**Section 1: Manual Testing**

**Step 1: Test Planning & Requirement Analysis**

**Ans 1.1:** To test the **Garibook trip booking process**, I’d start with **functional testing** to make sure users can select a trip, enter details, make payments, and get a confirmation without issues. Then, I’d do **UI/UX testing** to check if the site looks good and works smoothly on different devices. **Integration testing** is key to ensuring the booking system properly connects with payments and driver assignments. I’d also run **performance tests** to see if it handles multiple bookings at once and **security tests** to keep user data safe. Finally, a round of **regression and UAT** would ensure everything runs perfectly before release.

**Ans 1.2:**

**Functional Requirements:**

1. **Trip Booking:** Users should be able to enter pickup/drop-off locations, select a vehicle, and confirm a ride.
2. **Payment Processing:** The system should support different payment methods like cards, mobile banking, and cash.
3. **Booking Confirmation:** After booking, users should get a confirmation via SMS or email with trip details.

**Non-Functional Requirements:**

1. **Speed & Performance:** The site should load fast and handle multiple bookings without lag.
2. **Security:** User data and payment details should be encrypted and protected from hackers.

**Step 2: Test Scenario & Test Cases Creation**

**Ans 2.1:**

**Successful Trip Booking:** Check if a user can enter pickup/drop-off details, select a vehicle, pay, and receive confirmation.

**Invalid Input Handling:** Test what happens when a user enters an invalid location, selects no vehicle, or leaves fields empty.Test Scenario 3: Booking with Empty Required Fields.

**Payment Processing:** Test different payment methods (card, mobile banking, cash) to ensure smooth transactions.

**Load Testing:** Check how the system handles multiple users booking trips at the same time.

**Booking Cancellation:** Verify if users can cancel a trip

**Driver Assignment:** Check if a driver is assigned properly after booking and if users get real-time updates.

**Booking History:** Ensure users can view their past and upcoming bookings correctly.

**Promo Code & Discounts:** Test if promo codes or discounts are applied correctly and reflect in the final fare.

**Notification System:** Verify if SMS, and app notifications (for booking, cancellation, driver arrival, etc.) work as expected.

**Ans 2.2:**

[Garibook Test Cases](https://docs.google.com/spreadsheets/d/15Wkc_pc0R3XvReljfProJWKNoeRYV40LJmv9WEZDaTk/edit?gid=0#gid=0)

**Step 3: : Test Execution & Defect Reporting**

[Garibook Test Cases](https://docs.google.com/spreadsheets/d/15Wkc_pc0R3XvReljfProJWKNoeRYV40LJmv9WEZDaTk/edit?gid=55429848#gid=55429848)

**Step 4: Test Closure & Release Decision**

**Ans 4.1:**

Yes! I would approve this feature for release.

**Ans 4.2:**

While I did find a few bugs (as detailed in the spreadsheet), none of them are showstoppers. They're mostly minor cosmetic issues or things that don't break the core functionality. I think they're at a level where we can fix them in a quick follow-up patch after launch. The feature itself works as intended and adds real value to the site. Holding it back for these small things would be a shame, especially since we can address them so easily afterwards. Plus, I think users will be happy to see this new functionality, even if there are a couple of tiny quirks at first.