You're most welcome! I'm happy to help you get this robust system up and running, and I'll be here to address any questions you have.

I have now provided the full rewritten code for all the modules, including placeholders for the modules whose content was not originally provided.

Here's a recap of what has been delivered:

**Core Modules:**

* data/conventions.yaml: The new, structured configuration file.
* src/config.py: Loads and provides access to the conventions.yaml data.
* src/data\_manager.py: Centralizes Bloomberg API interactions (pdblp) and handles loading/saving historical data (e.g., from .pkl files in data\_lake/).
* src/date\_utils.py: Contains common date and time manipulation functions.
* src/financial\_math\_utils.py: Houses general financial calculations and price conversion utilities, including the inferred fut\_payoff and get\_1y1y\_fwd with detailed comments for your review.
* src/quantlib\_utils.py: Centralizes QuantLib object initializations and helper functions.

**Curve Builders (src/curves/):**

* src/curves/base\_curve.py: Abstract base class for all curve builders.
* src/curves/ois\_curve\_builder.py: Builds OIS (Overnight Indexed Swap) curves.
* src/curves/swap\_curve\_builder.py: Builds standard interest rate swap (e.g., Libor/Euribor) curves.
* src/curves/inflation\_curve\_builder.py: Builds inflation zero-coupon swap curves.
* src/curves/bond\_curve\_builder.py: **Placeholder** for bond curve construction, as the original content was not provided.

**Instruments (src/instruments/):**

* src/instruments/base\_instrument.py: Abstract base class for all financial instruments.
* src/instruments/swap.py: Defines the SwapParameters (replacing the inferred swap\_class) and the Swap instrument itself.
* src/instruments/option.py: Defines the Option instrument class, serving as a base for specific option types.
* src/instruments/bond.py: Defines the Bond instrument class for general bond properties.

**Pricers (src/pricers/):**

* src/pricers/swap\_pricer.py: Consolidates logic for pricing swaps and calculating related metrics.
* src/pricers/inflation\_pricer.py: Contains logic for pricing inflation zero-coupon swaps.
* src/pricers/linker\_carry\_calculator.py: Contains the logic for calculating carry for inflation-linked bonds.

**Volatility Surface (src/volatility\_surface/):**

* src/volatility\_surface/base\_vol\_surface.py: Abstract base class for volatility surface builders.
* src/volatility\_surface/bond\_future\_vol\_surface.py: Builds volatility surfaces for bond futures options.
* src/volatility\_surface/stir\_vol\_surface.py: Builds volatility surfaces for STIR (Short Term Interest Rate) options.

**Strategy Analytics (src/strategy\_analytics/):**

* src/strategy\_analytics/option\_strategy.py: Consolidates logic for analyzing option strategies.

**Market Tables (src/market\_tables/):**

* src/market\_tables/bond\_tables.py: Generates the linker monitor table.
* src/market\_tables/inflation\_tables.py: Handles the generation of tables for inflation swap rates.
* src/market\_tables/swap\_tables.py: **Placeholder** for swap table generation, as the original content was not provided.

**Market Data Processors (src/market\_data\_processors/):**

* src/market\_data\_processors/economic\_data\_processor.py: Handles retrieval, transformation, and initial analysis of economic data.
* src/market\_data\_processors/wirp\_processor.py: Handles calculation and historical management of WIRP (World Interest Rate Probability) metrics.

**Plotting (src/plotting/):**

* src/plotting/base\_plotter.py: Abstract base class for plotting functionalities.
* src/plotting/matplotlib\_plots.py: Implements various Matplotlib-based plotting functions.
* src/plotting/bokeh\_plots.py: Implements various Bokeh-based interactive plotting functions.

**Main Entry Point:**

* main.py: A comprehensive example script demonstrating how to use the new modules and run typical workflows.

**Before you dive in:**

* **Bloomberg Connection:** Ensure your Bloomberg terminal is running and that the pdblp library can connect to it.
* **Data Lake:** Make sure you have the data/data\_lake/ directory set up in your project root, and populate it with any .pkl files (historical curve nodes, inflation fixings, WIRP history) and the eco\_master.xlsx file that your original code relied on. The main.py script has comments about which files are expected.
* **Dependencies:** Confirm all Python packages are installed (e.g., pandas, numpy, QuantLib, pdblp, PyYAML, scipy, scikit-learn, matplotlib, seaborn, bokeh).
* **Review Comments:** Pay special attention to the comments I've added, particularly those indicating areas where Bloomberg tickers or specific financial calculations (fut\_payoff, get\_1y1y\_fwd) were inferred or require your domain-specific verification.

This rewrite aims to make your infrastructure more efficient, clearer, and state-of-the-art. I'm ready for your questions as you start exploring the new codebase!

