



Binance futures trading bot

R.Kumaravelu |date:09-07-2025

1. Project Overview

This Python CLI tool allows users to interact with the Binance Futures Testnet. It supports placing:

- Market orders
- Limit orders
- Stop-limit (STOP_MARKET) orders

It uses Binance's official API via the python-binance package and follows best practices including argument validation,

logging, and modular design

2. TECHNOLOGIES USED

Python 3.10+

- python-binance
- argparse
- logging
- Binance USDT-M Futures Testnet

3. PROJECT STRUCTURE

project/

---src/

----cli.py

----stop_limit_orders.py

----limit_order.py

----market_order.py

----logger.py

----utils.py

---logger.py

---bot.log

---README.md

---report.pdf

4. HOW IT WORKS

Example command to place a stop-limit (STOP_MARKET) order:

```
python src/cli.py stoplimit BTCUSDT BUY 0.01 --price 108400 --stop_price 108500
```

--stop_price: activation trigger

--price: used for logging

5. TESTING

- Connected to Binance Testnet
- Verified with valid/invalid orders
- Error-handling confirmed
- All activity logged

6. DESIGN HIGHLIGHTS

- Modular code
- Minimal dependencies
- Logging with timestamps
- Full traceback on failure
- Easily extendable

7. SAMPLE LOG OUTPUT

```
2025-07-08 16:30:12 - INFO - Stop-limit order placed: BTCUSDT BUY 0.01 | STOP=108500  
LIMIT=108400
```

```
2025-07-08 16:32:44 - ERROR - Failed to place order: APIError(code=-2021)...
```

8.PROJECT SNAPS

The screenshot displays the PyCharm IDE interface. The left sidebar shows the project structure for 'kumaravelu_binance_bot', including files like 'botLog', 'cli.py', 'limit_orders.py', 'logger.py', 'market_orders.py', 'stop_limit_orders.py', and 'utils.py'. The main editor window shows the 'utils.py' file with the following code:

```
1 import logging
2 import os
3 from binance.client import Client
4 from dotenv import load_dotenv
5 from binance.enums import *
6
7 def load_keys():
8     load_dotenv()
9     return os.getenv("API_KEY"), os.getenv("API_SECRET")
10
11 def init_logger():
12     logging.basicConfig(
13         filename="bot.log",
14         level=logging.INFO,
15         format="%(asctime)s - %(levelname)s - %(message)s"
16     )
17
18 def init_client(api_key, api_secret):
19     client = Client(api_key, api_secret)
```

The terminal window at the bottom shows the execution of the bot. It starts with a command to place a market order for BTC/USD with a price of 108488 and a stop price of 108550. The output shows the order was placed successfully with details like 'orderId', 'symbol', 'status', 'clientOrderId', 'price', 'avgPrice', 'origQty', 'executedQty', 'cumQty', 'cumQuote', 'timeInForce', 'type', 'reduceOnly', 'closePosition', 'side', 'positionSide', 'stopPrice', 'workingType', 'contractPrice', 'priceProtect', 'origType', 'priceMatch', 'selfTradePreventionMode', 'goodTillDate', and 'updateTime'. Subsequent log messages show the bot's activity, including an error for an invalid API key, IP, or permissions, and several successful market and limit orders.

8.CONCLUSION

Binance Futures Testnet Trading Bot - Submission Report

This project is a CLI-based trading tool for Binance Testnet. It demonstrates production-readiness with validation,modularity, logging, and extendability