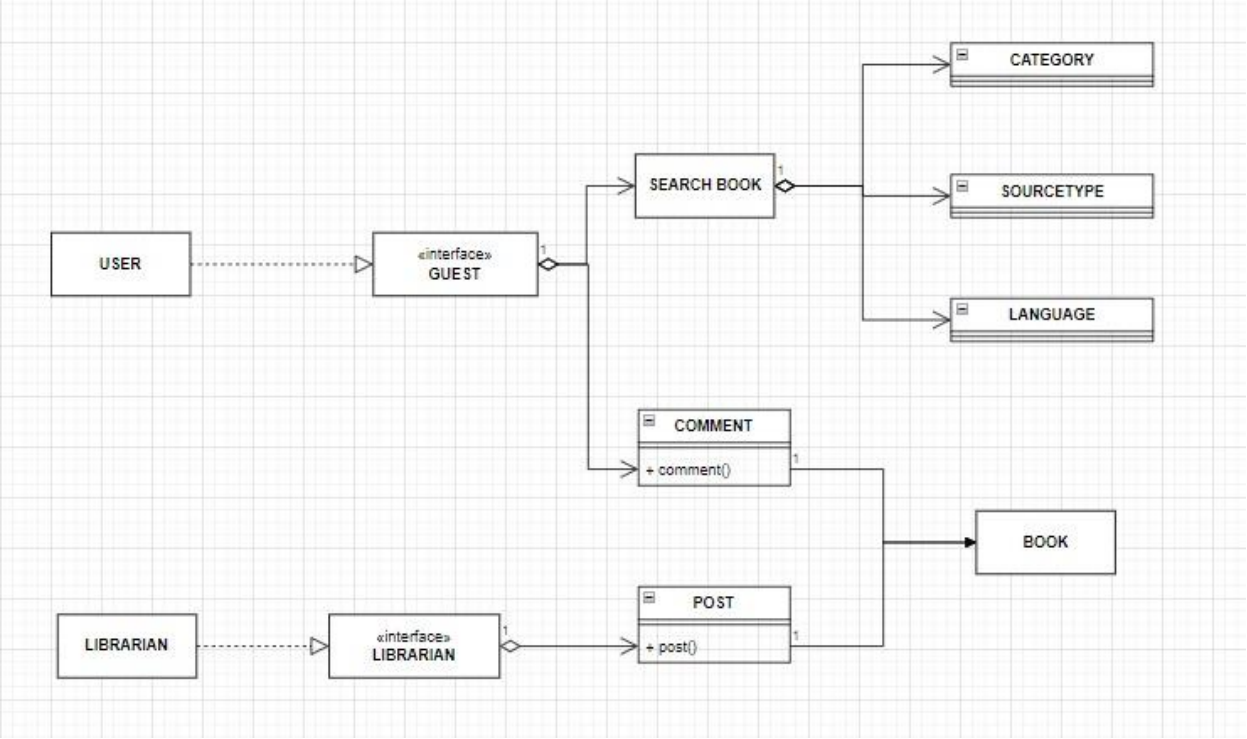


4.2 GUI Guest:

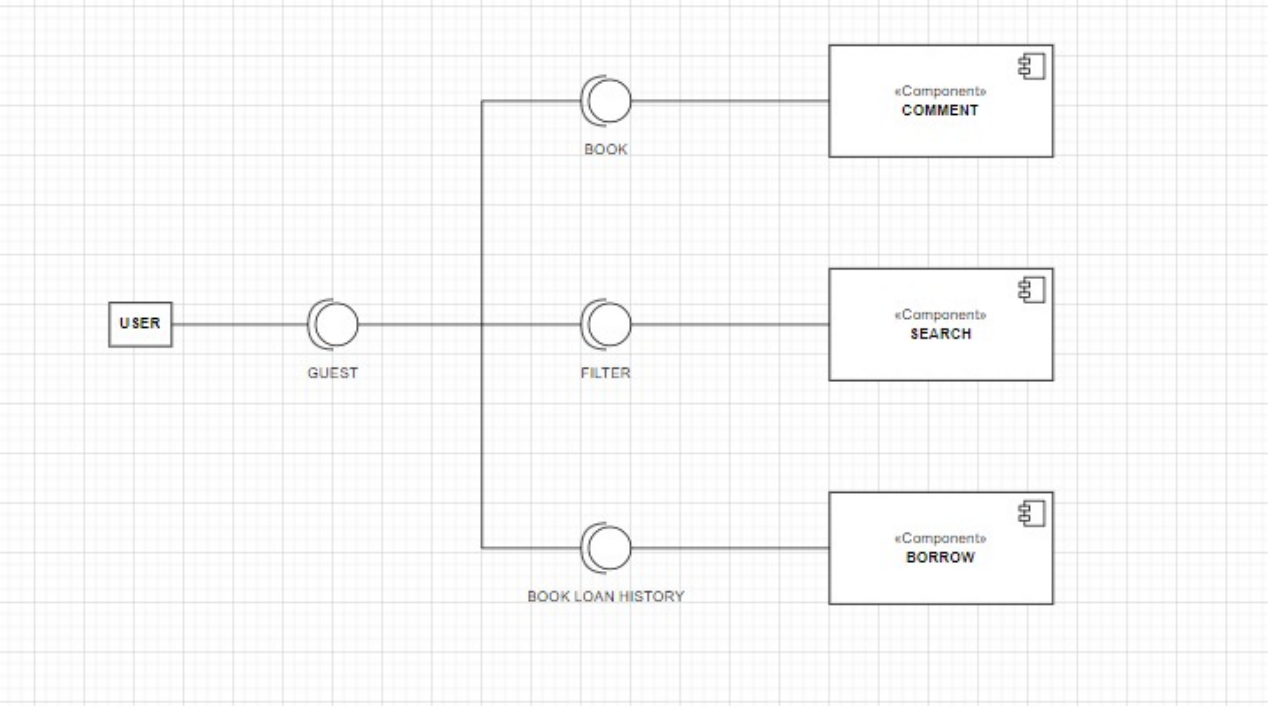
Reader can interact with books and utilize other components of the system to help with this interaction.

- **Search book:** Utilize systems functions to find book via models within the system like navigation bar and filters.
- **Comment:** Readers can comment on a specific book in the system adding more information and it's rating.
- **Borrowing:** Allows Guests (Readers that have a registered account) to borrow books from the library via interacting with the Librarian and view history via interface.

Class diagram:



Component diagram:

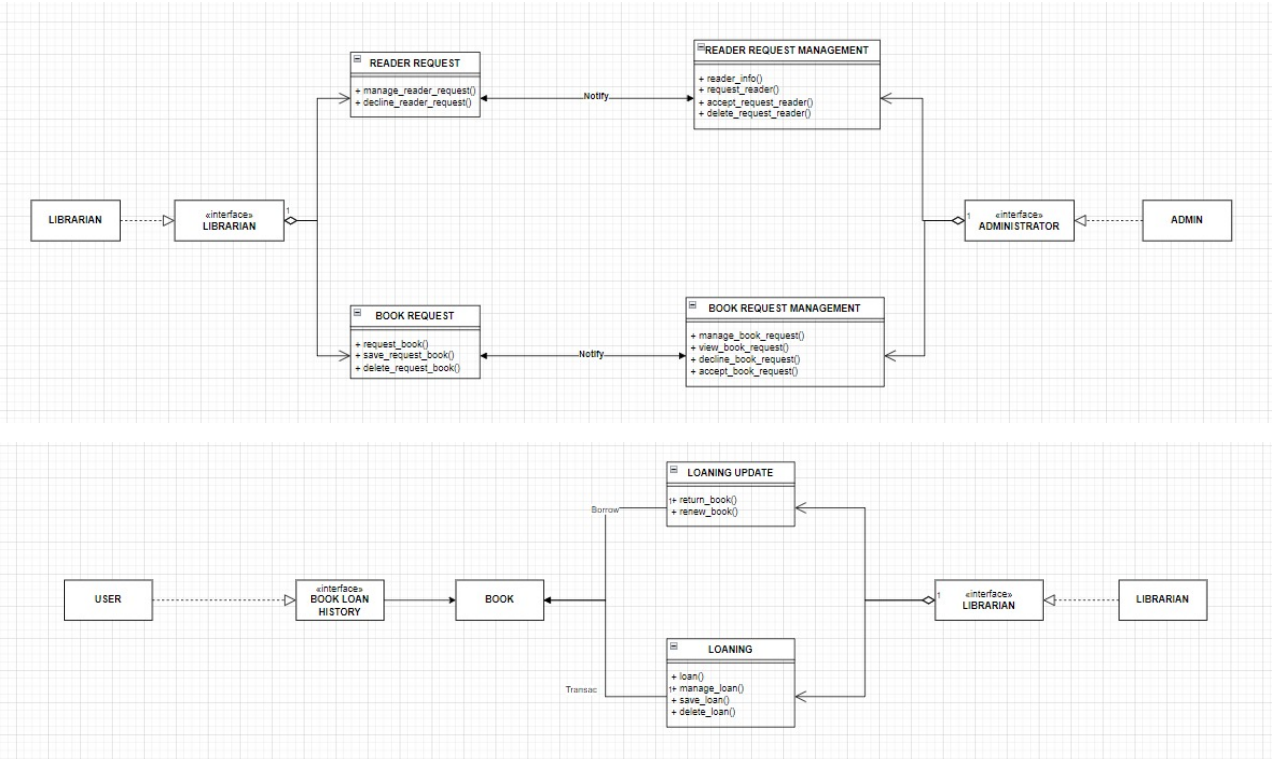


4.3 GUI Librarian:

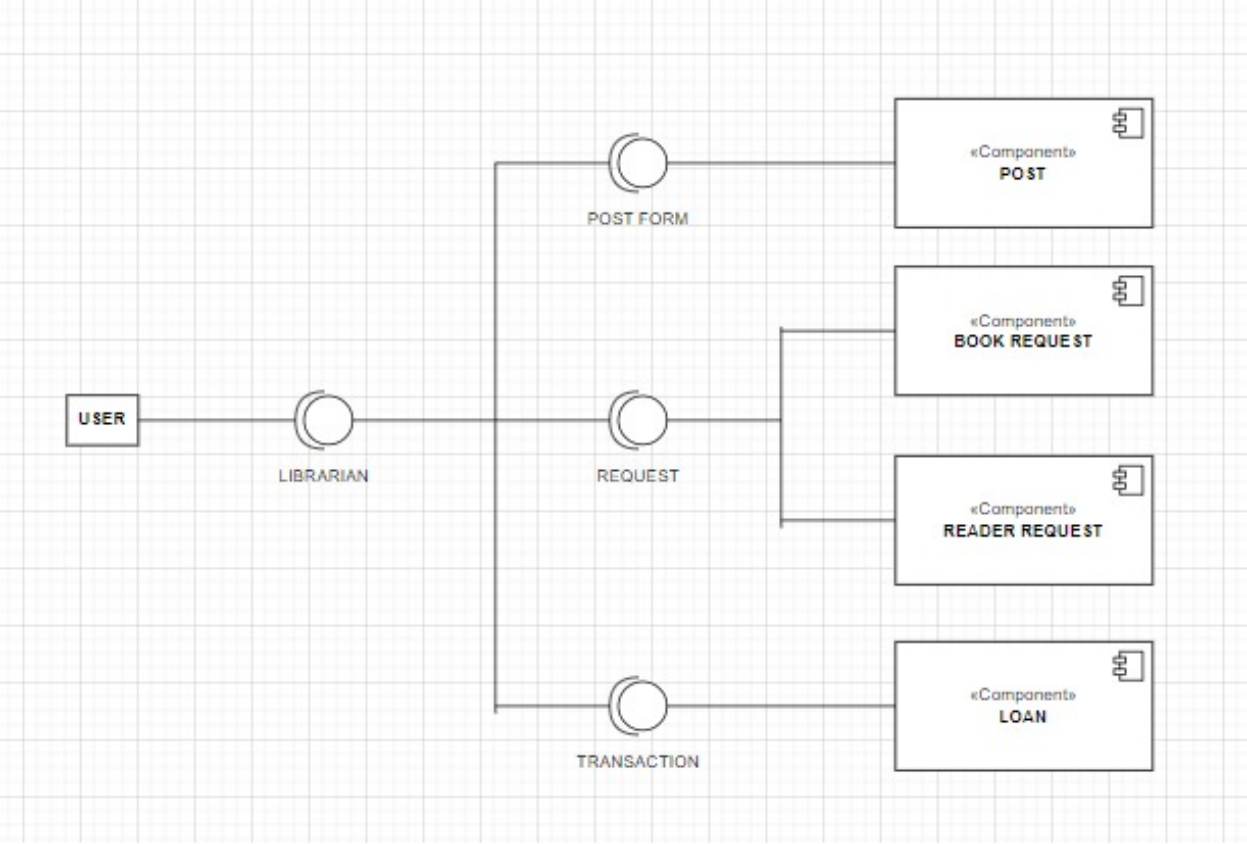
Librarian hold the role of viewing and updating books and accounts which can influence the Administrator's page.

- **Post:** Librarian can post news and articles and other information to the webpage.
- **Request Books:** Allows the Librarian to view and make request to add to the system's database.
- **Request Readers:** Allows the Librarian to view readers information and make request to either add their account to the system's database or to remove it, this request can be cancel.
- **Books Transaction (Loaning):** Librarian have the authority to make transactions with guest and keep track of it, this component allows for loaning out books and this transaction is reflected in the database.
    + *Loaning:* Loan out books for guest and keep track of it.
    + *Loaning Update:* Update the status of the loan, to either finish it by returning the book or extending the due date.

Class diagram:



Component diagram:



4.4 GUI Administrator:

Admin have the main role of managing the components that make up the system base functionalities as well as users accounts and books that could be recommended by the Librarian.

Models Management: Administrators can manage system's functionalities via the interface provided, which includes:

- **Models Books:** Main product of the system, have attributes and saved in the database as well as having many interaction with many components of the system
  - *Manage Books:* Can add, edit or delete and change book information.
  - *View Books:* View books details and it's attributes.
  - *Export Books:* Export book from the system.
  - *Import Books:* Import book to the system.
- **Models Category:** One of Books attributes as a accompanying component, present

in the search filter as the category section.

  + *Manage Category:* A Tool for Admin to make changes to the Category component, being able to add, edit, delete, etc…
  + *View Category:* Allows User to view details of the component via interface.
  + *Export Category:* Export a category from the system.
  + *Import Category:* Import a category to the system.

- **Models Source type:** One of Books attributes as a accompanying component, present in the search filter as the sourcetype section.
  + *Manage Source type:* A Tool for Admin to make changes to the Sourcetype component, being able to add, edit, delete, etc…
  + *View Source type:* Allows User to view details of the component via interface.
  + *Export Source type:* Export a source type from the system.
  + *Import Source type:* Import a source type from the system.
- **Models Language:** One of Books attributes as a accompanying component, present in the search filter as the language section.
  + *Manage Language:* A Tool for Admin to make changes to the Language component, being able to add, edit, delete, etc…
  + *View Language:* Allows User to view details of the component via interface.
  + *Export Language:* Export a language from the system.
  + *Import Language:* Import a language from the system.

User Management:

  + *Manage User:* Can add, edit or delete and change user's information.
  + *View User:* View user information via their account.
  + *Save User:* save user data to the database using the form functionality to check validation.

Book Request Management:

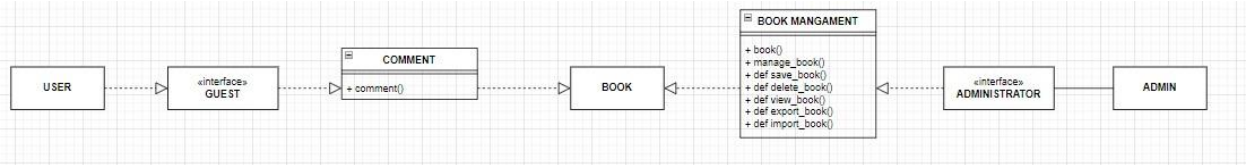
  + *View Book Request:* See the requests of changes recommended.
  + *Accept Book Request:* Accept a change and saves to the database.
  + *Decline Book Request:* Decline the change, removing it from the request list.

Reader Request Management:

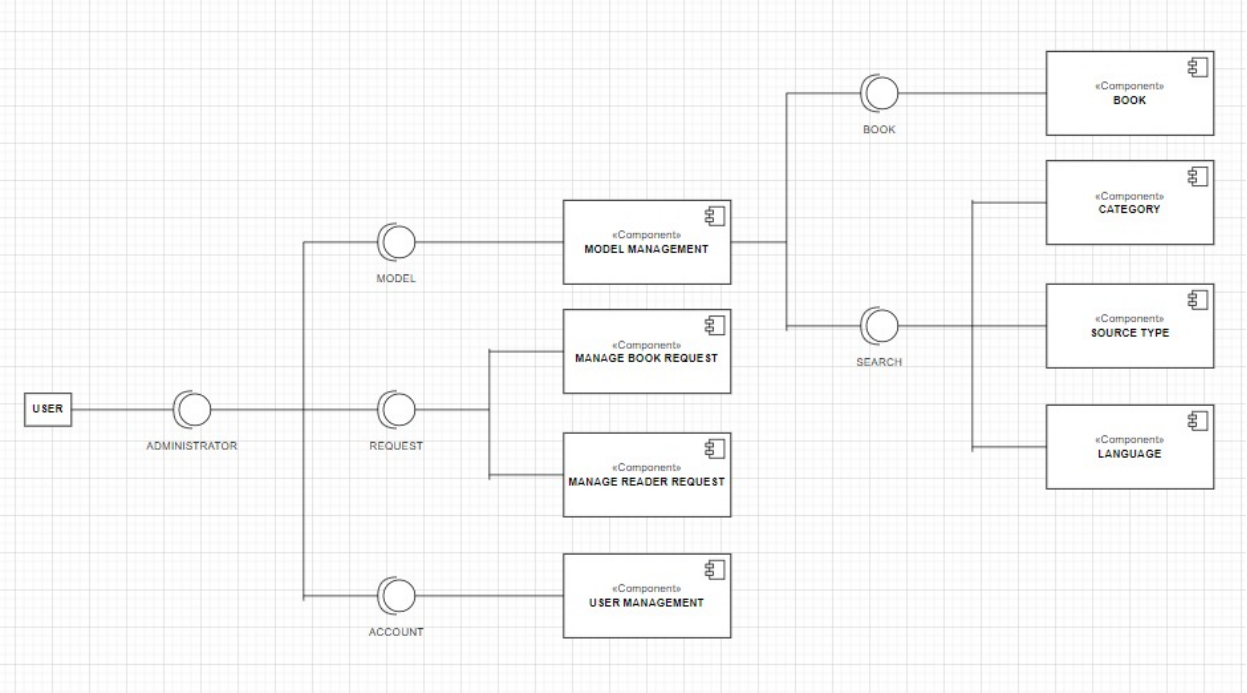
  + *View Reader Request:* See the requests to add accounts based on recommendation.

+ *Accept Reader Request:* Notify the librarian that the changes will be made.
+ *Decline Reader Request:* Notify the librarian that the changes is refused.

Class diagram:



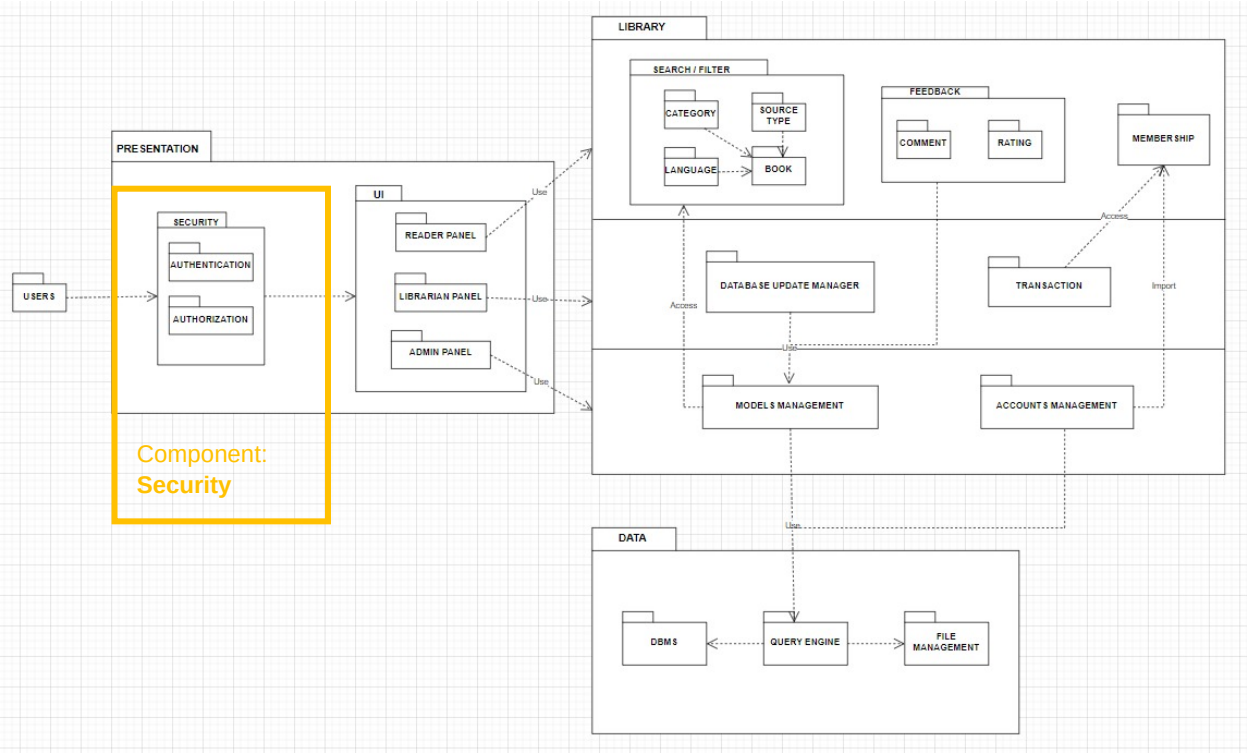
Component diagram:



PACKAGE COMPONENTS

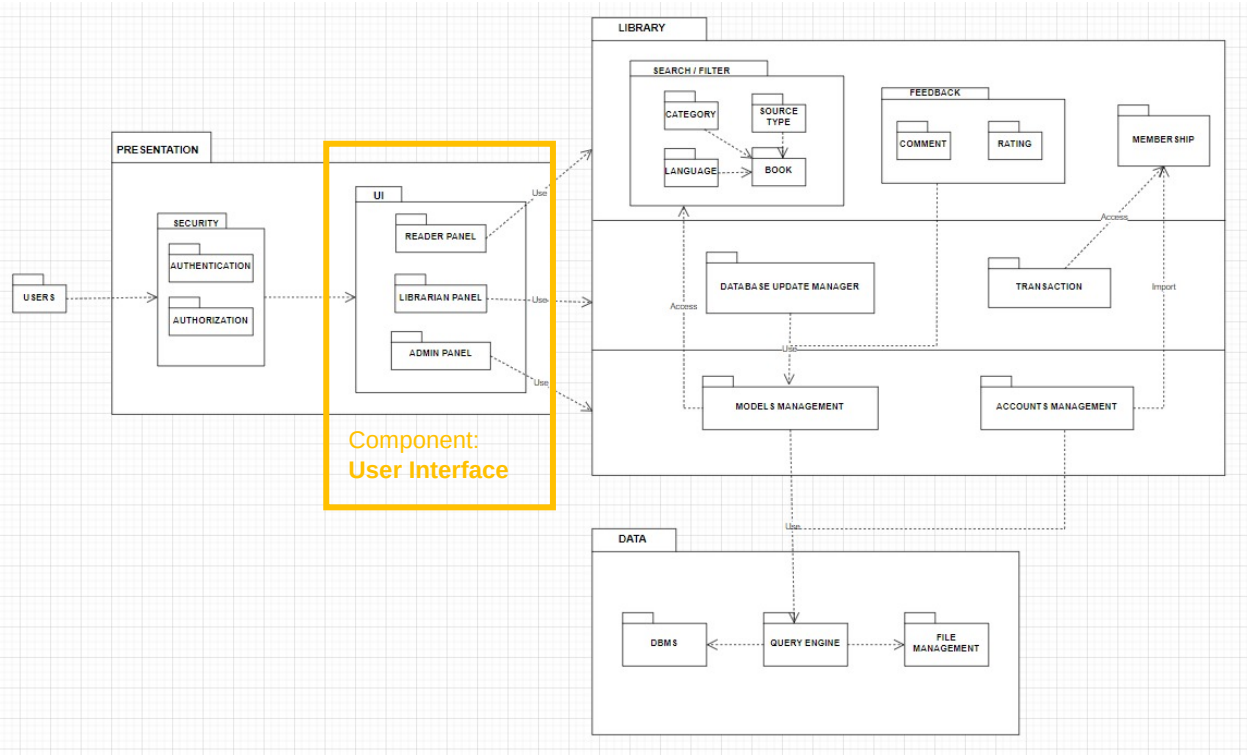
4.5 Component: Security

The system will asked users for ID, afterwards it will check the validation of said ID and sort users to their specific role, ensuring authentication and the correct authorization.



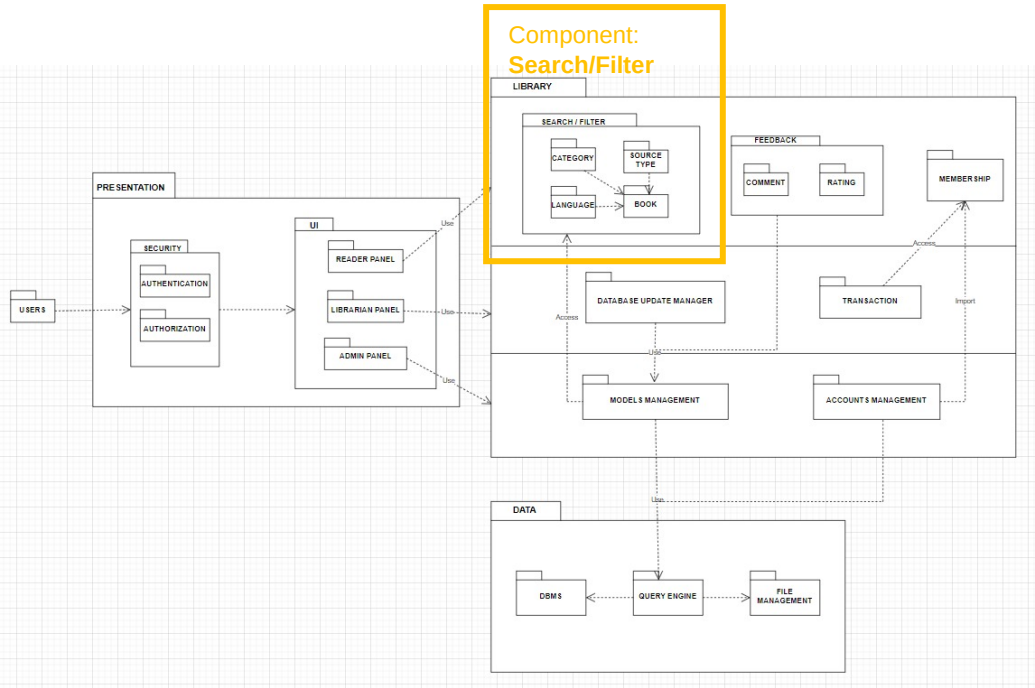
4.6 Component: User Interface

Due to website's purposes, the UI is separated depending on roles and will provide according services.

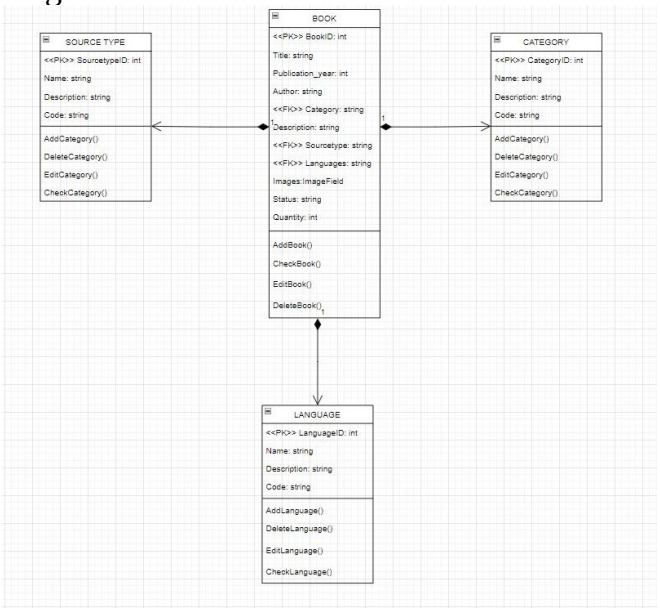


4.7 Component: Search/Filter tool

The main functionality of the library for readers, it is comprised of many classes that provides secondary traits to the main product of the library - books, this enable users to easily find the right book by sending data back through to the Database and running a check algorithm via the https protocol.

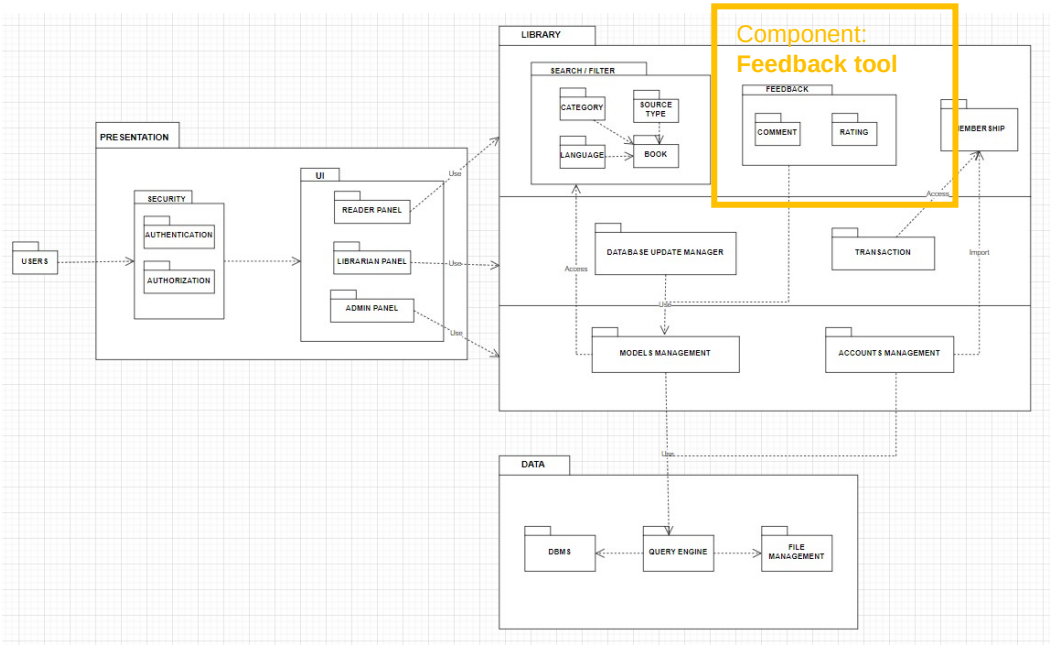


Accompanying class diagram:

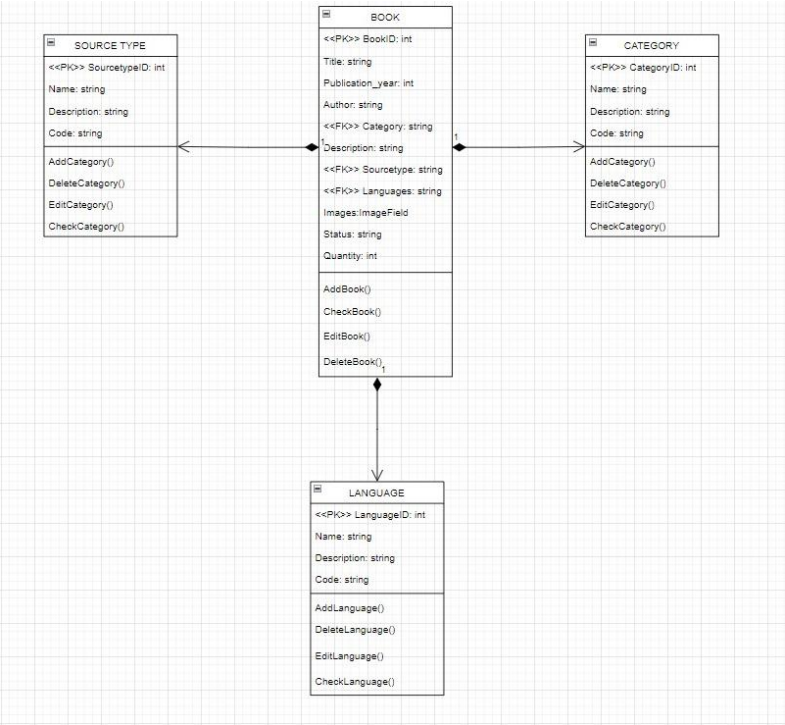


4.8 Component: Feedback tool

A readers exclusive tool, this give users the ability to rate and comment on certain books and will add the information directly to the data within the database also via the https protocol.

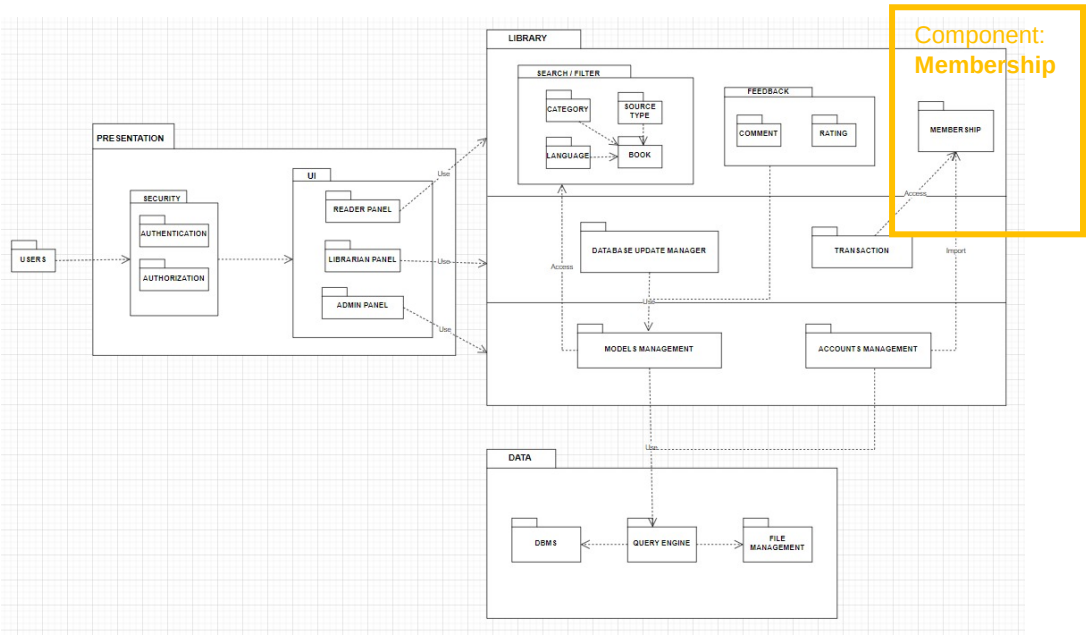


Accompanying class diagram:

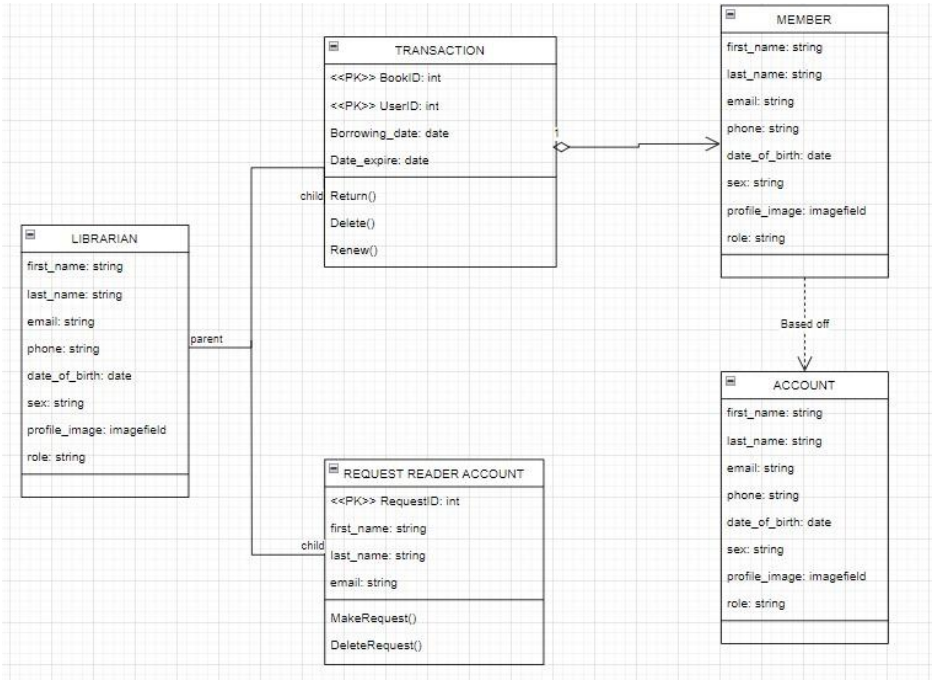


4.9 Component: Membership

A function that allows users to sign up to access more advance functionalities within the web page, it's status is directly connected to the user's account and will be updated accordingly when signing up is completed.

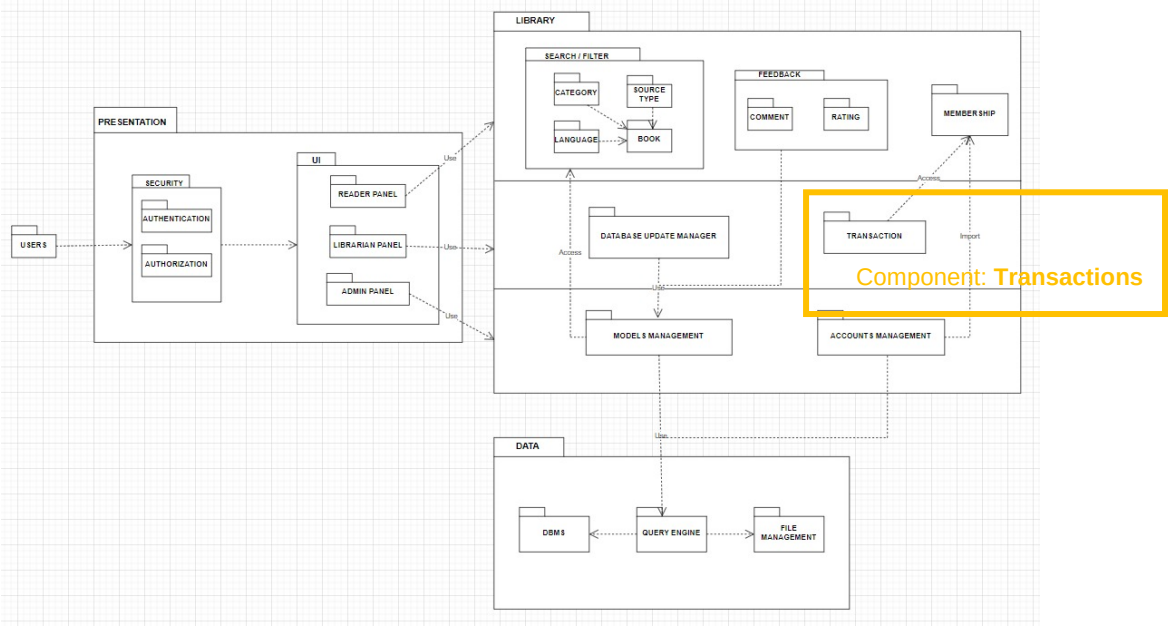


Accompanying class diagram:

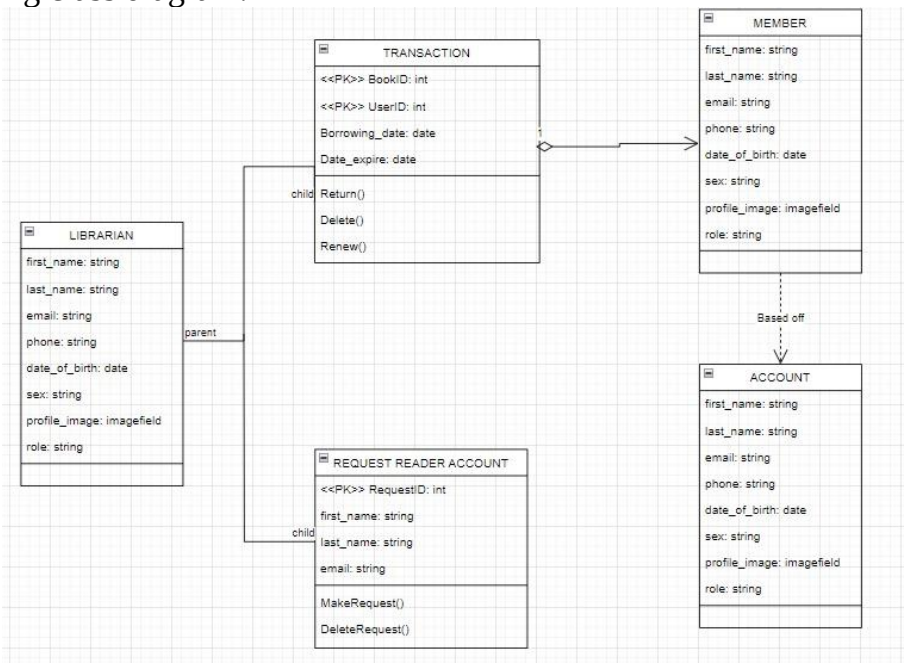


4.10 Component: Transactions

One of two librarians main objective within the system is to manage books within the system and lending them to users online in conjunction with the actual physical copy IRL, this is an advanced function that only unlocks for members of the website with abilities for the librarian to keep track of a loaned book as well as the member to return or extend the return date as they wish.

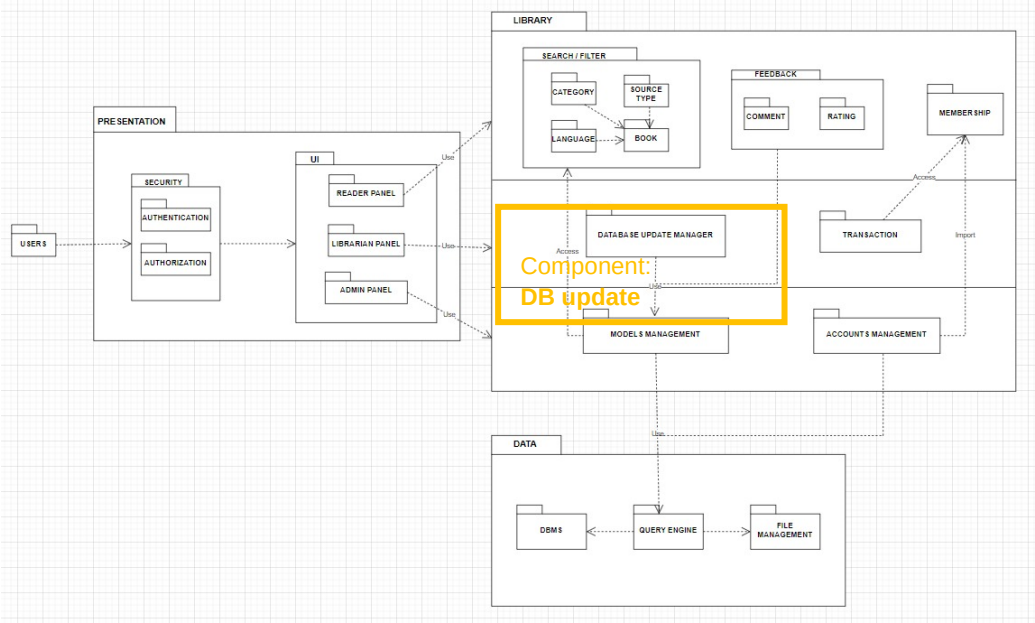


Accompanying class diagram:

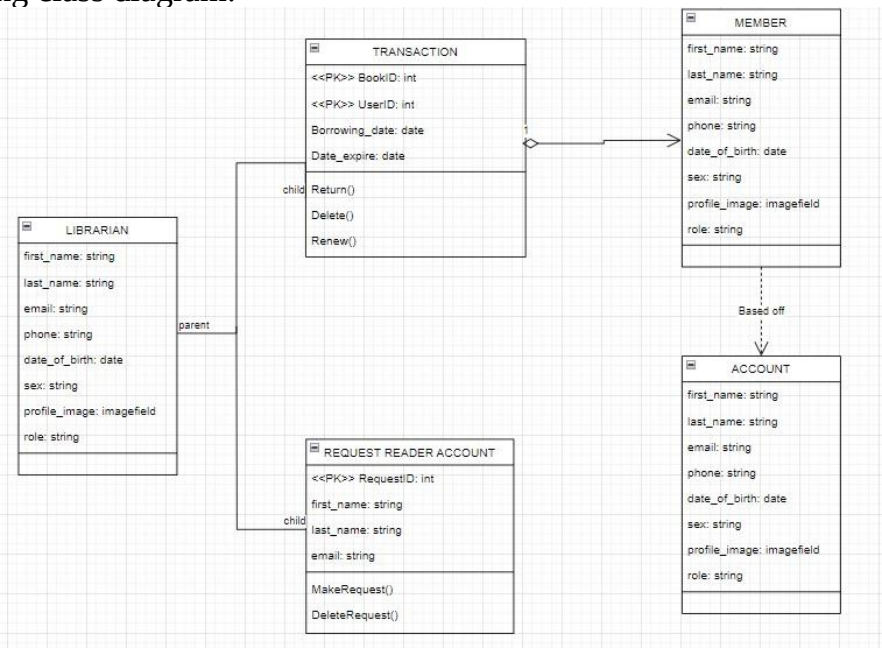


4.11 Component: Database update

To manage books efficiently, Librarians are tasked with keeping track of books, whether they are imported or exported, the need to make changes to the system's database is needed to be regulated accordingly, therefore this functionality provides librarians to observe the state of the library and make request to make changes if necessary.

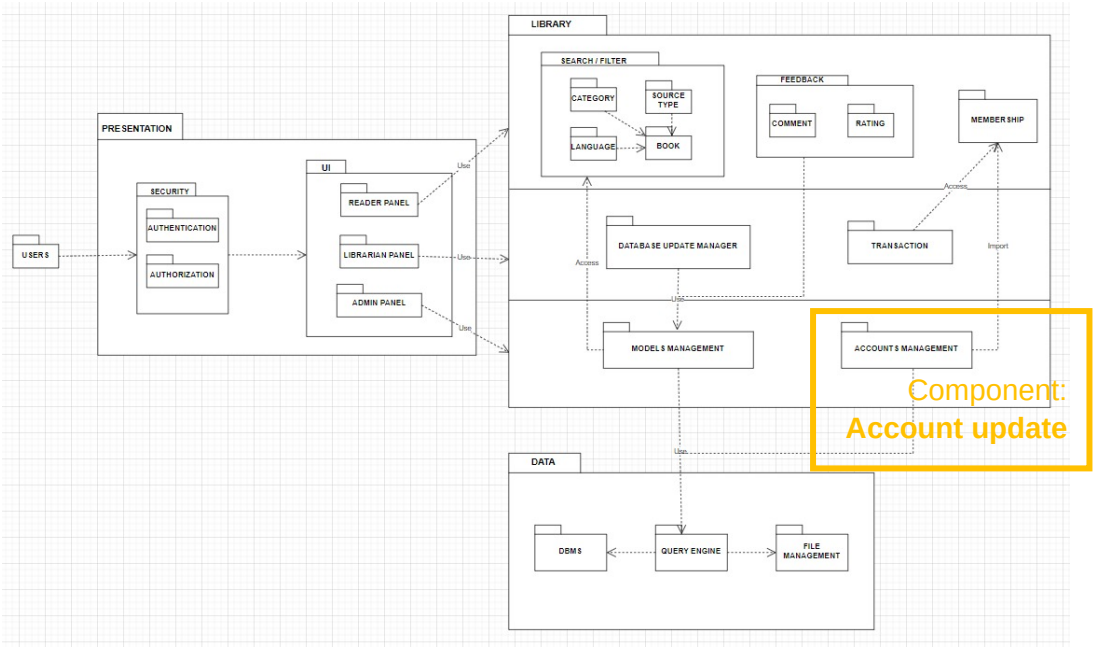


Accompanying class diagram:

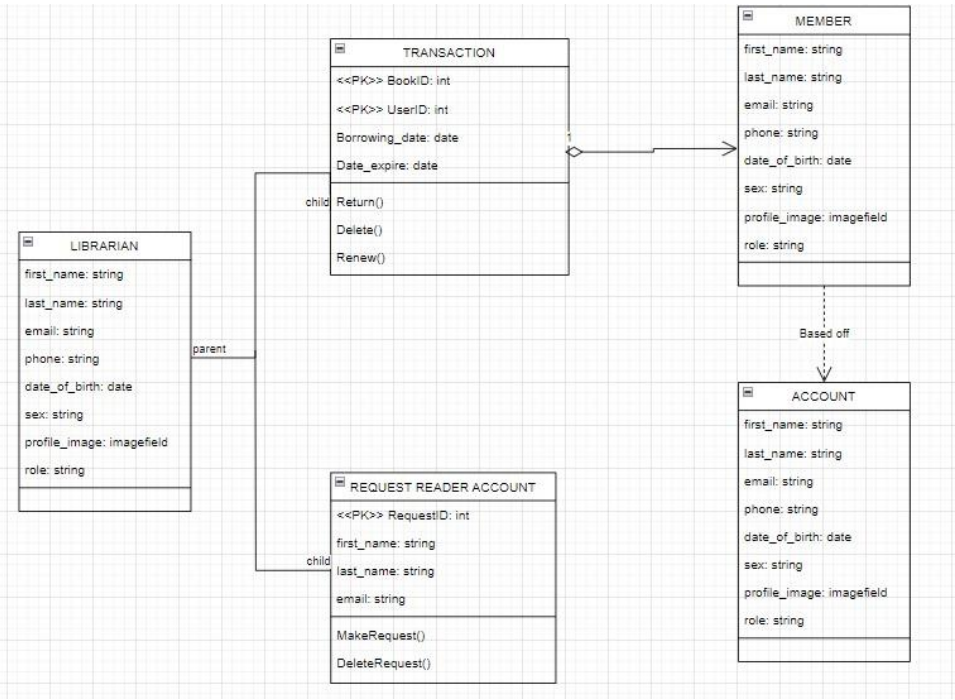


4.12 Component: Account update

Same as books, librarian also help mange users accounts and will make request directly to the Admins interface allowing them to accept and create accounts as per users need.

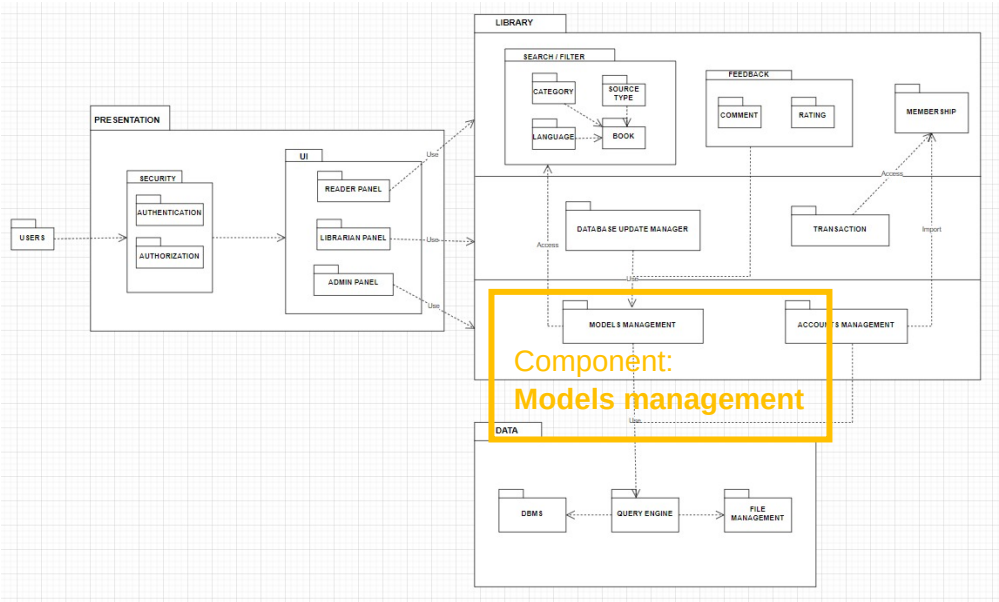


Accompanying class diagram:

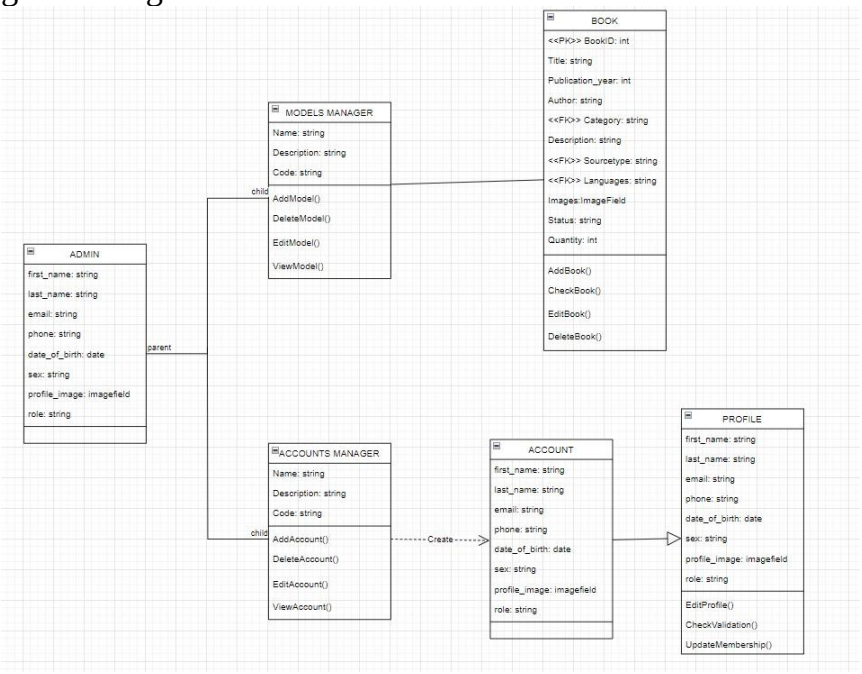


4.13 Component: Models Mangament

A UI specifically coded to easily add or remove many of the functionalities of the web page at a moment's notice. The administrator is provided with many ways to influenced other components of the systems as to make suitable changes to the library to fil the users needs, the changes will be transported to the database and the update will reflect immediately back on the website, accessible for the readers to utilize.

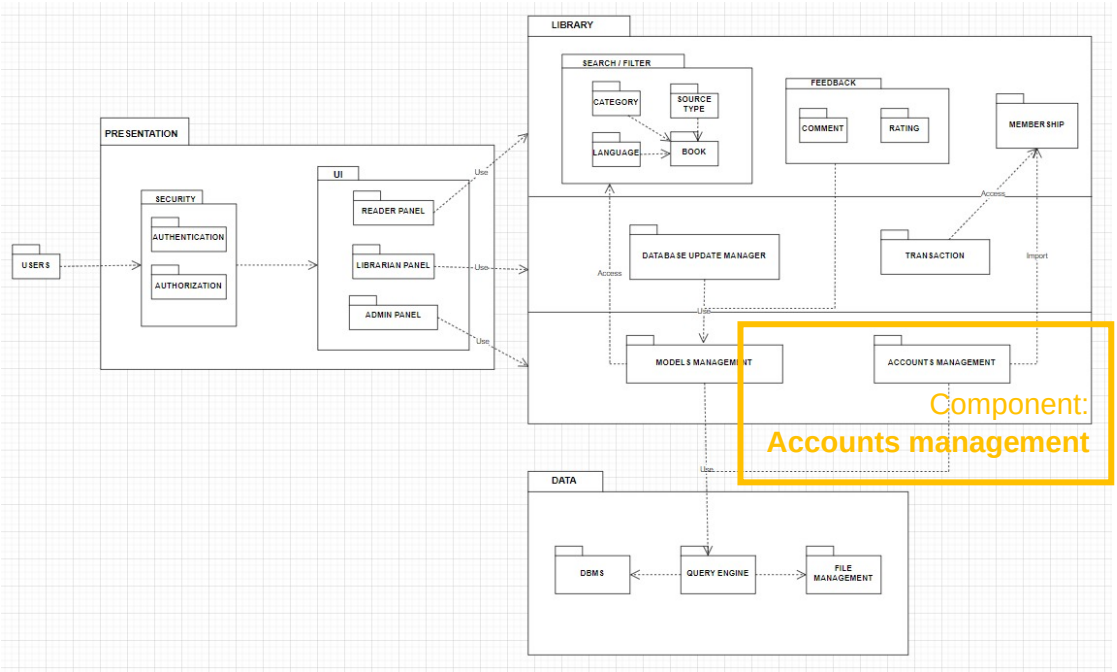


Accompanying class diagram:

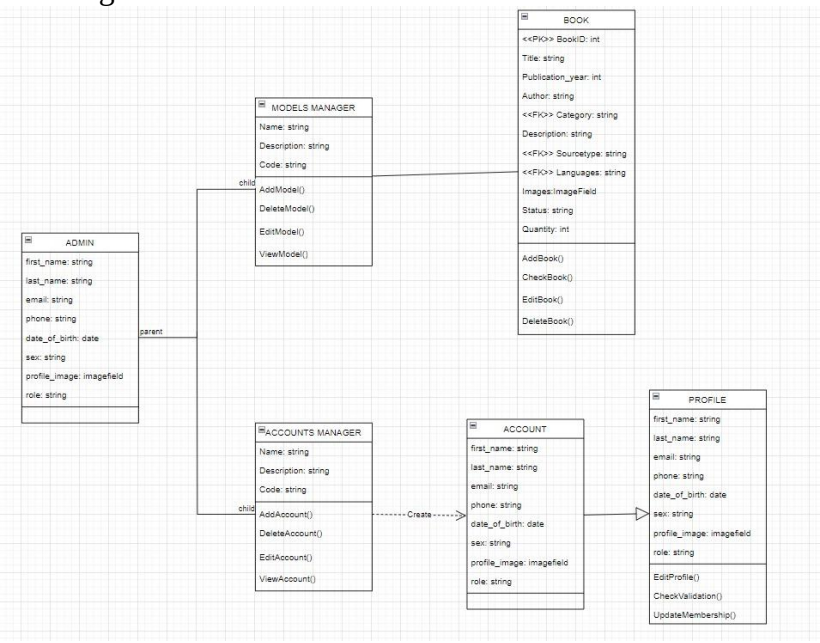


4.14 Component: Accounts Management

Readers of the web page will create an influx of accounts according to the amount user that traffic the website, therefore regulation is necessary and if permitted, the admin will create an account readily available with all of the existing profile created for the users to fill in and complete their accounts and having access to the library many services.

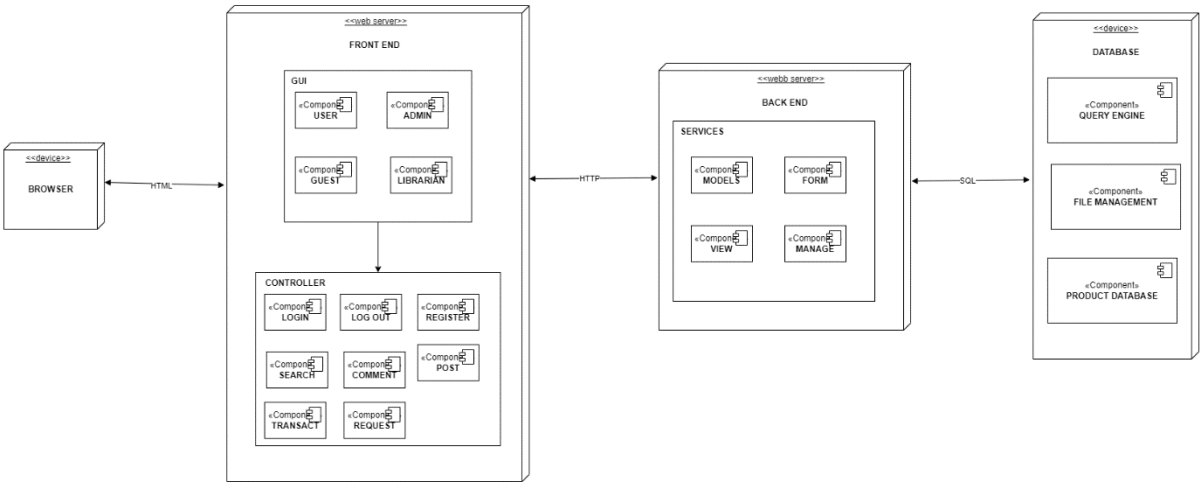


Accompanying class diagram:



5. Deployment

Deployment Diagram:



The Diagram is showcasing the interactions between a user and the system and the inner workings of it.

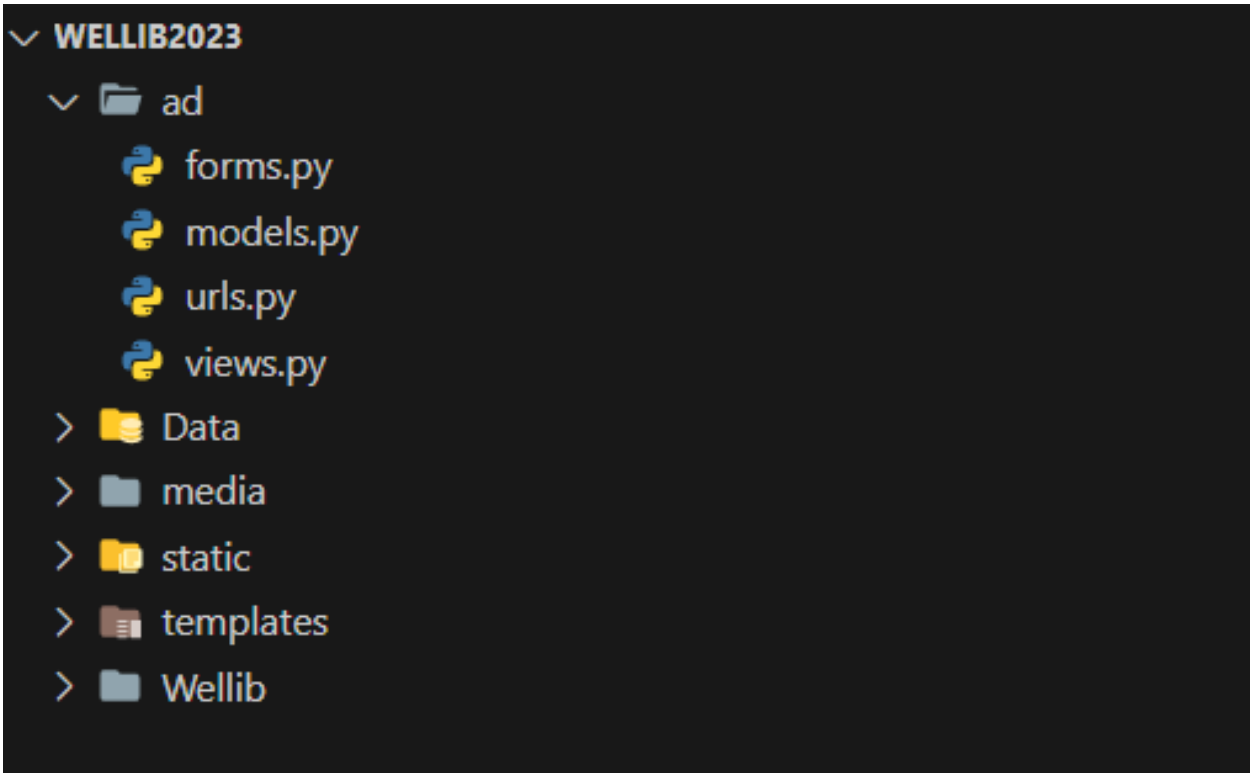
- User can access the website using an internet browser and connect to the website with HTML.
- The Server provides GUI for users to interact with and control based on their purpopes in the system.
- Depending on the action will the controllers at front end of the web server will communicate with the back end via the https protocol and access the predetermined features code specifically for the each component within the system's code.
- When dealing with the main product of the system, the data will be taken and reflected directly from the database and changes upon interaction.

6. Implementation View

The code of the system is organized through many folders of each corresponding purposes.

At the main source code folder includes the basic web framework of Django and is necessary to run the server and it also includes the main folders of the code: Data, Wellib, ad, media, static, templates.

- **Data folder:** includes the main information about the web page as well as the main information of attributes of main models in the system that consists of names, IDs and image files.
- **Wellib folder:** includes basic function that is imported in order for the system code to run
- **ad folder:** includes some basic py_cache and other basic migration install to run the code as well as the main frame to the functionality of the website represented by the main python code:
  - **forms.py:** includes the main template for most of the websites model so as to fill in easily when called upon via the interface, with the code design to fit the template of the component that the admin can create at a moment notice.
  - **models.py:** includes the models of the code and their functionalities .
  - **urls.py:** includes the url connecting the web server and is redirected for each component and their corresponding page.
  - **views.py:** Has the main functions of the code, includes all services and abilities provided by the management system.
  - **other files** are complimentary to the system's code.
- **media folder:** contains the main design and image folders to represent the display on the front end of the database with includes all models and more.
- **static folder:** exist smaller folder of which provides the base framework for the system with static data that make up the web server. These files includes: fonts, icons, images bootstrap, vendors folders, etc…..
- **templates folder:** is the html folder of the system that links main components and services of the library to their respective page and each individual functions of the web page.


WELLIB2023
ad
forms.py
models.py
urls.py
views.py
Data
media
static
templates
Wellib