BookingManager Test Cases:

Use case ID: TM01	**Use case name:** RegisteringUser
Test number: 1	
Objective: Tests the successful registration of a new user.	
Set up (must make sure that all listed below is satisfied before the test): 1. BookingManager system is initialized. 2. Calendar instance is available. 3. Ensures user is not currently registered in the system.	
Expected results: 1. New user is added to the system. 2. A success message is printed: “User [name] registered successfully.”	
Test: 1. The user enters their details. Example User Details: Name: Amy Lewis UserID: U289 2. The system calls registerUser(User) method. 3. The system adds the new user to the list and prints the confirmation.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TM02	**Use case name:** FindingUser
Test number: 2	
Objective: Tests the retrieval of a registered user.	
Set up (must make sure that all listed below is satisfied before the test): 1. Ensure there is an existing user with the user ID that is being searched for.	
Expected results: 1. If the user exists, the details are returned. 2. If the user does not exist, the system prints: “User not found.“	
Test: 1. The user searches for UserID. E.g UserID: U289. 2. The system calls findUser(U289). 3. If the user ID is found, the system returns the user object. 4. If the user ID is not found, the system prints an error message: “User not found.”	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TM03	**Use case name:** MakingBooking
Test number: 3	
Objective: Tests the successful creation of a booking.	
Set up (must make sure that all listed below is satisfied before the test): 1. Ensures a user exists to make the booking. 2. Ensures a FilmBooking instance is available. 3. The booking is confirmed.	
Expected results: 1. If the booking is created successfully, it is added to the user’s list. 2. If the user is a Friends of Lancaster Member, a discount is applied to the booking. 3. A confirmation message is printed once the booking has been created successfully: “Booking successfully added for [name].”	
Test: 1. The user selects a film for booking. 2. The system calls makeBooking(user, booking). 3. If the user is a Friends of Lancaster member, a discount message appears: “Discount applied for Friends of Lancaster member: [name]”. 4. If the booking is confirmed, the booking is added to the user’s list of bookings.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TM04	**Use case name:** CancellingBooking
Test number: 4	
Objective: Test the successful cancellation of a booking. (***Alternative flow*** of TM03: MakingBooking)	
Set up (must make sure that all listed below is satisfied before the test): 1. Ensure there is a valid user alongside a valid booking which can be cancelled. E.g: User U289 with a booking ID B001.	
Expected results: 1. If the booking is successfully cancelled, the booking is removed from the user’s list and the system prints: “Booking with ID [bookingID] removed successfully”. 2. If the booking cannot be found an error message is printed: “Booking with ID [bookingID] not found”.	
Test: 1. The user selects booking ID B001 for cancellation. 2. The system calls cancelBooking(user, bookingID). 3. If the booking exists, it is removed, and a success message is printed. Otherwise, if the booking does not exist, an error message is printed.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TM05	**Use case name:** ApplyingDiscount
Test number: 5	
Objective: Test the application of a Friends of Lancaster discount.	
Set up (must make sure that all listed below is satisfied before the test): 1. Ensure the user is a Friends of Lancaster member.	
Expected results: 1. If the user is a Friends of Lancaster member, the system applies a 10% discount and prints a success message: "Friends of Lancaster discount applied". 2. If the user is not a Friends of Lancaster member the original prices is returned with no discount applied.	
Test: 1. The system calls applyFriendsDiscount(memberID, originalPrice). 2. The system verifies the membership. 3. The original price is reduced by 10% if the member is active. Example data: Valid Friends of Lancaster member: U289 Original price: £100 Discounted price: £90	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Calendar Test Cases

Use case ID: TC01	**Use case name:** CheckDateAvailability
Test number: 1	
Objective: Tests to see if the system correctly identifies whether a date is available for booking.	
Set up (must make sure that all listed below is satisfied before the test): 1. A calendar instance is created with a calendarID and venue details. 2. Ensure all dates available for booking are stored in the availableDates list.	
Expected results: 1. The system should return true when the date is available. 2. The system should return false when the date is not available (which is not in the available dates list).	
Test: 1. Available date to be tested: 15-04-2025. Unavailable date to be tested: 16-04-2025. 2. Call addAvailableDate(15-04-2025). 3. Call isDateAvailable(15-04-2025). This should return true. 4. Call isDateAvailable(16-04-2025). This should return false.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TC02	**Use case name:** BookAvailableDate
Test number: 2	
Objective: Tests the system allows the booking of available dates.	
Set up (must make sure that all listed below is satisfied before the test): 1. A Calendar instance is created. 2. Ensure all dates available are present in the availableDates list.	
Expected results: 1. The system should allow booking for all available dates. 2. Once the dates have been booked, they should no longer be available.	
Test: 1. Date to be booked: 15-04-2025. 2. Call addAvailableDate(15-04-2025). 3. Call bookDate(15-04-2025). 4. Call isDateAvailable(15-04-2025). This should return false once the date has already been booked.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TC03	**Use case name:** BookUnavailableDate
Test number: 3	
Objective: Tests the system prevents booking a date that is already booked or unavailable. (***Alternative flow*** of TM02: BookAvailableDate).	
Set up (must make sure that all listed below is satisfied before the test): 1. A Calendar instance is created. 2. The unavailable date is not present in the availableDates list nor does it get added into it.	
Expected results: 1. The system should not allow booking for the unavailable date and prints the error message: “Date [date] is already booked or unavailable”.	
Test: 1. Example unavailable date: 15-04-2025 2. Call bookDate(15-04-2025).	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TC04	**Use case name:** RemoveBooking
Test number: 4	
Objective: Tests the system correctly removes a booking and makes the date available again.	
Set up (must make sure that all listed below is satisfied before the test): 1. A Calendar instance is created. 2. Ensure the date to be removed is a booked date currently in the list of availableDates.	
Expected results: 1. The system should allow removal for any booking. 2. Once a booking has been removed, that date should become available again after removal.	
Test: 1. Example booking date to be removed: 15-04-2025. 2. Call addAvailableDate(15-04-2025). 3. Call bookDate(15-04-2025). 4. Call removeBooking(15-04-2025). 5. Call isDateAvailable(15-04-2025), which should return true after the booking that was held here previously is removed.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	

Use case ID: TC05	**Use case name:** AddAvailableDate
Test number: 5	
Objective: Tests the system correctly adds new available dates to the calendar.	
Set up (must make sure that all listed below is satisfied before the test): 1. A Calendar instance is created.	
Expected results: 1. The system should allow dates to be added to the availableDates list. 2. When checking availability for the dates in the availableDates list, these added dates should be seen as available.	
Test: 1. Example date 15-04-2025 2. Call addAvailableDate(15-04-2025). 3. Call isDateAvailable(15-04-2025), which should return true once the new date has been added into the availableDates list.	
Test record: The expected behavior observed.	
Date: 17 March 2025	**Tester:** Impact Creators: Amina Khan
Result: Passed.	