|  |
| --- |
| Design Document for Shipment Tracker |

VERSION HISTORY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Reviewed by** | **Comment** |
| 1.0 | August 18, 2021 | Santosh Kumar Chaudhary, Vikas Kumar | Somarshi |  |

DOCUMENT DEFINITION

|  |  |
| --- | --- |
| **Name** | **Definition** |
| Shipment Tracker | This document describes about the project design of Shipment Tracker |

TABLE OF CONTENTS

1. Data Model: 3

2. User Story 1: 5

3. User Story 2: 5

5. User Story 3-5: 6

6. User Story 6-8: 7

7. User Story 9-12: 8

8. API 1 9

9. API 2 9

10. API 3 10

11. API 3 10

# **Data Model:**

**Collection: Order**

|  |
| --- |
| \_id |
| order\_id |
| shipment\_id<ref> |
| product\_id<ref> |
| orderQty |
| orderPrice |
| orderDate |
| creationDate |
| updatedDate |

**Collection: Customer**

|  |
| --- |
| \_id |
| cust\_id |
| cust\_name |
| address1 |
| address2 |
| address3 |
| cust\_email |
| Phoneno |
| CreationDate |
| UpdatedDate |

**Collection: Product**

|  |
| --- |
| \_id |
| ProductID |
| Name |
| Description |
| ManDate |
| ExpDate |
| AvailableQTY |
| CreationDate |
| UpdatedDate |

**Collection : Carrier**

|  |
| --- |
| \_id |
| carrier\_id |
| carrier\_name |
| Address |
| phoneno |
| description |

**Collection: ShipmentStatus**

|  |
| --- |
| \_id |
| shipment\_id |
| creationDate |
| updatedDate |
| Status{} |
|  |
| deliveryOTP |
| reasonOfDelay |
| order\_id<ref> |
| cust\_id<ref> |
| delivery\_id<ref> |

**Collection : Warehouse**

|  |
| --- |
| \_id |
| warehouse\_id |
| inchargename |
| phoneno |
| address1 |
| address2 |
| name |

**Collection : Delivery**

|  |
| --- |
| \_id |
| Expecteddeliverydate |
| Pickupdate |
| Reasonofdelay |
| delivery\_id |
| order\_id<ref> |
| wareouse\_id<ref> |
| carrier\_id<ref> |
| shipment\_id<ref> |

# **User Story 1:**

This user story starts once order is created.

On Order creation API will be called to create an entry to update shipment status collection with the new ordereid and status as “processing”.

Technical Specs :

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{type:"processing",date:<DateString>},

creationDate:<DateString>

delivery:{

carrierId:<carrier\_id\_ref>,

leadTime:<num\_of\_days>,

pickupDate:<DateString>,

warehouseId:<warehouse\_id\_ref>}}

# **User Story 2:**

Send an email via a new API once the above data entry is done with

subject: Your order delivery details...Use an

html template to mail the customer the below body:

Hi<userName>,

The delivery of your order has been created. Please find the below details.

OrderID, Carrier Name, Product Name, Qty, Class,

Expected DayofArrival, Status

Thanks,

Logisticsteam

+Tel:111111111

a. User name will be fetched from user customer collection

b. Product name will be fetched from Product collection

c. Product Id will be retrieved from Order collection

# **User Story 3-5:**

3. Once the order is picked up by the courier, add status to "pickedup" one existing status

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"pickedUp",

date:<DateString>

}

}

API will Return Successfully picked up from warehouse

1. If courier delays pick up beyond the pickup date, send an email to

Logistics team to look with the courier on the delay reason...

Update the "shipment" collection "delayed" and add a lead time of 3days (default) to the "delivery" collection

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"delayed",

date:<DateString

},

reason\_of\_delay:"Due to heavy rain"

}

1. Let the logistics team manually update via the API the "reason\_of\_delay" in the "delivery" collection, a scheduled job should run to update the user about the delay via email

# **User Story 6-8**

6. Once the courier brings the product to courier mainhub, then add

Status "deliveredToHub" and send an email to the user

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"deliveredToHub",

date:<DateString>

}

}

7. Once the courier is in transit from main hub to last mile delivery location, add status "inTransitFromHub"

And send an email to the user

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"inTransitFromHub",

date:<DateString>

}

}

8. Once the courier reaches the local hub, add status "deliveredToLocal" and send an email to the user

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"deliveredToLocal",

date:<DateString>

}

}

# **User Story 9-12:**

9. Once the courier is out for delivery from local hub, update the status to "outForDelivery" and

Send an email to the user with a generated OTP for

The user to show the delivery person

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"outForDelivery",

date:<DateString>

}

}

10. Once Delivered, add the status to "delivered" and send an email to the user

{

orderId:<order\_number\_ref>,

customerId:<customer\_id\_ref>,

status:{

type:"delivered",

date:<DateString>

}

}

11. At any point of time, there is an exception during delivery, update there as on and expected

Date accordingly as in point 4

12. Get delivery status at any point of time using

/orderId path parameter

# **API 1:**

This Api will be called once order is created successfully to Create an entry in ShipmentStatus Collection.

**Input parameter** : orderId, customerId, carrierID, warehouseId

**Output**: this API will return shipmentId,carrierID, expectedDeliveryDate and orderId.

* + ShipmentId will be generated randomely.
  + This API will update shimentStaus collection with below data:

orderId, deliveryId , customerId , shipmentId , status [ ] , creationDate, orderDate , updatedDate

* creationDate will be system time while first entry
* This API will update Deliver collection with below data:

Delivery Collections: carrierId, expectedDeliveryDate , pickupDate , orderId , warehouseId , reasonOfDelay [desc : " " , date : " " ]

**Assumptions:**

* + Order table will pre- populated
  + Other master data like product info, customer info, warehouse info will be pre populated in respective collections

1. Create and then updated api

# **API 1 :**

This Api is meant for sending a mail to user by using a mail template once the API 1 returns successfully.

**Input parameter** : orderId, customerId deliveryId, carrierID

**Output**: mail sent successfully.

**Mail Template:**

Hi<userName>,

The delivery of your order has been created. Please find the below details.

OrderID, Carrier Name, Product Name, Qty, Class,

Expected DayofArrival, Status

Thanks,

Logisticsteam

+Tel:111111111

* User name will be fetched from user customer collection
* Product name will be fetched from Product collection
* Product Id will be retrieved from Order collection

Separate api not required. Only method call will suffish.

# **API 2:**

This Api will be called to update status in ShipmentStatus Collection whenever shipment status changed.

**Input parameter** : orderId, customerId, carrierID, ShipmentID, statusType

**Output**: successfully updated.

This will call mailer function as well.

# **API 3:**

This Api will be called to update reasonOfDelay in ShipmentStatus Collection whenever delay happened.

**Input parameter** : orderId, customerId, carrierID, ShipmentID, reasonOfDelay

**Output**: successfully updated.

Get api : pass order id and get status of order

# 