

using System;

using System.Collections.Generic;

using System.Linq;

class Book

{

    public string Title { get; set; }

    public string Author { get; set; }

    public string ISBN { get; set; }

    public bool IsAvailable { get; set; } = true;

    public Book(string title, string author, string isbn)

    {

        Title = title;

        Author = author;

        ISBN = isbn;

    }

}

class Reader

{

    public string Name { get; set; }

    public List<Book> BorrowedBooks { get; private set; } = new List<Book>();

    private const int MaxBorrowLimit = 3;

    public Reader(string name)

    {

        Name = name;

    }

    public bool BorrowBook(Book book)

    {

        if (BorrowedBooks.Count >= MaxBorrowLimit)

        {

            Console.WriteLine($"{Name} не может взять больше {MaxBorrowLimit} книг.");

            return false;

        }

        if (!book.IsAvailable)

        {

            Console.WriteLine($"Книга \"{book.Title}\" уже арендована.");

            return false;

        }

        BorrowedBooks.Add(book);

        book.IsAvailable = false;

        Console.WriteLine($"{Name} взял(а) книгу \"{book.Title}\".");

        return true;

    }

    public void ReturnBook(Book book)

    {

        if (BorrowedBooks.Remove(book))

        {

            book.IsAvailable = true;

            Console.WriteLine($"{Name} вернул(а) книгу \"{book.Title}\".");

        }

        else

        {

            Console.WriteLine($"{Name} не может вернуть книгу \"{book.Title}\", так как она не арендована.");

        }

    }

}

class Librarian

{

    public string Name { get; set; }

    public Librarian(string name)

    {

        Name = name;

    }

    public void AddBook(Library library, Book book)

    {

        library.Books.Add(book);

        Console.WriteLine($"Библиотекарь {Name} добавил книгу \"{book.Title}\".");

    }

    public void RemoveBook(Library library, Book book)

    {

        if (library.Books.Remove(book))

        {

            Console.WriteLine($"Библиотекарь {Name} удалил книгу \"{book.Title}\".");

        }

        else

        {

            Console.WriteLine($"Книга \"{book.Title}\" не найдена в библиотеке.");

        }

    }

}

class Library

{

    public List<Book> Books { get; private set; } = new List<Book>();

    public void DisplayBooks(bool includeBorrowed = false)

    {

        foreach (var book in Books.Where(b => includeBorrowed || b.IsAvailable))

        {

            Console.WriteLine($"Название: {book.Title}, Автор: {book.Author}, Статус: {(book.IsAvailable ? "В наличии" : "Арендована")}");

        }

    }

    public Book FindBook(string title)

    {

        return Books.FirstOrDefault(b => b.Title.Equals(title, StringComparison.OrdinalIgnoreCase));

    }

}

class Program

{

    static void Main()

    {

        var library = new Library();

        var librarian = new Librarian("Алексей");

        var reader = new Reader("Иван");

        librarian.AddBook(library, new Book("Преступление и наказание", "Ф.М. Достоевский", "123456"));

        librarian.AddBook(library, new Book("Война и мир", "Л.Н. Толстой", "234567"));

        librarian.AddBook(library, new Book("Мастер и Маргарита", "М.А. Булгаков", "345678"));

        Console.WriteLine("Книги в библиотеке:");

        library.DisplayBooks();

        Console.WriteLine("\nИван берет книги:");

        reader.BorrowBook(library.FindBook("Преступление и наказание"));

        reader.BorrowBook(library.FindBook("Война и мир"));

        Console.WriteLine("\nКниги в библиотеке после аренды:");

        library.DisplayBooks();

        Console.WriteLine("\nИван возвращает книгу:");

        reader.ReturnBook(library.FindBook("Преступление и наказание"));

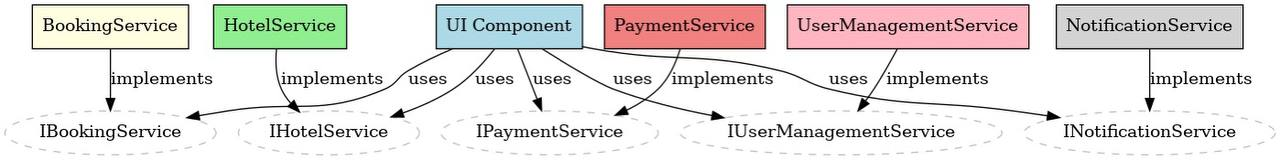
        Console.WriteLine("\nКниги в библиотеке после возврата:");

        library.DisplayBooks(true);

    }

}

//////////////////////////////////////////////////////////////////////////////////////////////////////////////



using System;

using System.Collections.Generic;

using System.Linq;

namespace HotelBookingSystem

{

    public interface IHotelService

    {

        List<Hotel> SearchHotels(string location, string roomClass, decimal maxPrice);

        void AddHotel(Hotel hotel);

    }

    public interface IBookingService

    {

        bool BookRoom(string userId, string hotelName, string roomClass, DateTime startDate, DateTime endDate);

        List<Booking> GetBookingsByUser(string userId);

    }

    public interface IPaymentService

    {

        bool ProcessPayment(string userId, decimal amount);

    }

    public interface INotificationService

    {

        void SendNotification(string userId, string message);

    }

    public interface IUserManagementService

    {

        void RegisterUser(string userId, string name);

        bool Login(string userId);

    }

    public class Hotel

    {

        public string Name { get; set; }

        public string Location { get; set; }

        public string RoomClass { get; set; }

        public decimal PricePerNight { get; set; }

    }

    public class Booking

    {

        public string UserId { get; set; }

        public string HotelName { get; set; }

        public string RoomClass { get; set; }

        public DateTime StartDate { get; set; }

        public DateTime EndDate { get; set; }

        public decimal TotalPrice { get; set; }

    }

    public class HotelService : IHotelService

    {

        private List<Hotel> hotels = new List<Hotel>();

        public List<Hotel> SearchHotels(string location, string roomClass, decimal maxPrice)

        {

            return hotels.Where(h => h.Location == location && h.RoomClass == roomClass && h.PricePerNight <= maxPrice).ToList();

        }

        public void AddHotel(Hotel hotel)

        {

            hotels.Add(hotel);

        }

    }

    public class BookingService : IBookingService

    {

        private List<Booking> bookings = new List<Booking>();

        public bool BookRoom(string userId, string hotelName, string roomClass, DateTime startDate, DateTime endDate)

        {

            if (bookings.Any(b => b.HotelName == hotelName && b.RoomClass == roomClass &&

                                  b.StartDate < endDate && b.EndDate > startDate))

            {

                return false;

            }

            bookings.Add(new Booking

            {

                UserId = userId,

                HotelName = hotelName,

                RoomClass = roomClass,

                StartDate = startDate,

                EndDate = endDate,

                TotalPrice = (endDate - startDate).Days \* 100

            });

            return true;

        }

        public List<Booking> GetBookingsByUser(string userId)

        {

            return bookings.Where(b => b.UserId == userId).ToList();

        }

    }

    public class PaymentService : IPaymentService

    {

        public bool ProcessPayment(string userId, decimal amount)

        {

            Console.WriteLine($"Payment of {amount:C} processed for user {userId}.");

            return true;

        }

    }

    public class NotificationService : INotificationService

    {

        public void SendNotification(string userId, string message)

        {

            Console.WriteLine($"Notification sent to {userId}: {message}");

        }

    }

    public class UserManagementService : IUserManagementService

    {

        private Dictionary<string, string> users = new Dictionary<string, string>();

        public void RegisterUser(string userId, string name)

        {

            if (!users.ContainsKey(userId))

            {

                users[userId] = name;

                Console.WriteLine($"User {name} registered successfully.");

            }

            else

            {

                Console.WriteLine($"User {name} already exists.");

            }

        }

        public bool Login(string userId)

        {

            if (users.ContainsKey(userId))

            {

                Console.WriteLine($"User {userId} logged in successfully.");

                return true;

            }

            Console.WriteLine($"Login failed for user {userId}.");

            return false;

        }

    }

    class Program

    {

        static void Main(string[] args)

        {

            IHotelService hotelService = new HotelService();

            IBookingService bookingService = new BookingService();

            IPaymentService paymentService = new PaymentService();

            INotificationService notificationService = new NotificationService();

            IUserManagementService userManagementService = new UserManagementService();

            hotelService.AddHotel(new Hotel { Name = "Oceanview", Location = "Hawaii", RoomClass = "Standard", PricePerNight = 200 });

            hotelService.AddHotel(new Hotel { Name = "Mountain Retreat", Location = "Alps", RoomClass = "Luxury", PricePerNight = 300 });

            userManagementService.RegisterUser("user1", "Alice");

            userManagementService.Login("user1");

            var hotels = hotelService.SearchHotels("Hawaii", "Standard", 250);

            Console.WriteLine("\nAvailable Hotels:");

            foreach (var hotel in hotels)

            {

                Console.WriteLine($"Hotel: {hotel.Name}, Location: {hotel.Location}, Room Class: {hotel.RoomClass}, Price: {hotel.PricePerNight:C}");

            }

            var success = bookingService.BookRoom("user1", "Oceanview", "Standard", DateTime.Now.AddDays(1), DateTime.Now.AddDays(5));

            if (success)

            {

                Console.WriteLine("\nBooking successful.");

                paymentService.ProcessPayment("user1", 200 \* 4);

                notificationService.SendNotification("user1", "Your booking has been confirmed.");

            }

            else

            {

                Console.WriteLine("\nRoom is not available.");

            }

            var bookings = bookingService.GetBookingsByUser("user1");

            Console.WriteLine("\nYour Bookings:");

            foreach (var booking in bookings)

            {

                Console.WriteLine($"Hotel: {booking.HotelName}, Room Class: {booking.RoomClass}, Dates: {booking.StartDate.ToShortDateString()} - {booking.EndDate.ToShortDateString()}, Total: {booking.TotalPrice:C}");

            }

        }

    }

}