EAST WEST UNIVERSITY

Hotel Management System

Submitted By

Rifat Hosain Fahim (2024-1-60-054)

Marium Islam Munni (2023-3-60-402)

Mst Samina Akter Tishi (2024-1-60-002)

Mayesha Ahmed (2024-1-60-185)

Swarna Ghosh (2023-3-60-304)

Submitted To

Md Sabbir Hossain

Lecturer, CSE Department

Introduction

The Hotel Management System (HMS) is designed to simplify the operations involved in managing a hotel. It provides an integrated solution for handling various aspects such as room bookings, customer management, billing, and staff management. The system leverages Java programming to deliver a robust, efficient, and user-friendly interface that ensures smooth operations and enhanced customer satisfaction. By automating routine tasks, the system reduces manual effort, minimizes errors, and allows hotel staff to focus more on delivering quality service.

Features of the System

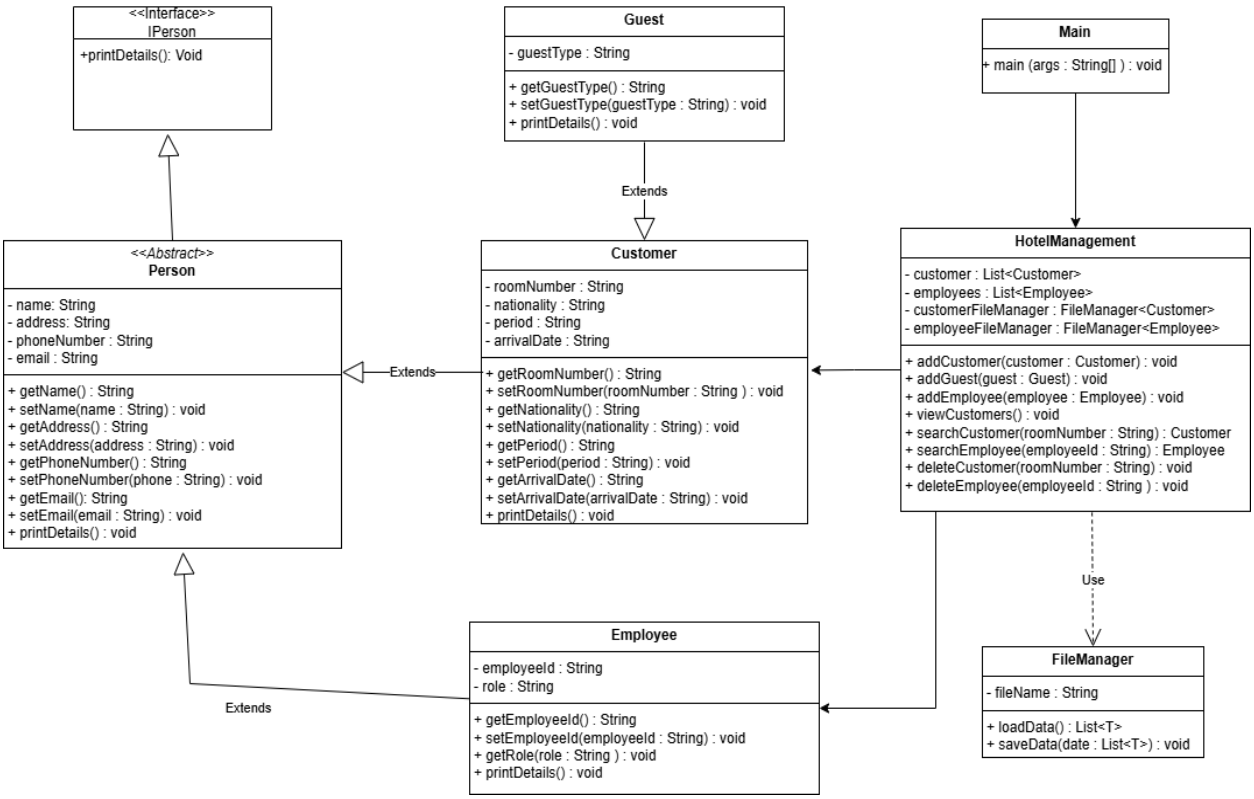
1. Room Management: Allows the addition, updating, and deletion of room details. Keeps track of room availability.

2. Customer Management: Facilitates customer check-in and check-out processes. Stores customer details for easy retrieval.

4. Billing System: Automatically generates bills based on customer usage and services availed. Ensures accurate and timely billing.

5. Employee Management: Maintains a database of employee details, including roles, and contact information.

UML Diagram


<<Interface>>
IPerson
+printDetails(): Void
Guest
- guestType : String
+ getGuestType() : String
+ setGuestType(guestType : String) : void
+ printDetails() : void
Main
+ main (args : String[]) : void
Extends
<<Abstract>>
Person
- name: String
- address: String
- phoneNumber : String
- email : String
+ getName() : String
+ setName(name : String) : void
+ getAddress() : String
+ setAddress(address : String) : void
+ getPhoneNumber() : String
+ setPhoneNumber(phone : String) : void
+ getEmail(): String
+ setEmail(email : String) : void
+ printDetails() : void
Extends
Customer
- roomNumber : String
- nationality : String
- period : String
- arrivalDate : String
+ getRoomNumber() : String
+ setRoomNumber(roomNumber : String) : void
+ getNationality() : String
+ setNationality(nationality : String) : void
+ getPeriod() : String
+ setPeriod(period : String) : void
+ getArrivalDate() : String
+ setArrivalDate(arrivalDate : String) : void
+ printDetails() : void
HotelManagement
- customer : List<Customer>
- employees : List<Employee>
- customerFileManager : FileManager<Customer>
- employeeFileManager : FileManager<Employee>
+ addCustomer(customer : Customer) : void
+ addGuest(guest : Guest) : void
+ addEmployee(employee : Employee) : void
+ viewCustomers() : void
+ searchCustomer(roomNumber : String) : Customer
+ searchEmployee(employeeId : String) : Employee
+ deleteCustomer(roomNumber : String) : void
+ deleteEmployee(employeeId : String) : void
Use
Employee
- employeeId : String
- role : String
+ getEmployeeId() : String
+ setEmployeeId(employeeId : String) : void
+ getRole(role : String) : void
+ printDetails() : void
Extends
FileManager
- fileName : String
+ loadData() : List<T>
+ saveData(date : List<T>) : void

Main.java

Analysis:

Customer Management:

- **CheckIn Customer:** Registers a new customer in the hotel system, capturing details like name, address, phone number, email, nationality, stay period, and arrival date.
- **Add Guest:** Allows adding guests associated with an existing room reservation. This might be useful for scenarios where multiple people are staying in the same room.
- **View Customers:** Provides a facility to view the list of registered customers.
- **Search Customer:** Enables searching for a specific customer by room number.
- **CheckOut Customer:** Processes a customer's check-out, potentially including finalizing charges, updating room availability, and handling any other necessary steps.

Employee Management:

- **Add Employee:** Creates a new employee record in the system, storing essential details like employee ID, name, contact information, and role (e.g., receptionist, manager, housekeeping).
- **View Employees:** Provides a way to view a list of all employees currently registered in the system.
- **Delete Employee:** Enables removing an employee's record from the system.
- **User Menu for Interaction:** The system presents a user-friendly menu with clear options for managing customers, employees, and exiting the program. This allows users to interact with the system and perform desired actions in an intuitive way.

IPerson.java

Analysis

Iperson is an interface and we took an abstract method named printDetails to print person details.

Person.java

Analysis

- **Person (abstract):** Base class for Customer and Employee, holding common attributes like name, address, phone number, and email.
- **Customer:** Extends Person with specific attributes like name, address, phone number, email.
- **Employee:** Extends Person with attributes like employee ID and role.

Employee .java

Analysis

- **Person (abstract)**: Base class for Customer and Employee, holding common attributes like name, address, phone number, and email.
- **Employee**: Extends Person with attributes like employee ID and role.
- **HotelManagement:** Core class managing customer and employee data, including methods for adding, searching, viewing, and deleting records.

Customer.Java

public class Customer extends Person: This declares a public class named Customer that extends the Person class.

public Customer(String roomNumber, String name, String address, String phoneNumber, String email, String nationality, String period, String arrivalDate): This is the constructor for the Customer class.

This initializes the roomNumber, nationality, period, and arrivalDate fields of the Customer object.

Getter and Setter Methods:

- The class provides getter and setter methods for the roomNumber, nationality, period, and arrivalDate fields, allowing access and modification of these properties.

printDetails() Method:

The method prints the specific details of the Customer, which are the roomNumber, nationality, period, and arrivalDate.

Guest.java

public class Guest extends Customer: This line declares a public class named Guest that extends the Customerclass.

public Guest(String roomNumber, String name, String address, String phoneNumber, String email, String nationality, String period, String arrivalDate, String guestType): This is the constructor for the Guest class.

The constructor initializes the guestType field of the Guest object.

Getter and Setter Methods:

- The class provides getter and setter methods for the guestType field, allowing access and modification of this property.

public void printDetails(): This method overrides the **printDetails()** method inherited from the Customer class.
The method prints the specific detail of the Guest, which is the guestType.

HotelManagement.Java

Data Management:

- Maintains lists of Customer and Employee objects.
- Uses a FileManager class to load and save customer and employee data to/from files.

Customer Operations:

- **AddCustomer():** Adds a new Customer object to the list.
- **addGuest()**: Adds a Guest object (which likely extends Customer) to the list.
- **viewCustomers():** Displays information about all registered customers.

- **searchCustomer()**: Searches for a specific customer by room number.
- **deleteCustomer()**: Removes a customer from the list.

Employee Operations:

- **addEmployee()**: Adds a new Employee object to the list.
- **viewEmployees()**: Displays information about all registered employees.
- **searchEmployee()**: Searches for a specific employee by employee ID.
- **deleteEmployee()**: Removes an employee from the list.

FileManager.Java

Analysis

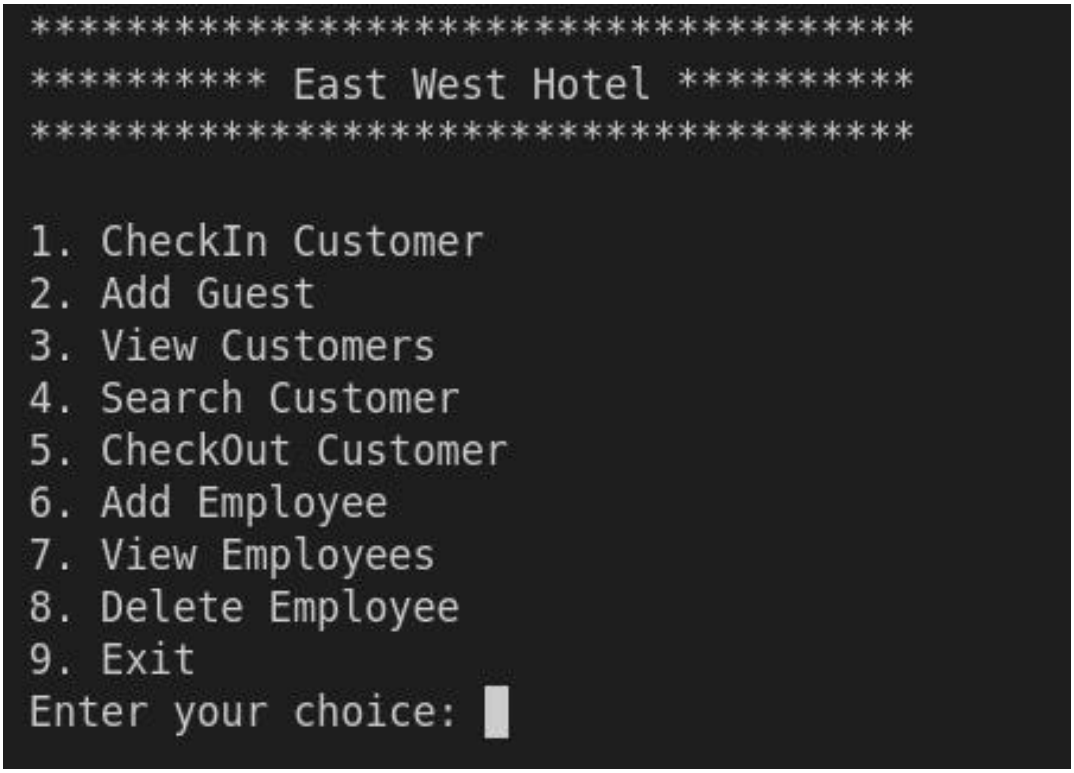
- **Generic Type:** The class uses a generic type T to allow storing various data types in the files.

- **File Name:** It maintains a fileName attribute to specify the file used for data storage.

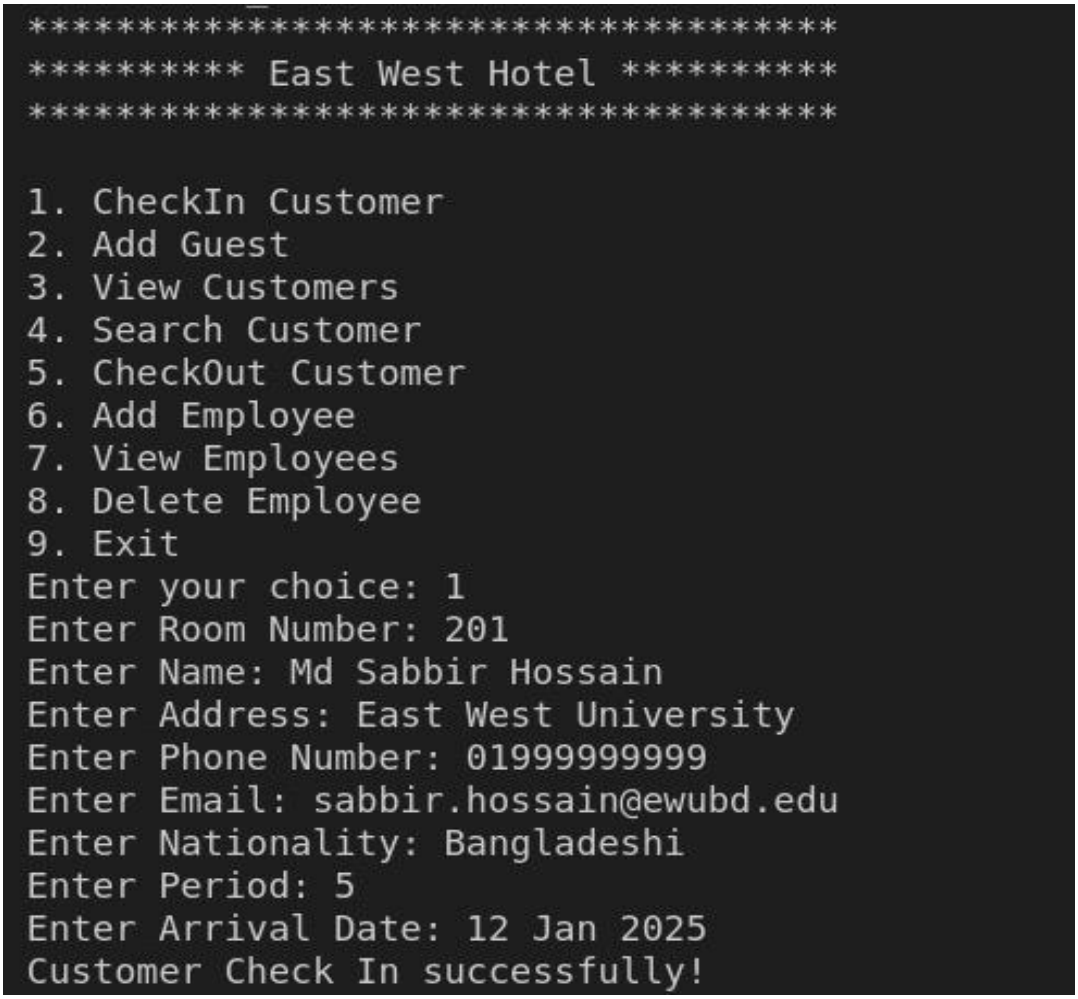
- **Loading Data:** The **loadData()** method attempts to read data from the specified file using an ObjectInputStream. If successful, it returns the loaded data as a List<T>. If there are any exceptions during loading, it returns an empty ArrayList.

- **Saving Data:** The **saveData()** method takes a list of data objects (List<T>) and writes them to the specified file using an ObjectOutputStream. Any exceptions encountered during saving are caught and a message is printed to the console.

Output:

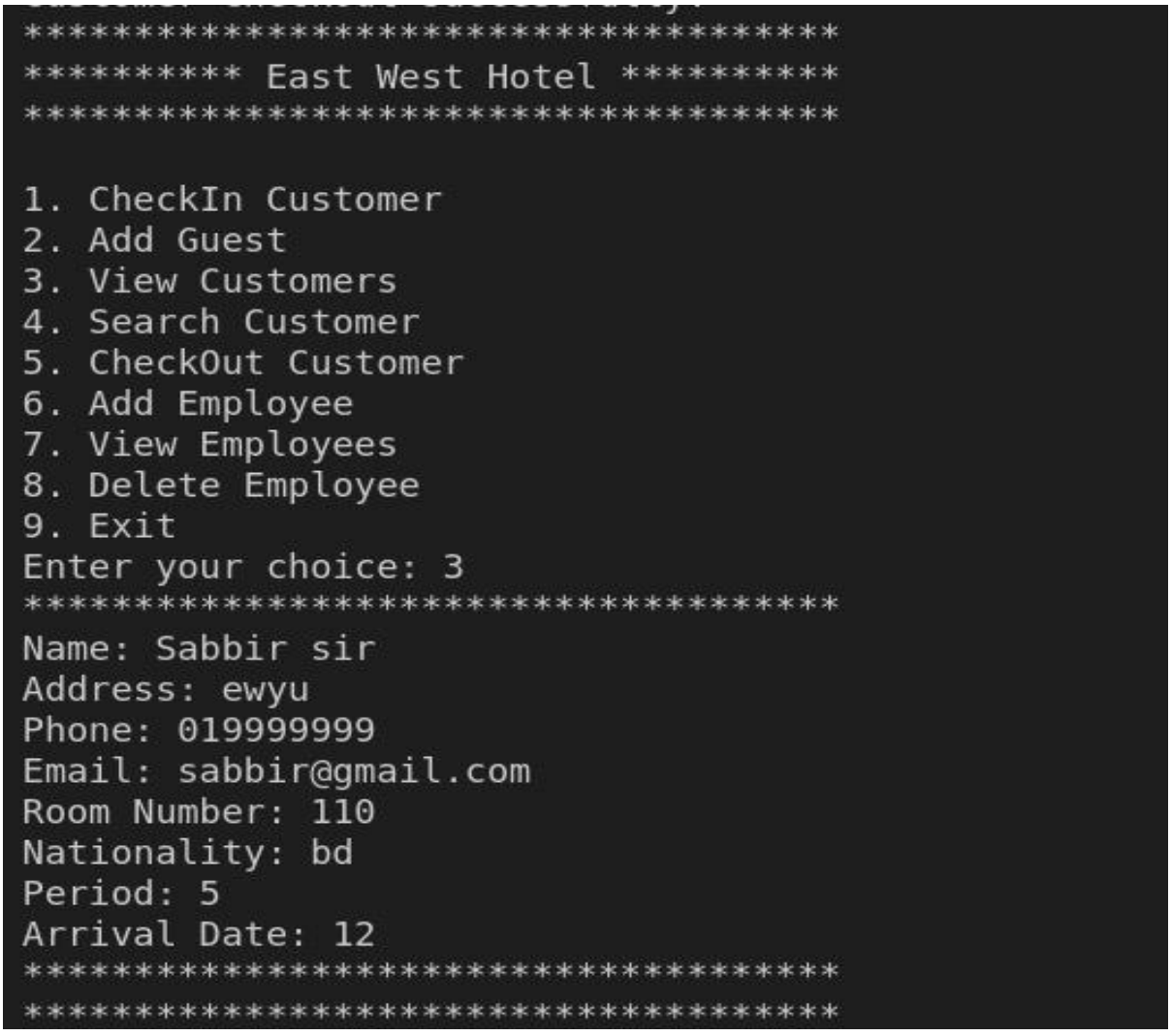


This is the output of this code and when you run the program it will be shown these above options and another option which is “enter your choice”.

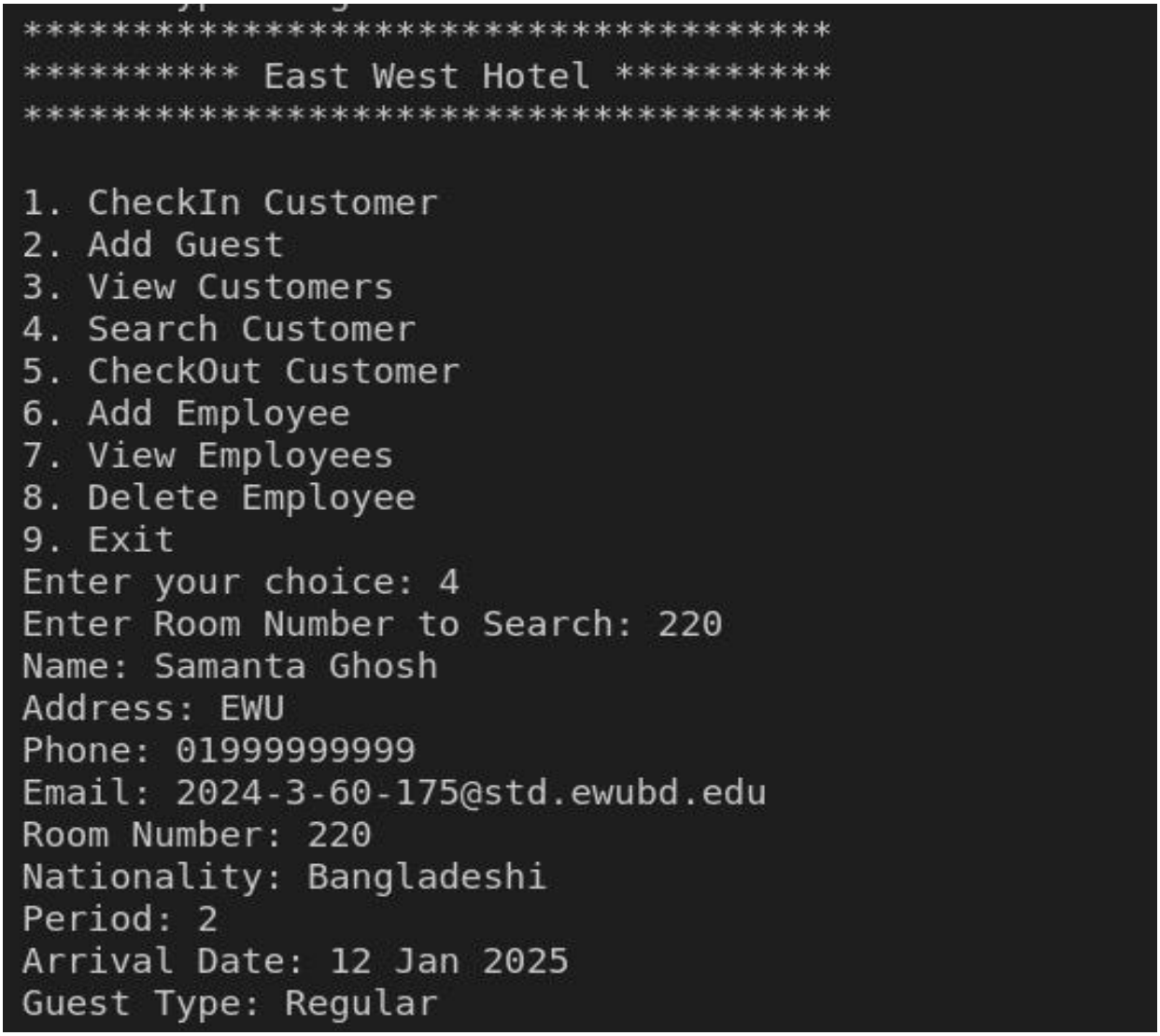


If a person clicks or choose option 1 which is CheckIn Customer then he/she will be shown few options and have to enter the details which are shown in the screen.

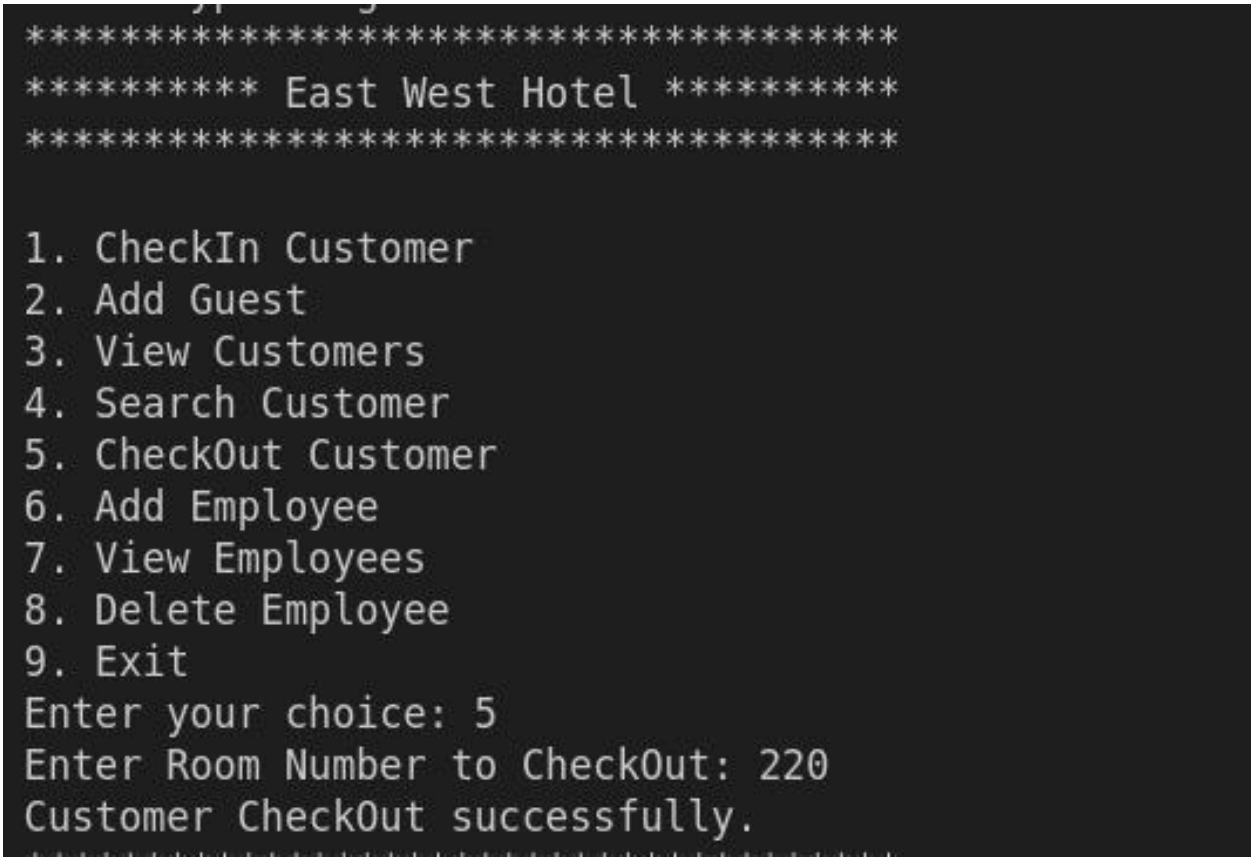
Option 2 indicates add guest where one will have to give all the details about guest and also guest type.



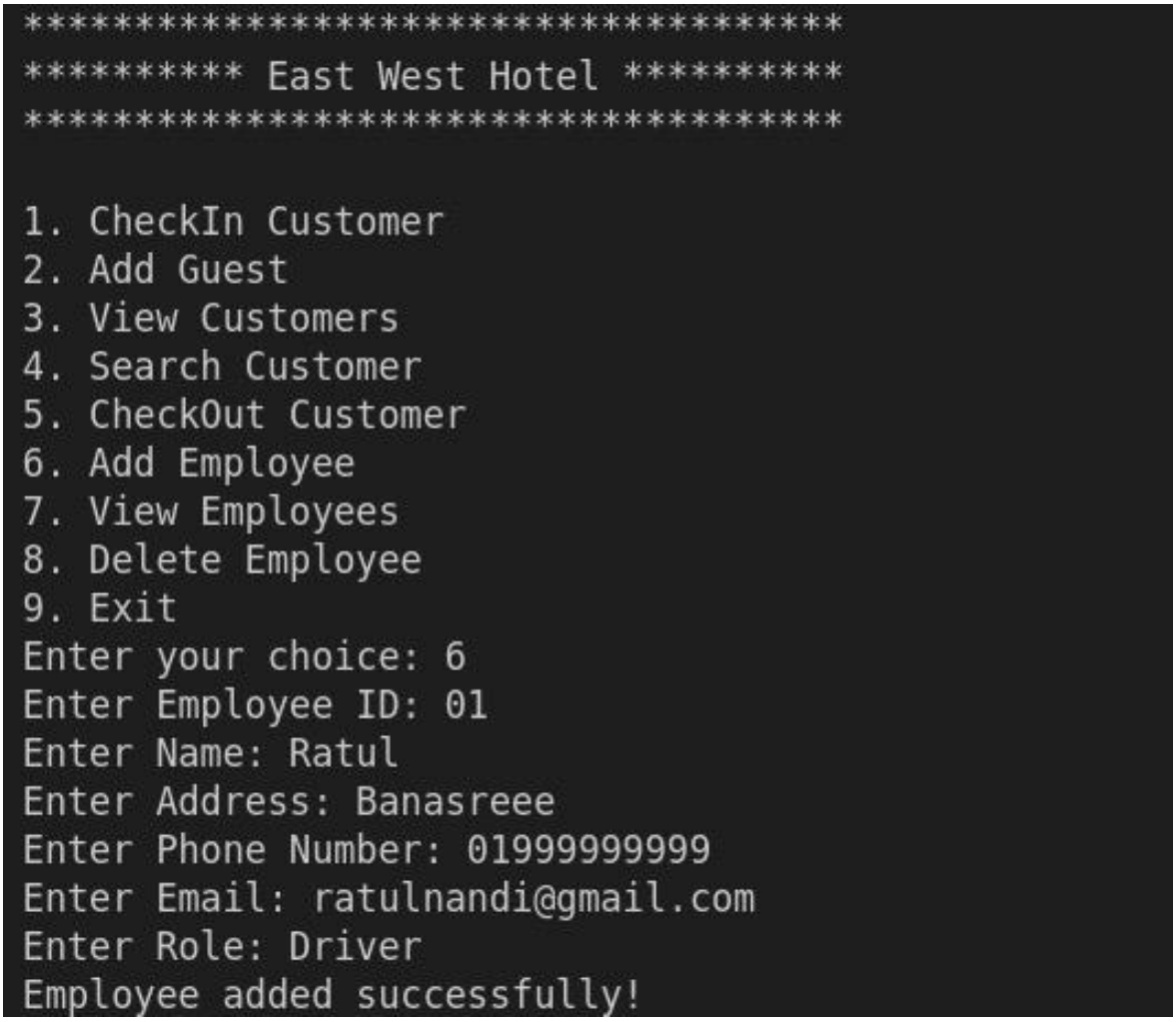
Option 3 indicates view customers where he/she will be able to view all the information's about customer.



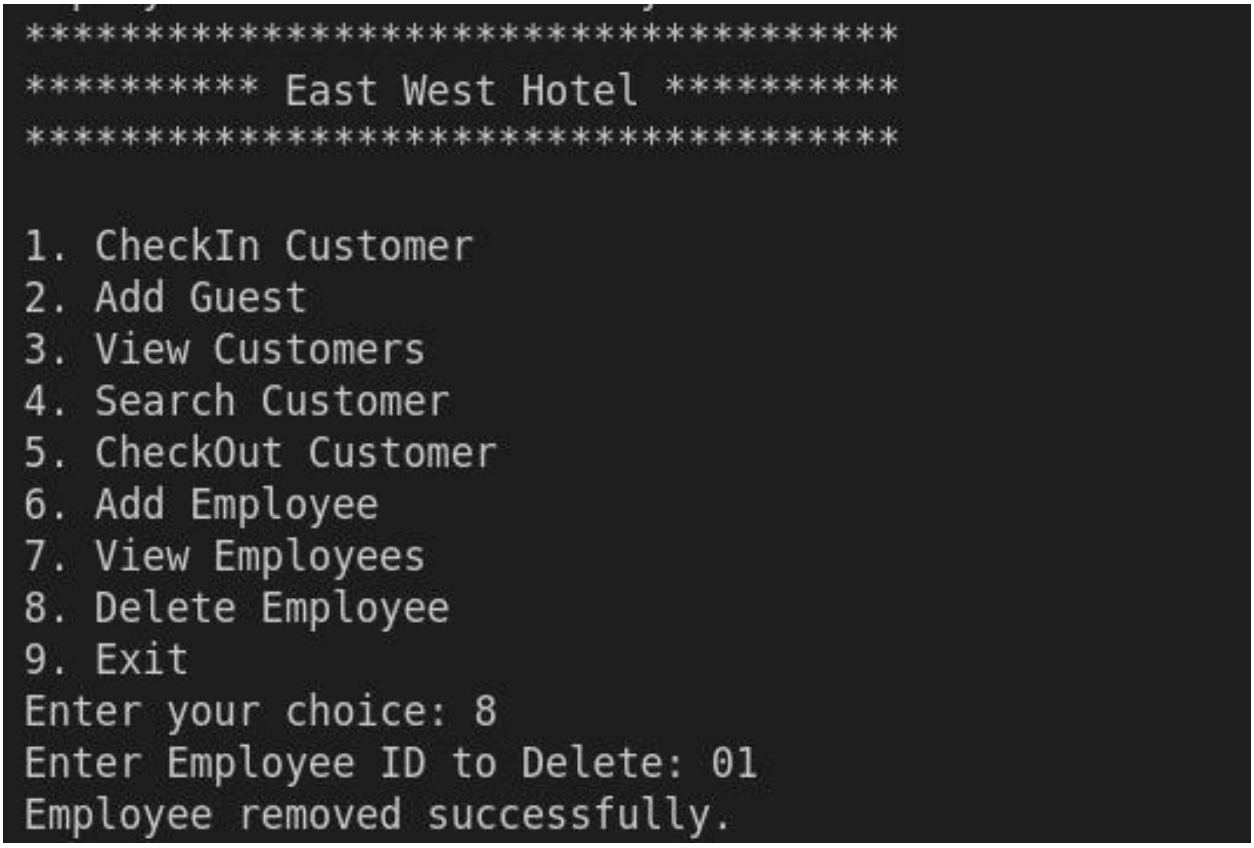
Option 4, is for search customer where he/she will be or the employee will be able to search for a customer details.



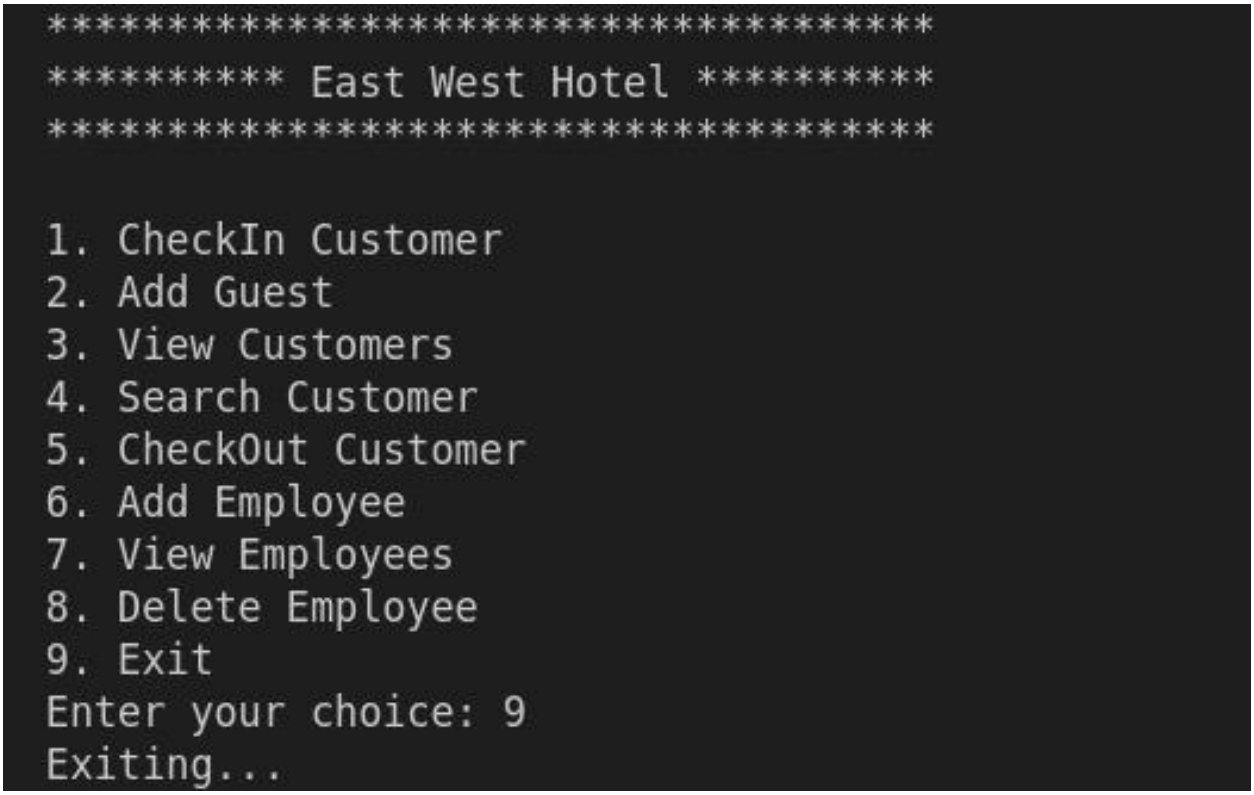
Option 5, is for Checkout Customer. If a customer wants to check out his/her room number will have to be inserted.



Option 6, is for adding an employee. For adding an employee, he/she has to give all the information about the employee.



Option 8, is or deleting an employee.



Option 9, Program ends here.