1. ProjectX Gateway API | ProjectX API Documentation
2. ProjectX Gateway API | ProjectX API Documentation
3. Search for Account | ProjectX API Documentation
4. Search for Trades | ProjectX API Documentation
5. Account | ProjectX API Documentation
6. API Reference | ProjectX API Documentation
7. Authenticate | ProjectX API Documentation
8. Getting Started | ProjectX API Documentation
9. Market Data | ProjectX API Documentation
10. Orders | ProjectX API Documentation
11. Positions | ProjectX API Documentation
12. Realtime Updates | ProjectX API Documentation
13. Trades | ProjectX API Documentation
14. Authenticate (with API key) | ProjectX API Documentation
15. Authenticate (for authorized applications) | ProjectX API Documentation
16. Connection URLs | ProjectX API Documentation
17. Placing Your First Order | ProjectX API Documentation
18. ProjectX Gateway API | ProjectX API Documentation
19. Real Time Data Overview | ProjectX API Documentation

ProjectX Gateway API | ProjectX API Documentation

Documentation for the ProjectX Gateway API

: https://gateway.docs.projectx.com

Skip to main content

ProjectX Gateway API | ProjectX API Documentation

Documentation for the ProjectX Gateway API

: https://gateway.docs.projectx.com/

Skip to main content

Search for Account | ProjectX API Documentation

API URL: /api/Account/search

: https://gateway.docs.projectx.com/docs/api-reference/account/search-accounts

On this page

Search for Account

API URL : POST https://gateway-api-demo.s2f.projectx.com/api/Account/search **API Reference** : **/api/account/search**

Description

Search for accounts.

Parameters

Name	Type	Description	Required	Nullable
onlyActiveAccounts	boolean	Whether to filter only active accounts.	Required	false

Example Usage

Example Request

- cURL Request

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Account/search' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"onlyActiveAccounts": true
}'
```

Example Response

- Success
- Error

```
 {
"accounts": [ { "id": 1, "name": "TEST_ACCOUNT_1", "balance": 50000,
"canTrade": true,
"isVisible": true
}
```

], "success": true, "errorCode": 0, "errorMessage": null }

Error: response status is 401

Search for Trades | ProjectX API Documentation

API URL: /api/Trade/search

: https://gateway.docs.projectx.com/docs/api-reference/trade/trade-search

On this page

Search for Trades

API URL : POST https://gateway-api-demo.s2f.projectx.com/api/Trade/search **API Reference** : **/api/Trade/search**

Description

Search for trades from the request parameters.

Parameters

Name	Type	Description	Required	Nullable
accountId	integer	The account ID.	Required	false
startTimestamp	datetime	The start of the timestamp filter.	Required	false
endTimestamp	datetime	The end of the timestamp filter.	Optional	true

Example Usage

Example Request

- cURL Request

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Trade/search' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"accountId": 203,
"startTimestamp": "2025-01-20T15:47:39.882Z",
"endTimestamp": "2025-01-30T15:47:39.882Z"
}'
```

Example Response

- Success
- Error

```
 {
"trades": [ { "id": 8604, "accountId": 203, "contractId": "CON.F.US.EP.H25",
"creationTimestamp": "2025-01-21T16:13:52.523293+00:00",
"price": 6065.250000000,
"profitAndLoss": 50.000000000,
"fees": 1.4000,
"side": 1,
"size": 1,
"voided": false,
"orderId": 14328
},
```

```
{
"id": 8603,
"accountId": 203,
"contractId": "CON.F.US.EP.H25",
"creationTimestamp": "2025-01-21T16:13:04.142302+00:00",
"price": 6064.250000000,
"profitAndLoss": null, //a null value indicates a half-turn trade
"fees": 1.4000,
"side": 0,
"size": 1,
"voided": false,
"orderId": 14326
}
```


], "success": true, "errorCode": 0, "errorMessage": null }

Error: response status is 401

Account | ProjectX API Documentation

: https://gateway.docs.projectx.com/docs/category/account

ðï¸ Search for AccountAPI URL: /api/Account/search

API Reference | ProjectX API Documentation

: https://gateway.docs.projectx.com/docs/category/api-reference

ðï¸ Account

ðï¸ Market Data## ðï¸ Orders

ðï¸ Positions## ðï¸ Trades

Authenticate | ProjectX API Documentation

This section outlines the process of authenticating API requests using JSON Web Tokens.

: https://gateway.docs.projectx.com/docs/category/authenticate

ðï¸ Authenticate (with API key)We utilize JSON Web Tokens to

authenticate all requests sent to the API. This process involves obtaining a session token, which is required for future requests.

ðï¸ Authenticate (for authorized applications)We utilize JSON Web Tokens to authenticate all requests sent to the API.

Getting Started | ProjectX API Documentation

We've designed a very robust API for accessing and managing all aspects of your firm.

: https://gateway.docs.projectx.com/docs/category/getting-started

ðï¸ Authenticate

ðï¸ Placing Your First OrderThis documentation outlines the process for placing your first order using our API. To successfully execute an order, you must have an active trading account associated with your user. Follow the steps below to retrieve your account details, browse available contracts, and place your order.## ðï¸ Connection URLs

---

Market Data | ProjectX API Documentation

Authorized users have access to order operations, allowing them to search for, modify, place, and cancel orders.

: https://gateway.docs.projectx.com/docs/category/market-data

ðï¸ Retrieve BarsAPI URL: /api/History/retrieveBars

ðï¸ Search for ContractsAPI URL: /api/Contract/search## ðï¸ Search for Contract by IdAPI URL: /api/Contract/searchById

---

Orders | ProjectX API Documentation

Authorized users have access to order operations, allowing them to search for, modify, place, and cancel orders.

: https://gateway.docs.projectx.com/docs/category/orders

ðï¸ Search for OrdersAPI URL: /api/Order/search

ðï¸ Search for Open OrdersAPI URL: /api/Order/searchOpen## ðï¸ Place an OrderAPI URL: /api/Order/place

ðï¸ Cancel an OrderAPI URL: /api/Order/cancel## ðï¸ Modify an OrderAPI URL: /api/Order/modify

---

Positions | ProjectX API Documentation

Authorized users have access to position operations, allowing them to search for, and close positions.

: https://gateway.docs.projectx.com/docs/category/positions

ðï¸ Close PositionsAPI URL: /api/Position/closeContract

ðï¸ Partially Close PositionsAPI URL: /api/Position/partialCloseContract## ðï¸ Search for PositionsAPI URL: /api/Position/searchOpen

---

Realtime Updates | ProjectX API Documentation

Realtime Updates for various events

: https://gateway.docs.projectx.com/docs/category/realtime-updates

ðï¸ Real Time Data OverviewThe ProjectX Real Time API utilizes SignalR library (via WebSocket) to provide real-time access to data updates involving accounts, orders, positions, balances and quotes.

Trades | ProjectX API Documentation

Authorized users have access to trade operations, allowing them to search for trades.

: https://gateway.docs.projectx.com/docs/category/trades

ðï¸ Search for TradesAPI URL: /api/Trade/search

Authenticate (with API key) | ProjectX API

Documentation

We utilize JSON Web Tokens to authenticate all requests sent to the API. This process involves obtaining a session token, which is required for future requests.

: https://gateway.docs.projectx.com/docs/getting-started/authenticate/authenticate-api-key

On this page

Authenticate (with API key) We utilize JSON Web Tokens to authenticate all requests sent to the API. This process involves obtaining a session token, which is required for future requests.

Step 1 To begin, ensure you have the following:

- An API key obtained from your firm. If you do not have these credentials, please contact your firm.
- The connection URLs, obtained **here**.

Step 2 API Reference: Login API Create a POST request with your username and API key.

- cURL Request

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Auth/loginKey' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"userName": "string",
"apiKey": "string"
}'
```

Step 3 Process the API response, and make sure the result is Success (0), then store your session token in a safe place. This session token will grant full access to the Gateway API.

- Response

```
{
  "token": "your_session_token_here",
  "success": true,
  "errorCode": 0,
  "errorMessage": null
}
```

Notes All further requests will require you to provide the session token in the "Authorization" HTTP header using the `Bearer` method. Session tokens are only valid for 24 hours. You must revalidate your

token to continue using the same session. The next step will explain how to extend / re-validate your session in case your token has expired.

Authenticate (for authorized applications) | ProjectX API Documentation

We utilize JSON Web Tokens to authenticate all requests sent to the API.

: https://gateway.docs.projectx.com/docs/getting-started/authenticate/authenticate-as-application

On this page

Authenticate (for authorized applications) We utilize JSON Web Tokens to authenticate all requests sent to the API.

Step 1 Retrieve the admin credentials (username and password, appId, and verifyKey) that have been provided for your firm. You will need these credentials to authenticate with the API. If you do not have these credentials, please contact your Account Manager for more information.

Step 2 API Reference: Login API Create a POST request with your username and password.

- cURL Request

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Auth/loginApp' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"userName": "yourUsername",
"password": "yourPassword",
"deviceId": "yourDeviceId",
"appId": "B76015F2-04D3-477E-9191-C5E22CB2C957",
"verifyKey": "yourVerifyKey"
}'
```

Step 3 Process the API response, and make sure the result is Success

(0), then store your session token in a safe place. This session token will grant full access to the Gateway API.

- Response

```
{
  "token": "your_session_token_here",
  "success": true,
  "errorCode": 0,
  "errorMessage": null
}
```

Notes All further requests will require you to provide the session token in the "Authorization" HTTP header using the `Bearer` method. Session tokens are only valid for 24 hours. You must revalidate your token to continue using the same session. The next step will explain how to extend / re-validate your session in case your token has expired.

Connection URLs | ProjectX API Documentation

: https://gateway.docs.projectx.com/docs/getting-started/connection-urls

Connection URLs

Select an Environment Select environmentAlpha TicksBlue GuardianBluskyE8XFunding FuturesThe Futures DeskFutures EliteFXIFY FuturesGoatFundedTickTickTraderTopOneFuturesTopstepXTX3Funding

---

Placing Your First Order | ProjectX API Documentation

This documentation outlines the process for placing your first order using our API. To successfully execute an order, you must have an active trading account associated with your user. Follow the steps below to retrieve your account details, browse available contracts, and place your order.

: https://gateway.docs.projectx.com/docs/getting-started/placing-your-first-order

On this page

Placing Your First Order This documentation outlines the process for placing your first order using our API. To successfully execute an order, you must have an active trading account associated with your user. Follow the steps below to retrieve your account details, browse available contracts, and place your order.

Step 1 To initiate the order process, you must first retrieve a list of active accounts linked to your user. This step is essential for confirming your account status before placing an order.

API URL : POST https://gateway-api-demo.s2f.projectx.com/api/account/search **API Reference** : **/api/account/search** * Request * Response * cURL Request

```
{
  "onlyActiveAccounts": true
}
```

```
 {
"accounts": [ { "id": 1, "name": "TEST_ACCOUNT_1", "canTrade": true,
"isVisible": true
}
```

], "success": true, "errorCode": 0, "errorMessage": null }

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Account/search' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"onlyActiveAccounts": true
}'
```

Step 2 Once you have identified your active accounts, the next step is to retrieve a list of contracts available for trading. This information will assist you in choosing the appropriate contracts for your order.

API URL : POST https://gateway-api-demo.s2f.projectx.com/api/contract/search **API Reference** : **/api/contract/search** * Request * Response * cURL Request

```
{
  "live": false,
  "searchText": "NQ"
}
```

```
 {
"contracts": [ { "id": "CON.F.US.ENQ.H25", "name": "ENQH25", "description": "E-mini NASDAQ-100: March 2025",
"tickSize": 0.25,
"tickValue": 5,
"activeContract": true
},
```

```
 {
"id": "CON.F.US.MNQ.H25",
"name": "MNQH25",
"description": "Micro E-mini Nasdaq-100: March 2025",
"tickSize": 0.25,
"tickValue": 0.5,
"activeContract": true
},
```

```
 {
"id": "CON.F.US.NQG.G25",
"name": "NQGG25",
"description": "E-Mini Natural Gas: February 2025",
"tickSize": 0.005,
"tickValue": 12.5,
"activeContract": true
},
```

```
{
  "id": "CON.F.US.NQM.G25",
  "name": "NQMG25",
  "description": "E-Mini Crude Oil: February 2025",
  "tickSize": 0.025,
  "tickValue": 12.5,
  "activeContract": true
}
```


], "success": true, "errorCode": 0, "errorMessage": null }

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Contract/search' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"live": false,
"searchText": "NQ"
}'
```

Final Step Having noted your account ID and the selected contract ID, you are now ready to place your order. Ensure that you provide accurate details to facilitate a successful transaction.

API URL : POST https://gateway-api-demo.s2f.projectx.com/api/order/place **API Reference** : **/api/order/place**

Parameters Name| Type| Description| Required| Nullable

---|---|---|---|--- accountId| integer| The account ID.| Required| false contractId| string| The contract ID.| Required| false type| integer| The order type: `1` = Limit `2` = Market `4` = Stop `5` = TrailingStop `6` = JoinBid `7` = JoinAsk| Required| false side| integer| The side of the order: `0` = Bid (buy) `1` = Ask (sell)| Required| false size| integer| The size of the order.| Required| false limitPrice| decimal| The limit price for the order, if applicable.| Optional| true stopPrice| decimal| The stop price for the order, if applicable.| Optional| true trailPrice| decimal| The trail price for the order, if applicable.| Optional| true customTag| string| An optional custom tag for the order.| Optional| true linkedOrderId| integer| The linked order id.| Optional| true * Request * Response * cURL Request


```
{
  "accountId": 1,
  "contractId": "CON.F.US.DA6.M25",
```

```
  "type": 2,
  "side": 1,
  "size": 1,
  "limitPrice": null,
  "stopPrice": null,
  "trailPrice": null,
  "customTag": null,
  "linkedOrderId": null
}
```

```
{
  "orderId": 9056,
  "success": true,
  "errorCode": 0,
  "errorMessage": null
}
```

curl -X 'POST' \ 'https://gateway-api-demo.s2f.projectx.com/api/Order/place' \ -H 'accept: text/plain' \ -H 'Content-Type: application/json' \

```
 -d '{
"accountId": 1,
"contractId": "CON.F.US.DA6.M25",
"type": 2,
"side": 1,
"size": 1,
"limitPrice": null,
"stopPrice": null,
"trailPrice": null,
"customTag": null,
"linkedOrderId": null
}'
```

ProjectX Gateway API | ProjectX API Documentation

Getting Started

: https://gateway.docs.projectx.com/docs/intro

On this page

ProjectX Gateway API

Getting Started ProjectX Trading, LLC - through its trading platform **ProjectX** , offers a complete end-to-end solution for prop firms and evaluation providers. These features include account customization, risk rules & monitoring, liquidations, statistics and robust permissioning. Our API utilizes the REST API architecture for managing your prop firm trader operations.

What you'll need

- An understanding of REST API
- cURL or Postman for making sample requests
- Getting Started
- What you'll need

Real Time Data Overview | ProjectX API Documentation

The ProjectX Real Time API utilizes SignalR library (via WebSocket) to provide real-time access to data updates involving accounts, orders, positions, balances and quotes.

: https://gateway.docs.projectx.com/docs/realtime

On this page

Real Time Data Overview The ProjectX Real Time API utilizes SignalR library (via WebSocket) to provide real-time access to data updates involving accounts, orders, positions, balances and quotes. There are two hubs: `user` and `market`.

- The user hub will provide real-time updates to a user's accounts, orders, and positions.
- The market hub will provide market data such as market trade events, DOM events, etc.

What is SignalR? SignalR is a real-time web application framework developed by Microsoft that simplifies the process of adding real-time functionality to web applications. It allows for bidirectional communication between clients (such as web browsers) and servers, enabling features like live chat, notifications, and real-time updates without the need for constant client-side polling or manual handling of connections. SignalR abstracts away the complexities of real-time communication by providing high-level APIs for developers. It supports various transport protocols, including WebSockets, Server-Sent Events (SSE), Long Polling, and others, automatically selecting the most appropriate transport mechanism based on the capabilities of the client and server. The framework handles connection management, message routing, and scaling across multiple servers, making it easier for developers to build scalable and responsive web

applications. SignalR is available for multiple platforms, including .NET and JavaScript, allowing developers to build real-time applications using their preferred programming languages and frameworks.

Further information on SignalR can be found here.

Example Usage

- User Hub
- Market Hub

// Import the necessary modules from @microsoft/signalr const { HubConnectionBuilder, HttpTransportType } = require('@microsoft/signalr');

// Function to set up and start the SignalR connection

```
 function setupSignalRConnection() {
const JWT_TOKEN = 'your_bearer_token';
const SELECTED_ACCOUNT_ID = 123; //your currently selected/visible account ID
const userHubUrl = 'https://gateway-rtc-demo.s2f.projectx.com/hubs/user?access_token=' + JWT_TOKEN;

// Create the connection
const rtcConnection = new HubConnectionBuilder()
.withUrl(userHubUrl, {
skipNegotiation: true,
transport: HttpTransportType.WebSockets,
accessTokenFactory: () => JWT_TOKEN, // Replace with your current JWT token
timeout: 10000 // Optional timeout
})
```

.withAutomaticReconnect() .build();

// Start the connection rtcConnection.start()

```
 .then(() => {
// Function to subscribe to the necessary events
const subscribe = () => {
rtcConnection.invoke('SubscribeAccounts');
rtcConnection.invoke('SubscribeOrders', SELECTED_ACCOUNT_ID); //you can call this function multiple times with differen
rtcConnection.invoke('SubscribePositions', SELECTED_ACCOUNT_ID); //you can call this function multiple times with diffe
rtcConnection.invoke('SubscribeTrades', SELECTED_ACCOUNT_ID); //you can call this function multiple times with differen
};
```

// Functions to unsubscribe, if needed

```
 const unsubscribe = () => {
rtcConnection.invoke('UnsubscribeAccounts');
rtcConnection.invoke('UnsubscribeOrders', SELECTED_ACCOUNT_ID); //you can call this function multiple times with differ
rtcConnection.invoke('UnsubscribePositions', SELECTED_ACCOUNT_ID); //you can call this function multiple times with dif
rtcConnection.invoke('UnsubscribeTrades', SELECTED_ACCOUNT_ID); //you can call this function multiple times with differ

};
```

// Set up the event listeners

```
 rtcConnection.on('GatewayUserAccount', (data) => {
console.log('Received account update', data);
});
```

```
 rtcConnection.on('GatewayUserOrder', (data) => {
console.log('Received order update', data);
});
```

```
 rtcConnection.on('GatewayUserPosition', (data) => {
console.log('Received position update', data);
});
```

```
 rtcConnection.on('GatewayUserTrade', (data) => {
console.log('Received trade update', data);
});
```

// Subscribe to the events subscribe();

// Handle reconnection

```
 rtcConnection.onreconnected((connectionId) => {
console.log('RTC Connection Reconnected');
subscribe();
});
```

})

```
 .catch((err) => {
console.error('Error starting connection:', err);
});
```

} // Call the function to set up and start the connection setupSignalRConnection();

// Import the necessary modules from @microsoft/signalr const { HubConnectionBuilder, HttpTransportType } = require('@microsoft/signalr');

// Function to set up and start the SignalR connection

```
 function setupSignalRConnection() {
const JWT_TOKEN = 'your_bearer_token';
const marketHubUrl = 'https://gateway-rtc-demo.s2f.projectx.com/hubs/market?access_token=' + JWT_TOKEN;
const CONTRACT_ID = 'CON.F.US.RTY.H25'; // Example contract ID


// Create the connection
const rtcConnection = new HubConnectionBuilder()
.withUrl(marketHubUrl, {
skipNegotiation: true,
transport: HttpTransportType.WebSockets,
accessTokenFactory: () => JWT_TOKEN, // Replace with your current JWT token
timeout: 10000 // Optional timeout
})
```

.withAutomaticReconnect() .build();

// Start the connection rtcConnection.start()

```
 .then(() => {
// Function to subscribe to the necessary events
const subscribe = () => {
rtcConnection.invoke('SubscribeContractQuotes', CONTRACT_ID);
rtcConnection.invoke('SubscribeContractTrades', CONTRACT_ID);
rtcConnection.invoke('SubscribeContractMarketDepth', CONTRACT_ID);
};
```

// Functions to unsubscribe, if needed

```
 const unsubscribe = () => {
rtcConnection.invoke('UnsubscribeContractQuotes', CONTRACT_ID);
rtcConnection.invoke('UnsubscribeContractTrades', CONTRACT_ID);
rtcConnection.invoke('UnsubscribeContractMarketDepth', CONTRACT_ID);
};
```

// Set up the event listeners

```
 rtcConnection.on('GatewayQuote', (contractId, data) => {
console.log('Received market quote data', data);
});
```

```
 rtcConnection.on('GatewayTrade', (contractId, data) => {
console.log('Received market trade data', data);
});
```

```
 rtcConnection.on('GatewayDepth', (contractId, data) => {
console.log('Received market depth data', data);
});
```

// Subscribe to the events subscribe();

// Handle reconnection

```
 rtcConnection.onreconnected((connectionId) => {
console.log('RTC Connection Reconnected');
subscribe();
});
```

})

```
 .catch((err) => {
console.error('Error starting connection:', err);
});
```

} // Call the function to set up and start the connection setupSignalRConnection();

- What is SignalR?
- Example Usage