Grubhub App - Functional E2E Test Scenarios

# Test Scenario 1: Reordering a Previously Ordered Item

* \*\*Objective:\*\* Validate the functionality of the “Reorder” feature.

\*\*Steps:\*\*

1. Launch the app and log in.
2. Navigate to Orders → Past Orders.
3. Select a previous order and tap Reorder.
4. Proceed to checkout.
5. Confirm the delivery address and payment.
6. Place the order.

\*\*Expected Result:\*\* The app should successfully repopulate the cart with all previously ordered items, preserve item customizations (e.g., extra toppings), and allow seamless checkout.

\*\*Potential Bugs to Catch:\*\*

* Missing items or incorrect item quantities in reorder.
* Crashes or freezes while repopulating the cart.

# Test Scenario 2: Filtering and Ordering from a Specific Cuisine

* \*\*Objective:\*\* Validate accurate filtering and ordering based on cuisine type.

\*\*Steps:\*\*

1. Launch the app.
2. Enter a valid delivery address.
3. Filter by “Italian” cuisine.
4. Select a listed restaurant.
5. Add an item to the cart and place the order.

\*\*Expected Result:\*\* Only Italian restaurants should be shown. The order should be placed successfully.

\*\*Potential Bugs to Catch:\*\*

* Irrelevant cuisine types being shown in the filtered list.
* Cart not updating with selected menu items.

# Test Scenario 3: Real-Time Order Tracking Verification

* \*\*Objective:\*\* Validate whether the real-time order tracking displays proper updates throughout the order lifecycle.

\*\*Steps:\*\*

1. Place an order and go to Order Tracking.
2. Monitor status changes (e.g., Confirmed → Preparing → On the way → Delivered).

\*\*Expected Result:\*\* The status should update in real time and correspond to the backend order status.

\*\*Potential Bugs to Catch:\*\*

* Status updates not reflecting correctly.
* Stuck or delayed status despite backend progression.

# Test Scenario 4: Placing an Order During Restaurant Off-Hours

* \*\*Objective:\*\* Validate business hour restrictions and error handling.

\*\*Steps:\*\*

1. Launch the app and select a restaurant.
2. Try placing an order outside the restaurant’s working hours.

\*\*Expected Result:\*\* The user should receive a clear message that the restaurant is closed, and ordering should be disabled.

\*\*Potential Bugs to Catch:\*\*

* User is allowed to add to cart and checkout despite the restaurant being closed.
* Missing or incorrect business hours information.

# Test Scenario 5: Applying a Discount Code During Checkout

* \*\*Objective:\*\* Validate discount code application and calculation logic.

\*\*Steps:\*\*

1. Add items to cart.
2. Proceed to checkout.
3. Apply a valid discount code.
4. Verify discount and final amount.

\*\*Expected Result:\*\* Discount is applied correctly, and final total reflects the discounted price.

\*\*Potential Bugs to Catch:\*\*

* Discount not being applied or removed after applying.
* Incorrect calculation in total or taxes.