CSBC 1000: Fundamentals for Back-End and Blockchain Development

Milestone 1

Submitted to:

Prof. Tomisin Jenrola

Submitted by:

Sanjana Rajesh Pai
Manik Jain
Deep Bhatt

Overview

We developed the backend for a peer to peer system for distributors and logistics companies. The client wanted two core functionalities to be implemented i.e. bidding and escrow payments, which we have been able to successfully implement.

The distributor would be able to create an auction for delivery jobs, along with choosing a delivery contract and payout a portion of the total delivery cost in ETH (wei).

The logistics company will be able to view all the jobs , place bids and update the status of delivery accordingly . They will be able to get paid a certain sum of the total amount as and when they proceed and provide status updates.

Use Case Acceptance Criteria and API calls

Distributor:

1. **Add distributor** : system should allow to add new distributor

   Input :
   a) name : string

   Acceptance criteria : A distributor with system generated id and default rating 0 should be added to the system.

   API call : POST /distributor

2. **View all operating logistics companies** : should return a list of all logistic companies with following details

   a) logistics_cmpy_pk: string
   b) name : string
   c) areaServed : [worldwide, specific location like North America]
   d) successfulOrders : number
   e) disputedOrders : number
   f) rating : number {5,4,3,2,1,0}
   g) freight : {trucks : number, planes : number, containers : number}

   Acceptance criteria : an array of logistic company details is returned

API call : GET /logisticsCompanies ⇒ LogisticCompany [ ]

3. **View logistics company details for a specific logistics company id**: system should return details for input logistics company, if it exists in the database

   Input : logistics_cmpy_pk: string

   Output :
   b) name : string
   c) areaServed : [worldwide, specific location like North America]
   d) successfulOrders : number
   e) disputedOrders : number
   f) rating : number {5,4,3,2,1,0}
   g) freight : {trucks : number, planes : number, containers : number}

   Acceptance criteria : logistics company's details are returned

   API call : GET /logisticsCompany/:pk

4. **Place order**: user places an order for delivery with a selected logistics company upon getting the quotation

   Input:
   a) quotationId
   b) preferred freight(optional)

   Output : If order has been confirmed, system returns with

   a) orderId : number
   b) invoiceId : number
   c) carrierId : number
   d) orderStatus : {pending, confirmed, declined}
   e) pickupDate : {timestamp, null}
   f) expectedDeliveryDate : {timestamp, null}

   Acceptance criteria : system should return the order specific details

   API call : PUT /order/:quotationId OrderInfo

5. **Track Order**: distributor can track the current status of order using the order id

   Input:

order id : number

Output :
a) orderId : number
b) invoiceId : number
c) logistics_cmpy_pk: string
d) orderStatus : {quotationRequested, awaiting Confirmation, pickUp, in-transit, delivered, delayed}- confirmation (20%), pick up(30%),in-transit , delivery(50%) ,delayed (20%)
e) pickupDate : timestamp

Acceptance criteria: system should return real time information about the order

API call : GET /order/track/:orderId

6. **Initiate Billing**: Once the order has been delivered, the system will automatically generate a billing receipt and assign rating to the carrier.

   Input:
   a) orderId : number
   b) rating : number
   c) status ['transaction initiated' / 'paid']

   Output :
   a) billId : number
   b) invoiceId : number
   c) carrierId : number
   d) orderStatus : {delivered, delayed}
   e) transactionId : number

   API call : PUT /order/bill BillInfo

7. **View all orders placed/quoted:** distributor is able to view the orders that they have placed/quoted

   Input :
   organisation id of current distributor
   a) orgId : number

   Acceptance criteria : an array of orders is returned

   API call : GET /order/:orgId

8. **Create auction :** Distributors are able to create an auction id for the logistics companies and view the bids put up by the logistics companies.

   a) distributor_pk: string,
   b) from: string,
   c) to: string,
   d) auction_end_date: timestamp,
   e) status: string

   API call: POST /auction

Logistics Company:

1. **Add Logistics Company :** system should allow to add new company

   Input :
   a) name : string
   b) areaServed : [worldwide, specific location like North America]
   c) freight : {trucks : number, planes : number, containers : number}

   Acceptance Criteria : A new logistics company with default 0 rating, successful and disputed order should be created in the system

   API call : POST /logistic_cmpy

2. **Update location served:** logistics company should be able to update the location area served

   Input:
   a) Logistic_cmpy_pk : string
   b) areaServed : String

   API call : PUT /logistic_cmpy/areaServed

3. **Update name :** logistics company should be able to update its name

   Input :
   a) logistic_cmpy_pk : string
   b) name : string

   API call : PUT /logistic_cmpy/name

4. **Update freight information:** logistics company should be able to update freight information

Input:
a) logistic_cmpy_pk : string
b) {trucks : number, planes : number, containers : number}

API call : PUT /logistic_cmpy/freight

5. **Bid for an auction:** Logistics companies should be able to add a bid for the auction that distributor is holding .

   f) distributor_pk: string,
   g) from: string,
   h) to: string,
   i) auction_end_date: timestamp,
   j) status: string

6. **Track order:** logistics company can track the current status of order using the order id

   Input:
   order id : number

   Output :
   a) orderId : number
   b) invoiceId : number
   c)logistic_cmpy_pk : string
   d) orderStatus : {quotationRequested, pickUp, in-transit, delivered, delayed}
   e) pickupDate : timestamp

   Acceptance criteria : system should return real time information about the order

   API call : GET /order/track/orderId

7. **Update order status:** logistics company should be able to update the status of the order. Possible values {'confirmed','picked up', 'in transit', 'delivered', 'delayed'}

   Payment per status (escrow) {'confirmed 20%','picked up 30%', 'in transit', 'delivered 50% '}

   Input:
   order id: number

Output:
OrderId : number
status: [‘confirmed’, ‘picked up’, ‘in transit’, ‘delivered’]

API call : PUT /order/status

8. **Rate distributor post billing:** Once order has been delivered and the billing transaction has been completed, a rating will be assigned to the distributor

Business Rule :
rating will be assigned as per the order delivery
a) excellent: 5
b) good: 4
c) average: 2
d) poor: 0

Input:
a) order id
b) distributor_pk
b) rating

API call : PUT /order/rating

9. **View All orders received :** carrier is able to view the orders that they have received.
Input : public key of current logistics company
a) orgId : number

Acceptance criteria : an array of orders received is returned

API call : GET /order/:orgId

Dependencies

- Language : ExpressJS v4.17.1
- Node : Node v8

- Dependencies:
  1. "@truffle/hdwallet-provider": "^1.1.0",
  2. "bignumber.js": "^9.0.1",
  3. "firebase-admin": "^8.10.0",
  4. "firebase-functions": "^3.6.1",
  5. "web3": "^1.3.0"