ReseacrhGate

Discover scientific knowledge and stay connected to the world of science · Discover research · Connect with your scientific community

Name	ID
ساره علاء محمد علي	201900332
محمود حسام الدين رشاد	201900765
ندى ايمن عبدالمجيد سيد	201900893
يسرى عزالدين محمد صالح	201900969

INDEX

INTRODUCTION

PURPOSE:

Many other ResearchGates have come into existence to enhance the services related to scientific researches, allowing authors to only upload there research to receive the opportunity that there researches are seen by many other scientists so they can take benefit from it. This Research Gate also provides the uploading of researches but in addition it also allows other scientists to comment and show their feedback on the research so it can modified and be able to help the scientific field more.

SCOPE:

Scientists.

DEFINITIONS:

User input: input from user UI.
System input: input from database or other function's output.

OVERVIEW:

This web application helps the process of communication between the scientists and help the spread of any new invention all over the world

OVERALL VIEW

PRODUCT PERSPECTIVE:

1- Reduce paper usage.
2- Reduce time for scientists and universities' staff.
3- Reduce overcrowding in libraries.
4- Help the scientists to be up to date to most of the new researches.

USER CHARACTERISTICS:

1 As an Admin, I need to login to the system to be able to:

1.1 - View my profile. So that, I can change password or edit profile.
1.2 Manage authors' as edit, delete them.
1.3 - View authors' profile to be able to get information about them.
1.4 - Logout from the system. So that, my account will be secure.

1 As an author, I need to login to the system to be able to:

1.1 - View my profile. So that, I can change password or edit profile.
1.2 - View other authors' profile so that I can read their research.
1.3 - View other authors' profile so that I can like or dislike their research.
1.4 - View other authors' profile so that I can write a comment on their research.
1.5- Logout from the system. So that, my account will be secure.

CONSTRAINTS:

1- The instructions from the ministry of Ministry of Higher Education, Scientific Research and Technology.
2- System constraints.

SPECIFIC REQUIREMENTS

FUNCTIONAL REQUIREMENTS:

USER REQUIREMENTS

The software has two modules, Admin, Author.

ADMIN

- Can login to his personal account using the ID and password.
- Can view their own profile.
- Can change their current password with new one whenever required.
- Can update their own profile (password, phone, and email).
- Can delete an author if required.
- Can update author info.
- Can view author info.
- Can logout.

Author

- Can login their personal account using email and password.
- Can view their own profile.
- Can upload their own Research.
- Can mention all participating authors in the research.
- Can change their current password with new one whenever required.
- Can search for other authors by e-mail or name.
- Can view other authors.
- Can view Researches on other authors profiles.
- Can comment on other authors researches.
- Can Like other authors researches.
- Can dislike other authors researches.
- Can logout.

System requirements

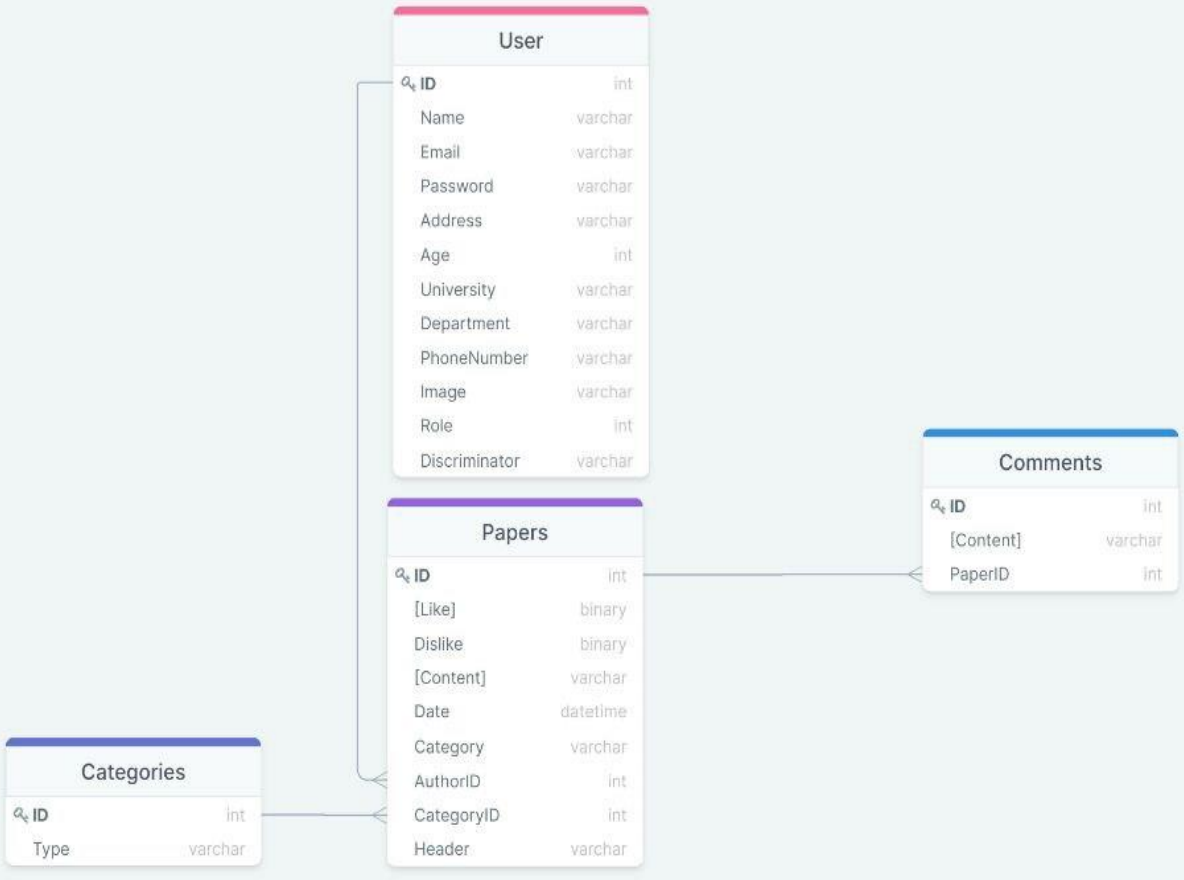
ADMIN

- Can login to his personal account.
  User input: e-mail and password.
  Output: Open home page.
  - **Situation:** Show error message in case of wrong e-mail/ password.
- Can view their own profile.
  System input: admin mail.
  Output: View profile.
- Can change their current password with new one whenever required.
  System input: admin e-mail.
  User input: current and new password.
  Output: Show message which prompts that password changed successfully.
  - **Situation:** Show error message if the new password is the same of current password. So, data will not be changed.
  - **Situation:** Show error message if the new password pattern is wrong. So, data will not be changed.
  - **Situation:** Show error message if the current password not matching with their password. So, data will not be changed.
- Can update their own profile (password, phone, and email).
  System input: admin e-mail.
  User input: password and new data to change.
  Output: Show message which prompts that [data] updated successfully.
  - **Situation:** Show error message if the new data's pattern not matching the current pattern. So, data will not be changed.
  - **Situation:** Show error message if the current password not matching with their password. So, data will not be changed.
- Can delete an author if required.
  System input: admin e-mail.
  User input: admin password and author's e-mail.
  Output: Show message which prompts that author deleted successfully.
  - **Situation**: Show error message if the current admin password not matching with their password. So, author will not be deleted.
- Can update author info (phone, email, address, password).
  System input: admin e-mail.
  User input: admin password and new data to change.
  Output: Show message which prompts that [data] updated successfully.
  - **Situation**: Show error message if the new data's pattern not matching the current pattern. So, data will not be changed.
  - **Situation**: Show error message if the current admin password not matching with their password. So, data will not be changed.
- Can view author info.
  User input: author's e-mail.
  Output: view aothor's profile.
  - **Situation:** Show error message if the author not found.
- Can logout.
  Output: open login page.

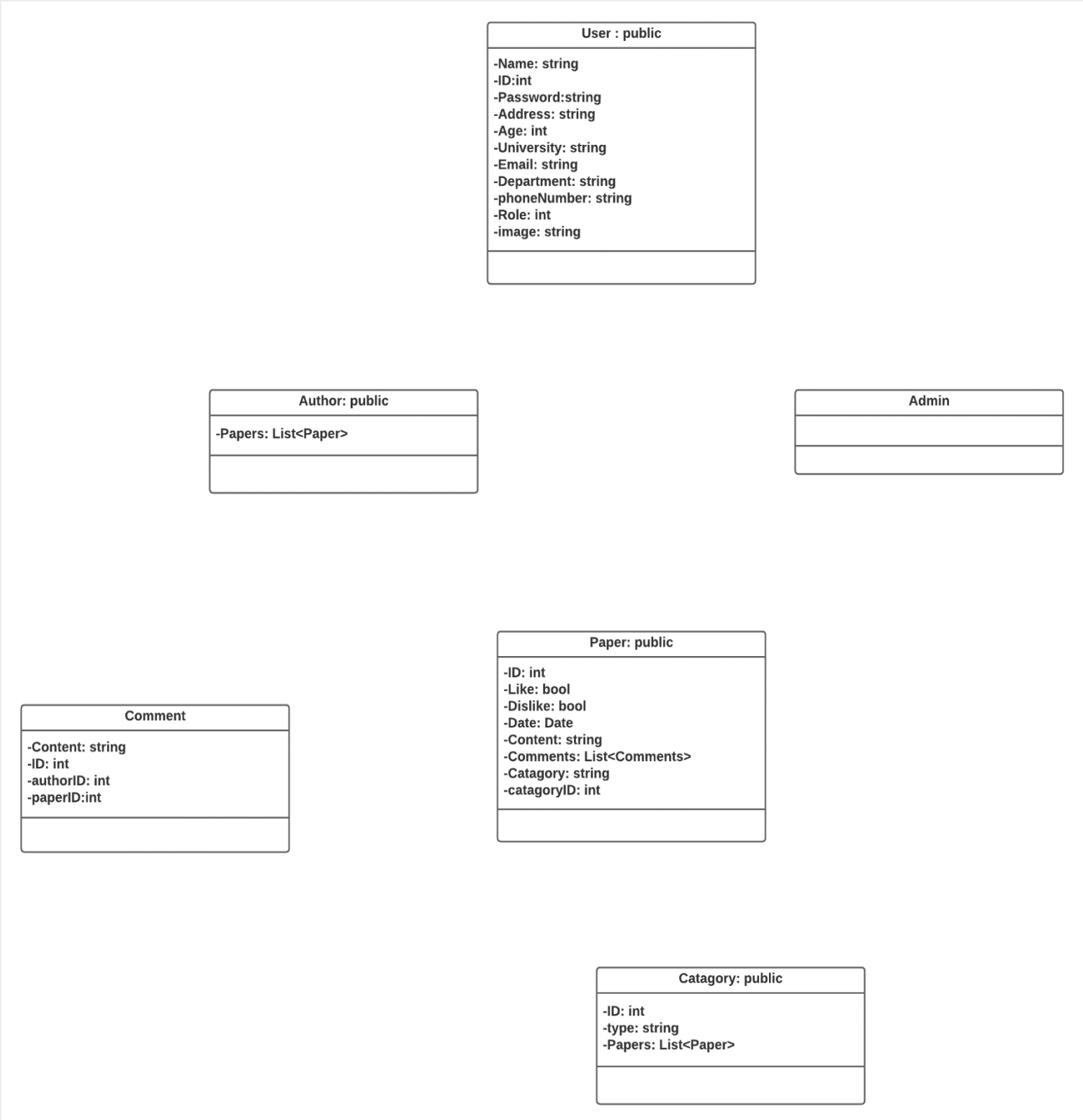
Author

- Can login their personal account using e-mail and password.
  User input: e-mail and password.
  Output: Open home page.
  - **Situation:** Show error message in case of wrong e-mail/ password.
- Can view their own profile.
  System input: author mail
  Output: View profile.
- Can change their current password with new one whenever required.
  System input: author e-mail.
  User input: current and new password.
  Output: Show message which prompts that password changed successfully.
  - **Situation:** Show error message if the new password is the same of current password. So, data will not be changed.
  - **Situation:** Show error message if the new password pattern is wrong. So, data will not be changed.
  - **Situation:** Show error message if the current password not matching with their password. So, data will not be changed.
- Can edit/ update their details (phone, email, DoB, address).
  System input: author e-mail.
  User input: password and new data to change.
  Output: Show message which prompts that [data] updated successfully.
  - **Situation:** Show error message if the new data's pattern not matching the current pattern. So, data will not be changed.
  - **Situation:** Show error message if the current password not matching with their password. So, data will not be changed.
- Can upload Research .
  User input: Research word file or pdf.
  Output: Show message which prompts that [Research] uploaded successfully.
  - **Situation:** Show error message if the file format not accepted.
- Can View other author's profile
  User input: author e-mail/name/university.
  Output: author's profile.
  Situation: Show error message if the e-mail/ name/ university is wrong that author doesn't exist.
- Can logout.
  Output: open login page.

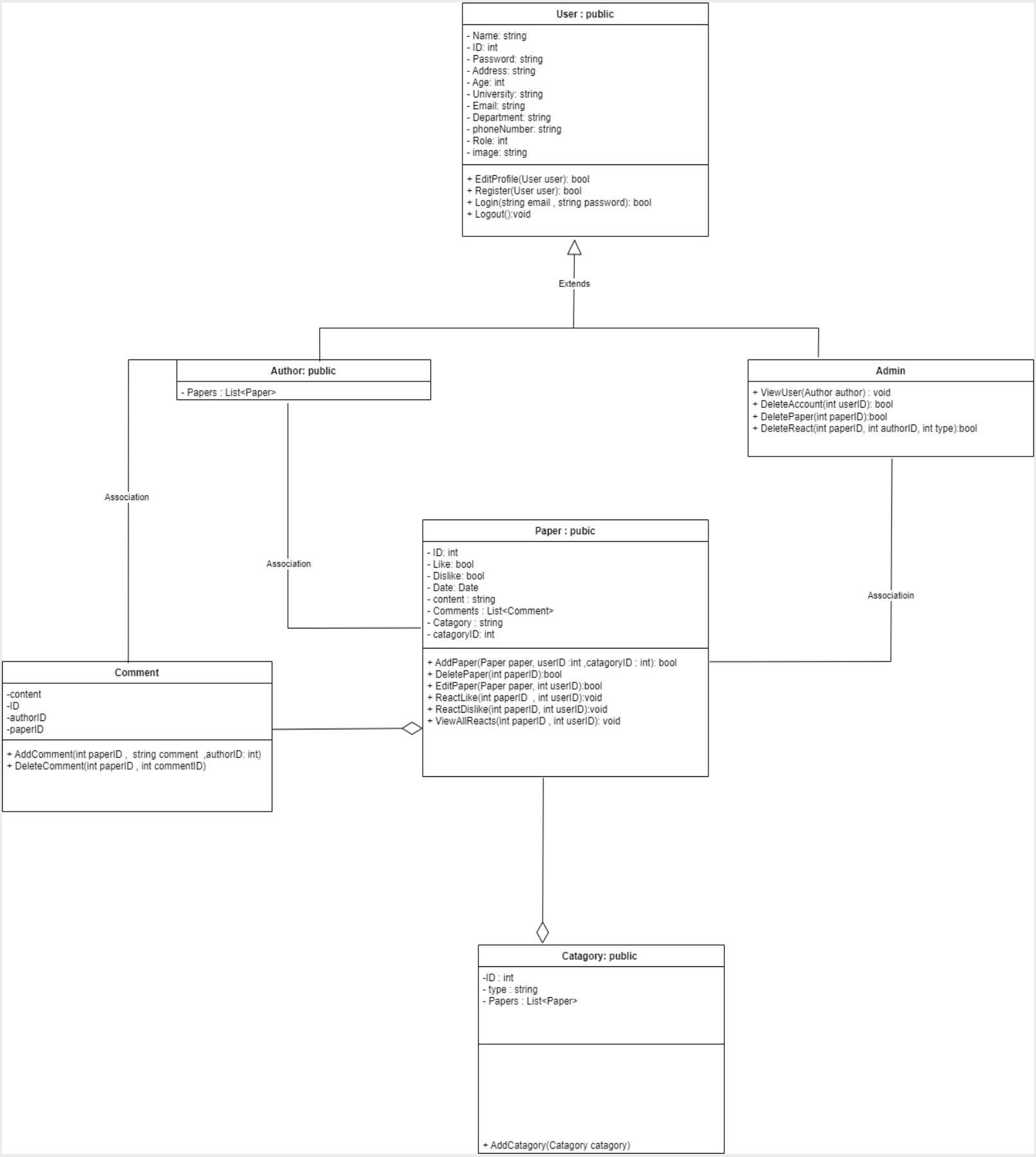
• Database Schema



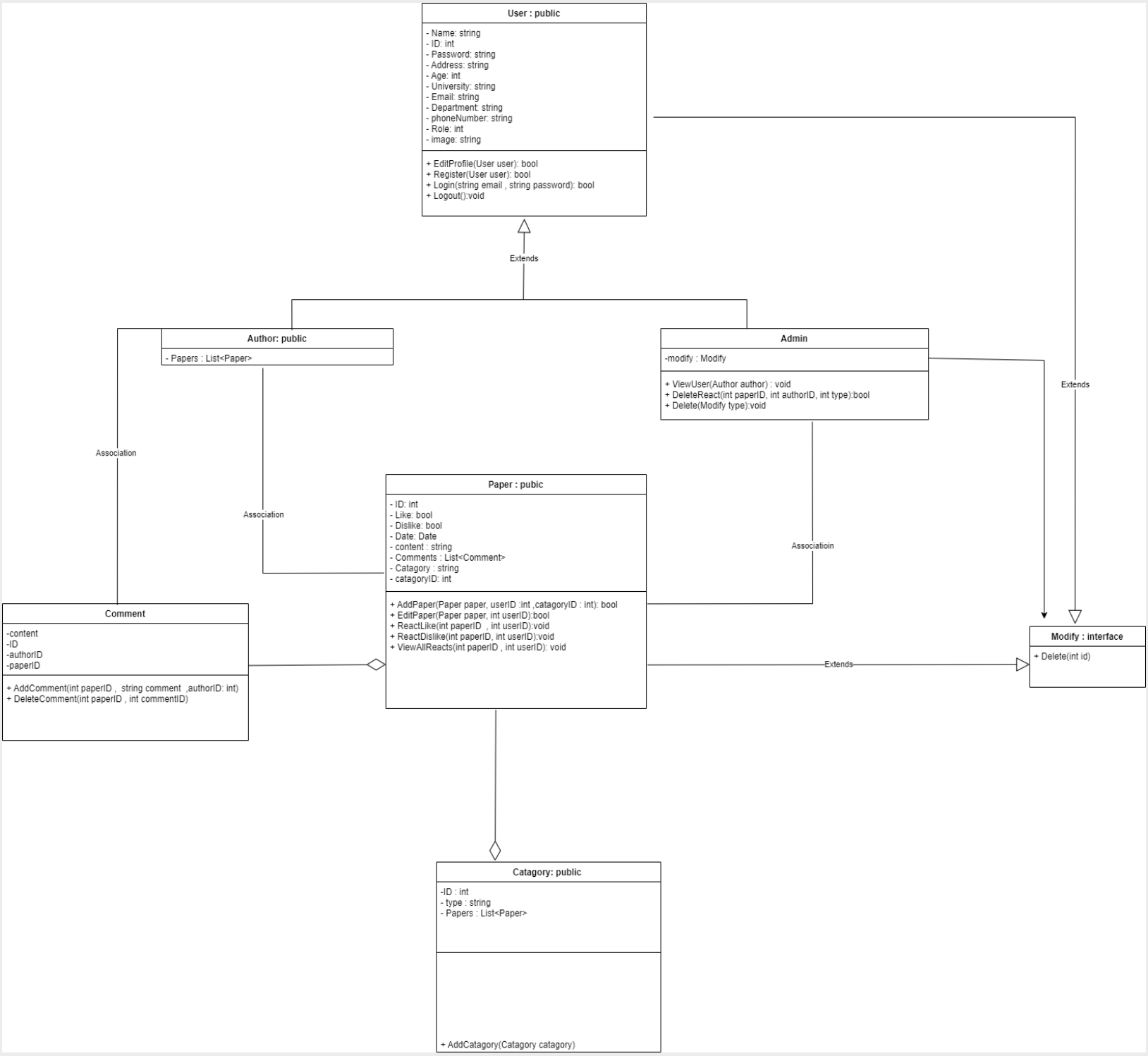
• Initial Class diagram



• Class diagram Intermediate



• Class diagram final version



❖Single Responsibility Principle (SRP)

- **Single Responsibility Principle (SRP): -** is one of five design principles of the SOLID design framework for object-oriented software design. The SRP dictates that classes should have only a single reason to change

- **Benefits: -**

- The class is more reusable, easier to maintain.
- When the class only does “one thing”, its interface The class is easier to understand usually has a small number of methods that are fairly self-explanatory.

❖Open Closed Principle (OCP)

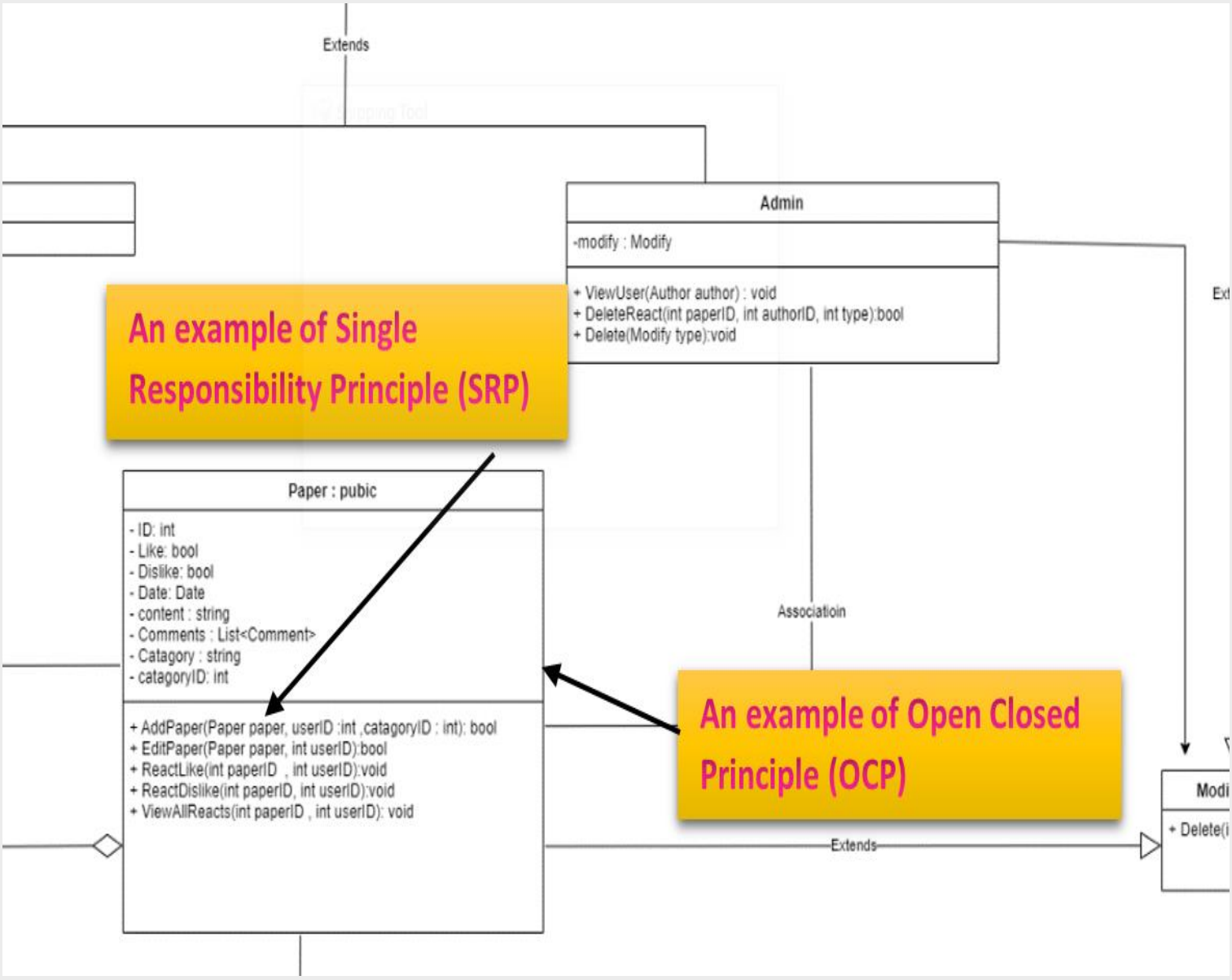
- **Open Closed Principle (OCP): -** In object-oriented programming, the open–closed principle states "software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification"; that is, such an entity can allow its behavior to be extended without modifying its source code.

- **Benefits: -**

The Open Close Principle encourages software developers to design and write code in a fashion that adding new functionality would involve minimal changes to existing code. Most changes will be handled as new methods and new classes.

❖The difference between SRP and OCP

- **SRP** states that a class must have only one reason to change.
- **OCP** states that the class must be closed for modification but open to extension



❖Liskov Substitution Principle (LSP)

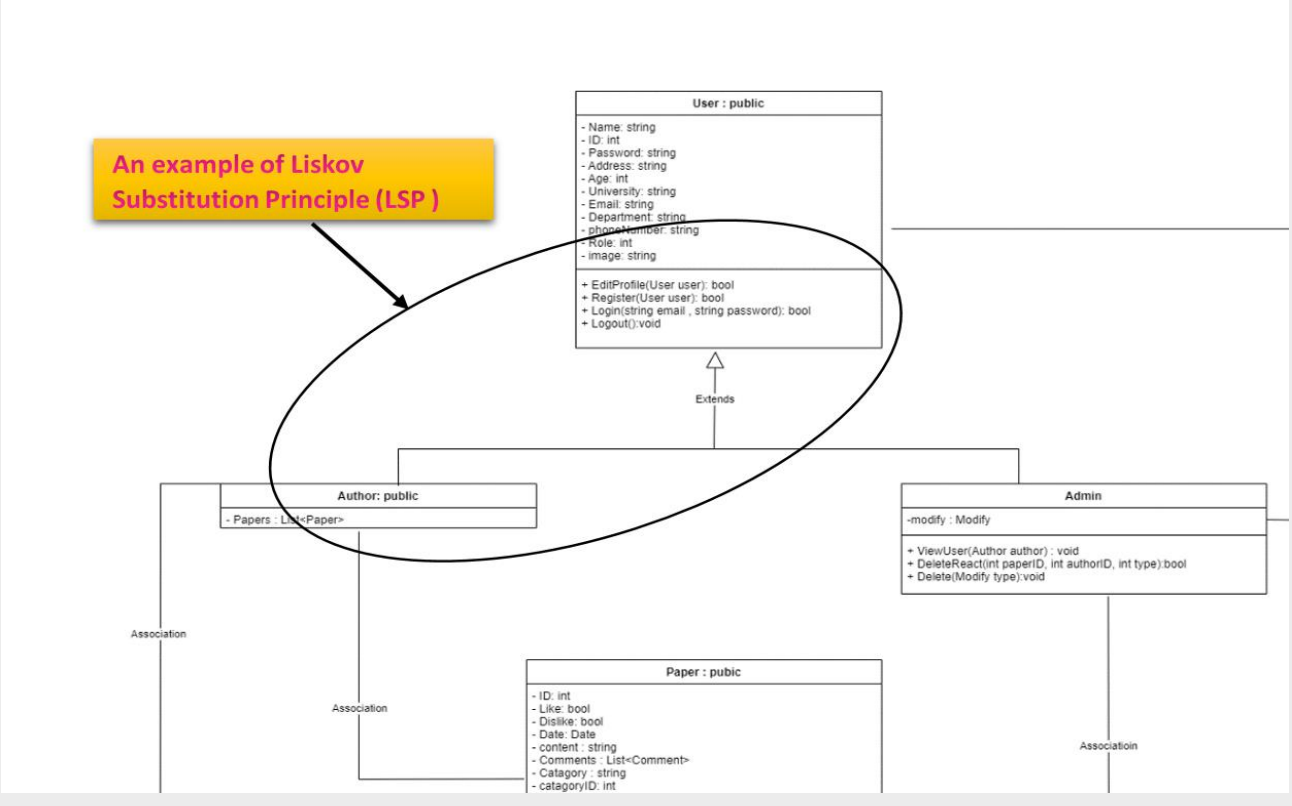
- **Liskov Substitution Principle (LSP):**

oriented design principle that puts some restrictions on the classes that inherit other classes or implement some interfaces. It is one of the five SOLID principles that aim to make the code easier to maintain and extend in the future.

- **Benefits:**

If your code adheres to the Liskov Substitution Principle you have many benefits.

These include: code re-usability, reduced coupling, and easier maintenance.

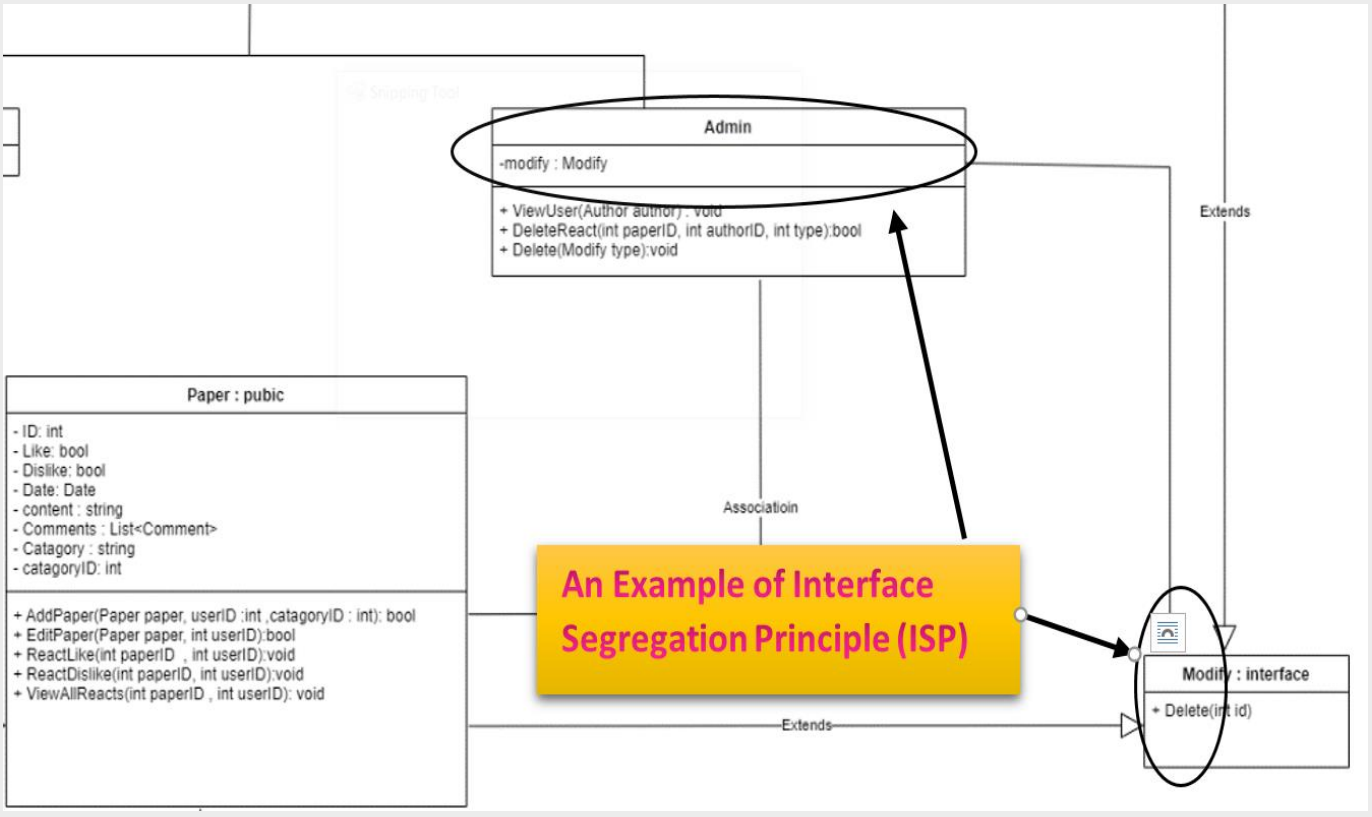


❖Interface segregation principle (ISP)

- **interface segregation principle (ISP): -** states that no code should be forced to depend on methods it does not use. ISP splits interfaces that are very large into smaller and more specific ones so that clients will only have to know about the methods that are of interest to them.

- **Benefits: -**

The Interface Segregation Principle increases the readability and maintainability of our code. We are reducing our class implementation only to required actions without any additional or unnecessary code.

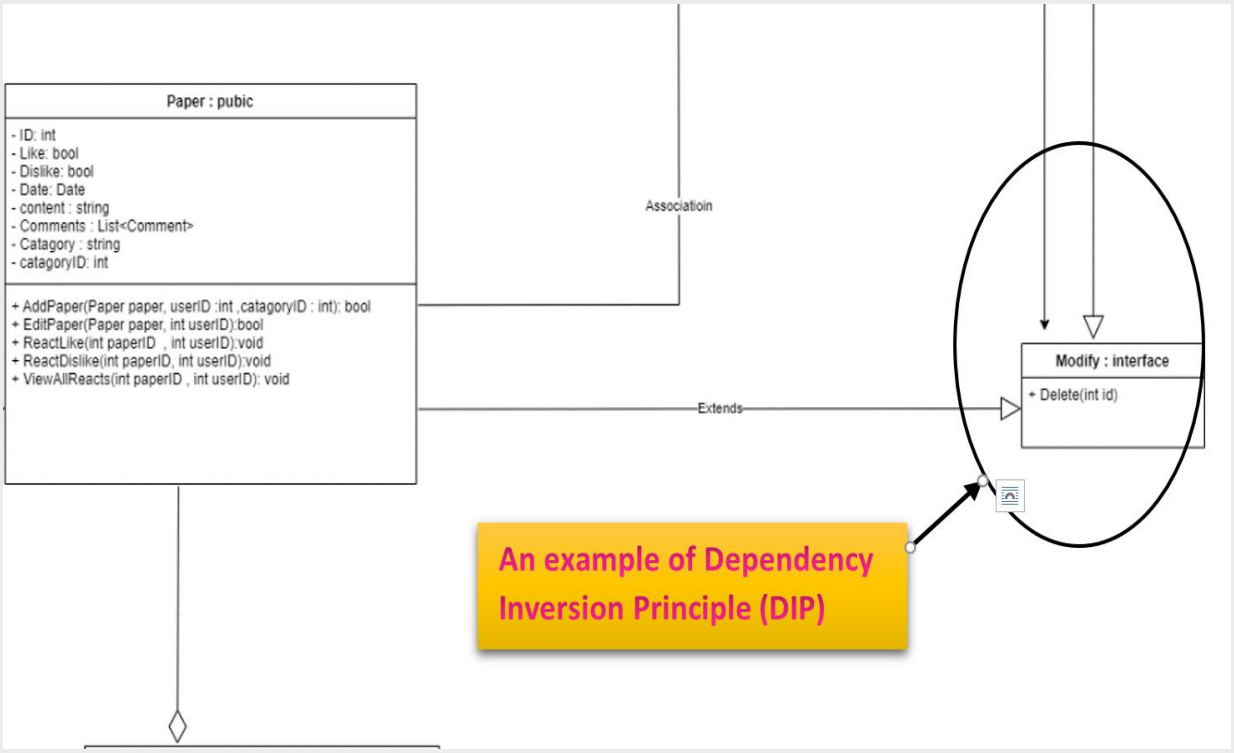


❖Dependency Inversion Principle (DIP)

- **Dependency Inversion Principle (DIP): -** In object-oriented design, the dependency inversion principle is a specific methodology for loosely coupling software modules. When following this principle, the conventional dependency relationships established from high-level, policy-setting modules to low-level, dependency modules are reversed, thus rendering high-level modules independent of the low-level module implementation details.

- **Benefits: -**

Dependency inversion well applied gives flexibility and stability at the level of the entire architecture of your application. It will allow your application to evolve more securely and stable.



❖Design Pattern

- **Factory Method**
  - **Factory Method: -** **Factory method is a creational design pattern which solves the problem of creating product objects without specifying their concrete classes. Factory Method defines a method, which should be used for creating objects instead of direct constructor call (new operator).**

- **Benefits:**
  - **Factory Method Pattern allows the sub-classes to choose the type of objects to create.**
  - **It promotes the loose-coupling by eliminating the need to bind application-specific classes into the code.**