Redkik 3 Embedded Insurance Software

Redkik 3 Embedded Insurance Software User Guide

Contents

Redkik 3 Embedded Insurance Software

Specifications

- **Product Name**: Redkik 3.0
- **Authentication**: OAuth 2.0
- **API Endpoint**: https://staging.app.redkik.com/api/v2/

Concepts

A Booking in Redkik represents coverage of a specific shipment. The lifecycle of a booking includes Setup, Quote, Purchase, Cancel, or Complete actions.

API Tokens

Redkik leverages OAuth 2.0 for authentication. API Tokens consist of a client_id and a secret_id keypair. To generate API Tokens, browse to Organizations, select your organization, and create a new API Token. Save the client_id and secret_id for future use.

Utilizing the Token and Listing Bookings

To list all bookings, use the provided access token in the Authorization header when making a GET request to the endpoint:

```
curl -X 'GET'
'https://staging.app.redkik.com/api/v2/quote/bookings' -H 'accept:
application/json' -H 'Authorization: Bearer
YOUR_ACCESS_TOKEN'
```

Generate a Quote and Purchasing It

Before generating a quote, retrieve setup information from the /quotes/setup endpoint. To create a quote, submit a JSON structure to the /Bookings/quote endpoint.

FAQ

- How do I refresh an access token?
- To refresh an access token, use the refresh token provided when generating the initial access token. Make a request to the token endpoint with the refresh token to obtain a new access token.
- What is the lifespan of an access token?
- The access token has a limited lifespan for security reasons. It is recommended to refresh the token before it expires to maintain continuous access to the API.

Redkik 3 API usage
Oct, 2023: Updated for Redkik 3.0

Introduction

Redkik has launched its third iteration of the Redkik platform, appropriately known as Redkik 3.0. This guide quickly walks through the basic concepts and basic actions available through the Redkik RESTful API and common patterns.

Concepts

Bookings

A Booking represents coverage of a specific shipment. The steps of creating a quote and then purchasing it are simple:

1. Generat
    1. Fields
        1. Commodity information
        2. Journey information
        3. Cargo Owner information
    2. retain the ID of the quote, returned with a successful quote
2. then use that ID to purchase the offer, which creates a booking. The Lifecycle of a Booking is:
    1. Setup
    2. Quote
    3. Purchase
    4. Cancel, or
    5. Complete

API Tokens

Redkik leverages OAUTH 2.0 for authentication strategies. To learn more about oAuth2, please go to:

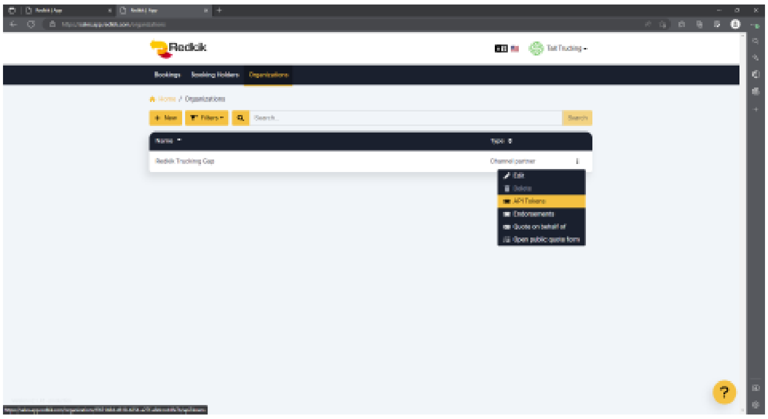
OAuth 2.0 — OAuth

The authentication flow is as follows

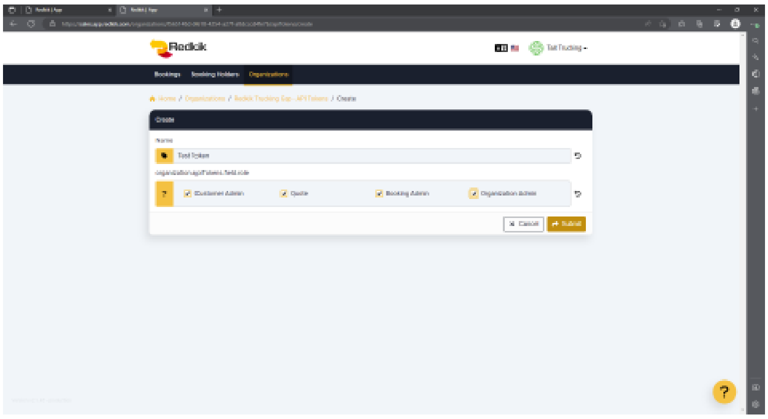
1. Generate API Tokens for an organization, typically Redkik's job.
2. Generate a one-time, limited-lifespan access token for endpoint access
3. Refresh Access Tokens for continued use of that authentication session

Generate API Tokens

API Tokens are generated for an organization, consisting of a client_id and a secret_id keypair, and are used to generate one-time access tokens with a limited time-to-live. Browse to Organizations, select your organization (or a child-org if creating a Quote "On Behalf Of" that child-org), right-click on the org's config button, and select API Tokens:



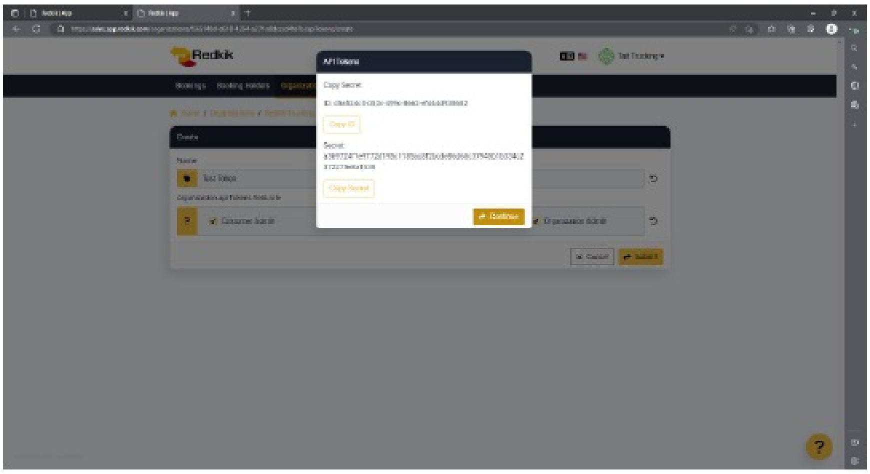
Add a new API Token:


Redkik
Bookings
Booking Holders
Organizations
Home / Organizations / Redkik Trucking Corp - API Tokens / Create
Create
Cancel
Submit

Name: Test Token

☑ Customer Admin ☑ Quote ☑ Booking Admin ☑ Organization Admin

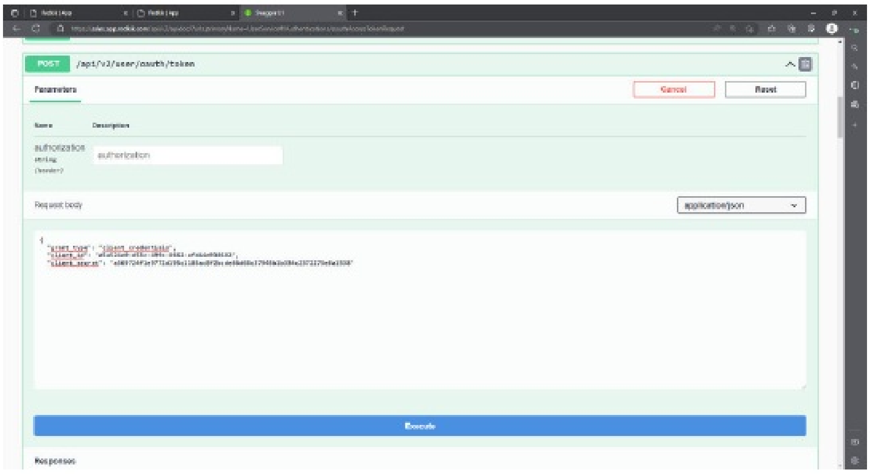
Upon Submission, you will have access to the client_id and the secret_id. Please ensure you save these off as this will be the only time you have access to them



ID: [illegible]

Secret: [illegible]

Test it out in Swagger, using the /oauth/token endpoint:



POST /api/v1/user/oauth/token

Parameters

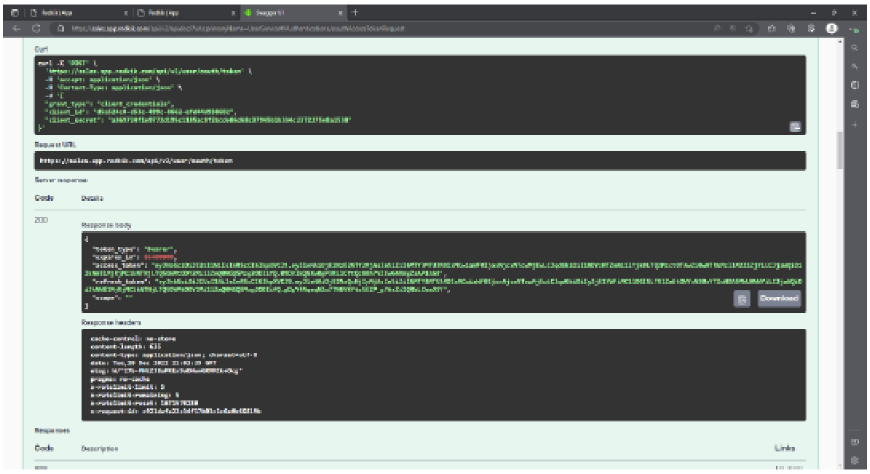
Name | Description

authorization
string
(header)

Request body: application/json

```
{
  "grant_type": "client_credentials",
  "client_id": "[illegible]",
  "client_secret": "[illegible]"
}
```

Producing an Access token and Refresh token



Curl

```
curl -X 'POST' \
  'https://sales.app.redkik.com/api/v1/user/oauth/token' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "grant_type": "client_credentials",
  "client_id": "[illegible]",
  "client_secret": "[illegible]"
}'
```

Request URL

```
https://sales.app.redkik.com/api/v1/user/oauth/token
```

Server response

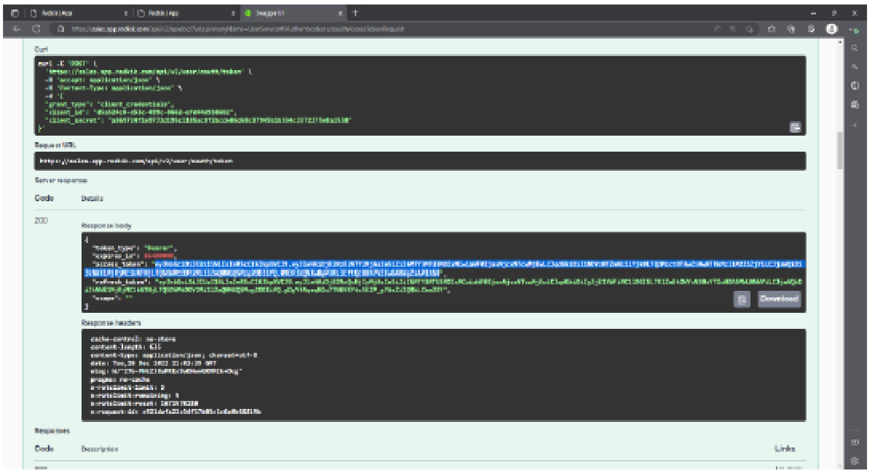
Code: 200

Response body

```
{
  "token_type": "Bearer",
  "expires_in": [illegible],
  "access_token": "[illegible]",
  "refresh_token": "[illegible]",
  "scope": ""
}
```

Response headers

```
cache-control: no-store
content-length: [illegible]
content-type: application/json; charset=utf-8
date: Tue,20 Dec 2022 [illegible] GMT
etag: [illegible]
pragma: no-cache
x-ratelimit-limit: [illegible]
x-ratelimit-remaining: [illegible]
x-ratelimit-reset: [illegible]
x-request-id: [illegible]
```



Curl

```
curl -X 'POST' \
  'https://sales.app.redkik.com/api/v1/user/oauth/token' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "grant_type": "client_credentials",
  "client_id": "[illegible]",
  "client_secret": "[illegible]"
}'
```

Request URL

```
https://sales.app.redkik.com/api/v1/user/oauth/token
```

Server response

Code: 200

Response body

```
{
  "token_type": "Bearer",
  "expires_in": [illegible],
  "access_token": "[illegible]",
  "refresh_token": "[illegible]",
  "scope": ""
}
```

Response headers

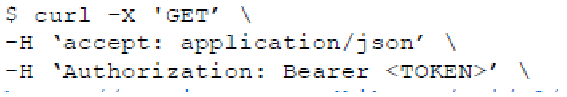
```
cache-control: no-store
content-length: [illegible]
content-type: application/json; charset=utf-8
date: Tue,20 Dec 2022 [illegible] GMT
etag: [illegible]
pragma: no-cache
x-ratelimit-limit: [illegible]
x-ratelimit-remaining: [illegible]
x-ratelimit-reset: [illegible]
x-request-id: [illegible]
```

Refresh Tokens

When API Tokens are used to generate an access token, they will also include a refresh token. This refresh token can be used to generate a new access token before the TTL of that access token expires, in order to keep that particular session alive. If the access token expires before it can be refreshed, the client_id and secret_id must be used again to generate a new access token.

Utilizing the token and Listing Bookings

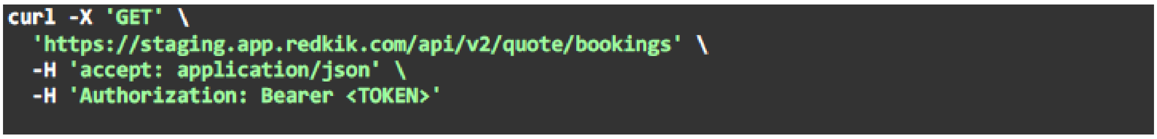
To GET bookings using the token: e.g.: Authorization: Bearer <TOKEN> For example, when using curl, one would need to include the token as a header specified on the command line



```
$ curl -X 'GET' \
-H 'accept: application/json' \
-H 'Authorization: Bearer <TOKEN>' \
```

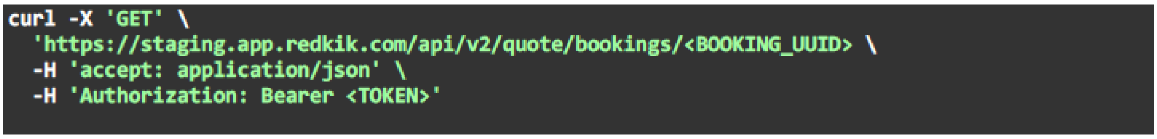
https://staging.app.redkik.com/api/v2/quote/bookings

Listing All Bookings



```
curl -X 'GET' \
  'https://staging.app.redkik.com/api/v2/quote/bookings' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer <TOKEN>'
```

Listing a single Booking's detailed information

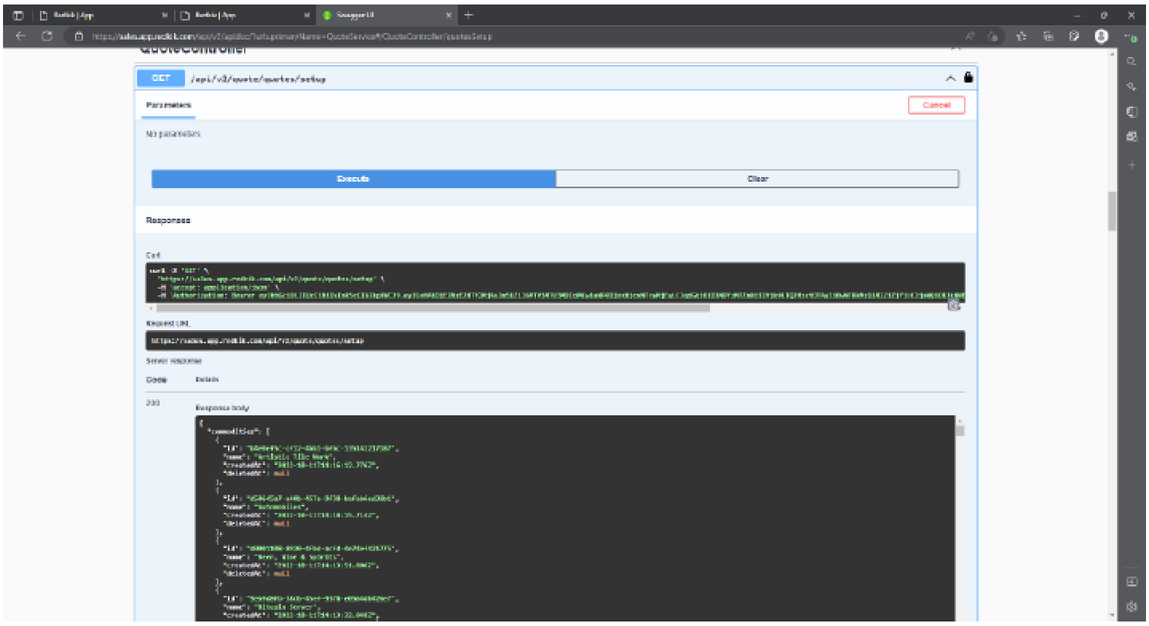


```
curl -X 'GET' \
  'https://staging.app.redkik.com/api/v2/quote/bookings/<BOOKING_UUID> \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer <TOKEN>'
```

Generate a Quote and Purchasing it

Getting Setup Information Before Generating a Quote

Optionally, we pre-populate a JSON structure with setup information around the policy context, available commodities, etc, which can help populate any UI forms upstream. This is not a required step, but it assists in setting the correct UUIDs for commodities and sundry. That endpoint is here: /quotes/setup



```
curl -X 'GET' \
  'https://[illegible].app.redkik.com/api/v2/quote/quotes/setup' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer [illegible]
```

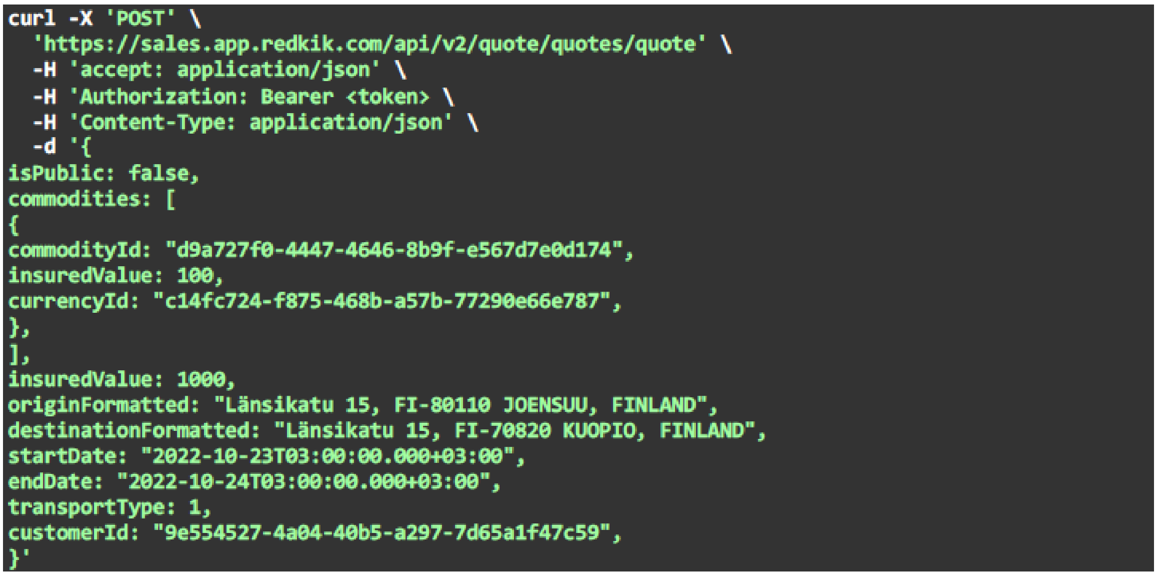
```
{
  "commodities": [
    {
      "id": "[illegible]",
      "name": "[illegible]",
      "createdAt": "[illegible]",
      "deletedAt": null
    },
    {
      "id": "[illegible]",
      "name": "Automobiles",
      "createdAt": "[illegible]",
      "deletedAt": null
    },
    {
      "id": "[illegible]",
      "name": "Beer, Wine & Spirits",
      "createdAt": "[illegible]",
      "deletedAt": null
    },
    {
      "id": "[illegible]",
      "name": "[illegible]",
      "createdAt": "[illegible]",
```

Creating a Quote

To create a quote, one must POST the following JSON structure to the /Bookings/quote endpoint (this is in its most basic form): /quote/quotes/quote:

/quote/quotes/quote:

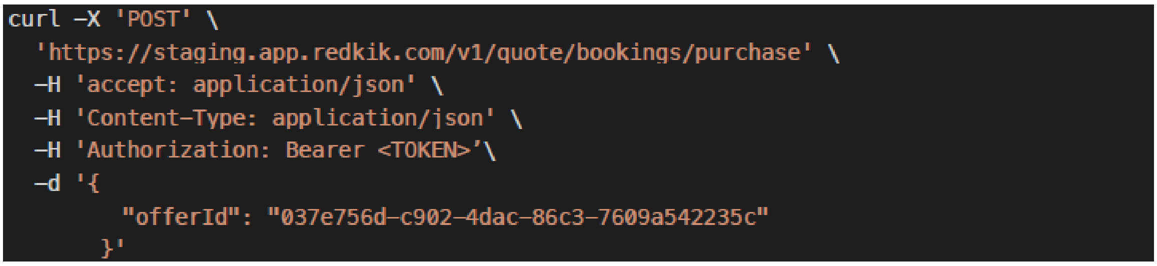
E.g.:



```
curl -X 'POST' \
  'https://sales.app.redkik.com/api/v2/quote/quotes/quote' \
  -H 'accept: application/json' \
  -H 'Authorization: Bearer <token> \
  -H 'Content-Type: application/json' \
  -d '{
isPublic: false,
commodities: [
{
commodityId: "d9a727f0-4447-4646-8b9f-e567d7e0d174",
insuredValue: 100,
currencyId: "c14fc724-f875-468b-a57b-77290e66e787",
},
],
insuredValue: 1000,
originFormatted: "Länsikatu 15, FI-80110 JOENSUU, FINLAND",
destinationFormatted: "Länsikatu 15, FI-70820 KUOPIO, FINLAND",
startDate: "2022-10-23T03:00:00.000+03:00",
endDate: "2022-10-24T03:00:00.000+03:00",
transportType: 1,
customerId: "9e554527-4a04-40b5-a297-7d65a1f47c59",
}'
```

Purchasing the Quote

To purchase the quote, one retains the ID (64-bit UUID) returned with the quote (in bold) and submits to the /purchase endpoint.



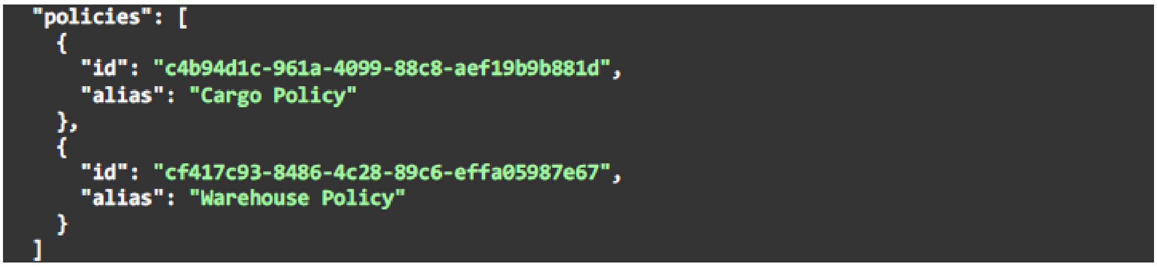
```
curl -X 'POST' \
  'https://staging.app.redkik.com/v1/quote/bookings/purchase' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -H 'Authorization: Bearer <TOKEN>'\
  -d '{
        "offerId": "037e756d-c902-4dac-86c3-7609a542235c"
      }'
```

Cancelling the Quote

To cancel a booking, use the /cancel endpoint Advanced Quoting Quoting from multiple policies

Finding Policy IDs

Some organizations have multiple policies from which they may quote. In those instances, the policies are listed in the /setup call.



```
  "policies": [
    {
      "id": "c4b94d1c-961a-4099-88c8-aef19b9b881d",
      "alias": "Cargo Policy"
    },
    {
      "id": "cf417c93-8486-4c28-89c6-effa05987e67",
      "alias": "Warehouse Policy"
    }
  ]
```

Quoting on behalf of

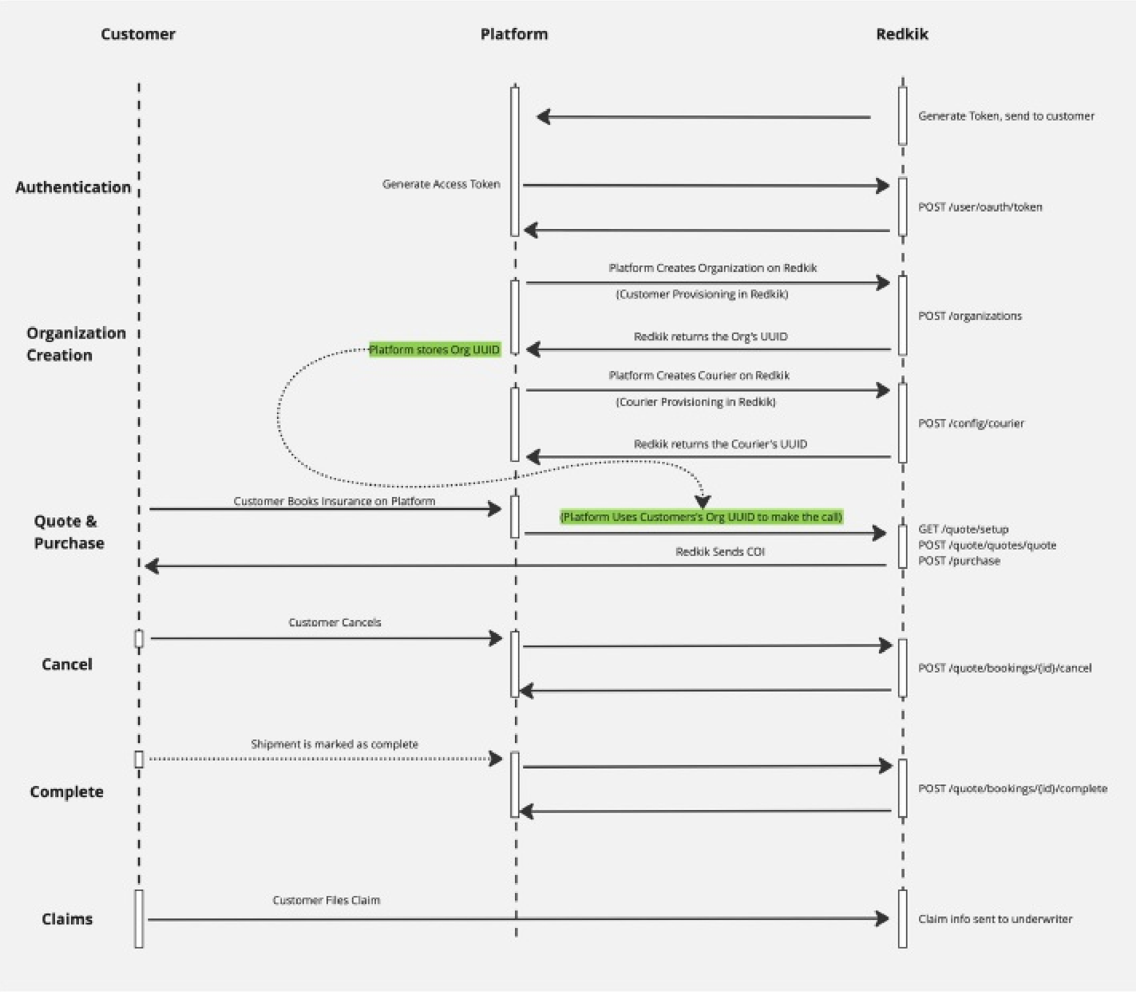
To create a booking on behalf of a child org, simply pass that child org’s id with the quote: “orgId”: “cf417c93-8486-4c28-89c6-effa05987e67”

Obtaining the Certificate of Insurance

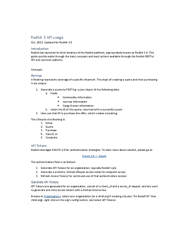
- TBD

- Referrals
- TBD
- Referring a Booking
- Listing Referred Bookings
- Accepting a Referred Booking (Underwriter)
- Purchasing a Referred Booking (Customer)

API Swimlane


Customer
Platform
Redkik
Authentication
Generate Token, send to customer
Generate Access Token
POST /user/oauth/token
Organization Creation
Platform Creates Organization on Redkik
(Customer Provisioning in Redkik)
POST /organizations
Platform stores Org UUID
Redkik returns the Org's UUID
Platform Creates Courier on Redkik
(Courier Provisioning in Redkik)
POST /config/courier
Redkik returns the Courier's UUID
Quote & Purchase
Customer Books Insurance on Platform
(Platform Uses Customers's Org UUID to make the call)
GET /quote/setup
POST /quote/quotes/quote
POST /purchase
Redkik Sends COI
Cancel
Customer Cancels
POST /quote/bookings/{id}/cancel
Complete
Shipment is marked as complete
POST /quote/bookings/{id}/complete
Claims
Customer Files Claim
Claim info sent to underwriter

Documents / Resources

	[Redkik 3 Embedded Insurance Software](#) [pdf] User Guide 3 Embedded Insurance Software, Embedded Insurance Software, Insurance Software, Software

References

- [staging.app.redkik.com/v1/quote/bookings/purchase](#)
- [User Manual](#)