**3. Test Cases Design/ test coverage and bugs reporting:**

**Part1:**

**Positive Test Cases:**

1. **TC01: Submit a plain text tweet with valid length (under 280 characters)**
   * Input: "This is a valid tweet."
   * Expected Result: Tweet is successfully posted.
2. **TC02: Submit a tweet with exactly 280 characters**
   * Input: "A tweet with exactly 280 characters...." (fill in with exact 280 characters)
   * Expected Result: Tweet is successfully posted.
3. **TC03: Submit a tweet with one image**
   * Input: "Tweet with an image" and attach one image.
   * Expected Result: Tweet is successfully posted with the image.
4. **TC04: Submit a tweet with four images**
   * Input: "Tweet with four images" and attach four images.
   * Expected Result: Tweet is successfully posted with four images.
5. **TC05: Submit a tweet with a GIF**
   * Input: "Tweet with a GIF" and attach a GIF.
   * Expected Result: Tweet is successfully posted with the GIF.
6. **TC06: Submit a tweet with a URL**
   * Input: "Check this out: [http://example.com](http://example.com/)"
   * Expected Result: Tweet is successfully posted with the URL.
7. **TC07: Submit a tweet with a poll**
   * Input: "What's your favorite season?" and add poll options (Spring, Summer, Fall, Winter).
   * Expected Result: Tweet is successfully posted with the poll.
8. **TC08: Create a thread by replying to your own tweet**
   * Input: Post a tweet "First tweet" and reply to it with "Second tweet".
   * Expected Result: A thread is created with the two tweets.
9. **TC09: Submit a tweet with a combination of text and a single image**
   * Input: "Check out this image!" and attach one image.
   * Expected Result: Tweet is successfully posted with the image.
10. **TC10: Submit a tweet with a combination of text and a GIF**
    * Input: "Check out this GIF!" and attach one GIF.
    * Expected Result: Tweet is successfully posted with the GIF.

**Negative Test Cases:**

1. **TC11: Submit a tweet with more than 280 characters**
   * Input: "A tweet with more than 280 characters..." (fill in with more than 280 characters)
   * Expected Result: Error message displayed, tweet not posted.
2. **TC12: Submit an identical tweet twice in a row**
   * Input: "This is a duplicate tweet."
   * Expected Result: First tweet is posted, but second tweet shows an error and is not posted.
3. **TC13: Submit a tweet with five images**
   * Input: "Tweet with five images" and attach five images.
   * Expected Result: Error message displayed, tweet not posted.
4. **TC14: Submit a tweet with both an image and a GIF**
   * Input: "Tweet with image and GIF" and attach one image and one GIF.
   * Expected Result: Error message displayed, tweet not posted.
5. **TC15: Submit a tweet with an image and a URL**
   * Input: "Check this out: [http://example.com](http://example.com/)" and attach one image.
   * Expected Result: Tweet is posted, but the URL thumbnail is not displayed.
6. **TC16: Submit a tweet with a GIF and a URL**
   * Input: "Check this out: [http://example.com](http://example.com/)" and attach one GIF.
   * Expected Result: Tweet is posted, but the URL thumbnail is not displayed.
7. **TC17: Submit a tweet with a broken image link**
   * Input: "Tweet with a broken image link" and attach a corrupted image file.
   * Expected Result: Error message displayed, tweet not posted.
8. **TC18: Submit a tweet with a broken GIF link**
   * Input: "Tweet with a broken GIF link" and attach a corrupted GIF file.
   * Expected Result: Error message displayed, tweet not posted.
9. **TC19: Submit a tweet while the network is disconnected**
   * Input: "This is a tweet without a network connection."
   * Expected Result: Error message displayed, tweet not posted.
10. **TC20: Submit a tweet with invalid URL format**
    * Input: "Check this out: invalid-url"
    * Expected Result: Tweet is posted, but the URL is not recognized as a valid link.

**Note (**Cross-Browser Testing Considerations):

Ensuring that the above test cases are executed on different web browsers to verify compatibility and consistent behaviour.

Common browsers to include are:

* Google Chrome
* Mozilla Firefox
* Microsoft Edge
* Safari

**Part 2**:

**Review of Requirements**

1. **Rate Management:**
   * Hotels can define multiple rates.
   * Rates are defined for specific room types, date ranges, and occupancy levels.
   * Example: Standard Room: From 1/12/2016 to 30/12/2016, $100 for 1 Adult, $110 for 2 adults.
2. **Synchronization with OTAs:**
   * Rates are sent to OTAs (e.g., Booking.com) when saved.
   * Example: On Booking.com, the rate for 1 adult is set to $150.
3. **Validation:**
   * No duplicate occupancy rules for the same dates within a rate.
4. **Functionality upon Saving:**
   * For date ranges where the end date is greater than the current date:
     + For each date within the range, check if the occupancy rule has changed since the last save.
     + If changed, update OTA with the new rate.

**Comments and Observations**

1. **Validation Requirement:**
   * Ensure the system prevents duplicate occupancy rules for the same date range. This validation is critical to maintain data integrity and prevent conflicts in rate definitions.
2. **Dynamic Rate Changes:**
   * The system needs to track the last updated timestamp to compare if occupancy rules have changed. This is essential for accurately identifying changes and syncing with OTAs.
3. **Rate Synchronization Logic:**
   * The synchronization process with OTAs must handle rate adjustments (discounts/supplements) and ensure accurate updates.
4. **Testing Considerations:**
   * The testing scope should include scenarios for rate creation, validation, synchronization, and dynamic rate changes.

**Positive Test Cases**

**Test Case 1.1: Create a rate with valid data**

* **Precondition:** No existing rate conflicts for the date range and occupancy.
* **Input:** Standard Room, 1/12/2016 to 30/12/2016, $100 for 1 Adult, $110 for 2 adults.
* **Expected Result:** Rate is saved successfully, and the rates are synchronized with OTAs.

**Test Case 1.2: Create multiple rates for different room types and periods**

* **Precondition:** No existing rate conflicts for the date ranges and occupancies.
* **Input:** Standard Room and Deluxe Room, multiple date ranges and occupancies.
* **Expected Result:** Rates are saved successfully without conflicts, and the rates are synchronized with OTAs.

**Test Case 1.3: Modify rate and verify synchronization for future dates**

* **Precondition:** Existing rate for Standard Room.
* **Input:** Modify rate for Standard Room, 1/12/2016 to 30/12/2016, $120 for 1 Adult.
* **Expected Result:** Rate is saved successfully, and the OTA shows the updated rate for the modified dates only.

**Test Case 1.4: Ensure validation for unique occupancy rules within the same dates**

* **Precondition:** Existing rate for Standard Room.
* **Input:** Standard Room, 1/12/2016 to 30/12/2016, $130 for 1 Adult, $140 for 2 adults.
* **Expected Result:** Rate is saved successfully without validation errors.

**Negative Test Cases**

**Test Case 2.1: Create a rate with duplicate occupancy rule for the same dates**

* **Precondition:** Existing rate for Standard Room.
* **Input:** Standard Room, 1/12/2016 to 30/12/2016, $100 for 1 Adult, $110 for 1 Adult.
* **Expected Result:** System displays an error message indicating duplicate occupancy rule.

**Test Case 2.2: Save rate with overlapping date ranges and different occupancies**

* **Precondition:** No existing rate conflicts for the date ranges and occupancies.
* **Input:** Standard Room, 1/12/2016 to 15/12/2016, $100 for 1 Adult; 10/12/2016 to 20/12/2016, $110 for 2 Adults.
* **Expected Result:** System correctly handles overlapping ranges and synchronizes the correct rates.

**Test Case 2.3: Attempt to create a rate with invalid date formats and ranges**

* **Precondition:** No existing rate conflicts for the date ranges and occupancies.
* **Input:** Standard Room, invalid dates or end date before start date.
* **Expected Result:** System displays appropriate error messages.

**Test Case 2.4: Verify no synchronization for past date changes**

* **Precondition:** Existing rate for Standard Room.
* **Input:** Modify rate for a past date range.
* **Expected Result:** No changes are synchronized to OTA.