Risk Engine Overview — Real-Time Options Trading Engine

A high-level explanation of how the system computes position size, stop loss, and target using Greeks, volatility, and defined risk parameters.

All examples and diagrams in this document use mock/sample data only.

What the Risk Engine Does

The Risk Engine is responsible for converting a selected strike and market conditions into **execution-ready trade parameters**, including:

- **Position Size**
- **Stop Loss (SL)**
- **Target (TP)**
- **Risk-Reward alignment**
- **Daily loss safeguards**

Why It Matters

This module ensures that every automated trade:

- Respects capital allocation limits
- Maintains consistent risk discipline
- Avoids oversized or unsafe positions
- Protects the system from high-volatility traps

Inputs to the Risk Engine

Core Inputs Used by the Risk Engine

- **Selected Strike** (from Strike Selector)
- **Live LTP (Premium)**
- **Delta, Gamma, Theta** (Greeks)
- **Volatility / SL Difference**
- **Capital Allocation %**
- **Max Loss per Trade**
- R**isk-Reward Ratio (RR)**

Position Sizing Logic

How the Risk Engine Calculates Position Size

Formula (Example – Generic)

Position Size = floor((Capital × Allocation%) / (Premium × LotSize))

The goal is to size positions consistently, so the engine never over-leverages or violates risk boundaries.

No proprietary formula is shown — only a conceptual example.

Stop Loss (SL) Computation

How SL Is Determined

The Stop Loss is derived using:

- **Delta sensitivity**
- **Gamma risk (curvature)**
- **Volatility-based SL distance**
- **Use of Black-Scholes Model**

$$C = SN(d_1) - Ke^{-rt}N(d_2)$$

where:

$$d_1 = \frac{ln\frac{S}{K} + (r + \frac{\sigma_s^2}{2})t}{\sigma_s\sqrt{t}}$$

and

$$d_2 = d_1 - \sigma_s\sqrt{t}$$

and where:

C = Call option price

S = Current stock (or other underlying) price

K = Strike price

r = Risk-free interest rate

t = Time to maturity

N = A normal distribution

Fig 1.1: **Black-Scholes Model**

SL = Premium – (Delta × Volatility Factor)

Purpose

- Avoid SLs that are too tight in volatile conditions
- Avoid SLs that are too wide to be inefficient
- Maintain stability during rapid price moves

(Reminder: This is an illustrative description only.)

Target Price (TP) Computation

Target Logic

The TP is derived directly from:

- SL distance
- Risk-Reward ratio (RR)

Example Representation

TP = Premium + (SL_Distance × RR)

Goal

Ensure every trade has a mathematically consistent reward–risk profile.

Safety Integration (Risk ←→ Safety Loop)

Risk Engine Outputs Are Re-Validated by Safety Framework

Before executing the trade, the Safety Framework checks:

- Daily max-loss limit
- PNL base capital allocation
- OI Support

Feedback Loop

The Safety Framework can return:

- Approval of the trade and execute
- Block the trade and show the reason
- Modify trade parameter and warns

This loop prevents unsafe trades even if the strike and SL/TP are technically valid.

Example Output (Mock Values)

For Strike: *24650 CE*

- Position Size: 75 qty (example)
- Stop Loss: 14.00 (example)
- Target: 28.00 (example)
- RR: 1:2
- Daily Loss Check: Passed
- Volatility Check: Passed
-

Summary

Risk Engine Summary

The Risk Engine is the core of safe automated trading:

- Converts strike + market data into safe parameters
- Ensures capital consistency
- Uses Greeks + volatility to size trades precisely
- Works with Safety Framework to prevent bad entries
- Outputs clean trade instructions for the Execution Layer

This ensures the system trades with discipline, consistency, and controlled risk.
