Employee Management System

Objects, classes and relationships

Primary Objects and Classes:

- **User**
  - **Attributes:** userID, name, email, password, position, department, role, contactDetails
  - **Methods:** login(), logout(), register()
- **Employee (Inherits from User)**
  - **Attributes:** employeeID (or use userID)
  - **Methods:** viewDetails(), updateDetails()
- **HRManager (Inherits from User)**
  - **Attributes:** hrManagerID (or use userID)
  - **Methods:** viewEmployeeList(), addEmployee(), updateEmployeeDetails(), deleteEmployee()
- **System**
  - **Attributes**: N/A
  - **Methods:** validateInput(), authenticateUser(), assignRole(), display()
- **Role**
  - **Attributes:** roleName, permissions
  - **Methods:** assignPermissions()
- **Authentication**
  - **Attributes:** userID, password
  - **Methods:** authenticate(), setPassword(), checkPassword()

Relationships:

- **Inheritance:** Both **Employee** and **HRManager** classes inherit from the **User** class because they share common attributes and methods like name, email, and the ability to log in and log out. This demonstrates a "is-a" relationship.
- **Dependency:** The System class has a dependency relationship with the User and Role classes because it uses these classes to authenticate users and assign roles. This is a "uses-a" relationship, indicating that the System class depends on the User and Role classes to function correctly.

- **Composition:** The Role class could be seen as part of a composition relationship with the User class, where Role is a part of User, and User can't exist independently of Role. This represents a "has-a" relationship.