## 

| **Notes to the Hackathon Participants**  1. This document is the primary document of reference for the ONDC Integration Sprint Challenges. 2. Please go through the details carefully and raise your queries before **11:00 AM** on **17th December 2021**. 3. Please post your queries [here](https://github.com/Open-network-for-digital-commerce/integration-sprint-dec21/discussions)  in the following format:   *<Network Participant Name> - <Clarification Serial Number> - <Refer Section No. of the document> - <Query Description>* |
| --- |

## **Use-Case Scenarios**

*Use case scenarios simulate real world scenarios of transactions envisaged on ONDC, with different buyer and seller distributed apps, connected in a decentralized mode.*

*These use-case scenario will cover 2 different domains - Retail and Logistics, and will have 3 different challenge statements:*

* *Retail Challenge Statement 1*
* *Retail Challenge Statement 2*
* *Logistics Challenge Statement*

*With the above, following* ***4 Scenarios*** *have been identified for the Integration sprint:*

1. *Retail Challenge Statement 1*
2. *Retail Challenge Statement 2*
3. *Retail Challenge Statement 1 + Logistics Challenge Statement*
4. *Retail Challenge Statement 2 + Logistics Challenge Statement*

*Any participant can decide to execute* ***one or more of these scenarios*** *based on the challenge statements defined below*.

## **Challenge Statements**

**2.1. Retail Challenge Statement 1**

Execute the user order journey described below:

Raj decides to order dinner for his family. Raj opens an app called “Open Commerce for All”.

a. Raj enters his location as “Koramangala, Bengaluru”. He types in the category of “pizza” and waits.

b. Raj sees a list of outlets that sell pizza. He sees that his favourite pizzas are available from a Mojo Pizza outlet near his location.

c. He selects the following by clicking “Add to Cart”:

* All Veggies Madness - Large - 1
* Italian Garden (hand tossed) - Medium - 1
* Mojo’s Chicken Special (pan crust) - Medium - 1

d. Raj adds these items to the cart and is about to checkout when he realizes he should also order garlic bread. He searches for the category “garlic bread”.

e. He sees that Mojo Pizza has several options for garlic bread.

f. He selects “Mexican stuffed garlic breadsticks” by clicking “Add to Cart”. He then verifies his order by clicking on “View Cart”.

g. Raj then proceeds with the following steps to confirm the order:

* Click on “Checkout”;
* Clicks on “Add Shipping Details” and provide all details including the landmark;
* Click on “Proceed to Pay”;
* Following payment options are available - “Cash on Delivery”, “UPI”. He selects “UPI” and clicks “Pay”. He is immediately redirected to his UPI app, “Paytm” and makes the payment.

h. After the payment is successful, he is redirected back to a screen that says, “Payment successful, confirming your order”. After confirmation of the order, Raj is redirected to an order screen that says, “Order confirmed. Packing your items”. A little while later, Raj receives a notification that says, “Searching for delivery agents near Mojo Pizza outlet”.

**2.2. Retail Challenge Statement 2**

Execute the user order journey described below:

Ram suddenly remembers at night that he needs personal care items for the next day. Ram opens an app called “Open Commerce for All”.

a. Ram enters his location as “Koramangala, Bengaluru”. He types in the category of “Personal Care” and waits.

b. He sees that a nearby supermarket has the items that he needs.

c. He selects the following by clicking “Add to Cart”:

* Closeup (small) - 1
* Sensodyne toothbrush (soft) - 3
* Patanjali Aloe Vera Gel - 1

Ram then verifies his order by clicking on “View Cart”.

d. Ram then proceeds with the following steps to confirm the order:

* Click on “Checkout”;
* Clicks on “Add Shipping Details” and verifies all details including the landmark;
* Click on “Proceed to Pay”;
* Following payment options are available - “Cash on Delivery”, “UPI”. He selects “UPI” and clicks “Pay”. He is immediately redirected to his UPI app, “Paytm” and makes the payment.

e. After the payment is successful, he is redirected back to a loader screen that says, “Payment successful, confirming your order”. After confirmation of the order, Raj is redirected to an order screen that says, “Order confirmed. Packing your items”.

**2.3. Logistics Challenge Statement**

Execute the Merchant delivery order journey described below:

Both the stores (Mojo Pizza outlet / SuperMarket) backend was pre-configured to automatically discover nearby logistics services, when an order gets confirmed based on the store location and the delivery location.

a. Upon confirmation, the store owner clicks the order.

b. The store owner sees that there are two logistics providers that deliver to the shipping address for that order.

c. The store owner quickly selects the cheapest option.

d. The store owner confirms the delivery.

e. The logistics provider assigns a delivery agent to that order and sends the delivery order details to the store backend which forwards it to the “Open Commerce” app. The buyer sees the delivery status “Delivery agent en-route to store” on his app.

The logistics provider continuously updates the delivery status to the store backend, that automatically forwards the status to the buyer.

## **Transaction Slices**

**A slice is a combination of an API call and its corresponding callback.** *e.g. a combination of confirm and on\_confirm is a slice.*

Each challenge statement defined above comprises many such slices. The different steps, in each slice, and its applicability to the buyer app or seller app is defined below.

A **round-trip transaction** *goes through the entire lifecycle, starting from search and going all the way to confirmation of an order*. There are at least 2 participants involved in a round-trip transaction. A successful round-trip transaction requires each participant to successfully complete their steps in each slice that is a part of the round-trip transaction.

Each challenge statement translates into a round trip transaction and covers multiple slices, with each slice having multiple steps that are applicable to the buyer app, seller app or in very specific cases, the gateway.

The challenge statement-wise list of slices are provided in **Section 4, 5 and 6** of this document. Below Table represents a summary of the mapping of the slices to the challenge.

|  | search, on\_search | select, on\_select | init, on\_init | confirm, on\_confirm | status, on\_status | track, on\_track |
| --- | --- | --- | --- | --- | --- | --- |
| Retail Challenge Statement 1 | 4a | 4b | 4c | 4d | 4e | 4f |
| Retail Challenge Statement 2 | 5a | 5b | 5c | 5d | 5e | 5f |
| Logistics Challenge Statement | 6a |  | 6b | 6c | 6d |  |

## **Transaction Slices for Retail Challenge Statement 1**

**4. a. Retail Network search / on\_search**, entailing following steps:

1. Buyer App : Add search intent, location and call search to ONDC Gateway
2. ONDC Gateway : broadcast search
3. Seller App 1 : add list of providers with matching items and return on\_search to ONDC Gateway
4. Seller App 2 : add list of providers with matching items and return on\_search to ONDC Gateway
5. Seller App 3 : add list of providers with matching items and return on\_search to ONDC Gateway
6. ONDC Gateway : forward on\_search from Seller App 1
7. ONDC Gateway : forward on\_search from Seller App 2
8. ONDC Gateway : forward on\_search from Seller App 3
9. Buyer App : view catalog of any provider from Seller App 1 (direct search to Seller App)
10. Buyer App : view catalog of any provider from Seller App 2 (direct search to Seller App)
11. Buyer App : view catalog of any provider from Seller App 3 (direct search to Seller App)
12. Seller App 1 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
13. Seller App 2 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
14. Seller App 3 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
15. Buyer App : select category ID inside Seller App 1 provider catalog and call search (direct search to Seller app)
16. Buyer App : select category ID inside Seller App 2 provider catalog and call search (direct search to Seller app)
17. Buyer App : select category ID inside Seller App 3 provider catalog and call search (direct search to Seller app)
18. Seller App 1 : add items under the selected category ID and return on\_search
19. Seller App 2 : add items under the selected category ID and return on\_search
20. Seller App 3 : add items under the selected category ID and return on\_search

**4. b. Retail Network select / on\_select,** entailing following steps:

1. Buyer App: add Seller App 1 item to cart and call select
2. Buyer App: add Seller App 2 item to cart and call select
3. Buyer App: add Seller App 3 item to cart and call select
4. Seller App 1: add item, calculate quote and return on\_select
5. Seller App 2: add item, calculate quote and return on\_select
6. Seller App 3: add item, calculate quote and return on\_select
7. Buyer App: remove Seller App 1 item from cart and call select
8. Buyer App: remove Seller App 2 item from cart and call select
9. Buyer App: remove Seller App 3 item from cart and call select
10. Seller App 1: remove item from cart, calculate quote and return on\_select
11. Seller App 2: remove item from cart, calculate quote and return on\_select
12. Seller App 3: remove item from cart, calculate quote and return on\_select

**4. c. Retail Network init / on\_init,** entailing following steps:

1. Buyer App: add billing & shipping details for Seller App 1 and call init
2. Buyer App: add billing & shipping details for Seller App 2 and call init
3. Buyer App: add billing & shipping details for Seller App 3 and call init
4. Seller App 1: recalculate quote, add payment terms and return on\_init
5. Seller App 2: recalculate quote, add payment terms and return on\_init
6. Seller App 3: recalculate quote, add payment terms and return on\_init

**4. d. Retail Network confirm / on\_confirm,** entailing following steps:

1. Buyer App: add promise / proof of payment to Seller App 1 and call confirm
2. Buyer App: add promise / proof of payment to Seller App 2 and call confirm
3. Buyer App: add promise / proof of payment to Seller App 3 and call confirm
4. Seller App 1: add order ID, update fulfilment state and return on\_confirm
5. Seller App 2: add order ID, update fulfilment state and return on\_confirm
6. Seller App 3: add order ID, update fulfilment state and return on\_confirm

**4. e. Retail Network status / on\_status,** entailing following steps:

1. Buyer App : get latest status of fulfillment by calling status to Seller App 1
2. Seller App 1: Send following status to Buyer App: “Agent has been Assigned” via on\_status when Delivery Buyer App receives it from Delivery Seller App
3. Seller App 1: Send following status to Buyer App: “Agent at store” via on\_status when Delivery Buyer App receives it from Delivery Seller App
4. Seller App 1: Send following status to Buyer App: “Agent has picked up items” via on\_status when Delivery Buyer App receives it from Delivery Seller App
5. Seller App 1: Send following status to Buyer App: “Agent is en-route to drop” via on\_status when Delivery Buyer App receives it from Delivery Seller App
6. Seller App 1: Send following status to Buyer App: “Agent is at drop location” via on\_status when Delivery Buyer App receives it from Delivery Seller App
7. Seller App 1: Send following status to Buyer App: “Order delivered” via on\_status when Delivery Buyer App receives it from Delivery Seller App
8. Buyer App : get latest status of fulfillment by calling status to Seller App 2
9. Seller App 2: Send following status to Buyer App: “Agent has been Assigned” via on\_status when Delivery Buyer App receives it from Delivery Seller App
10. Seller App 2: Send following status to Buyer App: “Agent at store” via on\_status when Delivery Buyer App receives it from Delivery Seller App
11. Seller App 2: Send following status to Buyer App: “Agent has picked up items” via on\_status when Delivery Buyer App receives it from Delivery Seller App
12. Seller App 2: Send following status to Buyer App: “Agent is en-route to drop” via on\_status when Delivery Buyer App receives it from Delivery Seller App
13. Seller App 2: Send following status to Buyer App: “Agent is at drop location” via on\_status when Delivery Buyer App receives it from Delivery Seller App
14. Seller App 2: Send following status to Buyer App: “Order delivered” via on\_status when Delivery Buyer App receives it from Delivery Seller App

The above steps for Seller App 2 also repeat for Seller App 3

**4. f. Retail Network track / on\_track,** entailing following steps:

1. Buyer App : get tracking info from Seller app by calling track
2. Seller App 1 : fetch tracking info from Logistics Seller App by calling track via the Logistics Buyer App
3. Seller App 1 : Forward tracking info if received via on\_track from the Logistics Seller App to the Retail Buyer App

The above steps repeat for Seller App 2, Seller App 3

## **5. Transaction Slices for Retail Challenge Statement 2**

**5. a. Retail Network search / on\_search,** entailing following steps:

1. Buyer App : Add search intent, location and call search to ONDC Gateway
2. ONDC Gateway : broadcast search
3. Seller App 1 : add list of providers with matching items and return on\_search to ONDC Gateway
4. Seller App 2 : add list of providers with matching items and return on\_search to ONDC Gateway
5. Seller App 3 : add list of providers with matching items and return on\_search to ONDC Gateway
6. ONDC Gateway : forward on\_search from Seller App 1
7. ONDC Gateway : forward on\_search from Seller App 2
8. ONDC Gateway : forward on\_search from Seller App 3
9. Buyer App : view catalog of any provider from Seller App 1 (direct search to Seller App)
10. Buyer App : view catalog of any provider from Seller App 2 (direct search to Seller App)
11. Buyer App : view catalog of any provider from Seller App 3 (direct search to Seller App)
12. Seller App 1 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
13. Seller App 2 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
14. Seller App 3 : send catalog of selected provider to Buyer App with matching items from original search (direct on\_search to Buyer App)
15. Buyer App : select category ID inside Seller App 1 provider catalog and call search (direct search to Seller app)
16. Buyer App : select category ID inside Seller App 2 provider catalog and call search (direct search to Seller app)
17. Buyer App : select category ID inside Seller App 3 provider catalog and call search (direct search to Seller app)
18. Seller App 1 : add items under the selected category ID and return on\_search
19. Seller App 2 : add items under the selected category ID and return on\_search
20. Seller App 3 : add items under the selected category ID and return on\_search

**5. b. Retail Network select / on\_select,** entailing following steps:

1. Buyer App: add Seller App 1 item to cart and call select
2. Buyer App: add Seller App 2 item to cart and call select
3. Buyer App: add Seller App 3 item to cart and call select
4. Seller App 1: add item, calculate quote and return on\_select
5. Seller App 2: add item, calculate quote and return on\_select
6. Seller App 3: add item, calculate quote and return on\_select
7. Buyer App: remove Seller App 1 item from cart and call select
8. Buyer App: remove Seller App 2 item from cart and call select
9. Buyer App: remove Seller App 3 item from cart and call select
10. Seller App 1: remove item from cart, calculate quote and return on\_select
11. Seller App 2: remove item from cart, calculate quote and return on\_select
12. Seller App 3: remove item from cart, calculate quote and return on\_select

**5. c. Retail Network init / on\_init,** entailing following steps:

1. Buyer App: add billing & shipping details for Seller App 1 and call init
2. Buyer App: add billing & shipping details for Seller App 2 and call init
3. Buyer App: add billing & shipping details for Seller App 3 and call init
4. Seller App 1: recalculate quote, add payment terms and return on\_init
5. Seller App 2: recalculate quote, add payment terms and return on\_init
6. Seller App 3: recalculate quote, add payment terms and return on\_init

**5. d. Retail Network confirm / on\_confirm,** entailing following steps:

1. Buyer App: add promise / proof of payment to Seller App 1 and call confirm
2. Buyer App: add promise / proof of payment to Seller App 2 and call confirm
3. Buyer App: add promise / proof of payment to Seller App 3 and call confirm
4. Seller App 1: add order ID, update fulfilment state and return on\_confirm
5. Seller App 2: add order ID, update fulfilment state and return on\_confirm
6. Seller App 3: add order ID, update fulfilment state and return on\_confirm

**5. e. Retail Network status / on\_status,** entailing following steps:

1. Buyer App : get latest status of fulfillment by calling status to Seller App 1
2. Seller App 1: Send following status to Buyer App: “Agent has been Assigned” via on\_status when Delivery Buyer App receives it from Delivery Seller App
3. Seller App 1: Send following status to Buyer App: “Agent at store” via on\_status when Delivery Buyer App receives it from Delivery Seller App
4. Seller App 1: Send following status to Buyer App: “Agent has picked up items” via on\_status when Delivery Buyer App receives it from Delivery Seller App
5. Seller App 1: Send following status to Buyer App: “Agent is en-route to drop” via on\_status when Delivery Buyer App receives it from Delivery Seller App
6. Seller App 1: Send following status to Buyer App: “Agent is at drop location” via on\_status when Delivery Buyer App receives it from Delivery Seller App
7. Seller App 1: Send following status to Buyer App: “Order delivered” via on\_status when Delivery Buyer App receives it from Delivery Seller App
8. Buyer App : get latest status of fulfillment by calling status to Seller App 2
9. Seller App 2: Send following status to Buyer App: “Agent has been Assigned” via on\_status when Delivery Buyer App receives it from Delivery Seller App
10. Seller App 2: Send following status to Buyer App: “Agent at store” via on\_status when Delivery Buyer App receives it from Delivery Seller App
11. Seller App 2: Send following status to Buyer App: “Agent has picked up items” via on\_status when Delivery Buyer App receives it from Delivery Seller App
12. Seller App 2: Send following status to Buyer App: “Agent is en-route to drop” via on\_status when Delivery Buyer App receives it from Delivery Seller App
13. Seller App 2: Send following status to Buyer App: “Agent is at drop location” via on\_status when Delivery Buyer App receives it from Delivery Seller App
14. Seller App 2: Send following status to Buyer App: “Order delivered” via on\_status when Delivery Buyer App receives it from Delivery Seller App

**5. f. Retail Network track / on\_track,** entailing following steps:

1. Buyer App: get tracking info from Seller app by calling track
2. Seller App 1 : fetch tracking info from Logistics Seller App by calling track via the Logistics Buyer App
3. Seller App 1 : Forward tracking info if received via on\_track from the Logistics Seller App to the Retail Buyer App

Seller App 3: add order ID, update fulfilment state and return on\_confirm

## **6. Transaction Slices for Logistics Challenge Statement**

**6. a. Logistics Network search / on\_search (Logistics Challenge),** entailing following steps:

1. Buyer App : Upon confirming Retail Order, add pickup location, drop location and package details and call search to ONDC Gateway with logistics context
2. ONDC Gateway : broadcast search from Buyer App
3. Seller App 1 : add catalog and return on\_search to ONDC Gateway
4. Seller App 2 : add catalog and return on\_search to ONDC Gateway
5. ONDC Gateway : forward on\_search from Seller App 1 to Buyer App
6. ONDC Gateway : forward on\_search from Seller App 2 to Buyer App

**6. b. Logistics Network init / on\_init (Logistics Challenge),** entailing following steps:

1. Buyer App 1 : add billing details, shipping details and call init to Seller App 1
2. Buyer App 2 : add billing details, shipping details and call init to Seller App 2
3. Seller App 1 : recalculate quote and return on\_init to Buyer App 1
4. Seller App 2 : recalculate quote and return on\_init to Buyer App 2

**6. c. Logistics Network confirm / on\_confirm (Logistics Challenge),** entailing following steps:

1. Buyer App 1 : add proof of payment and call confirm to Seller App 1
2. Buyer App 2 : add proof of payment and call confirm to Seller App 2
3. Seller App 1 : assign rider, update fulfilment status and return on\_confirm to Buyer App 1
4. Seller App 2 : assign rider, update fulfilment status and return on\_confirm to Buyer App 2

**6. d. Logistics Network status / on\_status (Logistics Challenge),** entailing following steps:

1. Buyer App 1 : get latest status of fulfillment by calling status to Seller App 1
2. Seller App 1 : Send following status to Buyer App: “Agent has been Assigned” via on\_status
3. Buyer App 1 : Forward “Agent has been Assigned“ status to Retail Buyer App via on\_status
4. Seller App 1 : Send following status to Buyer App: “Agent at store” via on\_status
5. Buyer App 1 : Forward “Agent at store“ status to Retail Buyer App via on\_status
6. Seller App 1 : Send following status to Buyer App: “Agent has picked up items” via on\_status
7. Buyer App 1 : Forward “Agent has picked up items“ status to Retail Buyer App via on\_status
8. Seller App 1 : Send following status to Buyer App: “Agent is en-route to drop” via on\_status
9. Buyer App 1 : Forward “Agent is en-route to drop”status to Retail Buyer App via on\_status
10. Seller App 1: Send following status to Buyer App: “Agent is at drop location” via on\_status
11. Buyer App 1 : Forward “Agent is at drop location” status to Retail Buyer App via on\_status
12. Seller App 1 : Send following status to Buyer App: “Order delivered” via on\_status
13. Buyer App 1 : Forward “Order delivered” status to Retail Buyer App via on\_status
14. Steps 1-13 above repeat for <Logistics Buyer App 2, Logistics Seller App 2>

**6. e. Retail - Logistics Network integration (Logistics Challenge),** entailing following steps:

1. Retail Seller App / Logistics Buyer App : Call search to logistics network as soon as Retail Seller App receives init
2. Logistics Buyer App : select any Seller App from the delivery search results and call on\_init with added delivery fee to Retail Buyer App against the same transaction\_id
3. Retail Seller App / Logistics Buyer App : Send on\_status to retail Buyer App after receiving on\_confirm from Logistics Seller App 1 against the same order ID created previously for Retail Buyer App
4. Retail Seller App / Logistics Buyer App : Send on\_status to retail Buyer App after receiving on\_confirm from Logistics Seller App 2 against the same order ID created previously for retail Buyer App

## 

## **7. Evaluation**

Each scenario will be evaluated for 2 categories of awards which are defined below, along with the process for evaluating each category.

**Award Category 1: Recognise Individual Contributions**

*Evaluation Process:*

1. A participant submits a claim along with a video recording that a particular slice is complete and is ready for demonstrating the completion
2. The submission will be done in a participant-specific sheet [here](https://drive.google.com/drive/folders/1AkdlUWdapXhNo1XJVLhOtaleIhXrchWH?usp=sharing)
3. The inspection of the claim will be based on the video submission and any subsequent inquiries by the Hackathon judging team and accordingly accept/reject the slide.
4. Any submissions based on tool simulated API calls (e.g Postman simulated API calls) instead of actual implementation of APIs will be rejected.
5. For every Scenario : One winner between BAPs and BPPs based on Maximum number of slices completed. The award will be split in case of a tie
6. Total Cash Prize per scenario: INR 95,000 for winner of each scenario

**Award Category 2: Recognize ONDC’s first ever round-trip full order transaction (simulated)**

*Evaluation Process:*

1. A participant submits a claim along with a video recording that a particular slice is complete and is ready for demonstrating the completion
2. The claim submission will be done in a sheet [here](https://docs.google.com/spreadsheets/d/12FmasdxG2mZkYmQ1cLqQi8r2iI_StIs6hmcZ9h4_a3M/edit?usp=sharing). The submission must clearly indicate the names of ALL participants involved in the round-trip transaction.
3. The inspection of the claim will be based on the video submission and any subsequent inquiries by the Hackathon judging team and accordingly accept/reject the slide.
4. Any submissions based on tool simulated API calls (e.g Postman simulated API calls) instead of actual implementation of APIs will be rejected.
5. First round-trip reported among Scenario 1 or 2 and First round-trip reported among Scenario 3 or 4, based on verification by Hackathon judging team, will be recognised as the winning submissions for this category
6. Total Cash Prize per scenario:
   1. First round-trip reported among Scenario 1 or 2: INR 2.0 lakhs split equally among all participants involved
   2. First round-trip reported among Scenario 3 or 4: INR 4.2 lakhs split equally among all participants involved