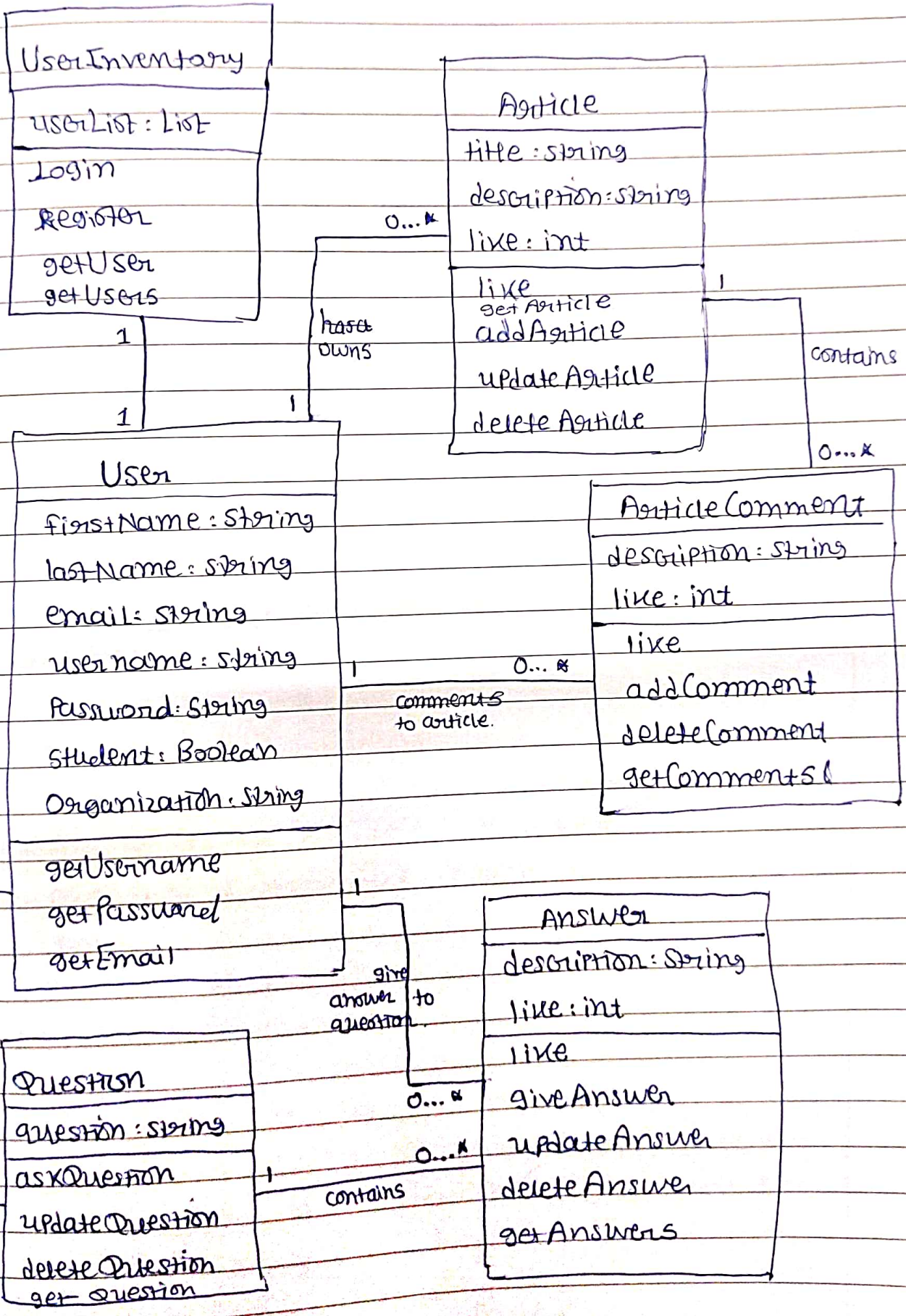
OOSE - Lab 5

CE121

Jwalit Shah

11 | 1 | 2

Class Diagram



Page No.:

Modified Class Diagram & it's Relationship explained

- Here links and associations are used to define the relationship.

Links:- it is a physical or conceptual connection among objects. for example, Swalit works-for Google company. Most links are relate two objects, but some links relate three or more objects.

Association:- it is a description of a group of links with common structure & semantics.

Relationships

1). User 1 —— 1 UserInventory.

- it is one to one mapped association.
- each user is associated with the one UserInventory only.

2). User 1 —— 0..* Question

- it is one to many mapped relationship.
- where each user can ask 0 or multiple questions.
- or each user can post any number of questions

3). User 1 ——— 0..* Article

- it is one to many mapped relationship.
- where each user can post multiple articles to th on the system.

4). User 1 ——— 0..* Article Comment

- it is one to many mapped relationship.
- where each user can comment multiple comments on any articles.

5). User 1 ——— 0..* Answer

- it is one to many mapped relationship.
- where each user can have their multiple posted answers on the questions in the system.

6). Article 1 ——— 0..* Article Comment

- it is one to many mapped relationship.
- where one article can have multiple Article comments associated with each.

7). Question 1 ——— 0..* Answer

- it is one to many mapped relationship.
- where for one question, there can be multiple associated answers.

Basic structure of classes.

```
Public class User
{       Public int userId;
        Public int String firstName;
        Public String lastName;
        Public String email;
        Public String username;
        Public String password;
        Public String organization,
        Public boolean student;

        Public String getEmail();
        Public String getPassword();
        Public String getUsername();

}

Public class UserInventory
{
    Public list userList;

    Public void login ( String username,
                        String password );

    Public void register ( String fname. String lName,
                           String email, String username,
                           String password, String org,
                           boolean student);

    Public User getUser ( int String username);
    Public User[] getUsers ()

}
```

```
Public class Article
{
    Public int ArticleId;
    Public String title, description;
    Public int like=0, userId;

    Public void like();
    Public Article getArticle(int id);
    Public void addArticle(String title,
                           String description);
    Public void updateArticle(String title,
                              String description,
                              int id);
    Public void deleteArticle(int id);
}

Public class ArticleComment
{
    Public int articleCommentId,
               userId,
               articleId;
    Public String description;
    Public int like=0;

    Public void like();
    Public ArticleComment[] getComments
                            (int articleId,
                             int userId);
    Public void addComment(String desc);
    Public void updateComment(String desc, int id);
    Public void deleteComment(int id);
}
```

Page No.:

```
Public class Question
{
    Public int questionId, userId;
    Public String question;

    Public void askQuestion (String question);
    Public void updateQuestion (int id, String question);
    Public void deleteQuestion (int id);
    Public Question getQuestion (int id);

}

Public class Answer
{
    Public int answerId, userId, questionId, like = 0;
    Public String description;

    Public void like();
    Public void giveAnswer (String description);
    Public void updateAnswer (int id, String
                              description);

    Public void deleteAnswer (int id);
    Public Answer getAnswer (int id);
    Public Answer[] getAnswers (int userId,
                                int questionId);

}
```

* Plan for remaining implementation.

- now after designing all the use cases, classes and after structuring all the class, usecase and state diagram, the implementation of creating database and creating POJO classes will be done.

- ~~the after~~ firstly all classes will be abstract and then according to the need we'll gradually move forwards to the implementation of methods.

- then front end templates, controller ~~or~~ will be included.

State Chart Diagram.

* States

- Logged In
- Loggedout
- Waiting for UserInput
- Response to user

Page No.:

* characteristics of states

- Logged In:
  - when user has entered correct credentials then user is allowed to access all the authorized contents.

- Logged out / idle:
  - when user has successfully logged out of the system then user will not be allowed to access all the authorized contents.

- Waiting for user input:
  - when system is having form to fillup any data then state is waiting infinitely for the ~~user to~~ input given by the user.

(Respond)

- Response to user:
  - ~~in~~ after accepting input data, system will goes to the Response to user state from waiting for user input, depending upon the data provided by the user. & in this state system will show error or success message depending on what the data is has entered by the user.

State Diagram.