BLACK BOX TEST CASES

Group Name: TISM Tech
Ayaan Khan - i220832
Ayaan Mughal - i220861
Mishal Ali - i221291

US02: Booking a Hotel Room

Test Case ID: TC-US02-01

- **Test Case Name**: Booking with valid data
- **User Story**: US02: Booking a Hotel Room
- **Input**: Valid guest information, valid card details, valid check-in/check-out dates
- **Expected Output**: Booking is confirmed and receipt is generated
- **Actual Output**: Booking is confirmed and receipt is generated
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Navigate to a hotel listing.
    2. Select an available room.
    3. Fill in valid guest details.
    4. Provide correct payment information.
    5. Click "Book Now".
    6. Verify booking confirmation and receipt.

Test Case ID: TC-US02-02

- **Test Case Name**: Booking with missing invalid name
- **User Story**: US02: Booking a Hotel Room
- **Input**: Guest details filled, Name set to someone else
- **Expected Output**: System shows "Enter correct guest details" error
- **Actual Output**: Booking Confirmed
- **Status**: Failed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Navigate to a hotel listing.
    2. Select a room.

3. Fill in guest information, leave payment details empty.
4. Attempt to book.
5. Verify that an error appears.

Test Case ID: TC-US02-03

- **Test Case Name**: Booking with invalid email
- **User Story**: US02: Booking a Hotel Room
- **Input**: Valid guest information, email doesn’t follow format “hgmail.com”
- **Expected Output**: System rejects transaction with "Invalid email syntax" message
- **Actual Output**: Rejects booking and gives error
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Select a hotel and a room.
    2. Enter valid guest details.
    3. Enter invalid credit card number (e.g., 1234).
    4. Attempt to book.
    5. Verify that payment is rejected.

Test Case ID: TC-US02-04

- **Test Case Name**: Booking with incomplete guest details
- **User Story**: US02: Booking a Hotel Room
- **Input**: Missing fields in guest info e.g missing name
- **Expected Output**: System displays validation error
- **Actual Output**: Validation error
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Select hotel and room.
    2. Leave name field blank.
    3. Attempt to book.
    4. Observe system validation.

Test Case ID: TC-US02-05

- **Test Case Name**: Booking on current date (minimum valid check-in)
- **User Story**: US02: Booking a Hotel Room
- **Input**: Check-in date = Today's date
- **Expected Output**: Booking is allowed

- **Actual Output**: Room not available
- **Status**: Failed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Book hotels with today's date.
    2. Select room.
    3. Complete booking process.
    4. Verify booking success.

Test Case ID: TC-US02-06

- **Test Case Name**: Booking with check-in date in the past (invalid)
- **User Story**: US02: Booking a Hotel Room
- **Input**: Check-in date = Yesterday's date
- **Expected Output**: System should reject with "Invalid date" error or Can’t select date
- **Actual Output**: Can’t select date
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Book hotels with check-in = yesterday.
    2. Attempt to proceed.
    3. Observe system error.

Test Case ID: TC-US02-07

- **Test Case Name**: Booking minimum stay (1 night)
- **User Story**: US02: Booking a Hotel Room
- **Input**: Check-in = 10-May, Check-out = 11-May
- **Expected Output**: Booking allowed (1-night stay)
- **Actual Output**: Booking allowed
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Select check-in = 10-May, check-out = 11-May.
    2. Book a room.
    3. Verify success.

Test Case ID: TC-US02-08

- **Test Case Name**: Booking checkout allowed same day
- **User Story**: US02: Booking a Hotel Room
- **Input**: checkout = checkin
- **Expected Output**: Booking not allowed
- **Actual Output**: Booking not allowed
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Select a check-in and check-out gap of 30 nights.
    2. Complete booking.
    3. Verify booking accepted.

US11: Manage Hotels

Test Case ID: TC-US11-01

- **Test Case Name**: Adding a new hotel with all valid information
- **User Story**: US11: Manage Hotels
- **Input**: Hotel name, location, contact info all valid
- **Expected Output**: Hotel successfully added to the system
- **Actual Output**: Hotel successfully added to system
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Login as Admin.
    2. Navigate to "Add Hotel" form.
    3. Fill all fields correctly.
    4. Click "Save".
    5. Verify that hotel appears in listings.

Test Case ID: TC-US11-02

- **Test Case Name**: Adding hotel with missing required fields
- **User Story**: US11: Manage Hotels
- **Input**: Missing hotel name or contact info
- **Expected Output**: System displays validation error
- **Actual Output**: Validation error
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning

- **Steps to Execute**:
    1. Leave hotel name blank.
    2. Attempt to submit form.
    3. Verify error for missing field.

Test Case ID: TC-US11-03

- **Test Case Name**: Adding a duplicate hotel
- **User Story**: US11: Manage Hotels
- **Input**: Hotel name already exists
- **Expected Output**: System rejects duplicate entry
- **Actual Output**: Hotel successfully added to system
- **Status**: Failed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Add a hotel with an existing name.
    2. Submit.
    3. Confirm error message about duplication.

Test Case ID: TC-US11-04

- **Test Case Name**: Hotel name minimum characters
- **User Story**: US11: Manage Hotels
- **Input**: Hotel name = 1 character
- **Expected Output**: Hotel added to system
- **Actual Output**: Hotel added to system
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Input a 1-character name.
    2. Save.
    3. Check result.

Test Case ID: TC-US11-05

- **Test Case Name**: Hotel contact number has letter in it
- **User Story**: US11: Manage Hotels
- **Input**: Hotel contact number has letter in it
- **Expected Output**: Rejected

- **Actual Output**: Accepted
- **Status**: Failed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Input a number with a letter
    2. Save.
    3. Confirm correct behavior.

Test Case ID: TC-US11-06

- **Test Case Name**: Contact phone number boundary test
- **User Story**: US11: Manage Hotels
- **Input**: Phone number with minimum and maximum allowed digits
- **Expected Output**: Validation success or failure based on phone rules
- **Actual Output**: Hotel Booked
- **Status**: Failed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Enter less than 7-digit number (minimum boundary).
    2. Enter more than 15-digit number (maximum boundary).
    3. Try to save and observe validations.

US09: Manage Rooms

Test Case ID: TC-US09-01

- **Test Case Name**: Adding a new room with all valid fields
- **User Story**: US09: Manage Rooms
- **Input**: Room type, price, capacity, special features correctly filled
- **Expected Output**: Room added successfully
- **Actual Output**: Room added successfully
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Navigate to Room Management.
    2. Fill all room fields correctly.
    3. Save.
    4. Verify room listing update.

Test Case ID: TC-US09-02

- **Test Case Name**: Adding a room with missing price field
- **User Story**: US09: Manage Rooms
- **Input**: Room fields filled but price left blank
- **Expected Output**: System asks to add price
- **Actual Output**: Asks to enter price
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Fill room details.
    2. Leave price blank.
    3. Save.
    4. Verify error handling.

Test Case ID: TC-US09-03

- **Test Case Name**: Updating room details successfully
- **User Story**: US09: Manage Rooms
- **Input**: Existing room edited (new price)
- **Expected Output**: Room details updated successfully
- **Actual Output**: Room details updated successfully
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Find existing room.
    2. Edit price.
    3. Save changes.
    4. Confirm updates.

Test Case ID: TC-US09-04

- **Test Case Name**: Removing a room
- **User Story**: US09: Manage Rooms
- **Input**: Select an existing room and remove
- **Expected Output**: Room removed successfully
- **Actual Output**: Room removed
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:

1. Open Room List.
2. Select a room.
3. Remove.
4. Verify deletion.

Test Case ID: TC-US09-05

- **Test Case Name**: Room capacity minimum limit
- **User Story**: US09: Manage Rooms
- **Input**: Capacity = 1 person
- **Expected Output**: Room added successfully
- **Actual Output**: Room added
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Add room with capacity = 1.
    2. Save.
    3. Confirm acceptance.

Test Case ID: TC-US09-06

- **Test Case Name**: Room capacity negative number
- **User Story**: US09: Manage Rooms
- **Input**: Capacity = -10
- **Expected Output**: Validation error
- **Actual Output**: Validation error
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Enter capacity = -1
    2. Try saving.
    3. Confirm error.

US08: Register/Login

Test Case ID: TC-US08-01

- **Test Case Name**: Successful user registration

- **User Story**: US08: Register/Login
- **Input**: Valid email , strong password
- **Expected Output**: Account created successfully
- **Actual Output**: Account created
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Open the registration page.
    2. Enter valid email, strong password
    3. Click "Register."
    4. Observe that account is created.

Test Case ID: TC-US08-02

- **Test Case Name**: Login with valid credentials
- **User Story**: US08: Register/Login
- **Input**: Registered email and correct password
- **Expected Output**: User successfully logged in and redirected to homepage
- **Actual Output**: User logged in and redirected to homepage
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Open login page.
    2. Enter registered email and correct password.
    3. Click "Login."
    4. Verify that user is logged in and redirected.

Test Case ID: TC-US08-03

- **Test Case Name**: Registration with invalid email
- **User Story**: US08: Register/Login
- **Input**: Invalid email format (e.g. usergmail.com) and strong password
- **Expected Output**: Error message: "Enter valid email"
- **Actual Output**: Error message
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Open registration form.
    2. Enter invalid email format.
    3. Submit the form.
    4. Check that validation error appears.

Test Case ID: TC-US08-04

- **Test Case Name**: Password minimum length boundary
- **User Story**: US08: Register/Login
- **Input**: Password = exactly 6 characters
- **Expected Output**: Password accepted, registration successful
- **Actual Output**: Registration successful
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Open registration page.
    2. Enter password exactly 6 characters long.
    3. Register.
    4. Verify success.

Test Case ID: TC-US08-05

- **Test Case Name**: Password just below minimum length
- **User Story**: US08: Register/Login
- **Input**: Password = 5 characters
- **Expected Output**: Validation error "Password too short"
- **Actual Output**: Error
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Try registering with 5-character password.
    2. Submit.
    3. Verify error shown.

Test Case ID: TC-US08-06

- **Test Case Name**: Login attempt with wrong password
- **User Story**: US08: Register/Login
- **Input**: Correct email, wrong password
- **Expected Output**: "Incorrect password" error
- **Actual Output**: Error
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:

1. Open login page.
2. Enter valid email but wrong password.
3. Try to login.
4. Observe error message.

US01: Searching and Filtering Hotels

Test Case ID: TC-US01-01

- **Test Case Name**: Valid hotel search
- **User Story**: US01: Searching and Filtering Hotels
- **Input**: City = "New York"
- **Expected Output**: List of hotels matching criteria
- **Actual Output**: “Grand Plaza”, “Tipton Hotel”
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Open the search page.
    2. Enter “New York”
    3. Click Search.
    4. Observe hotel listings.

Test Case ID: TC-US01-02

- **Test Case Name**: Invalid city search
- **User Story**: US01: Searching and Filtering Hotels
- **Input**: City = "@#$%^", valid dates
- **Expected Output**: Error "No hotels found" or Empty
- **Actual Output**: Empty
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Enter invalid characters in city.
    2. Click Search
    3. Observe Output

Test Case ID: TC-US01-03

- **Test Case Name**: Search with minimum characters
- **User Story**: US01: Searching and Filtering Hotels
- **Input**: City = "N”
- **Expected Output**: Hotels in Cities starting from N
- **Actual Output**: “Grand Plaza”, “Tipton Hotel”
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Search using 1-letter city.
    2. Click Search.
    3. Verify behavior.

Test Case ID: TC-US01-04

- **Test Case Name**: Search with maximum input length
- **User Story**: US01: Searching and Filtering Hotels
- **Input**: City = long name (50+ chars)
- **Expected Output**: Empty
- **Actual Output**: Empty
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Enter maximum length city.
    2. Search.
    3. Check if system processes or truncates input.

US12: Manage Accounts

Test Case ID: TC-US12-01

- **Test Case Name**: Creating a new user account successfully
- **User Story**: US12: Manage Accounts
- **Input**: Valid username, email, and password
- **Expected Output**: User account created successfully
- **Actual Output**: User account created successfully
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Admin logs in.
    2. Go to API using postman

3. Fill all fields correctly.
4. Save.
5. Verify account in Database

Test Case ID: TC-US12-02

- **Test Case Name**: Username minimum length validation
- **User Story**: US12: Manage Accounts
- **Input**: Username = 1 character
- **Expected Output**: User registered
- **Actual Output**: User registered
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Try creating a user with 1-character username.
    2. Save.
    3. Verify

US13: Adjust Prices

Test Case ID: TC-US13-01

- **Test Case Name**: Successfully updating room price
- **User Story**: US13: Adjust Prices
- **Input**: Update price from $100 to $120
- **Expected Output**: Price updated successfully
- **Actual Output**: Price updated
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Admin logs in.
    2. Edit a room's price field.
    3. Save changes.
    4. Verify updated price shown.

Test Case ID: TC-US13-02

- **Test Case Name**: Price minimum value validation

- **User Story**: US13: Adjust Prices
- **Input**: Enter price = 0
- **Expected Output**: Validation error
- **Actual Output**: Room price changed
- **Status**: Failed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Set room price = 0.
    2. Try to save.
    3. Confirm system error.

US03: Managing Bookings

Test Case ID: TC-US03-01

- **Test Case Name**: Cancel Booking
- **User Story**: US03: Managing Bookings
- **Input**: Logged-in user with bookings
- **Expected Output**: Booking no longer in list
- **Actual Output**: Booking still in list
- **Status**: Failed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Login as user.
    2. Open "My Bookings".
    3. Cancel booking
    4. Verify

Test Case ID: TC-US03-02

- **Test Case Name**: Update Booking details
- **User Story**: US03: Managing Bookings
- **Input**: logged in user with bookings
- **Expected Output**: Booking details updated
- **Actual Output**: Booking details updated
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Login with a new account.
    2. Open "My Bookings".

3. Select Booking and update details
4. Verify

US10: Viewing Booking Data

Test Case ID: TC-US10-01

- **Test Case Name**: Retrieve guest booking details successfully
- **User Story**: US10: Viewing Booking Data
- **Input**: Search by valid guest name
- **Expected Output**: Guest bookings displayed
- **Actual Output**: Bookings displayed
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Staff logs in.
    2. Search guest by name.
    3. Confirm booking data displayed.

Test Case ID: TC-US10-02

- **Test Case Name**: Search with empty input (boundary)
- **User Story**: US10: Viewing Booking Data
- **Input**: Empty search field
- **Expected Output**: Validation error
- **Actual Output**: Validation error
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Leave search field empty.
    2. Try searching.
    3. Verify error.

US05: Facilitating Group Bookings

Test Case ID: TC-US05-01

- **Test Case Name**: Successfully book multiple rooms
- **User Story**: US05: Facilitating Group Bookings
- **Input**: 2 rooms booked together
- **Expected Output**: Group booking success
- **Actual Output**: Group booking success
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:
    1. Login as Travel Agent.
    2. Select 2 rooms.
    3. Complete group booking.

Test Case ID: TC-US05-02

- **Test Case Name**: Booking minimum allowed rooms
- **User Story**: US05: Facilitating Group Bookings
- **Input**: 0 rooms booked
- **Expected Output**: Booking rejected
- **Actual Output**: Booking rejected
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Book exactly 0 rooms.
    2. Complete booking.
    3. Confirm system allows.

US07: Enrolling in Loyalty Program

Test Case ID: TC-US07-01

- **Test Case Name**: Enroll into loyalty program after eligibility
- **User Story**: US07: Enrolling in Loyalty Program
- **Input**: User meets booking criteria
- **Expected Output**: Enrollment success
- **Actual Output**: Enrollment success
- **Status**: Passed
- **Testing Method**: Equivalence Class Partitioning
- **Steps to Execute**:

1. Complete required number of bookings.
2. Enroll in loyalty program.
3. Verify confirmation.

Test Case ID: TC-US07-02

- **Test Case Name**: Loyalty points redemption boundary
- **User Story**: US07: Enrolling in Loyalty Program
- **Input**: Redeem exactly 500 points (minimum)
- **Expected Output**: Discount applied successfully
- **Actual Output**: Discount applied successfully
- **Status**: Passed
- **Testing Method**: Boundary Value Analysis
- **Steps to Execute**:
    1. Try redeeming 500 points.
    2. Complete booking.
    3. Confirm discount applied.
