Hotel Reservation System

Project Description

Implement a simple hotel reservation system in **Python**.
The system should simulate the hotel rooms, employees, customers, and reservations in form of object-oriented classes. The system must contain:

- Class **Employee**
  Has employee count and each employee has an id, name, SSN, and password.
  Functions:
  - signIn(…) return true if password is correct.
  - __str__(…) return the employee data in form of string
    "id: 0, name: William, SSN: 123456789, password: Apollo"

- Class **Customer**
  Has customer_count and each customer has an id, name, SSN, and ress (list of references to the customer's old and new reservations).
  Functions:
  - addReservation(…) add a reservation to the array
  - __str__(…) return the customer data in form of string. (And reservation's ID)
    "id: 0, name: Jack, SSN, 78945645, reservations: [0, 2]"

- Class **Room**
  Has room_count and each room has a number, type and curr_res (reference to class Reservation).
  There exist 4 types of rooms (Royal, Presidential, Suite and BB or Bed and Breakfast).
  Functions:
  - isReserved(…) return true if the room is not empty
  - reserve(…) to reserve the room.
  - __str__(…) return the room data in form of string
    "number: 0, type: Royal, reserved: false"
    "number: 2, type: Presidential, reserved: true"
  - free(…) set the reservation to **None** to free the room

- Class **Reservation**
  Has res_count and each reservation has an id, date_created, duration, room (reference to class Room), customer (reference to class Customer) and employee or the creator (reference to class Employee). In addition to the __str__(…) Functions.

Notes: **Encapsulation** should be applied to all attributes (if needed).
Apply inheritance to any given chance (**Customer** and **Employee** may share some attributes….).
Your system must maintain modularity.

When the system starts, it prompts the user to enter his id and password, after signing in, a menu appears with the options below:

1. **View All rooms**

   Print all rooms data (number, type and reservation state).
2. **Add new employee**

   Ask the user for the employee's name, password and create a new employee with the input data then print his new information after creation.
3. **View all employees**

   Print all employees' data (id, name, SSN, and password).
4. **Delete employee by id**

   Ask the user for the employees' id to delete it (he cannot delete himself).
5. **Make new reservation**

   Create a new reservation with customer id, room number and duration.
6. **View all reservations**

   Print all reservations data (id, customer id, room id, date created, duration).
7. **Delete a reservation**

   Ask the user for the reservation id to delete.
8. **Add new customer**

   Create a new customer with id and name then print his new information.
9. **View all customer**

   Print all customers' data (id, name and reservations id).
10. **Delete customer by id**

    Ask the user for the customer id to delete.
11. **Sign Out**

    Sign out and back go to the sign in.
12. **Exit and Save**

    Exist the system.

When the system starts, it calls the function load which load the data from a saved file.
When the user exits the system, it calls the function save which saves or overwrite the current saved file.
You are free in applying any technique for saving and loading the saved file.

Protect options 2, 3, 4 and 7 with a master key (different from the employee password)

BONUS: apply hashing for all passwords and master key (no secret keys should be saved as plain text)

Your program should handle all possible exceptions.

In case of exceptions the system should **raise** a custom exception.

NameDuplicateError if the employee's or customer's name is already taken.
EmployeeNotFoundError if the employee is not found.
CustomerNotFoundError if the customer is not found
RoomNotFoundError if the room is not found.
ReservationNotFoundError if the reservation is not found
RoomNotAvailableError if the room is not available (reserved).
AccessDeniedError if the employee's password or master password is incorrect.

Test Script:

Create a test script with:

- List of 10 rooms (initialized with 1 royal, 2 presidential, 3 suite and 4 BB).
- List of employees (initialized with 1 employee).
- List customers (initialized empty).
- List of reservations (initialized empty).
- Menu loop

Perform a simple test:

1. Sign in with employee account
2. Create new customer with any name
3. Make new reservation with previously created customer id (printed on screen)
4. Create a new employee
5. Sign out
6. Sign in with another account
7. Delete previous reservation
8. Add new employee
9. Delete old employee
10. Exit the system
11. Start the system again
12. Display all rooms
13. Display all employees
14. Display all customers
15. Display all reservations

Hotel Reservation System

Rubric:

Points	Percentage
Classes Structure	25%
Encapsulation	7.5%
Inheritance	7.5%
Functionalities	30%
Extra functionalities	10%
Exceptions	10%
Modularity	5%
Clean and readable code	5%
Total	**100%**
BONUS	15%