## Objective:

Design and implement a User Account Management System to demonstrate your understanding of Object-Oriented Programming (OOP) principles.

## Background:

Our application requires a user account management capability to allow users to access the system. As an administrator (end user), one should be able to perform the following tasks.

## Requirements:

1. **User Class:**

* Attributes: **userID, username, password, email, accountStatus** (e.g., Active, Suspended, Deleted).
* Methods:
  + **createUser():** Adds a new user.
  + **deleteUser():** Removes an existing user.
  + **suspendUser():** Temporarily disables a user from logging in.
  + **reactivateUser():** Re-activates a previously suspended user.

1. **Admin Class (inherits from User Class):**

* Additional Methods:
  + **viewAllUsers():** Displays a list of all users and their status.
  + **resetPassword():** Resets the password of a given user.

1. **System Class:**

* Attributes: **usersList** (a list that stores instances of User).
* Methods:
  + **login():** Allows a user (or admin) to log in using their credentials.
  + **logout():** Allows a logged-in user (or admin) to log out.

1. **Functionalities to Implement:**

* Account Creation: A user should be able to create an account.
* Login/Logout: After creating an account, the user should be able to log in and out of the system.
* Admin Capabilities: Admins should be able to view all users, suspend/reactivate accounts, and reset passwords.

## Guidelines:

* Implement encapsulation by making attributes private and using public methods to access or modify them.
* Use polymorphism, where necessary, to handle different user actions.
* Incorporate inheritance and the principle to minimize repetition.
* Please include comments in your code to explain the logic behind your implementation choices.

## Submission:

* Create a console-based application (or a simple GUI if you're comfortable).
* Use any OOP-supported language of your choice (e.g., Python, Java, C#).
* Submit your code files along with a brief document explaining your design decisions and a guide on how to run your application.

## Evaluation Criteria:

* Correctness and completeness of the implemented functionalities.
* Adherence to OOP principles.
* Code clarity, organization, and comments.
* Ability to handle potential errors or exceptional cases.

Good luck! We're interested in seeing your approach to problem-solving and your grasp of object-oriented design principles.