Core Components Overview

A. Backtester (Main Orchestrator)

- **Class: Backtester (File: backtester folder -> __init__.py)**
    - **Responsibilities**:
        1. **Manages agents**: Adds, removes, and clears agents.
        2. **Orchestrates data flow**: Fetches historical market data via DataProvider.
        3. **Runs simulations**: Allocates portfolio weights using agents' models.
        4. **Evaluates performance**: Applies benchmarks (e.g., Sharpe ratio, PnL, etc.) using evaluate_agents
        5. **Exports results**: Converts simulation results into structured formats like Excel.
    - **Key Methods**:
        - run: Orchestrates data fetching and agent weight allocation.
        - evaluate_agents: Evaluates agents' performance against benchmarks.
        - results_to_excel2: Exports backtesting results.

B. Agent

- **Class: Agent (File: agent folder -> init__.py)**
    - **Responsibilities**:
        1. **Interface to portfolio models**: Uses a WeightAllocationModel subclass (e.g., HRP) to predict portfolio weights.
        2. **Manages weight predictions**: Resamples and aligns weight predictions with market data.
    - **Key Attributes**:
        - model: Instance of a subclass of WeightAllocationModel.
        - weight_predictions: Stores weight allocations over time.
    - **Key Methods**:
        - weights_allocate: Uses the model to calculate portfolio weights. Returns a dataframe with the weights. Each row corresponds to each time we updated weights.
        - date_data_needed: Determines historical data requirements based on the model's configuration.

C. Weight Allocation Models

- **Base Class: WeightAllocationModel (File: model_base.py)**
    - **Responsibilities:**
        1. Provides the abstract interface for models (e.g., weights_allocate, date_data_needed).
        2. Defines shared behavior for all models.
- **Subclass Example: HRP (File: HRP_allocation.py)**
    - Implements the Hierarchical Risk Parity (HRP) strategy.
    - Determines historical data requirements (date_data_needed) and performs periodic weight allocation (weights_allocate) using the HRP algorithm**.**

D. Data Provider

- **Class: DataProvider (File: DataProvider.py)**
    - **Responsibilities:**
        1. Fetches historical market data for a given date range and ticker list using Yahoo Finance (yfinance).
        2. Cleans data (handles missing values) and calculates returns.
    - **Key Methods:**
        - **fetch:** Downloads raw data.
        - **clean:** Cleans the dataset.

E. Benchmarks

- **Base Class: Benchmark (File: evaluation_base.py)**
    - **Responsibilities:**
        1. Provides the interface for all benchmarks (e.g., calculate).
        2. Defines a frequency utility (groupby_freq for aggregating by frequency; handles correctly scalar and additive metrics).
- **Examples of Benchmarks (File: evaluation.py):**
    - **PNL**
    - **Sharpe**
    - **Volatility**

Backtester -> Agent: Manages agents and their weight allocation models.

Agent -> Model: Uses models to predict portfolio weights.

Backtester -> DataProvider: Fetches historical data for simulations.

Backtester -> Benchmarks: Evaluates performance using benchmark metrics.