# ONDC Integration Hackathon

## Notes to the Hackathon Participants

1. This document is the primary document of reference for the ONDC Integration Hackathon.
2. Please go through the details carefully and raise your queries before **4 PM** on **1st July 2022**.
3. Please post your queries in the Discord channel.

   Steps to join Discord > Go to Discord.com > Go to Add Server use link : https://discord.gg/qpBUpqCe >> then join channel >> Discord channel.

## 1. Instructions to Participants

Participants are requested to follow these instructions that will be applicable during the Integration Hackathon.

- **Setup entry in ONDC registry**
  - Participants should setup an instance with URL that is publicly accessible and make an entry in the ONDC registry;
  - Please fill the Google form to get yourself added to the registry;
- **Reference application, sample code:**
  - ONDC Buyer Reference App (Web)
- **ONDC Buyer Reference App (APK)**
  - ONDC Buyer Reference App APK can be downloaded from here
- **Signing & Verification**
  - Generating key pairs, using authorization headers for digital signature
  - Generate keys using this link
- **ONDC Protocol Specs & Postman collection**
  - SwaggerHub Link
- **Transaction Contracts**
  - Retail (Same will applicable for Agriculture domain)
  - Logistics
- **Communication channels:**
  - Discord Channel
  - GitHub

# 2. Overview of 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 scenarios will cover 2 different domains (Agriculture [same as retail] and Logistics), and will include the following:*

- ***Scenario 1 - Execute a complete transaction, where a farmer is able to place an order for farm inputs from a producer, resulting in order confirmation;***
- ***Scenario 2 - Execute a complete transaction, where a farmer is able to place an order for farm inputs from a producer, resulting in order confirmation and fulfillment by any one logistics provider;***
- ***Scenario 3 - Execute a complete transaction, where an FPO / Mandi trader / other wholesaler is able to place an order for farm produce, resulting in order confirmation and fulfillment by more than one logistics provider;***

***Note: Authorization header validations for APIs will be enabled to ensure that participants get a chance to test their signing & verification functionality as well.***

*The user journey and detailed transaction steps, for each scenario, is defined below. Any participant can decide to execute* ***one or more of these scenarios****.*

# 3. Detailing - Scenario 1

## 3.1 User Journey

Krishna, a farmer in Bengaluru rural, decides to order farm inputs (including seeds, fertilizers & other specialty nutrients, saplings, pesticides, etc.) for himself & other farmers for sowing & plantation of fruits, vegetables, pulses, grains, commercial crops, etc. Krishna opens an app called “Open Commerce for All”.

1. Krishna enters his location as “Bengaluru, Karnataka”. He types in the category of ‘Seeds” and waits.
2. Krishna sees a list of sellers that sell seeds. He then selects the seeds he requires for himself & other farmers for their plantation. He identifies different producers that have the variety of seeds they need.
3. He selects the seeds required for planting cauliflower, brinjal, watermelon and cucumber, in various quantities.
4. Krishna adds these items to the cart and is about to checkout when he realizes he should also order fertilizers / crop nutrients, fungicide. He searches for these categories.
5. He sees that there are several options for each of these categories from Syngenta and other providers.
6. He selects fertilizers, fungicide in various quantities by clicking “Add to Cart”. He then verifies his order by clicking on “View Cart”.
7. Krishna views the commodity prices based on the order quantity placed.
8. Krishna then proceeds with the following steps to confirm the order:
    a. Click on “Checkout”;
    b. Clicks on “Add Shipping Details” and provide all details including the landmark;
    c. Click on “Proceed to Pay”;
    d. Following payment options are available - “Cash on Delivery”. He selects this option and clicks “Confirm”.
9. After confirmation of the order, Krishna is redirected to an order screen that says, “Order confirmed”.
10. The order will be self fulfilled by the producers. Krishna is able to trace the shipment from these producers to rural Bengaluru.

## 3.2 Transaction Steps

A. **Retail Network search / on_search**, resulting in following steps (note - this will be repeated once per category):
    - Buyer App : add search intent, location and call search to ONDC Gateway
    - ONDC Gateway : broadcast search
    - Seller App(s) : add list of providers with matching items and return on_search to ONDC Gateway
    - ONDC Gateway : forward on_search from Seller App(s)
    - Buyer App : view catalog of any provider from Seller App(s)
    - Seller App(s) : send catalog of selected provider to Buyer App with matching items from original search

B. **Retail Network select / on_select,** resulting in following steps:
    - Buyer App: add Seller App(s) item to cart and call select
    - Seller App(s): add item, calculate quote and return on_select
    - Buyer App: remove Seller App(s) item from cart and call select

- Seller App(s): remove item from cart, calculate quote and return on_select

C. **Retail Network init / on_init,** resulting in following steps:
   - Buyer App: add billing & shipping details for Seller App(s) and call init
   - Seller App(s): recalculate quote, add payment terms and return on_init

D. **Retail Network confirm / on_confirm,** resulting in following steps:
   - Buyer App: add promise / proof of payment to Seller App(s), add order ID and call confirm
   - Seller App(s): update fulfillment state and return on_confirm

# 4. Detailing - Scenario 2

## 4.1 User Journey

Suresh, a farmer in Bengaluru rural, decides to order farm inputs (including seeds, fertilizers & other specialty nutrients, saplings, pesticides, etc.) for himself & other farmers for sowing & plantation of fruits, vegetables, pulses, grains, commercial crops, etc and also have the orders fulfilled by logistics providers on the network. Suresh opens an app called “Open Commerce for All”.

1. Suresh enters his location as “Bengaluru, Karnataka”. He types in the category of ‘Seeds” and waits.
2. Suresh sees a list of sellers that sell seeds. He then selects the seeds he requires for himself & other farmers for their plantation. He identifies different producers in Mysuru that have the variety of seeds they need.
3. He selects the seeds required for planting cauliflower, muskmelon, bhendi and watermelon, in various quantities.
4. Suresh adds these items to the cart and is about to checkout when he realizes he should also order fertilizers / crop nutrients, fungicide. He searches for these categories.
5. He sees that there are several options for each of these categories from Syngenta and other providers in Mysore .
6. He selects fertilizers, fungicide in various quantities by clicking “Add to Cart”. He then verifies his order by clicking on “View Cart”.
7. Suresh views the commodity prices based on the order quantity placed.
8. Suresh then proceeds with the following steps to confirm the order:
    a. Click on “Checkout”;
    b. Clicks on “Add Shipping Details” and provide all details including the landmark;
    c. Click on “Proceed to Pay”;
    d. Following payment options are available - “Cash on Delivery”. He selects this option and clicks “Confirm”.
9. After confirmation of the order, Suresh is redirected to an order screen that says, “Order confirmed”. A little while later, Suresh receives a notification that says, “Searching for a logistics provider near Mysore”.
   Rahman’s seller app backend was pre-configured to automatically discover nearby logistics providers, when an order gets confirmed based on the pickup location and the delivery location.
10. Upon confirmation, Rahman clicks the order.
11. Rahman sees that there are two logistics providers that deliver to the shipping address for that order.
12. Rahman quickly selects the cheapest option.
13. Rahman confirms the delivery.
14. 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. Suresh 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 Suresh.

## 4.2 Transaction Steps

A. **Retail Network search / on_search**, resulting in following steps (note - this will be repeated once per category) :
   - Buyer App : add search intent, location and call search to ONDC Gateway
   - ONDC Gateway : broadcast search
   - Seller App(s) : add list of providers with matching items and return on_search to ONDC Gateway
   - ONDC Gateway : forward on_search from Seller App(s)
   - Buyer App : view catalog of any provider from Seller App(s)
   - Seller App(s) : send catalog of selected provider to Buyer App with matching items from original search

B. **Retail Network select / on_select,** resulting in following steps:
   - Buyer App: add Seller App(s) item to cart and call select
   - Seller App(s): add item, calculate quote and return on_select
   - Buyer App: remove Seller App(s) item from cart and call select
   - Seller App(s): remove item from cart, calculate quote and return on_select

C. **Logistics Network search / on_search,** resulting in following steps:
   - Buyer App : **init** of retail network triggers the Logistics Buyer App - add pickup location, drop location and package details and call search to ONDC Gateway with logistics context
   - ONDC Gateway : broadcast search from Buyer App
   - Seller App(s) : add catalog and return on_search to ONDC Gateway
   - ONDC Gateway : forward on_search from Seller App(s) to Buyer App

D. **Retail Network init / on_init,** resulting in following steps:
   - Buyer App: add billing & shipping details for Seller App(s) and call init
   - Seller App(s): recalculate quote, add payment terms and return on_init

E. **Retail Network confirm / on_confirm,** resulting in following steps:
   - Buyer App: add promise / proof of payment to Seller App(s) and call confirm
   - Seller App(s): add order ID, update fulfillment state and return on_confirm

F. **Logistics Network init / on_init,** resulting in following steps:
   - Buyer App : **confirm** of retail network triggers Logistics Seller App - add billing details, shipping details and call init to Seller App
   - Seller App : recalculate quote and return on_init to Buyer App

G. **Logistics Network confirm / on_confirm,** resulting in following steps:
   - Buyer App : add proof of payment and call confirm to Seller App
   - Seller App : assign delivery agent, update fulfillment status and return on_confirm to Buyer App

H. **Logistics Network status / on_status,** resulting in following steps:
   - Buyer App : get latest status of fulfillment by calling status to Seller App
   - Seller App : send following status to Buyer App: “Agent has been Assigned” via on_status
   - Buyer App : forward “Agent has been Assigned” status to Retail Buyer App via on_status
   - Seller App : send following status to Buyer App: “Agent at store” via on_status
   - Buyer App : forward “Agent at store” status to Retail Buyer App via on_status
   - Seller App : send following status to Buyer App: “Agent has picked up items” via on_status
   - Buyer App : forward “Agent has picked up items” status to Retail Buyer App via on_status

- Seller App : send following status to Buyer App: “Agent is en-route to drop” via on_status
- Buyer App : forward “Agent is en-route to drop” status to Retail Buyer App via on_status
- Seller App : send following status to Buyer App: “Agent is at drop location” via on_status
- Buyer App : forward “Agent is at drop location” status to Retail Buyer App via on_status
- Seller App : send following status to Buyer App: “Order delivered” via on_status
- Buyer App : forward “Order delivered” status to Retail Buyer App via on_status

I. **Retail Network track / on_track,** resulting in following steps:

- Buyer App : get tracking info from Seller app by calling track
- Seller App : fetch tracking info from Logistics Seller App by calling track via the Logistics Buyer App
- Seller App : forward tracking info, if received, via on_track from the Logistics Seller App to the Retail Buyer App

# 6. Detailing - Scenario 3

## 6.1 User Journey

Salim, on behalf of a group of farm & dairy wholesalers in Bengaluru, decides to order fruits, vegetables, dry fruits, cane sugar and milk through local farm & dairy cooperatives in various parts of Karnataka. Salim opens an app called “Open Commerce for All”.

1. Salim enters his location as “Bengaluru”. He types in the category of ‘Fruits” and waits for the results.
2. Salim sees a list of sellers that sell fruits. He then selects different varieties of apple & mango to filter out the sellers that sell the specific variety. He identifies that a farmer cooperative, near Chamarajanagar, has the required variety of fruits.
3. He selects different quantities of these fruits, adds these items to the cart and then repeats this for vegetables, dry fruits, cane sugar (jaggery) and milk. He then verifies his order by clicking on “View Cart”.
4. Salim views the pricing for the commodities in the order.
5. Salim then proceeds with the following steps to confirm the order:
    a. Click on “Checkout”;
    b. Clicks on “Add Shipping Details” and provide all details including the landmark;
    c. Click on “Proceed to Pay”;
    d. Following payment options are available - “Credit card”. He selects this option and clicks “Confirm”.
6. After confirmation of the order, Salim is redirected to an order screen that says, “Order confirmed”. A little while later, Salim receives a notification that says, “Searching for delivery agents near Chamarajanagar”.
7. The farm & dairy cooperative seller app backend was pre-configured to automatically discover nearby logistics services, when an order gets confirmed based on the farm location and the delivery location.
8. Upon confirmation, the order is accepted.
9. There are multiple logistics providers that could deliver to the shipping address for the order. The farmer cooperative backend was configured to quickly select the cheapest option & the option for quickest delivery and thereafter confirm the delivery.
10. Each of the logistics providers 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. Salim sees the delivery status “Delivery agent en-route to store” on his app.

The logistics providers continuously update the delivery status to the store backend, which automatically forwards the status to Salim.

Static attributes to be updated by the Seller for the following:

11. Salim shall be able to track the shipment
12. Salim should be able to verify the weight of the produce provided by Raju. Salim can request for details from the weighbridge services.
13. Salim should be able to verify the quality of the produce delivered by the farmer cooperative, through either a quality service provider on ONDC to perform the assessment or by verifying the quality certificate shared with the shipment by the farm & dairy cooperative.

These static values will be fetched from the various service providers similar to the information sourced from the logistics service providers.

## 6.2 Transaction Steps

- **Retail Network search / on_search**, resulting in following steps (note - this will be repeated once per category) :
    - Buyer App : add search intent, location and call search to ONDC Gateway
    - ONDC Gateway : broadcast search
    - Seller App(s) : add list of providers with matching items and return on_search to ONDC Gateway
    - ONDC Gateway : forward on_search from Seller App(s)
    - Buyer App : view catalog of any provider from Seller App(s)
    - Seller App(s) : send catalog of selected provider to Buyer App with matching items from original search
- **Retail Network select / on_select,** resulting in following steps:
    - Buyer App: add Seller App(s) item to cart and call select
    - Seller App(s): add item, calculate quote and return on_select
    - Buyer App: remove Seller App(s) item from cart and call select
    - Seller App(s): remove item from cart, calculate quote and return on_select
- **Logistics Network search / on_search,** resulting in following steps:
    - Buyer App : **init** of retail network triggers the Logistics Buyer App - add pickup location, drop location and package details and call search to ONDC Gateway with logistics context
    - ONDC Gateway : broadcast search from Buyer App
    - Seller App(s) : add catalog and return on_search to ONDC Gateway
    - ONDC Gateway : forward on_search from Seller App(s) to Buyer App
- **Retail Network init / on_init,** resulting in following steps:
    - Buyer App: add billing & shipping details for Seller App(s) and call init
    - Seller App(s): recalculate quote, add payment terms and return on_init
- **Retail Network confirm / on_confirm,** resulting in following steps:
    - Buyer App: add promise / proof of payment to Seller App(s) and call confirm
    - Seller App(s): add order ID, update fulfillment state and return on_confirm
- **Logistics Network init / on_init,** resulting in following steps:
    - Buyer App : **confirm** of retail network triggers Logistics Seller App - add billing details, shipping details and call init to Seller App
    - Seller App : recalculate quote and return on_init to Buyer App
- **Logistics Network confirm / on_confirm,** resulting in following steps:
    - Buyer App : add proof of payment and call confirm to Seller App
    - Seller App : assign delivery agent, update fulfillment status and return on_confirm to Buyer App
- **Logistics Network status / on_status,** resulting in following steps:
    - Buyer App : get latest status of fulfillment by calling status to Seller App
    - Seller App : send following status to Buyer App: “Agent has been Assigned” via on_status
    - Buyer App : forward “Agent has been Assigned” status to Retail Buyer App via on_status
    - Seller App : send following status to Buyer App: “Agent at store” via on_status
    - Buyer App : forward “Agent at store” status to Retail Buyer App via on_status
    - Seller App : send following status to Buyer App: “Agent has picked up items” via on_status

    - Buyer App : forward “Agent has picked up items” status to Retail Buyer App via on_status
    - Seller App : send following status to Buyer App: “Agent is en-route to drop” via on_status
    - Buyer App : forward “Agent is en-route to drop” status to Retail Buyer App via on_status
    - Seller App : send following status to Buyer App: “Agent is at drop location” via on_status
    - Buyer App : forward “Agent is at drop location” status to Retail Buyer App via on_status
    - Seller App : send following status to Buyer App: “Order delivered” via on_status
    - Buyer App : forward “Order delivered” status to Retail Buyer App via on_status
- **Retail Network track / on_track,** resulting in following steps:
    - Buyer App : get tracking info from Seller app by calling track
    - Seller App : fetch tracking info from Logistics Seller App by calling track via the Logistics Buyer App
    - Seller App : forward tracking info, if received, via on_track from the Logistics Seller App to the Retail Buyer App

# 7. Evaluation

The evaluation and awarding process is defined below.

***Evaluation Process:***

1. To qualify for an award, a participant needs to complete **all steps for a use-case scenario**.
2. A participant submits a claim along with a video recording that all steps for a use-case scenario are complete.
3. For submission, a participant should fill the submission sheet, and generate a pull request in the integration hackathon github repository for the filled sheet.
4. The submission will be done in a participant-specific sheet. Submission deadline is 3rd July 2022, midnight.
5. The inspection of the claim will be based on the video submission and any subsequent inquiries by the Hackathon judging team and will accordingly accept/reject the submission.
6. Any submissions based on tool simulated API calls (e.g Postman simulated API calls) instead of actual implementation of APIs will be rejected.
7. Any reference app will not be evaluated for an award.