eeStec LC Zagreb

ALGOTRADE hackathon 2025

Powered by WINCENT

AlgoTrade 2025 Competition Guide

Competition Overview

Welcome to **AlgoTrade 2025** - a high-frequency algorithmic trading competition where teams compete to build the most profitable trading strategies for derivatives markets. You'll be trading futures and options in a simulated environment that mirrors real-world trading conditions.

Your Mission

Maximize your portfolio's profit by developing algorithmic trading strategies that can:

- Trade futures and options contracts
- React to real-time market data
- Manage risk and positions effectively
- Outperform other competing teams (you all are in the same market)

Available Instruments

Underlying instruments
- **THESE ARE NOT TRADEABLE**
- **Use case**: all derivatives settle based on their price at a certain time.

Futures Contracts [Wiki]
- **ID format**: `[underlying]_future_[expiry second]`
  (e.g. *$CARD_future_1400*)
- **Use case**: Direct exposure to delivery at a certain time in the future
- **Expiry**: Number of seconds since the start of the round at which settlement starts
- **Payoff**: You get the cash amount equal to the price at that moment in time

Options Contracts [Wiki]
- **Call Options ID format**: `[underlying]_call_[strike price]_[expiry second]` (various strike prices) (e.g. *$CARD_call_39000_1400*)
- **Put Options ID format**: `[underlying]_put_[strike price]_[expiry second]`
(various strike prices) (e.g. *$CARD_put_39000_1400*)
- **Expiry**: Number of seconds since the start of the round at which settlement happens
- **Payoff for call**: You get max(price_at_expiry - strike_price, 0)
- **Payoff for put**: You get max(strike_price - price_at_expiry, 0)

Rate Limits & Rules

Connection Limits
- Maximum **of 10 connections at a time** per team
- Maximum **of 10 connections per second** per team
- Maximum **of 300 messages per second** per team
- Maximum **of 600 pending orders at a time** per team

Trading Rules
- **Starting Capital**: Each team begins with a set amount, 1 000 000$
- **Settlement**: Contracts settle automatically at expiry
- **Performance Metric**: Absolute net worth at the end of each round

Interacting with the Platform

Web Interface (Recommended to try first)

- **URL**: `192.168.100.10`
- **Features**: Real-time charts, order placement, portfolio tracking
- **Best for**: Getting to know the platform, gaining intuition and visual referencing
- **WARNING**: Each open tab of the web interface consumes one of your team's connections, so use it carefully

WebSocket API

- **Endpoint**: `ws://[URL]:9001/trade?team_secret=YOUR_TEAM_SECRET`
- **Protocol**: JSON-based WebSocket messages
- **Best for**: Scored trading rounds

Getting Your Credentials

Each team receives:

- **Team Name**: Your unique team identifier
- **Team ID**: Numeric identifier for your team
- **Team Secret**: Authentication key for API access

You will receive these physically at the competition, as well as the credentials to connect to the Wi-Fi.

Collocation and delay server

- In trading, **collocation** means placing servers near an exchange to **reduce latency and execute trades faster**. It's used to gain a **speed advantage** in high▪ frequency trading.
- In our competition, collocation is simulated using Raspberry Pis. Each team has its own Raspberry Pi that you can connect to using SSH
- Exact instructions and credentials for connecting to your Raspberry Pi will be printed out and given to you at the competition
- Your direct connection to the server **has a certain delay**. During the rounds **some additional delay** will be added. On the other hand, **Raspberry Pis have 0 delay**
- Delay **will impact** how fast your orders will be received and handled by the server

Test rounds

- Test rounds **do not** count towards final standings
- Each test round lasts for 10 minutes (600 seconds)

Scored rounds & Competition Format

- There will be **5 scored rounds**. These rounds **count towards your final standings**.
- Each scored round lasts for **30 minutes (1800 seconds)**.
- Each round has a **different coefficient** that determines its impact on your final score.
- Each round has a **different delay** that will impact your bots.
- Your score in each round is normalized based on **your relative performance** between the best and worst teams.
- Your **total score** is the sum of the scores from all rounds, calculated using the formula:

$$\sum_{i=1}^{5} coef_i * \frac{S_i - S_{min}}{S_{max} - S_{min}}$$

Where:
- S_i = your team's net worth in the i-th round
- S_min = net worth of the lowest-ranked team in that round
- S_max = net worth of the highest-ranked team in that round

Round #No	Time	Coefficient	Delay
1	23:00 - 23:30	1000	0 ms
2	06:00 - 06:30	1500	100 ms
3	09:00 - 09:30	2000	200 ms
4	13:00 - 13:30	2500	350 ms
5	16:00 - 16:30	3000	500 ms

Troubleshooting

Common Issues

- **401 Unauthorized**: Check your team secret
- **429 Rate Limited**: Reduce message frequency
- **Order Rejected**: Check balance and/or inventory
- **Connection Drops**: Implement reconnection logic

Good luck, and may the best algorithm win!

For technical support during the competition, contact the organizers by shouting loudly

---

@Tech Support on Discord works as well

Appendix: WebSocket API Quick Start

1. Connect to the Trading API

```
const ws = new WebSocket("ws://[URL]/trade?team_secret=YOUR_
SECRET");

ws.onopen = () => {
  console.log("Connected to AlgoTrade 2025!");
};

ws.onmessage = (event) => {
  const message = JSON.parse(event.data);
  console.log("Received:", message);
};
```

2. Place Your First Order

```
const order = {
  "type": "add_order",
  "user_request_id": "0000000000", // Same entry is returned
in the reply
  "instrument_id": "$CARD_call_82630_360",
  "price": 10, // Price in cents, always an integer
  "expiry": 58233560, Date.now() + 3600000, // Expires in 1
hour, auto capped to option expiry
  "side": "bid", // "bid" to buy, "ask" to sell
  "quantity": 1 // Number of contracts
}

ws.send(JSON.stringify(order));
```

Response sample:

```
{
  "type": "add_order_response",
  "user_request_id": "0000000000",
  "success": true,
  "data": { "order_id": 34 }
}
```

3. Monitor Your Portfolio

```
const inventoryRequest = {
  type: “get_inventory”,
  user_request_id: “my_req1”,
};

ws.send(JSON.stringify(inventoryRequest));

const ordersRequest = {
  type: “get_pending_orders”,
  user_request_id: “my_req2”,
};

ws.send(JSON.stringify(ordersRequest));
```

Example responses:

```
{
  “type”: “get_inventory_response”,
  “user_request_id”: “my_req1”,
  “data”: { “$”: [1760, 1000000] }
}

 {
  “type”: “get_pending_orders_response”,
  “user_request_id”: “my_req2”,
  “data”: {
   “$CARD_future_60”: [
     [
       // bids
       {
          “orderID”: 25,
          “teamID”: 1,
          “price”: 1,
          “time”: 5694,
          “expiry”: 60000,
          “side”: “BID”,
          “unfilled_quantity”: 1,
          “total_quantity”: 1,
          “live”: true
       }
     ],
     [] // asks
   ]
 }
}
```

4. Cancel Orders

```
const cancelOrder = {
  type: “cancel_order”,
  user_request_id: “my_cancel”,
  order_id: 23,
  instrument_id: “$CARD_call_100676_75”,
};

ws.send(JSON.stringify(cancelOrder));
```

Response:

```
{
  “type”: “cancel_order_response”,
  “user_request_id”: “my_cancel”,
  “success”: true
}
```

Real-Time Market Data

The platform automatically streams:

- **Price Updates**: Real-time instrument prices
- **Order Book**: Live bid/ask depth
- **Trade Executions**: Your successful trades
- **Market Events**: Settlement notifications

```
ws.onmessage = (event) => {
  const data = JSON.parse(event.data);
  switch (data.type) {
  case “market_data_update”:
   console.log(`${data.instrument_id}: $${data.price /
100}`);
   break;

  case “add_order_response”:
   if (data.success) {
    console.log(`Order placed: ID ${data.data.order_id}`);

   } else {
    console.log(`Order failed: ${data.data.message}`);
   }

   break;

  case “trade_update”:
   console.log(`Trade executed: ${data.quantity} @ $${data.
price / 100}`);
   break;
 }
};
```