Title	**Facebook Management System**
Description	Have you ever thought about how the Facebook could be designed? We want to figure out the OOP design for such a project so think about its classes and use the hints below.
Deliverables	1. java code using all the OOP concepts 2. UML class diagram 3. Valid data stored in files for testing on delivery day 4. Documentation that contains a. your own system description b. input and output scenarios
Bonus	GUI
Details	● A FB user does have an email, name, password, gender, and a birthdate. ● The user can write an endless number of posts. ● A post has an ID, users' comments, tagged users and it should have two privacy options (Public or Friends only). ● A comment has an ID and users' replies. ● And each of the posts, comments, and replies do have a number of reactors or likers. ● The user also can do more than one conversation. ● Each conversation has an ID, comprises a number of messages and has a number of participants. ● Each user can have any number of friends. ● The user friends can be a regular friend or a restricted one. **The user can:** 1. Create an account 2. Log in his account if the password provided was correct 3. Like/write posts 4. Tag people to posts 5. Like/write comments on posts 6. Like/Reply to a comment 7. See posts of other user according to the posts' privacy level 8. See the friendship between any two users by using + operator (this should show all the common posts between them)

	9. See the mutual friends between any two users by using & operator 10. Search for/Add friends 11. Send messages 12. A restricted friend of the user can see his public posts only
The suggested Flow	● *Read the files that contain all your system details.* ● Welcome screen that shows two options 1. Register an account 2. Log in an account ● In case of 'register choice' provide the user with the fields required to create an account ● In case of 'logging in' ask the user for his email and password, check for the password if it matches the correct one. ● When logging in successfully show the user a list of actions he can do. 1. Write post 2. Search for a friend 3. See friend's posts . . . ● Let the user choose the action he wants to do and help him with providing the required data to keep your system consistent and robust.
Notes:	1. You should implement all concepts of OOP. 2. Each member **MUST** work on at least one of the required classes besides file processing or GUI. (**Individual marks**) 3. The evaluation will be mainly based on the student's ability to use and apply OOP concepts and the explanation of the code. 4. You must deliver the Class Diagram for the project. 5. You must apply exception handling. 6. Using Files is mandatory (Not Database) 7. Any project must have at least **8 classes** 8. Regarding **files:** • You must have only two functions for file reading and writing.

	• You should read data once at the beginning of your run then do your operations and access the code then save in files at the end of your program.