# Backend Technical Assessment: Order Transaction

# Objective

Design and implement Service(s) for handling Order Transaction with domain driven approach. Please use the following requirement and rule when working on the assessment. Your assessment will be scored based on the key indicators stated in *Assessment Aspects*section.

The service will be serving as a backend API for a client app in a RESTful manner with JSON as data format. It's best to focus on the main domain here: Order Transaction. You are free to add your assumption to ease your work, for any additional assumption please include in your *readme* file.

Develop the task with the mindset that it must be ready for production. A great plus if the app is deployed to hosting (heroku/aws/azure/digital ocean). Please send source code in zip file and we will review the codes.

## Requirement

The situation in an online store are stated below. This scenario we would like to focus on basic transactions that happen in general online stores in Indonesia.

1. Order transaction involves the following actors: customer and admin.
2. Product dictionary → free to define product metadata and values as necessary, can be hardcoded,
   1. Product has quantity; product with quantity 0 can not be ordered
3. Order transaction process flow and verification; single transaction has the following steps:
   1. Customer can add product to an order
   2. Customer can submit an order and the order is finalized
   3. Customers can only pay via bank transfer
   4. When placing an order the following data is required: name, phone number, email, address
   5. When an order is submitted, the quantity for the ordered product will be reduced based on the quantity.
   6. An order is successfully submitted if all ordered products are available.
   7. After an order is submitted, customer will be required to submit payment proof
   8. After an order is submitted, the order is accessible to admin and ready to be processed
   9. Admin can view order detail
   10. Admin can verify the validity of order data: customer name, phone, email, address, payment proof
       1. Given an order is valid, then Admin will prepare the ordered items for shipment
       2. Given and order is invalid, then Admin can cancel the order
   11. Admin can mark the order as shipped and update the order with Shipping ID
   12. Customer can check the order status for the submitted order
   13. Customer can check the shipment status for the submitted order using Shipping ID

## Instruction

* Please write your project in NodeJS, or Java. Using framework is encouraged.
* Please use first commit to initialise framework/libraries only, and the subsequent commits with proper commit logs
* You must submit your solution in zip file
* Your application must be able to run locally or deployed to cloud infra that compatible with \*NIX/Linux environment
* If your application must be run locally, please have reasonable step (no more than 5 steps) and provide necessary readme
* If you decide to deploy your solution in cloud infra, please make sure it's still up and accessible until the next 2 weeks after your submission

## Assessment Aspects

* Code cleanliness
* Feature completeness
* Application design and abstraction layer
* Quality assessment with unit test and or functional API test
* Runnable locally in other local development machine and or deployed to cloud infrastructure