**DHL Next Generation Freight Station**

**(NGFS)**

**Revision 1.1**

**Client:**

DHL Global Forwarding

11601 West Touhy Avenue

Building 895

Upper Express Drive

Chicago, IL 60666

United States

Revision History

|  |  |  |
| --- | --- | --- |
| **RELEASE NUMBER** | **DATE** | **DESCRIPTION** |
| 1.0 | 4/20/2020 | Initial Release |
| 1.1 | 5/14/2020 | Added description for most of the sections |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Contents

[1.1 Project Summary 4](#_Toc40597856)

[1.2 Project Scope 4](#_Toc40597857)

[1.3 System Context 6](#_Toc40597858)

[1.4 High Level Architecture 8](#_Toc40597859)

[1.5 NGFS Sequence Diagram 11](#_Toc40597860)

[Sequence Component 12](#_Toc40597861)

[1.5.1 Antena 12](#_Toc40597862)

[1.5.2 UWB Module 12](#_Toc40597863)

[1.5.3 Compute Module 12](#_Toc40597864)

[1.5.4 WIoT 12](#_Toc40597865)

[1.5.5 Track And Trace component 13](#_Toc40597866)

[1.5.6 Device Status Component 13](#_Toc40597867)

[1.5.7 Asset Tracking socket 13](#_Toc40597868)

[1.5.8 UI Dashboard 14](#_Toc40597869)

[1.6 NGFS Deployment Topology 15](#_Toc40597870)

[1.7 NGFS Dashboard Architecture 16](#_Toc40597871)

[1.7.1 *Warehouse* Layout component : 17](#_Toc40597872)

[*1.7.2* *Shipment Tracking component:* 19](#_Toc40597873)

[*1.7.3* *Asset Tracking component:* 19](#_Toc40597874)

[*1.7.4* *Heat Map component:* 20](#_Toc40597875)

[*1.7.5* *Exception component:* 21](#_Toc40597876)

[*1.7.6* *Admin component:* 21](#_Toc40597877)

[1.7.7 *Troubleshooting component:-* 22](#_Toc40597878)

[1.8 NGFS Dashboard 23](#_Toc40597879)

[1.9 NGFS Entity Diagram 24](#_Toc40597880)

## Project Summary

The Solution will track, trace, and monitor expensive freight - up to 3500 products - throughout the warehouse environment within two (2) meters of accuracy by combining AR/Dead Reckoning, Visualization, Reporting/Analytics, Mapping, and Alerts/Events/Notifications technology to create a visible, smart warehouse environment. The Solution will allow DHL to detect, accurately calculate, transmit & process freight location in real time 24 hours a day 7 days a week

As part of the overall solution proposed, Watson IoT solution services identified as the backend operational platform that receives information from SmartCone Device Environment systems in the solution. For visualization of shipment , asset etc, it uses cutting edge visual technologies like React.js,Node.js and IBM carbon design system which gives insight into the warehouse site to make better operation decisions to shop floor manager and supervisor.

## Project Scope

Under this project IBM will provide services to DHL for the development and support of the Solution to enable DHL to achieve accuracy of their indoor freight location solution to determine the location of DHL’s assets within two (2) meters of accuracy in up to six U.S. designated facilities

The Solution shall meet the following requirements:

**Forklift RFID System (Edge Device) Requirements**

a) connectivity to DHL provided WIFI

b) use of HTTP Proxy to send HTTP data

c) use of Socks proxy for MQTT data and non-HTTP data

d) ability to read RFID tags that start with "D\_"

e) calculate the location at the time of the RFID tag read

f) transfer location data, RFID tag data, forklift ID, date time to IoT platform within 60 seconds

g) connect to DHL provided forklift power source

h) durability for warehouse use (withstand a reasonable level of vibration, temperature)

**Indoor Locating System (Beacon) Requirements**

a) coverage of 460,000 square feet

b) connectivity to single DHL provided network

**Device Management Requirements**

a) ability to be configured remotely

b) ability to be monitored remotely

c) display automated dashboard alerts for beacon and RFID malfunctioning or power loss

d) Edge Device monitoring to be performed remotely

e) automated alerts for edge device malfunctioning or power loss

f) firmware updates to both Beacons & Edge devices can be performed remotely

g) security updates to Beacons & Edge devices can be performed remotely

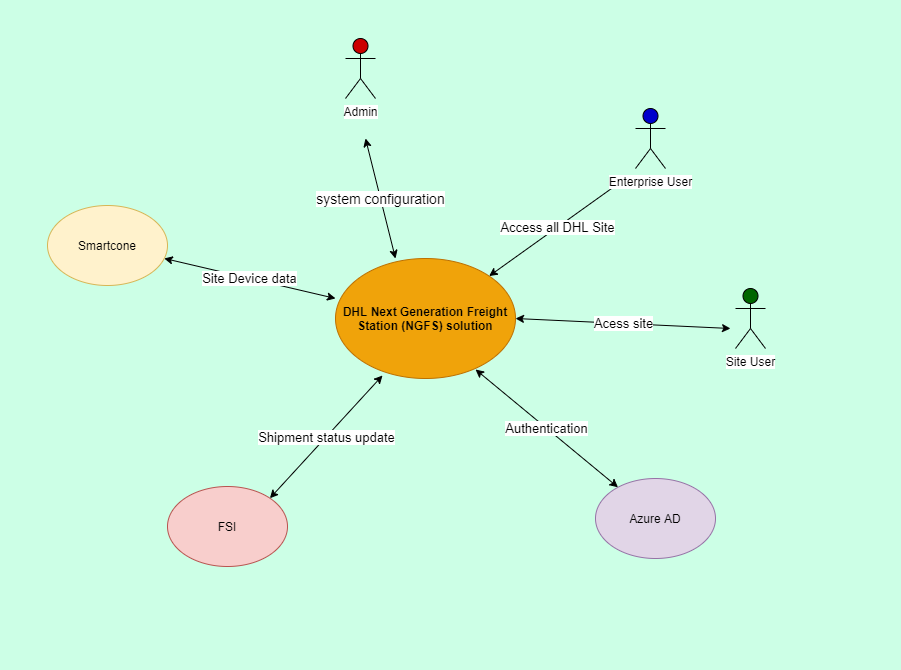
**IBM Cloud Requirements**

a) IBM Cloud will store the data transmitted by edge devices

b) IoT platform has available API's to interface with the platform and additional APIs may be needed to manage the stored DHL data

c) IoT platform analytics service to provide advanced analytics capability

## System Context



|  |  |  |
| --- | --- | --- |
| Sr No | Actor/System | Description |
| 1 | Admin | Admin User will carry out following system configuration   * User Management * Layout Management * System Properties * Troubleshooting |
| 2 | Enterprise User | Enterprise User can access all DHL site and can access following:-   * Shipment Tracking * Asset tracking * Heatmap * KPI * Shipment playback * Exception |
| 3 | Site User | Site User can access particular DHL site assigned to him and can access following:-   * Shipment Tracking * Asset tracking * Heatmap * KPI * Shipment playback * Exception |
| 4 | SmartCone | Smart cone will provide following device data to WIoT   * Forklift data with shipment * Forklift data without shipment * Device Status data |
| 5 | FSI | NGFS Solution will update shipment location data to FSI  FSI will update shipment status to NGFS solution |
| 6 | Azure AD | NGFS Solution will authenticate using Azure AD |

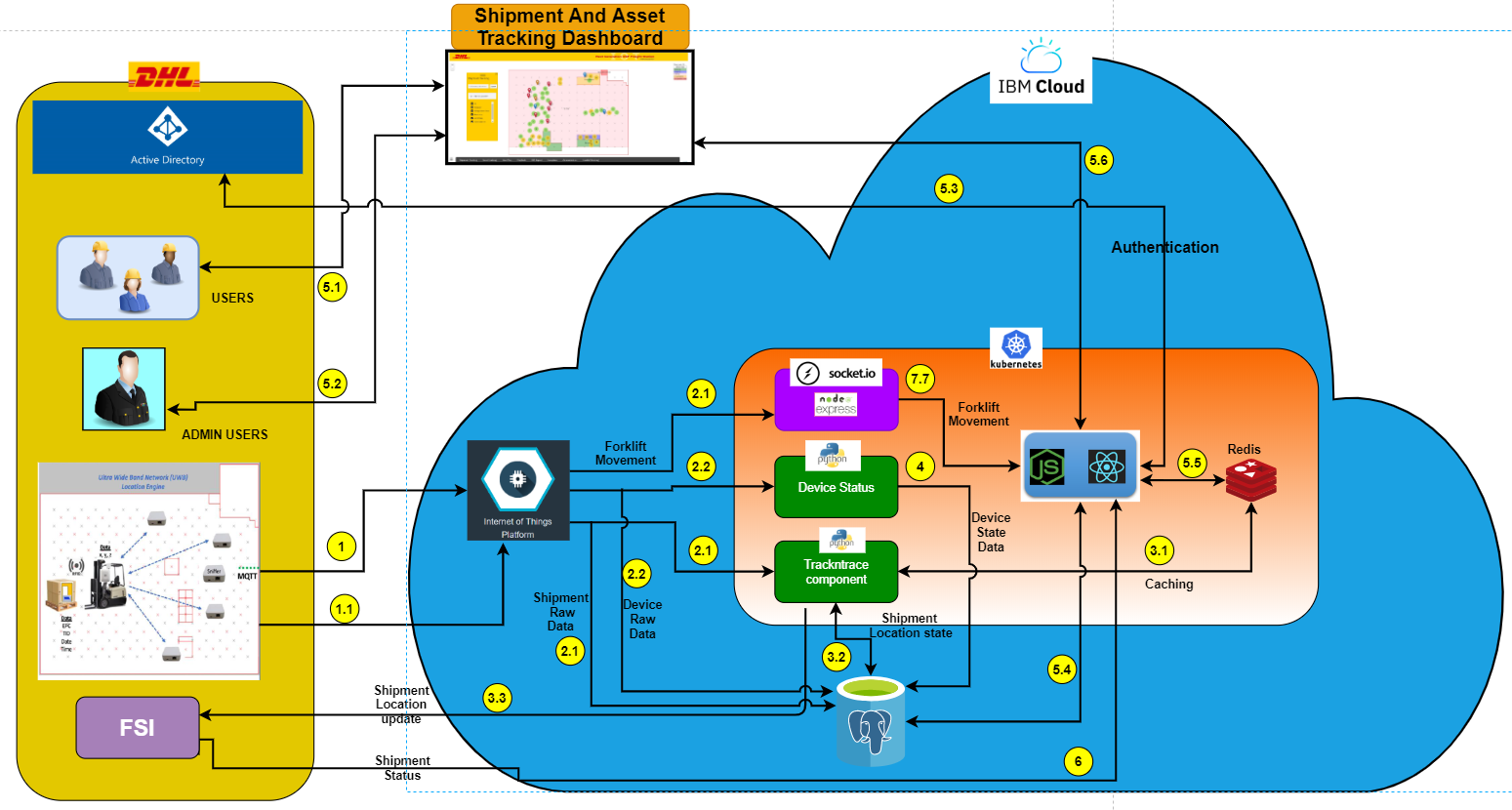
## High Level Architecture

The architecture is combination of hardware, firmware and software deployed across DHL N/W and IBM Cloud.

DHL N/W consists of DHL’s applications, SmartCone devices that provides forklift location as well shipment EPC details.

IBM Cloud consists of fully managed Maximo Worker Insight’s IoT platform, storage services and Kubernetes container service to deploy containerized apps.

Below is the architecture to provide smart warehouse environment



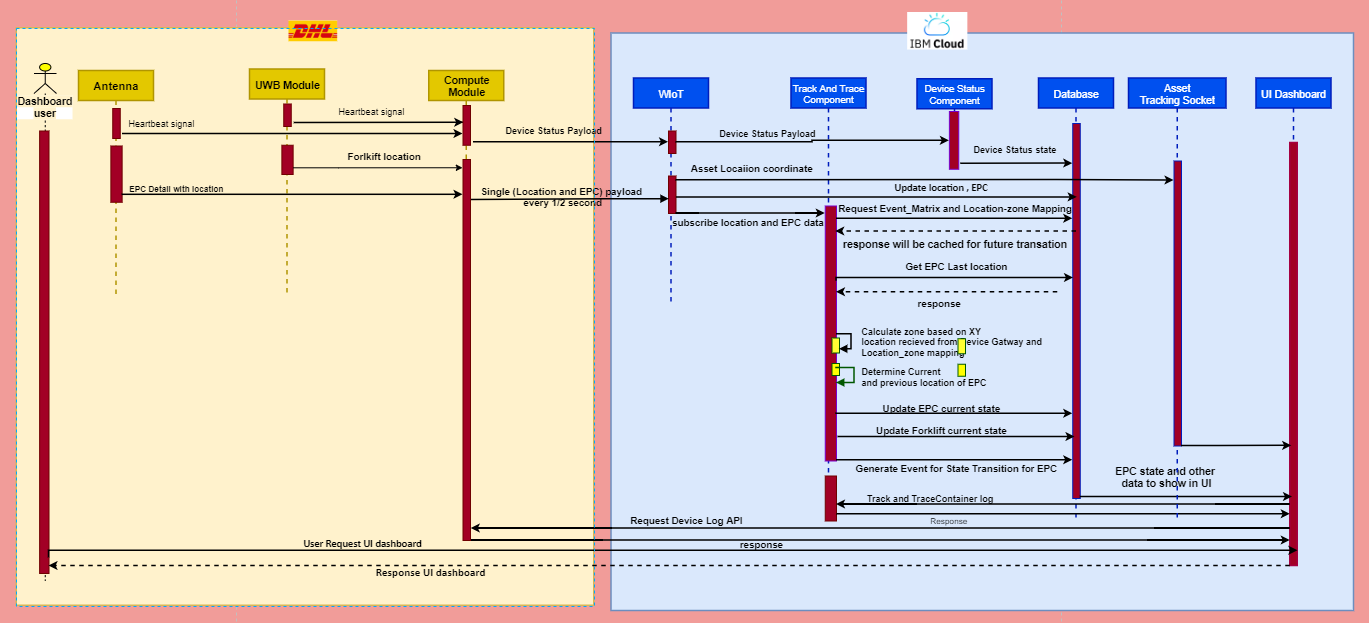
Various interactions in the architecture can be details as:

|  |  |  |
| --- | --- | --- |
| **Steps** | **Description** | **Remark** |
| 1 | SmartCone Cone publishes forklift with and without EPC payload to WIoT’s forklift Gateway | Example Payload  {"EPC": "444835363636363137335f313632", "fromLocation": {}, "toLocation": {},"forkliftLocation": {"X": 50, "Y": 52}, "timestamp": "2020-03-15T08:51:48.111Z","ConeID": "anchor\_tag\_1"} |
| 1.1 | SmartCone Cone publishes devise status payload to WIoT’s devicestatus Gateway | Example Payload :-  {"ConeID":"ORDF01","DeviceType":"forklift","Mode":"Charging","LocationTag":{"ID":1,"status":"ON"},  "RFIDReader":{"ID":1,"status":"IDLE"},"ComputeModule":{"ID":1,"status":"ON",  "telemetry":{"ram":"50%","cpu":"20%","storage":"95%"}},"timestamp":"2019-04-03T13:35:54Z",  "GWtimestamp":"2019-04-03T13:36:04Z"} |
| 2.1 | WIoT’s forklift Gateway will consume by following component   * Sockect.io component which will push data directly to Dashboard UI * Trackntrace component to do shipment and forklift state management. |  |
| 2.2 | WIoT’s devicestatus Gateway will consume by Device Status component to do device status state management |  |
| 3.1 | Trackntrace component will load master data like layout and system properties from redis cache |  |
| 3.2 | Trackntrace component will determine current state of shipment and forklift along with its current location. After determining state, data will persist into Database |  |
|  |  |  |
| 3.3 | After determining location of shipment ,Trackntrace component will send location data of shipment to FSI |  |
| 4 | Device Status component will determine current state of devices. After determining state, data will persist into Database |  |
| 5.1 | Site User will access dashboard from browser |  |
| 5.2 | Admin User will access dashboard from browser |  |
| 5.3 | User will be authenticated using Azure Active directory |  |
| 5.4 | After Dashboard authentication node.js express server will load UI related shipment data from database |  |
| 5.5 | Also cached data like layout ,system properties etc data will load from redis cache |  |
| 5.6 | Retrieved data from database and cache will show in Dashboard UI |  |
| 5.7 | When user will access asset tracking in dashboard the it will connect to socket.io component to receive continuous forklift movement | Example Payload  {"EPC": "444835363636363137335f313632", "fromLocation": {}, "toLocation": {},"forkliftLocation": {"X": 50, "Y": 52}, "timestamp": "2020-03-15T08:51:48.111Z","ConeID": "anchor\_tag\_1"} |
| 6 | FSI will update shipment shipped status to exposed API by node.js express |  |

## NGFS Sequence Diagram

The information flow between DHL N/W and IBM Cloud can be broken into following sub flows:

1. SmartCone publishes Forklift location details along with shipment details carried by the forklift. This information is published every 0.5 seconds from SmartCone to IBM Cloud
2. SmartCone publishes device (Antenna, Anchor, Device Gateway) status every 1 minute. The helps IBM Cloud to report near real time status of the forklift.
3. DHL user is authenticated against DHL LDAP before logging into the dashboard hosted in IBM Cloud



|  |  |  |
| --- | --- | --- |
| Sequence Component | **Detail** | **Remark** |
| Antena | Antena will publish following message to compute module   1. Hearbeat signal 2. Shipment EPC on forklift |  |
| UWB Module | UWB Module will publish following message to compute module   1. Hearbeat signal 2. Forklift location coordinate |  |
| Compute Module | Compute Module will do following operation:   1. Prepare device status message and pushed to devicestatus Gateway of WIoT 2. Prepare Shipment with EPC messagealong with forklift location pushed to forklift Gateway of WIoT |  |
| WIoT | WIoT will persist device status and shipment data with forklift location into data base | Possible device status is:  Active, Offline, Service,Idle,Fault |
| Track And Trace component | Trackntrace component will receive epc with location payload in every .5 second for each forklift.Component will do following operation   1. Load location zone data from database 2. Get last location of shipment 3. Calculate current location of shipment based on forklift location coordinate 4. Determine if location has changed based on comparing step 2 and 3 location 5. If location changed then persist new location for particular shipment .Also store forklift current location current |  |
| Device Status Component | Device status will receive message from WIoT and persist last state of each device |  |
| Asset Tracking socket | Asset Tracking socket component subscribe forklift Gateway to get current location of forklift .When dashboard’s user open asset tracking page then socket connection will established between user and this component. User will get continuous feed of forklift movement |  |
| UI Dashboard | Dashboard will get data of shipment ,forklift etc from database to show user in browser. It will also access logs from track and Trace component and Compute Module for trouble shooting purpose |  |
| Dashboard User | Dashboard User will access UI Dashboard component to see track shipment and forklift in browser |  |

Payload from SmartCone devices:

DeviceID: [**coneID]**

DeviceType: **forklift**

Frequency: every half a sec ( when freight is on the forklift)

{

"EPC": ["445F36363633323732343031","445F36363633323732343031"],

"fromLocation": {"X": 1,"Y": 2,"Z": 3},

"toLocation": {"X": 2,"Y": 3,"Z": 4},

"forkliftLocation":{"X": 2,"Y": 3,"Z": 4},

"timestamp": "2019-04-03T13:35:54Z"

}

Frequency: every 10 sec (When freight is not on the forklift)

{

"EPC": [],

"fromLocation": null,

"toLocation": null,

"forkliftLocation":{"X": 2,"Y": 3,"Z": 4},

"timestamp": "2019-04-03T13:35:54Z"

}

## NGFS Deployment Topology

Below are the instance details of services provisioned in IBM Cloud:

1. Maximo Worker Insight’s IoT platform – The platform provides managed services to ingest and persist the Warehouse data for downstream analysis against the DHL site metadata. There are two environment provisioned – DEV/QA and Prod.

The URL of the instance are:

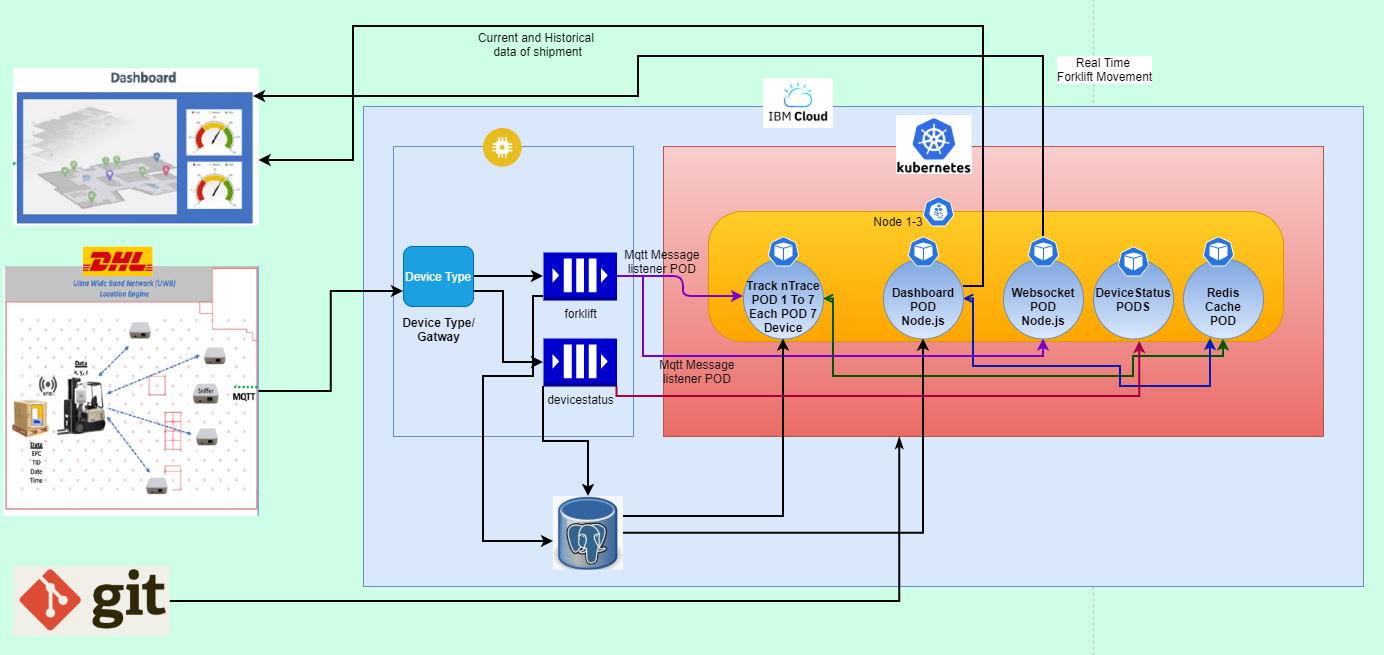
DEV&QA – <https://dashboard-us.connectedproducts.internetofthings.ibmcloud.com/preauth?tenantid=MWI-DHL>

Prod - <https://dashboard-us.connectedproducts.internetofthings.ibmcloud.com/preauth?tenantid=MWI-DHLProd>

1. Kubernetes cluster and its pods – These are managed services that process the raw data ingested by Watson IoT platform against business conditions and displays the contextual information published by SmartCone devices.

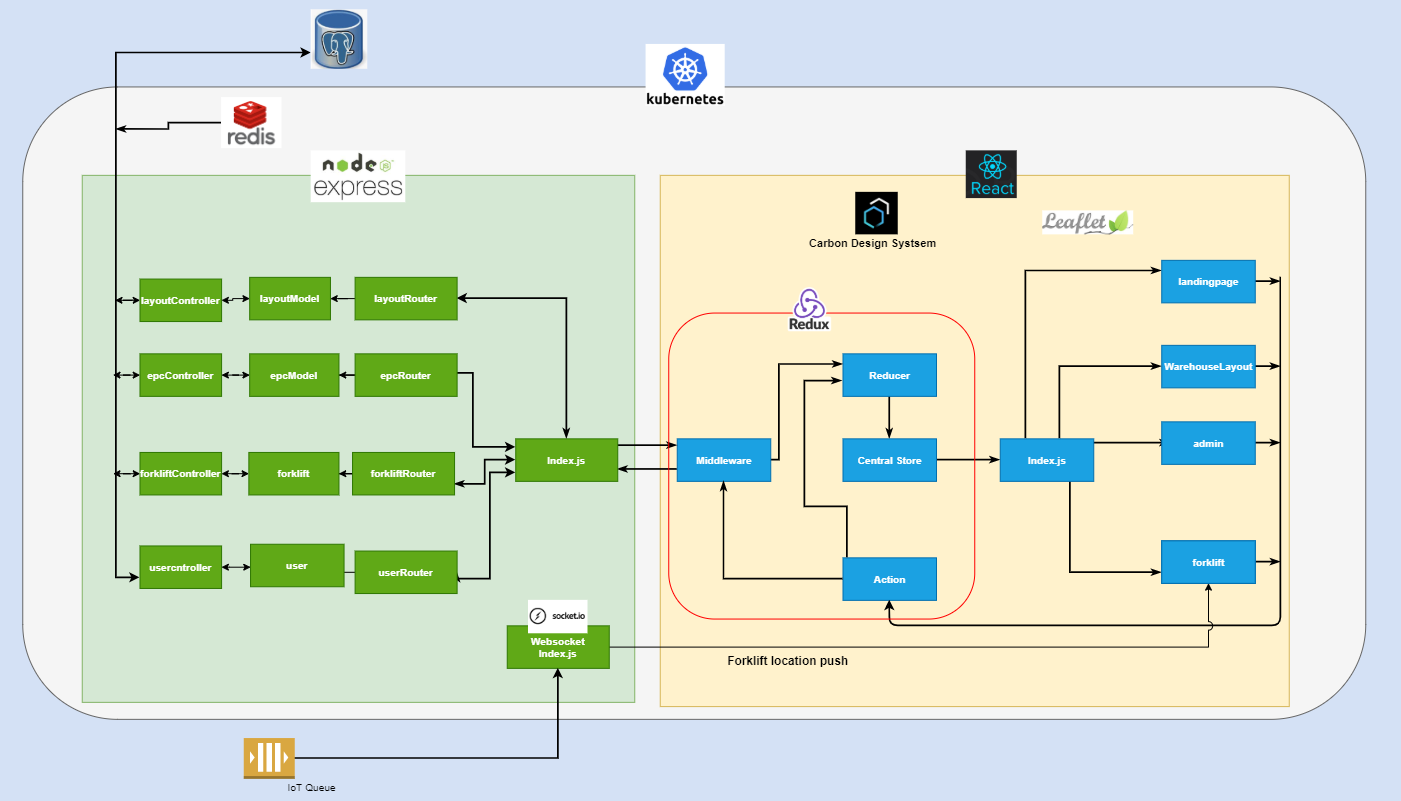
The Cloud tenant ID for DHL instance is 1927383 - DHL Global Forwarding and can be accessed using URL: <https://cloud.ibm.com>

Below diagram provides details of components and services provisioned in IBM Cloud.



**IoT platform**

## NGFS Dashboard Architecture

NGFS dashboard is hosted in Kubernetes cluster of IBM Cloud. The pods deploy node express and React framework to realize the dashboard. The dashboard is based on Carbon design system. Carbon is IBM’s open-source design system for products and experiences. With the IBM Design Language as its foundation, the system consists of working code, design tools and resources, human interface guidelines, and a vibrant community of contributors.

Following UI Component are designed as part of react framework:

### Warehouse Layout component :

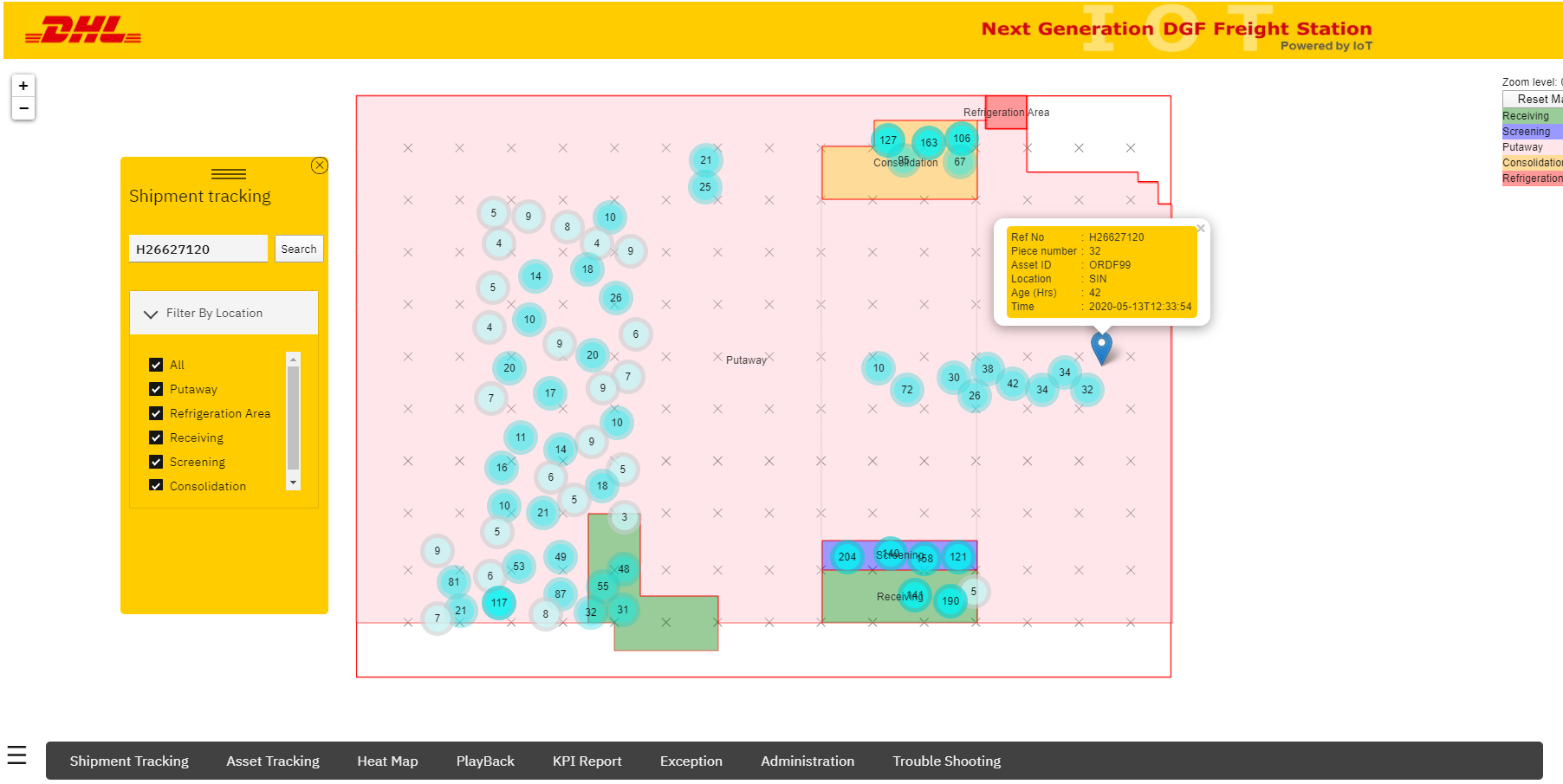
This component will get layout data exposed by node.js express’s layoutController component API from database.This will show layout in zone , area and subarea level as shown below.

|  |  |
| --- | --- |
| **Layout** | **UI design** |
| Zone |  |
| Subarea |  |
| area |  |

### Shipment Tracking component:

User can view and search shipment using Shipment Tracking as shown below:

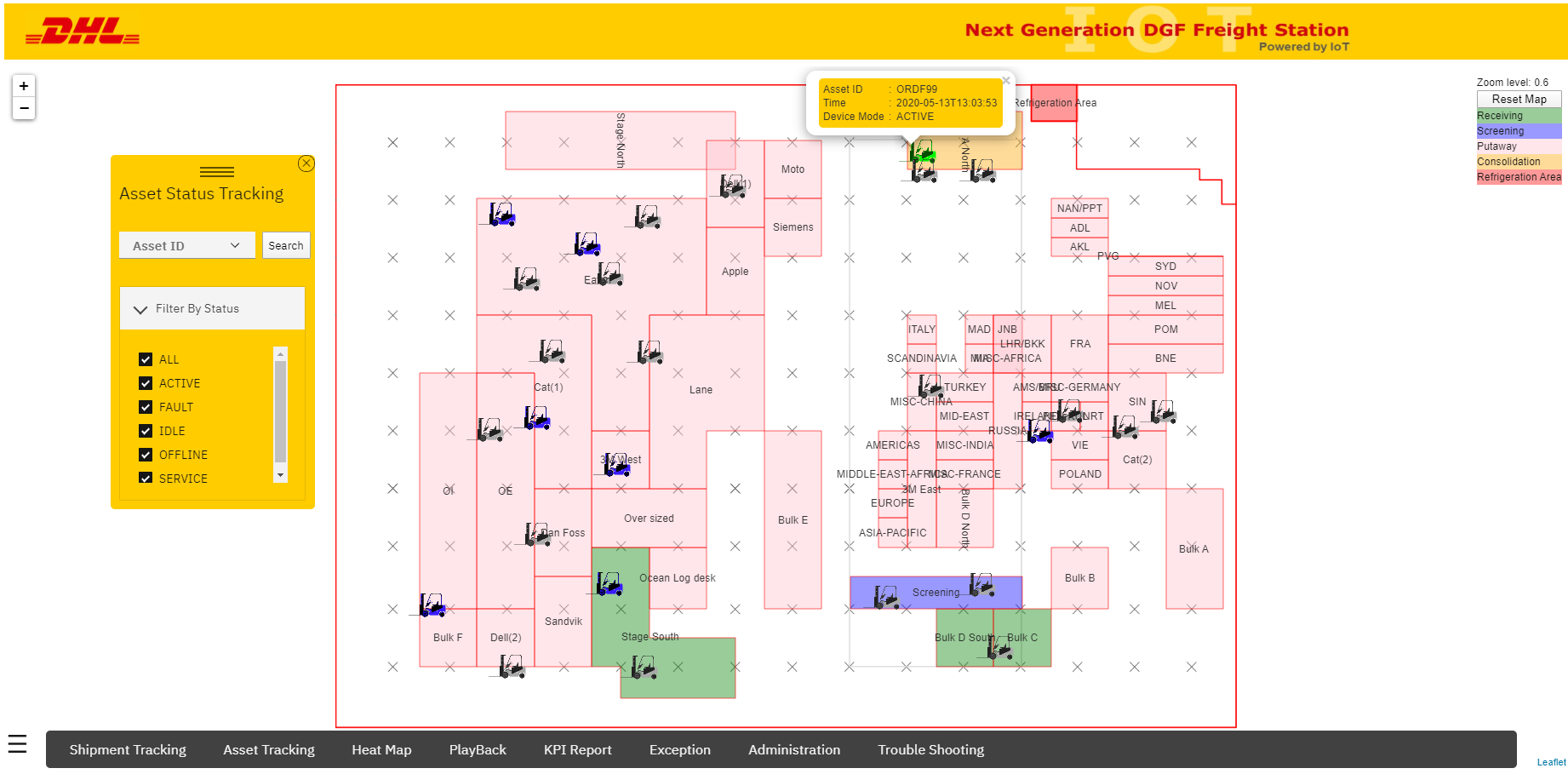
Node.js express’s epcStateController component expose API to get shipment state from database



### Asset Tracking component:

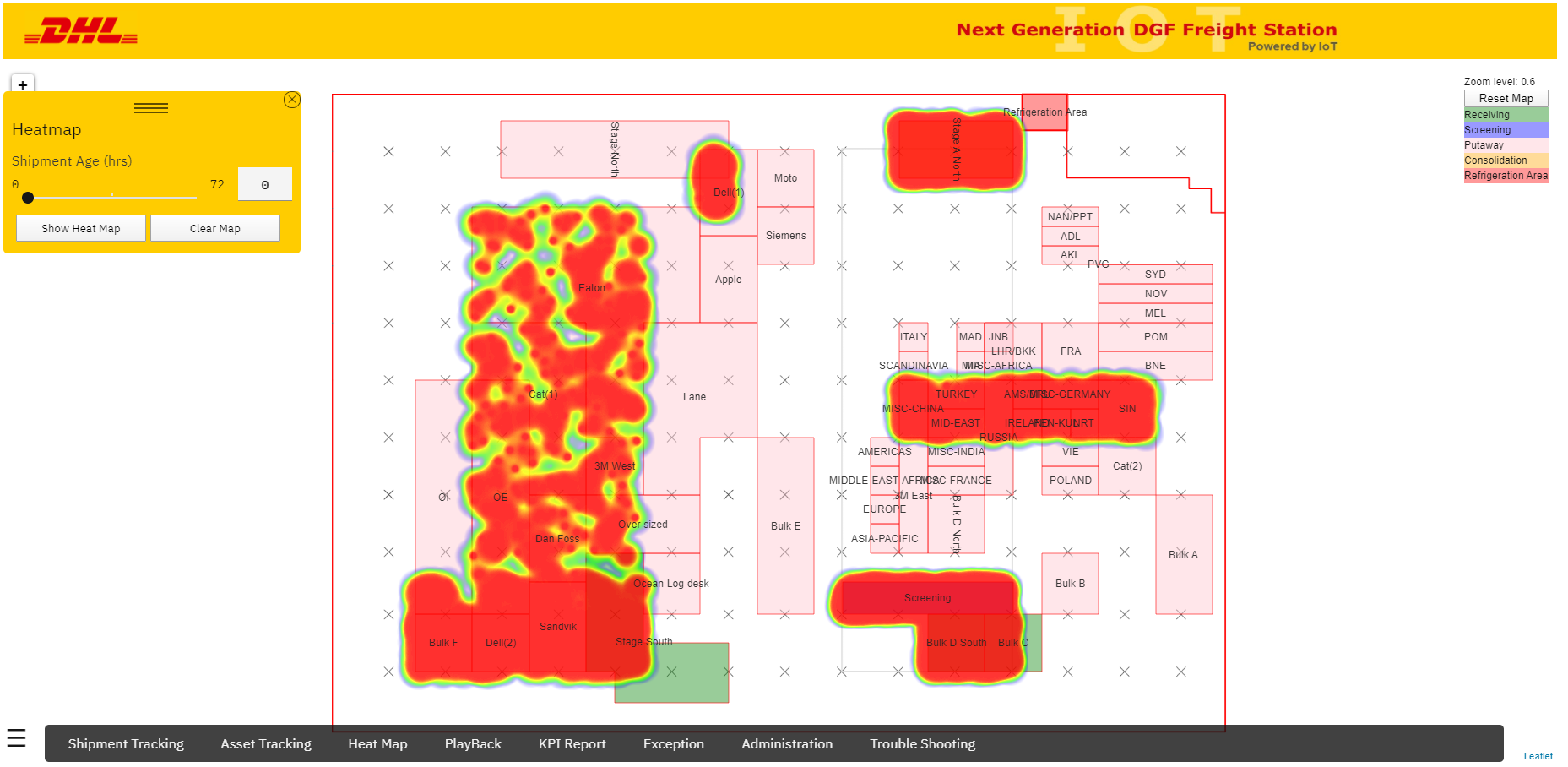
User can track real time movement of forklift inside ware house also can see asset status along with tracking

Node.js express’s forkliftStateController component expose API to get forklift state from database.Also for real time forklift movement socket.io websocket open connection with dashboard asset tracking compomnent



### Heat Map component:

Shipment heat map is a data visualization technique that shows magnitude of a phenomenon as color in two dimensions. The variation in color may be by hue or intensity, giving obvious visual cues to the user about how the phenomenon is clustered or varies over space

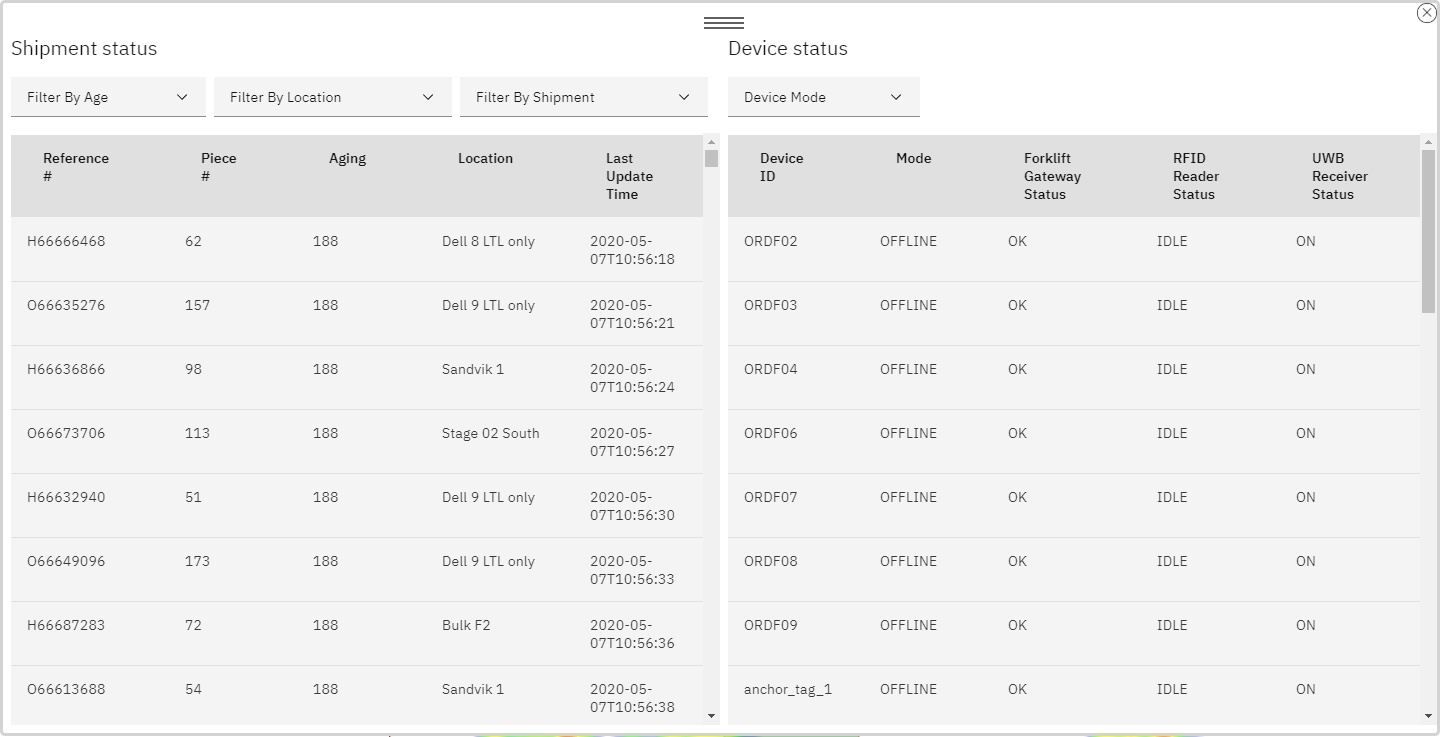


### Exception component:

User can view shipment status based on shipment age in warehouse.User can filter all shipment with more than 72 hrs and between 48 and 72 hrs passed in ware house.

User can also view Device status as part of this component

Node.js Expres’s expose deviceStateController component API to get device status from database

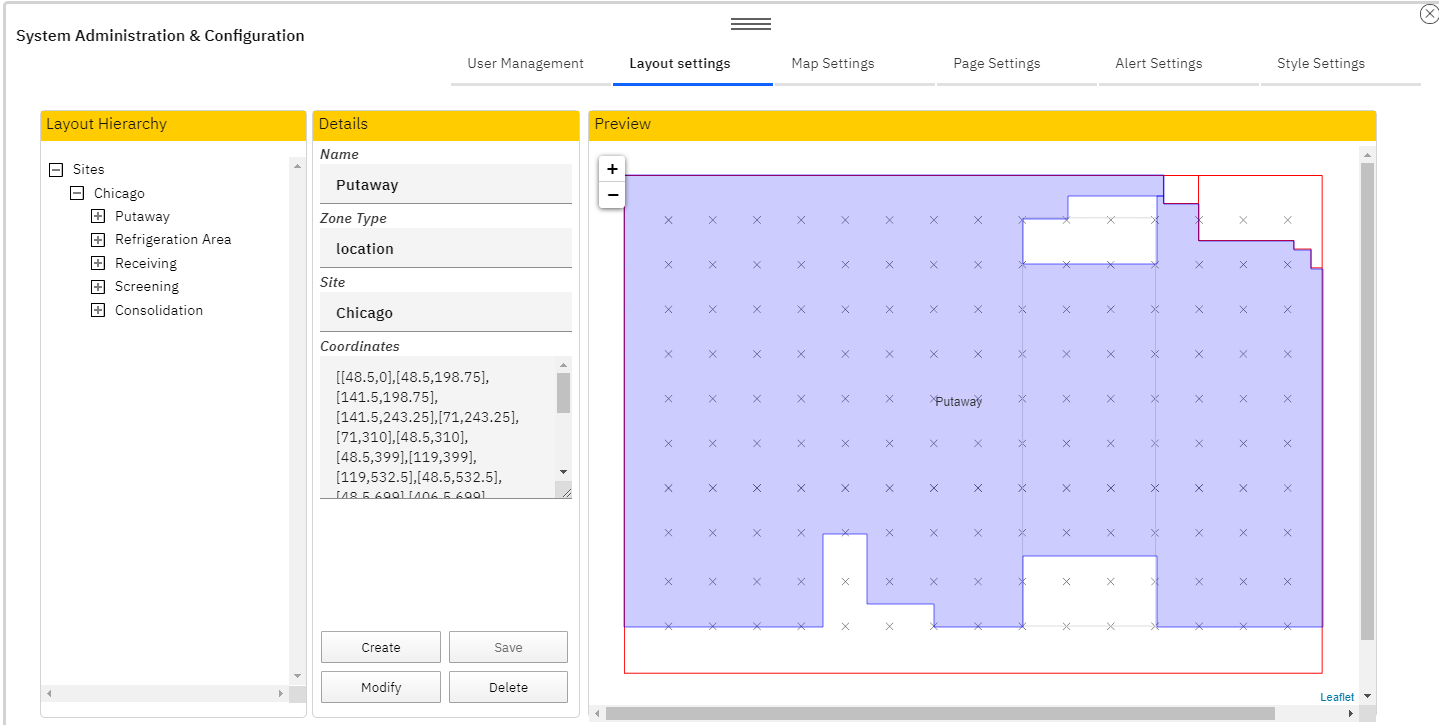


### Admin component:

NGFS administrator can view and modify system configuration using admin component.

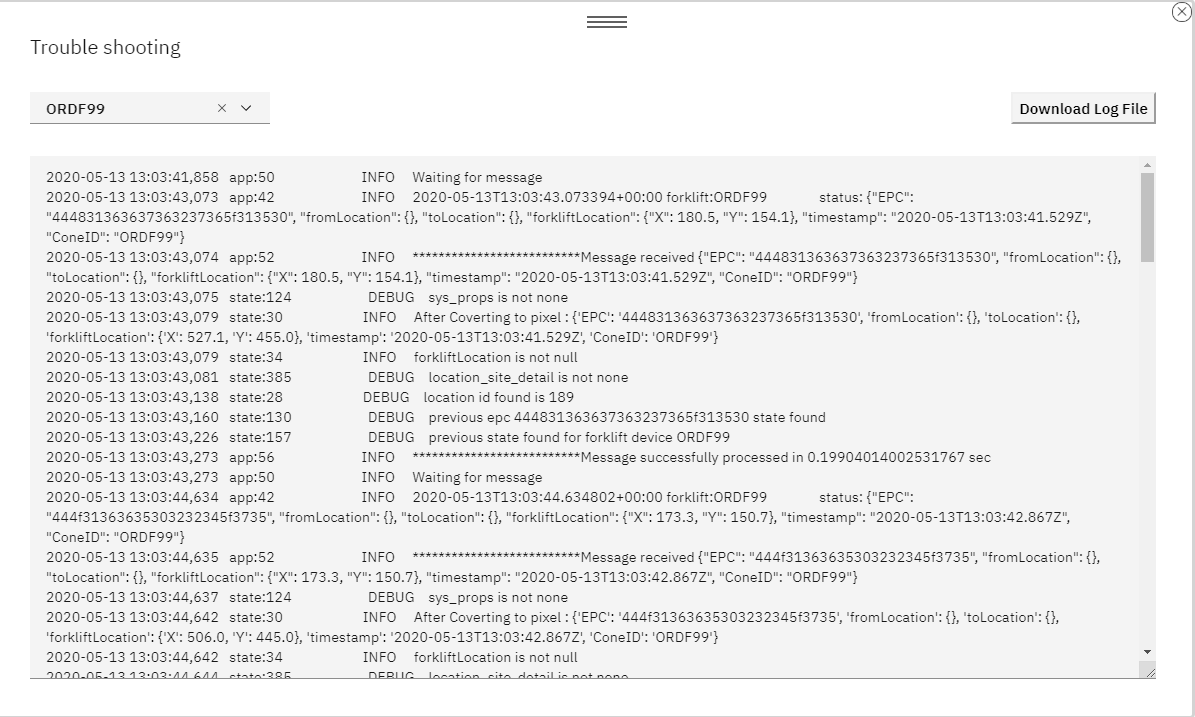
Admin can do user management, Layout management using this component.Also user can set system properties for Map ,Page and alert.

Node.js express’s userMgmtController, sysPropController and layoutController components API are exposed to get data from database



### Troubleshooting component:-

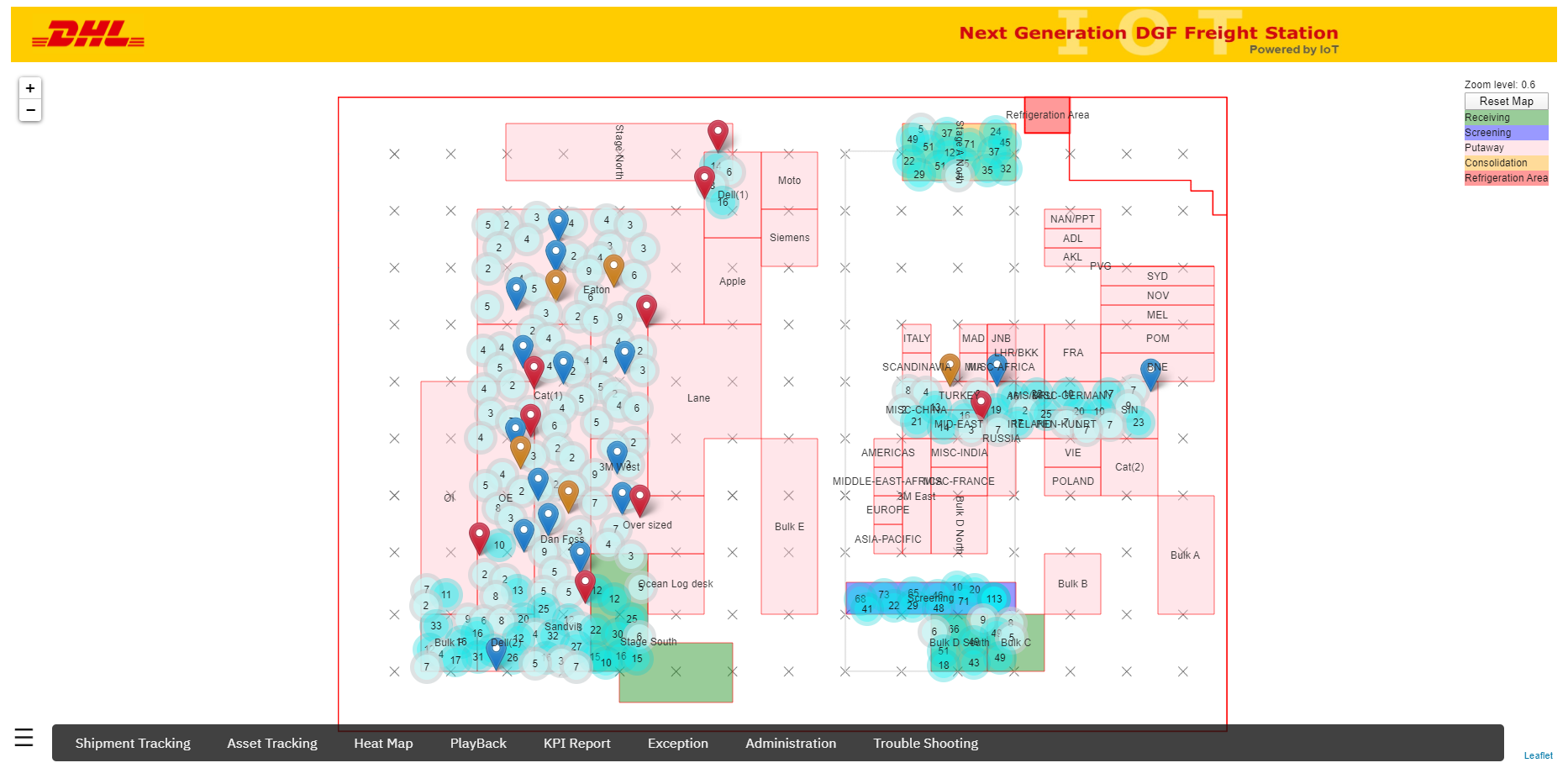
User can view,download and analyze logs for any issue from Track and Trace component along with device logs

****

## NGFS Dashboard

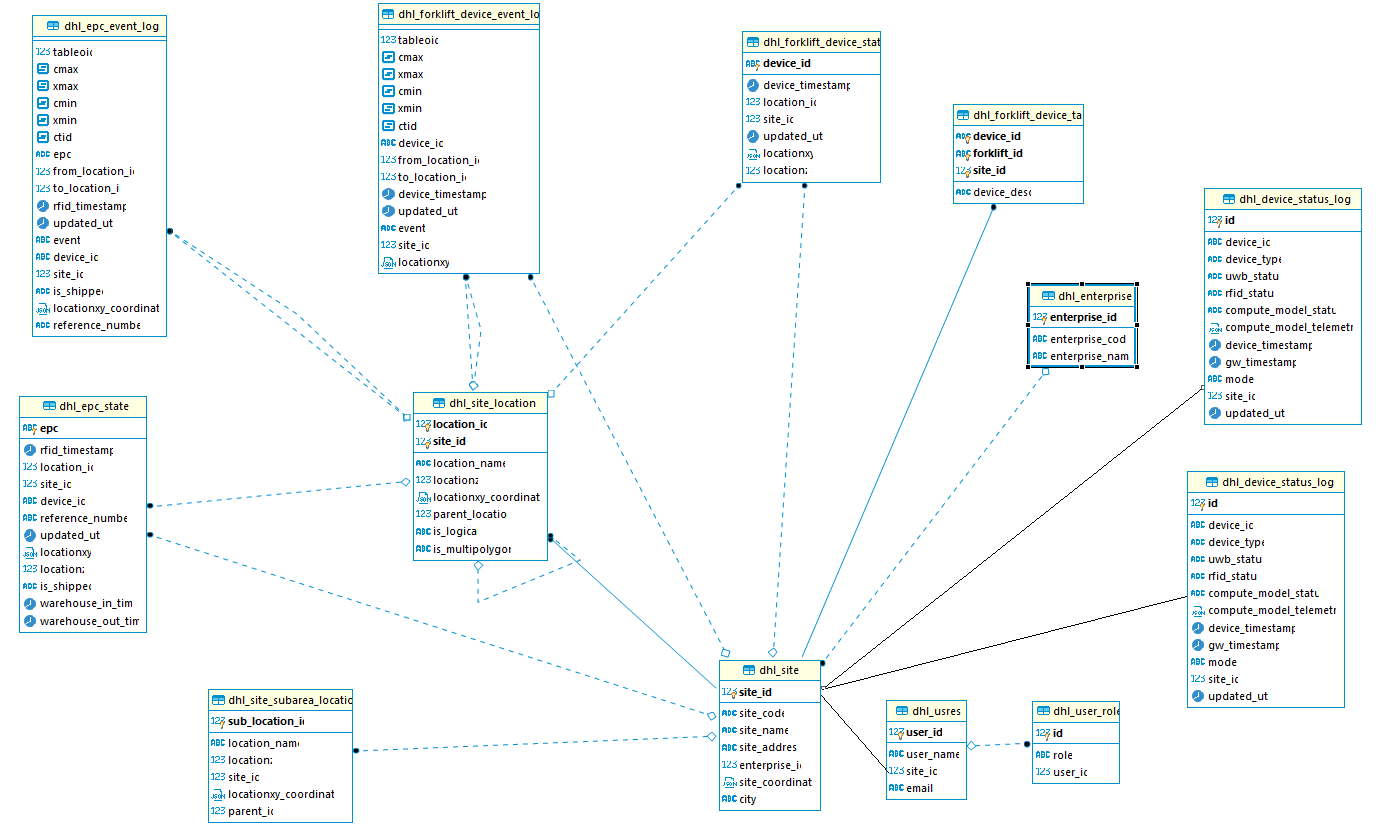
The URL of the NFGS solution dashboard is :

<https://global-dhl-423761-0a85938276edbbcfb58a5c4bf680f8e3-0000.us-south.containers.appdomain.cloud/>



## NGFS Entity Diagram

The metamodel to correlated various events published by SmartCone devices is shown below:



|  |  |  |  |
| --- | --- | --- | --- |
| TABLE | Table Definition | Description | Remark |
| dhl\_enterprise | **CREATE** **TABLE** dhl\_enterprise (  enterprise\_id **serial** **NOT** **NULL**,  enterprise\_code **varchar**(50) **NULL**,  enterprise\_name **varchar**(50) **NULL**,  **CONSTRAINT** dhl\_enterprise\_pkey **PRIMARY** **KEY** (enterprise\_id)  ); | Table for enterprise level detail |  |
| dhl\_site | **CREATE** **TABLE** dhl\_site (  site\_id **serial** **NOT** **NULL**,  site\_code **varchar**(50) **NULL**,  site\_name **varchar**(50) **NULL**,  site\_address **varchar**(1000) **NULL**,  enterprise\_id **int4** **NULL**,  site\_coordinate **jsonb** **NULL**,  city **varchar**(50) **NULL**,  **CONSTRAINT** dhl\_site\_pkey **PRIMARY** **KEY** (site\_id)  );  -- public.dhl\_site foreign keys  **ALTER** **TABLE** public.dhl\_site **ADD** **CONSTRAINT** dhl\_site\_enterprise\_id\_fkey **FOREIGN** **KEY** (enterprise\_id) **REFERENCES** dhl\_enterprise(enterprise\_id); | The site table was created to store general information for the site. It store site with spatial reference |  |
| dhl\_site\_location | **CREATE** **TABLE** dhl\_site\_location (  location\_id **serial** **NOT** **NULL**,  location\_name **varchar**(100) **NULL**,  locationz **numeric**(6,2) **NULL**,  site\_id **int4** **NOT** **NULL**,  locationxy\_coordinate **jsonb** **NULL**,  parent\_location **int4** **NULL**,  is\_logical **varchar**(10) **NULL**,  is\_multipolygon **varchar**(10) **NULL** **DEFAULT** 'F'::**character** **varying**,  **CONSTRAINT** dhl\_site\_location\_location\_id\_key **UNIQUE** (location\_id),  **CONSTRAINT** dhl\_site\_location\_pkey **PRIMARY** **KEY** (location\_id,site\_id)  );  -- public.dhl\_site\_location foreign keys  **ALTER** **TABLE** public.dhl\_site\_location **ADD** **CONSTRAINT** dhl\_site\_location\_site\_id\_fkey **FOREIGN** **KEY** (site\_id) **REFERENCES** dhl\_site(site\_id);  **ALTER** **TABLE** public.dhl\_site\_location **ADD** **CONSTRAINT** parentlocfk **FOREIGN** **KEY** (parent\_location) **REFERENCES** dhl\_site\_location(location\_id); | The table was created to store warehouse layout with its zone,area and subarea x,y coordinate.It is self referencing table to create parent child relation between zone area and subarea |  |
| dhl\_epc\_state | **CREATE** **TABLE** dhl\_epc\_state (  epc **varchar**(200) **NOT** **NULL**,  rfid\_timestamp **timestamp** **NULL**,  location\_id **int4** **NULL**,  site\_id **int4** **NULL**,  device\_id **varchar**(200) **NULL**,  reference\_number **varchar**(200) **NULL**,  updated\_utc **timestamp** **NULL** **DEFAULT** timezone('UTC'::**text**, **now**()),  locationxy **jsonb** **NULL**,  locationz **numeric**(6,2) **NULL**,  is\_shipped **varchar**(10) **NULL** **DEFAULT** 'F'::**character** **varying**,  warehouse\_in\_time **timestamp** **NULL**,  warehouse\_out\_time **timestamp** **NULL**,  **CONSTRAINT** dhl\_epc\_state\_pkey **PRIMARY** **KEY** (epc)  );  -- public.dhl\_epc\_state foreign keys  **ALTER** **TABLE** public.dhl\_epc\_state **ADD** **CONSTRAINT** dhl\_epc\_state\_location\_id\_fkey **FOREIGN** **KEY** (location\_id) **REFERENCES** dhl\_site\_location(location\_id);  **ALTER** **TABLE** public.dhl\_epc\_state **ADD** **CONSTRAINT** dhl\_epc\_state\_site\_id\_fkey **FOREIGN** **KEY** (site\_id) **REFERENCES** dhl\_site(site\_id); | The table was created to store shipment state data. Here shipment’s EPC will be unique and store last state with its location.It also store forklift carrying the same |  |
| dhl\_epc\_event\_log | CREATE TABLE dhl\_epc\_event\_log (  epc varchar(200) NULL,  from\_location\_id int4 NULL,  to\_location\_id int4 NULL,  rfid\_timestamp timestamp NULL,  updated\_utc timestamp NULL DEFAULT timezone('UTC'::text, now()),  "event" varchar(50) NULL,  device\_id varchar(200) NULL,  site\_id int4 NULL,  is\_shipped varchar(10) NULL DEFAULT 'F'::character varying,  locationxy\_coordinate jsonb NULL,  reference\_number varchar(100) NULL  );  -- public.dhl\_epc\_event\_log foreign keys  ALTER TABLE public.dhl\_epc\_event\_log ADD CONSTRAINT dhl\_epc\_event\_log\_from\_location\_id\_fkey FOREIGN KEY (from\_location\_id) REFERENCES dhl\_site\_location(location\_id);  ALTER TABLE public.dhl\_epc\_event\_log ADD CONSTRAINT dhl\_epc\_event\_log\_to\_location\_id\_fkey FOREIGN KEY (to\_location\_id) REFERENCES dhl\_site\_location(location\_id); | The table was created to store every state of shipment state data. This can be used to playback shipment in dashboard |  |
| dhl\_forklift\_device\_state | CREATE TABLE dhl\_forklift\_device\_state (  device\_id varchar(200) NOT NULL,  device\_timestamp timestamp NULL,  location\_id int4 NULL,  site\_id int4 NULL,  updated\_utc timestamp NULL DEFAULT timezone('UTC'::text, now()),  locationxy jsonb NULL,  locationz numeric(6,2) NULL,  CONSTRAINT dhl\_forklift\_device\_state\_pkey PRIMARY KEY (device\_id)  );  -- public.dhl\_forklift\_device\_state foreign keys  ALTER TABLE public.dhl\_forklift\_device\_state ADD CONSTRAINT dhl\_forklift\_device\_state\_location\_id\_fkey FOREIGN KEY (location\_id) REFERENCES dhl\_site\_location(location\_id);  ALTER TABLE public.dhl\_forklift\_device\_state ADD CONSTRAINT dhl\_forklift\_device\_state\_site\_id\_fkey FOREIGN KEY (site\_id) REFERENCES dhl\_site(site\_id); | The table was created to store forklift state data. Here forklift device id will be unique and store last state with its forklift location. |  |
| dhl\_forklift\_device\_event\_log | **CREATE** **TABLE** dhl\_forklift\_device\_event\_log (  device\_id **varchar**(200) **NULL**,  from\_location\_id **int4** **NULL**,  to\_location\_id **int4** **NULL**,  device\_timestamp **timestamp** **NULL**,  updated\_utc **timestamp** **NULL** **DEFAULT** timezone('UTC'::**text**, **now**()),  "event" **varchar**(50) **NULL**,  site\_id **int4** **NULL**,  locationxy **jsonb** **NULL**  );  -- public.dhl\_forklift\_device\_event\_log foreign keys  **ALTER** **TABLE** public.dhl\_forklift\_device\_event\_log **ADD** **CONSTRAINT** dhl\_forklift\_device\_event\_log\_from\_location\_id\_fkey **FOREIGN** **KEY** (from\_location\_id) **REFERENCES** dhl\_site\_location(location\_id); | The table was created to store every movement of forklift state data. |  |
| dhl\_device\_status\_state | CREATE TABLE dhl\_device\_status\_state (  device\_id varchar(50) NOT NULL,  device\_type varchar(20) NULL,  uwb\_status varchar(10) NULL,  rfid\_status varchar(10) NULL,  compute\_model\_status varchar(10) NULL,  compute\_model\_telemetry jsonb NULL,  device\_timestamp timestamp NULL,  gw\_timestamp timestamp NULL,  "mode" varchar(10) NULL,  site\_id int4 NULL,  updated\_utc timestamp NULL,  CONSTRAINT dhl\_device\_status\_state\_pkey PRIMARY KEY (device\_id)  ); | The table was created to store device status last state. |  |
| dhl\_device\_status\_log | CREATE TABLE dhl\_device\_status\_log (  id serial NOT NULL,  device\_id varchar(50) NULL,  device\_type varchar(20) NULL,  uwb\_status varchar(10) NULL,  rfid\_status varchar(10) NULL,  compute\_model\_status varchar(10) NULL,  compute\_model\_telemetry jsonb NULL,  device\_timestamp timestamp NULL,  gw\_timestamp timestamp NULL,  "mode" varchar(10) NULL,  site\_id int4 NULL,  updated\_utc timestamp NULL,  CONSTRAINT dhl\_device\_status\_log\_pkey PRIMARY KEY (id)  ); | The table was created to log device status’s every record. |  |
| dhl\_system\_properties | CREATE TABLE dhl\_system\_properties (  id serial NOT NULL,  "name" varchar(100) NULL,  description varchar(200) NULL,  properties jsonb NULL,  site\_id int4 NULL  ); | This table will log record as system properties which admin will set from dashboard |  |
| dhl\_usres | **CREATE** **TABLE** dhl\_usres (  user\_id **serial** **NOT** **NULL**,  user\_name **varchar**(50) **NULL**,  site\_id **int4** **NULL**,  email **varchar**(100) **NULL**,  **CONSTRAINT** dhl\_usres\_pkey **PRIMARY** **KEY** (user\_id)  ); | This table will create user which dhl will authenticate in Azure AD |  |
| dhl\_user\_role | CREATE TABLE dhl\_user\_role (  id serial NOT NULL,  "role" varchar(50) NULL,  user\_id int4 NULL,  CONSTRAINT dhl\_user\_role\_pkey PRIMARY KEY (id)  );  -- public.dhl\_user\_role foreign keys  ALTER TABLE public.dhl\_user\_role ADD CONSTRAINT constraint\_user\_id FOREIGN KEY (user\_id) REFERENCES dhl\_usres(user\_id) ON DELETE CASCADE; | This table will create role for authorization to access dashboard operation based on allowable access |  |