

LTL Quote API

Integration Documentation

Introduction

Integrating with GoShip's LTL Quote APIs can provide small businesses with numerous benefits, including a reduction in operational costs and overhead. GoShip's APIs automate many of the manual processes involved in quoting and booking shipments. As a result, businesses can save time, money, and resources, which can be used for other areas of the business. Furthermore, GoShip's APIs can also improve the overall customer experience, by providing real-time shipping information, which is essential for small businesses looking to grow and expand.

Table of Contents

<u>Authentication</u>	3
<u>Generating the API Token / Key</u>	3
<u>Endpoints and Requests</u>	5
<u>Using the Document Explorer</u>	7
<u>Integration using NodeJS</u>	9
<u>Getting the Rates (Explorer)</u>	10
<u>Preparing the Order</u>	11
<u>Finalizing a Shipment Booking via Vendor Account</u>	12

Authentication

All GraphQL queries require a valid access key. Include your key as an **X-GoShip-API-Key** header on all queries. Example screenshots are provided further down in this document.

Generating the API Token / Key

In order to generate an access key you must have a GoShip account.

1. To create an account (If you don't already have one)

- a. Visit quotes.goship.com/sign-up

The screenshot shows a web browser window with the URL 'quotes.goship.com/sign-up'. The page title is 'GO SHIP'. The main content is a 'Create New Account' form. It includes fields for FIRST NAME, LAST NAME, EMAIL, COMPANY, PHONE NUMBER, CREATE PASSWORD, and CONFIRM PASSWORD. Below the form is a note about transit times and payment terms, followed by a 'CREATE ACCOUNT' button.

- b. Enter your account details
- c. Click 'Create an Account'
- d. Once an account has been created you will receive a verification email at the email address you specified during account creation
- e. After verifying your email and logging in, you will be able to progress to the next step (Generate your API Key)

2. Generate your API key

- Once logged in, navigate to quotes.goship.com/apikeys

Api Key Manager		
NAME	STATUS	DEACTIVATE
Test Key	Active	<button>deactivate</button>
<button>Add New</button>		

- Click the 'Add New' button to add a new API key.

Create new API key

Name

Create Cancel

- Enter a name for your API Key and then click the 'Create' button. The system will automatically generate your secure key. **Please make sure to copy the secure key, and keep it safe.**

⚠ Treat the API Key like a password ⚠

The API key named "Admin-key" has been
created

The value of the key is
7 38
Copy

Save this value somewhere secure and private. You will not be able to retrieve it later.

Done

Endpoints and Requests

All GraphQL endpoints are handled from this URL: <https://nautilus.goship.com/broker/graphql>

To test a request and authentication key, open up a GraphQL explorer. For example, we have two explorers as an example.

1. GraphiQL (preferred)

The screenshot shows the GraphiQL interface. At the top, there is a header with "GraphQL Endpoint" set to <https://nautilus.goship.com/broker/graphql>, "Edit HTTP Headers" button, and "Method" dropdown set to "POST". Below the header, the main area has tabs for "GraphiQL" (selected), "Prettify", and "History". The "GraphiQL" tab contains a code editor with the following text:

```
1 # Welcome to GraphiQL
2 #
3 # GraphiQL is an in-browser tool for writing, validating, and
4 # testing GraphQL queries.
5 #
6 # Type queries into this side of the screen, and you will see
7 # typeheads aware of the current GraphQL type schema and live
8 # validation errors highlighted within the text.
9 #
10 # GraphQL queries typically start with a "{" character. Lines
11 # with a # are ignored.
```

On the right side, there is a "Documentation Explorer" panel with a search bar ("Search Schema...") and a description: "A GraphQL schema provides a root type for each kind of operation." It also lists "ROOT TYPES", "query: Query", and "mutation: Mutation".

2. Altair (Google Chrome Extension)

The screenshot shows the Altair GraphQL Client extension in a Google Chrome browser window. The title bar says "Altair GraphQL Client | chrome-extension://flnheeeellpciglgpaodhkhmapeljopja/...". The main interface has a toolbar with icons for back, forward, and refresh, and a status bar indicating "No environment". The main area is a request builder with tabs for "POST", "Query", "Pre-request", "Post-request", "Result", and "Response headers". The "Result" tab is selected. On the right, there is a "Docs" section with a "Send Request" button, a search bar ("Search docs..."), and sections for "Query" and "Mutation".

Point the GraphQL explorer to the URL <https://nautilus.goship.com/broker/graphql> and set the header for authentication

1. Name: **X-GoShip-API-Key**
2. Value: <secret key copied from API keys page—created in Step 2>

GraphQL

Edit HTTP Headers

Header name	Header value
X-GoShip-API-Key	<.....>

+ Add Header

Edit Delete

Altair

Set Headers https://nautilus.goship.com/broker/graphql

Query Pre-request Post-request

```
1 # Welcome to Altair GraphQL Client.  
2 # You can send your request using CmdOrCtrl + Enter.  
3  
4 # Enter your graphQL query here.
```

HEADERS Add, edit and remove headers in your request X

<input checked="" type="checkbox"/> X-GoShip-API-Key	0	x
<input checked="" type="checkbox"/> Header key	Header value	x

Forbidden headers will not work in the browser environment. If you want to use them, use the desktop apps. [Learn more](#)

Add header Save

After saving the header, you can explore the broker API.

Using the Document Explorer

For elaborate requests and responses, both GraphQL and Altair have a Documentation Explorer section where all the queries and mutations can be viewed along with their request and response parameters.

1. GraphQL

On the right side of the explorer, there is a section with the heading 'Documentation Explorer' which contains queries and mutations along with request and response parameters

The screenshot displays two panels of the Documentation Explorer:

Left Panel (Schema Overview):

- Documentation Explorer** (Title)
- Search Schema...** (Search bar)
- A GraphQL schema provides a root type for each kind of operation.
- ROOT TYPES**
 - query:** Query
 - mutation:** Mutation

Right Panel (Query Details):

- Query** (Title)
- Search Query...** (Search bar)
- No Description
- FIELDS**
 - requestLTLQuote(**rfq:** LtlRfqInput!): [LtlQuote]
 - requestFTLQuote(**rfq:** FtlRfqInput!): FtlQuote
 - getLoadDocuments(**loadId:** String!, **shipmentType:** SHIPMENT_TYPE!): JSON

Bottom Left Panel (Input Type Details):

- LtlRfqInput** (Title)
- Search LtlRfqInput...** (Search bar)
- No Description
- FIELDS**
 - origin:** RfqEndpointInput!
 - destination:** RfqEndpointInput!
 - pickupDate:** String!
 - items:** [ItemInput]!!

Bottom Right Panel (Input Type Details):

- ItemInput** (Title)
- Search ItemInput...** (Search bar)
- No Description
- FIELDS**
 - quantity:** Int!
 - packaging:** PACKAGING_TYPES!
 - sizeUoM:** DISTANCE_UOM!
 - length:** Int!
 - width:** Int!
 - height:** Int!
 - weightUoM:** WEIGHT_UOM!
 - weight:** Int!
 - freightClass:** FREIGHT_CLASS!

2. Altair

Similar to GraphQL, document explorer can be viewed on the right panel. If it is not visible there is a 'Docs' button to the left of 'Send Request' that will toggle the document explorer

The screenshot shows the Altair interface with the 'Docs' button selected. The right panel displays a document explorer with the following sections:

- Search docs...**: A search bar.
- Query**: A section for defining queries.
- Mutation**: A section for defining mutations.
- FIELDS**: A list of available fields:
 - requestLTLQuote (rfq [LtlRfqInput!](#))
[[LtlQuote!](#)!]
 - requestFTLQuote (rfq [FtlRfqInput!](#))
[[FtlQuote](#)]
 - getLoadDocuments (loadId [String!](#),
shipmentType [SHIPMENT_TYPE!](#))
[JSON](#)

The screenshot shows the Altair interface with the 'Docs' button selected. The right panel displays two detailed field definitions:

LtlRfqInput

FIELDS: A list of available fields:

- origin [RfqEndpointInput!](#)
- destination [RfqEndpointInput!](#)
- pickupDate [String!](#)
- items [[ItemInput!](#)]!

ItemInput

FIELDS: A list of available fields:

- quantity [Int!](#)
- packaging [PACKAGING_TYPES!](#)
- sizeUoM [DISTANCE_UOM!](#)
- length [Int!](#)
- width [Int!](#)
- height [Int!](#)
- weightUoM [WEIGHT_UOM!](#)
- weight [Int!](#)
- freightClass [FREIGHT_CLASS!](#)
- value [Float!](#)

ADD FRAGMENT: A button to add a fragment.

Integration using NodeJS

1. Library

- a. apollo ^2.33.4
- b. graphql ^15.5.1

2. Reference: (<https://dev.to/lico/apollo-client-understanding-fetch-policy-with-usequery-1gfa>)

React Apollo: Understanding Fetch Policy with useQuery
I've been working on an Apollo client project these days. I wasn't used to GraphQL so I struggled to understand it at the beginning. In the app I'm [DEV https://dev.to/lico/apollo-client-understanding-fetch-policy-with-usequ...](https://dev.to/lico/apollo-client-understanding-fetch-policy-with-usequ...)



3. Setting up Apollo Client

```
const httpLink = createHttpLink({  
  uri: GRAPHQL_ENDPOINT,  
});  
  
const authLink = setContext(async (_, { headers }) => {  
  return {  
    headers: {  
      ...headers,  
      'X-GoShip-API-Key': '<secret key copied from API keys page>',  
    },  
  };  
});  
  
export default new ApolloClient({  
  link: authLink.concat(httpLink),  
  cache: new InMemoryCache(),  
});
```

Getting the Rates (Explorer)

To request shipping rates, open the explorer and pass the required information in the query field. An example query is below. Query parameters can be changed to see different results.

```
query{
  requestLTLQuote(rfq:{
    origin:{
      postalCode:"16066"
      addressType: RESIDENTIAL
    }
    destination:{
      postalCode:"16066"
      addressType: BUSINESS_W_DOCK
    }
    pickupDate:"2022-11-12" //this must be a future date
    items:{
      quantity:1
      packaging: PALLET
      sizeUoM:in
      length:10
      width:10
      height:10
      weightUoM:lbs
      weight:200
      freightClass:CLASS_50
      value:100
      itemCondition:NEW
      stackable:false
      hazardous:false
      description:"rest"
      pieces:1
      country:USA
    }
  )|{
    cost
    carrier{
      name
      id
    }
    id
    deliveryDate
  }
}
```

Preparing the Order

When checkout is done by the customer after selecting a quote, the request needs to be sent as a mutation to store the shipment order. The mutation only stores the quote information and it **does not actually process a shipment. A shipment must be confirmed and paid for by the vendor using the GoShip Vendor Orders UI. The steps and screenshots are below.**

Mutation

```
mutation{
  prepareOrder(input:{
    quoteId:"283cdf9e-2e52-439b-b066-1eb4f594387e" //this is the id returned from the
    orderNumber:"tes_z133"
    origin:{
      address:{
        streetAddress:"1234 Main Street"
        postalCode:"12345"
        city:"Schenectady"
        state:"NY"
        country:USA
        firstName:"John"
        lastName:"Doe"
        companyName:"JohnDoeCompany"
        phone:{
          areaCode:"888"
          countryCode:"1"
          number:"1234567"
        }
        email:"john@doe.com"
      }
      startTime:"2023-11-12 10:53:00.888 -0800"
      endTime:"2023-11-12 10:53:00.888 -0800"
    }
    destination:{
      address:{
        streetAddress:"1234 Main Street"
        postalCode:"12345"
        city:"Schenectady"
        state:"NY"
        country:USA
        firstName:"John"
        lastName:"Doe"
        companyName:"JohnDoeCompany"
        phone:{
          areaCode:"888"
          countryCode:"1"
          number:"1234567"
        }
        email:"john@doe.com"
      }
      startTime:"2023-12-12 10:53:00.888 -0800"
      endTime:"2023-12-12 10:53:00.888 -0800"
    }
  )}){
  vendorOrder{
    id
  }
  error
}
```

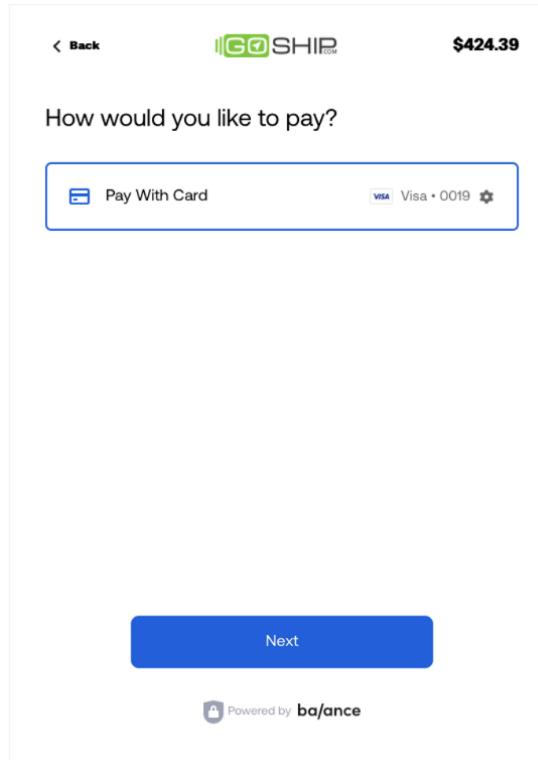
Finalizing a Shipment Booking via Vendor Account

1. Navigate to <https://quotes.goship.com/booking/vendor-orders>, this page is also accessible from the GoShip Shipping Dashboard.

The screenshot shows the GoShip Shipping Dashboard with a sidebar on the left containing links: Request Quote, View Orders, Vendor Orders (highlighted in green), Account, and Logout. The main area displays two vendor orders in a table format.

NUMBER	DATE	ORIGIN	DESTINATION	TOTAL
te_z1_33	2022-11-07	Drive Chicago, IL 60601	Street Chicago, IL 60608	\$424.39
ITEMS				
1 x Pail (1 x 1 x 1)		PICKUP DATE		
		2022-12-30		
ITEMS				
tes_Z133	2022-11-09	803	808	\$418.99
ITEMS				
		PICKUP DATE		
		2022-11-24		

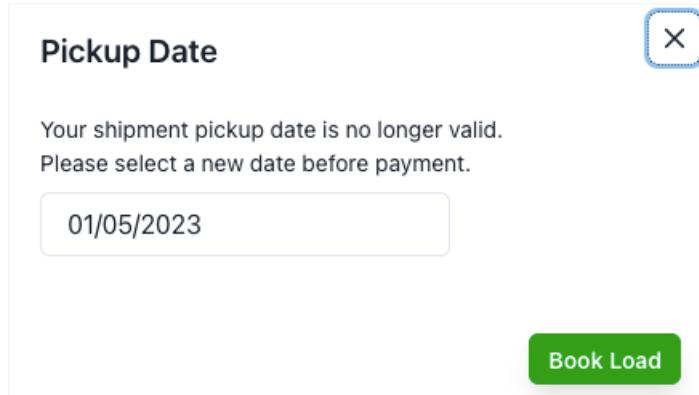
2. After an order has been placed and the mutation call for prepareOrder returns a successful response, the order can be seen in the vendor orders table. Payment can be made using the green 'checkmark' button. After the button is clicked, a payment modal will open up. Enter the credit card details and submit the payment.



4. Querying the Rates

```
const GET_RATES = gql`  
query RequestLTLQuote($rfq: RFQ!) {  
  requestLTLQuote(rfq: $rfq) {  
    cost  
    carrier{  
      name  
      id  
    }  
    id  
    deliveryDate  
  }  
}  
`;  
  
export default async function requestLTLQuote(rfq: RFQ) {  
  const resp = await graphqlClient.query<RequestLTLQuote>({  
    query: GET_RATES,  
    variables: { rfq },  
  });  
  
  const rates = resp?.data?.requestLTLQuote;  
  return { rates };  
}
```

In the case where payment is being submitted after the estimated ship date, an expired pickup date prompt will be displayed. Enter a new future pickup date. **It must be a future pickup date to book the shipment.**



3. After a successful payment, a shipment booking confirmation email is sent to the email address used to create the vendor account. The email contains the BOL document and details of the shipment.
4. Vendors can track and view shipment documents (e.g. BOL and Invoices) for all shipments from their GoShip View Orders Dashboard.

Shipping Dashboard Resources

My Order History

Request Quote

Earn \$100 in GoShip credit! Get Started

View Orders

Search by BOL

Vendor Orders

STATUS ^ BOL ^ TYPE ^ CARRIER ^ TRACKING # ^ PICKUP ^ DELIVERY ^

Carrier	Tracking #	Type	Pickup Date	Delivery Date
ABF FREIGHT SYSTEM, INC.	1401113	LTL	Oct 5	Oct 6
ABF FREIGHT SYSTEM, INC.	1401116	LTL	Sep 29	Sep 30
ABF FREIGHT SYSTEM, INC.	1401111	LTL	Sep 16	Sep 19
OLD DOMINION FREIGHT LINE INC	1401110	LTL	Aug 5	Silverthorne, CO Aug 12