

Grid Martingale EA – Developer Brief (MQL5)

This document defines the functional and structural design for an MQL5 Expert Advisor that executes a price-level grid strategy with independent buy/sell cycles, manual martingale scaling, hidden take-profit logic, and a hedge-reset feature. The EA runs continuously without entry signals.

Input Parameters

```
// --- EA Input Parameters ---
input double   GridSizePips      = 30;           // Grid spacing in pips
input string   LotSizes         = "0.01;0.02;0.03;0.06;0.09;0.18;0.33;0.57;0.75";
input double   ProfitCashTarget = 10;           // Hidden basket profit target in account currency
input double   ProfitPipsTarget = 5;           // Hidden profit target from weighted average price
input int      HedgeLevel       = 10;          // Grid count before hedge trigger
input double   MaxSpreadPips    = 1;           // Max allowed spread for entry
input long     MagicNumber      = 20251022;    // Unique identifier
input int      SlippagePoints   = 20;          // Max slippage
input bool     AllowHedgeReset  = true;        // Enable hedge logic
input bool     AllowNewCycle    = true;        // Enable/disable new cycle after hedge
input string   CommentTag      = "GridMartingaleEA"; // Order comment prefix
```

Internal Variables & Structures

```
double anchor_price;           // Starting price of current cycle
int grid_index;                // Current grid offset relative to anchor
bool hedge_triggered;          // Hedge active flag
int cycle_id;                  // Identifier for current trading cycle

// Arrays to track open positions
struct TradeLeg {
    ulong   ticket;
    int     grid_level;
    double  lot;
    double  open_price;
    string  side;                // "BUY" or "SELL"
};
TradeLeg  buy_legs[];
TradeLeg  sell_legs[];

int lot_step_index_buy;
int lot_step_index_sell;
```

Core Functions

```
//-----
// EA Initialization
//-----
int OnInit() {
    InitGlobals();
    OpenInitialOrders();
    return(INIT_SUCCEEDED);
}

//-----
// Tick Handler
```

```

//-----
void OnTick() {
    if(!AllowNewCycle) return;
    double current_price = SymbolInfoDouble(_Symbol, SYMBOL_BID);
    int new_index = ComputeGridIndex(current_price);

    if(CheckSpreadTooHigh()) return;

    // Manage Buy/Sell sides separately
    ManageBuyCycle(new_index);
    ManageSellCycle(new_index);

    // Hedge logic
    if(CheckHedgeTrigger(new_index)) ExecuteHedgeReset();
}

//-----
// Grid Index Computation
//-----
int ComputeGridIndex(double price) {
    double pip_size = (SymbolInfoInteger(_Symbol, SYMBOL_DIGITS) == 3 ||
                      SymbolInfoInteger(_Symbol, SYMBOL_DIGITS) == 5) ? 0.0001 : 0.01;
    double grid_pips = GridSizePips * pip_size;
    return(int)MathRound((price - anchor_price) / grid_pips);
}

//-----
// Hidden Take Profit Logic
//-----
bool CheckBasketProfit(string side) {
    double totalProfit = 0;
    for(int i=0; i<ArraySize((side=="BUY")?buy_legs:sell_legs); i++)
        totalProfit += PositionGetDouble(POSITION_PROFIT);
    return (totalProfit >= ProfitCashTarget);
}

```

Trade Management Logic

- On startup: open one Buy and one Sell at market using the first lot size.
- Each grid move opens a new leg only if price crosses into a new grid level.
- If price moves up: the Buy side closes base TP and reopens; the Sell side adds a martingale leg.
- If price moves down: the Sell side closes base TP and reopens; the Buy side adds a martingale leg.
- Each basket closes once combined floating profit \geq cash or pip target.
- At the hedge trigger: open an opposite order equal to total net exposure, then reset all cycles.

Error Handling & Broker Safety

```

// Retry logic for trade execution
bool SafeOrderSend(MqlTradeRequest &request, MqlTradeResult &result) {
    for(int i=0; i<3; i++) {
        if(OrderSend(request, result)) return true;
        if(result.retcode == TRADE_RETCODE_REQUOTE || result.retcode == TRADE_RETCODE_PRICE_CHAN

```

```
        Sleep(500);  
    }  
    return false;  
}
```

Summary

This EA maintains a two-sided grid system where both Buy and Sell sides operate independently. The side moving with price takes small hidden profits, while the opposing side builds martingale legs until the basket reaches a target profit or a hedge trigger resets the cycle. The design includes full recovery logic, spread guards, and structured order management for stability and transparency in execution.