Instructions: How to Run the Fedex\_OrderStatus Module

# Module Installation

* Copy the module to your Magento installation:  
   app/code/Fedex/OrderStatus
* Enable and install the module using Magento CLI:  
   php bin/magento module:enable Fedex\_OrderStatus  
   php bin/magento setup:upgrade  
   php bin/magento setup:di:compile  
   php bin/magento cache:flush
* Verify the module status:  
   php bin/magento module:status Fedex\_OrderStatus

# Admin Usage

* Navigate to: Admin Panel > Sales > Order Status
* From this grid, you can:  
   - View,or delete order statuses  
   - Enable or disable, delete multiple statuses using mass actions

# REST API Usage (Update Order Status)

* Endpoint:  
   POST /V1/fedex/order/status/update
* Authentication:  
   Use Magento's Bearer Token (Admin or Integration token)
* Request Example (JSON):  
   {  
   "increment\_id": "100000001",  
   "new\_status": "custom\_approved"  
   }
* Sample Response:  
   {  
   "message": "Order status updated successfully."  
   }

# Email Notifications

* When an order is marked as shipped, an email notification is sent to the customer automatically.
* Email Template:  
   - view/frontend/email/order\_status\_shipped.html  
   - Customizable via admin (Transactional Emails)
* Trigger:  
   - Event: sales\_order\_save\_after  
   - Fires when status changes to 'shipped'

# Database Tables

* fedex\_order\_status:  
   - id, status, is\_active, created\_at, updated\_at
* fedex\_order\_status\_log:  
   - log\_id, order\_id, old\_status, new\_status, created\_at
* Created automatically during: php bin/magento setup:upgrade

# Caching for Status List

* getAvailableStatuses() uses Magento’s Cache API.
* Cached with custom cache key/tag:  
   Fedex\OrderStatus\Model\Status::CACHE\_TAG
* Cache auto invalidates on add/update/delete.

# Events & Observers

* Observer for: sales\_order\_save\_after
* Actions:  
   - Log status to fedex\_order\_status\_log  
   - Email customer if marked as shipped

# Testing

* Admin: Test grid, add/edit/delete/enable/disable statuses
* API: Test with Postman or REST Client using Bearer Token

# Summary

* Admin grid with mass actions
* Custom DB table via db\_schema.xml
* API endpoint for external updates
* Email notifications on ship
* Event logging
* Performance-optimized with caching