Renewable Asset Valuation

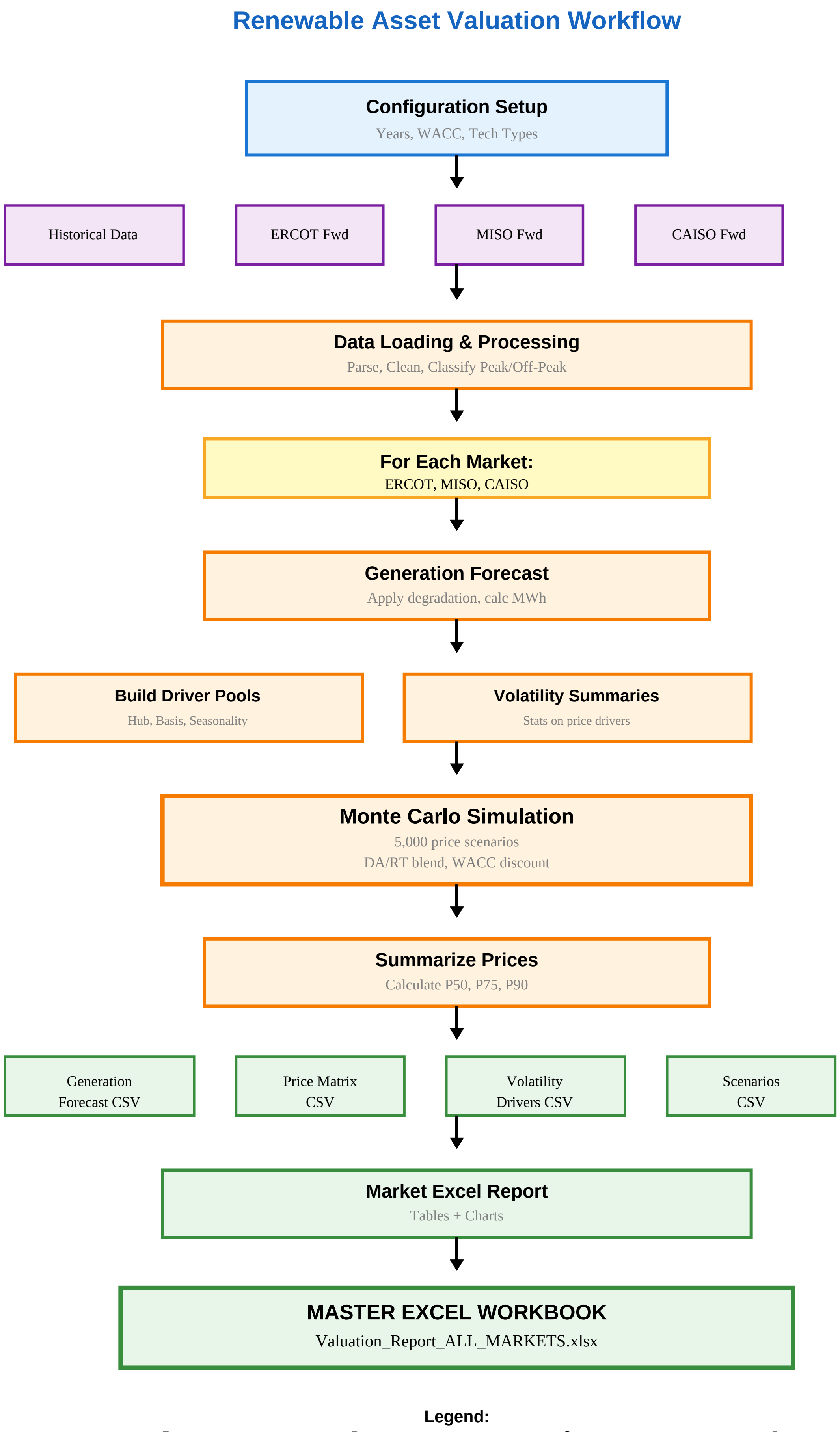
Workflow Schematic

Monte Carlo Simulation Framework

for Wind and Solar Asset Pricing

Script Type:	Python Energy Market Valuation
Markets:	ERCOT, MISO, CAISO
Technologies:	Wind & Solar
Analysis Period:	2026-2030
Simulation Method:	Monte Carlo (5,000 scenarios)
Discount Rate:	7% WACC
Price Weighting:	80% Day-Ahead / 20% Real-Time

Workflow Schematic Diagram



Detailed Workflow Breakdown

Phase 1: Initialization

The script begins by loading configuration parameters and validating input files. Key parameters include the analysis years (2026-2030), WACC discount rate (7%), technology types for each market (Wind for ERCOT/MISO, Solar for CAISO), and degradation rates (0.7% annually for wind, 0.5% for solar).

<b>File</b>	<b>Description</b>
HackathonDataset.xlsx	Historical hourly generation and price data
Forward price_ERCOT.xlsx	ERCOT forward price curves
Forward price_MISO.xlsx	MISO forward price curves
Forward price_CAISO.xlsx	CAISO forward price curves

Phase 2: Data Loading & Processing

load_sheet(): Parses each market's sheet from the historical workbook. The function intelligently detects various timestamp formats (datetime, date+HE, date+time) and handles different column naming conventions.

build_ts() + tidy_hourly(): Standardizes the data by creating consistent timestamp columns, cleaning price data (handling currency symbols and negative values), and classifying hours as Peak (P) or Off-Peak (OP) based on market-specific rules:

<b>Market</b>	<b>Peak Definition</b>
ERCOT	Mon-Fri, HE 7-22
MISO	Mon-Fri, HE 8-23
CAISO	Mon-Sat, HE 7-22

load_forwards(): Processes forward price curves by parsing time periods, extracting peak and off-peak prices, and cleaning numerical data. Handles multiple date formats including 'MMM-YY' notation.

Phase 3: Market-Level Analysis Loop

The script iterates through each market (ERCOT, MISO, CAISO) and performs identical analysis workflows. This modular approach ensures consistency while accommodating market-specific characteristics.

monthly_gen_forecast(): Projects future generation by:

- Computing historical capacity factors by month and peak period
- Applying technology-specific degradation curves
- Calculating expected generation (MW) and total energy (MWh)

build_driver_pools(): Analyzes historical price relationships to capture volatility:

- Hub price distributions (Day-Ahead and Real-Time)
- Node-Hub basis patterns (DA Busbar - DA Hub, RT Busbar - RT Hub)
- Seasonal variations by month and peak period

volatility_driver_summaries(): Computes statistical measures (mean, std, min, max, percentiles) for all price components to inform the simulation.

Phase 4: Monte Carlo Simulation

simulate_term(): The core valuation engine that runs 5,000 price scenarios using:

<b>Component</b>	<b>Description</b>
Forward Anchor	Hub forward prices provide the central tendency
Basis Sampling	Historical node-hub basis distributions add locational spread
DA/RT Blending	Weighted combination (80% DA / 20% RT) of price streams
Revenue Calculation	Generation forecast × Nodal price by period
NPV Discounting	7% WACC applied to future cash flows
Quantile Analysis	P50, P75, P90 risk-adjusted fair prices extracted

Phase 5: Valuation & Summarization

summarize_prices(): Aggregates simulation results by calculating percentile-based fair prices. The P50 (median) represents the expected value, while P75 and P90 provide risk-adjusted pricing for conservative valuation scenarios.

Phase 6: Output Generation

For each market, the script generates comprehensive outputs:

- **expected_generation_2026_2030.csv:** Monthly generation forecasts
- **fixed_price_matrix_P50_P75_P90.csv:** Risk-adjusted fair prices
- **volatility_driver_summaries.csv:** Statistical analysis of price drivers
- **scenario_unit_PV_distribution.csv:** Full distribution of 5,000 scenario outcomes

export_market_excel(): Creates an Excel workbook with multiple worksheets including formatted tables, embedded charts (column charts for fair prices, line charts for generation and forwards), and detailed component breakdowns.

Phase 7: Master Aggregation

The final step consolidates results from all three markets into a single master Excel workbook (**Valuation_Report_ALL_MARKETS.xlsx**) containing dedicated sheets for each market's fair prices, generation forecasts, volatility drivers, and scenario distributions. This provides stakeholders with a comprehensive view of all assets in one file.

Key Features & Capabilities

Robust Data Handling

- Flexible timestamp parsing across multiple formats
- Automatic column detection with fuzzy matching
- Handling of currency symbols, negative values, and missing data
- Market-specific peak/off-peak classification rules

Advanced Valuation Methodology

- Monte Carlo simulation with 5,000 scenarios for robust statistical analysis
- Technology-specific degradation curves (0.7% wind, 0.5% solar)
- Historical volatility-informed basis distributions
- Dual market blending (Day-Ahead 80% / Real-Time 20%)
- NPV calculation with configurable WACC discount rate

Professional Output Suite

- Individual market Excel reports with charts
- Consolidated master workbook across all markets
- CSV exports for downstream analysis
- Risk-adjusted pricing at multiple confidence levels (P50/P75/P90)

Primary Use Cases

<b>Use Case</b>	<b>Application</b>
Asset Valuation	Price renewable energy projects for acquisition or sale
PPA Negotiation	Establish fair market value baselines for power purchase agreements
Risk Management	Quantify price volatility and basis risk exposure
Portfolio Optimization	Compare relative value across multiple markets and technologies
Investment Analysis	Support due diligence with probabilistic revenue forecasts
Budget Planning	Generate expected revenue scenarios for financial planning

Technical Requirements

- **Python Libraries:** numpy, pandas, pathlib, re
- **Input Files:** Excel (.xlsx) historical data and forward price curves
- **Data Format:** Hourly timeseries with generation, DA/RT hub and busbar prices
- **Forward Prices:** Monthly peak/off-peak forward curves by market
- **Runtime:** Approximately 1-5 minutes per market depending on data volume

Generated from Hackathon.py script analysis