Binance Futures CLI Trading Bot – Technical Report

Developer: *Bhaskara Sri Vaishnavi*
Environment: Binance USDT-M Futures **Testnet**
Submission: Assignment – Backend / Python Binance Bot

1. Objective

The goal of this project is to design and implement a **command-line trading bot** for **Binance USDT-M Futures** with support for multiple order types, clear logging, validation, modular code design, and documentation.

The bot allows automated and semi-automated trade execution while ensuring safety by using Binance Testnet (virtual assets only).

2. System Architecture

```
binance_bot/
├── src/
│   ├── main.py            (CLI entry point)
│   ├── client.py          (Binance client + .env integration)
│   ├── logger.py          (Central structured logger)
│   ├── market_orders.py     (Market order logic)
│   ├── limit_orders.py      (Limit order logic)
│   ├── advanced/
│       ├── stop_limit.py      (Stop-Limit execution)
│       ├── oco.py           (Take-profit + Stop-loss pair)
│       ├── twap.py          (Time-Weighted Average Price)
│       ├── grid_strategy.py   (Grid trading execution)
├── .env                   (API keys – not committed)
├── bot.log                 (Execution and error logs)
├── README.md
└── report.pdf
```

3. Supported Order Types

Order Type	Description
Market Order	Executes immediately at best market price
Limit Order	Executes only if specified price is hit

Stop-Limit Order	Triggers a limit order only when stop price is reached
OCO (One-Cancels-Other)	Places take-profit and stop-loss simultaneously
TWAP Strategy	Splits trade into chunks executed over time
Grid Trading	Executes orders at multiple price levels in a range

All orders use **structured logging**, and both **success and failure events** are captured in bot.log.

4. Logging System

The logging module records:

- Timestamp
- Log level
- Action context
- Success or error reason
- Order IDs returned by Binance (if applicable)

Example log entries:

```
MARKET ORDER SUCCESS | BUY 0.01 ETHUSDT | OrderId: XXXXX
LIMIT ORDER SUCCESS | BUY 0.01 ETHUSDT @ 2155 | OrderId: XXXXX
STOP-LIMIT FAILED | Error: Order would immediately trigger
TWAP EXECUTED CHUNK 2/5 | BUY 0.01 ETHUSDT
GRID LEVEL 4/5 | Target Price=2180.0 | Order Qty=0.01
```

The log serves as an auditable execution trail.

5. Safety & Risk Control

To ensure **zero financial risk**, the bot operates exclusively on **Binance Futures Testnet**, not Mainnet.

.env file stores API keys securely and is excluded from version control.

6. Conclusion

The bot fulfills and exceeds assignment requirements:

Requirement	**Status**
Market / Limit Orders	✓
Advanced Orders (Stop-Limit, OCO)	✓
Algorithmic Strategies (TWAP + Grid)	✓
Structured Logging	✓

CLI-Based Execution	✓
Testnet Safety	✓

This implementation is modular, extendable, and production-aligned, demonstrating backend development skills in API integration, logging, automation, and system design.

End of Report

Bhaskara Sri Vaishnavi
Binance Futures CLI Bot — Technical Submission