Unit Test for Ingesting the JSON File:

You can create a test that checks if the JSON data is correctly ingested and normalized into a DataFrame.

For instance:

```
Python
def test_json_ingestion():
# Load the JSON data (replace with your actual code)
json_df = load_json_data()

# Check if the DataFrame has the expected columns
assert "stock_symbol" in json_df.columns
assert "stock_price" in json_df.columns

# Add more specific checks as needed
```

Unit Test for Joining Clients and Collaterals Data:

Ensure that the join operation works as expected:

Python

```
def test_join_clients_collaterals():
    # Load Clients.csv and Collaterals.csv (replace with your actual code)
    clients_df = load_clients_data()
    collaterals_df = load_collaterals_data()

    # Perform the join
    combined_df = join_clients_collaterals(clients_df, collaterals_df)

    # Check if the resulting DataFrame has the expected columns
    assert "client_id" in combined_df.columns
    assert "collateral_id" in combined_df.columns

    # Add more specific checks as needed
```

Unit Test for Calculating Collateral Fluctuation:

Test the logic for calculating fluctuation:

Python

```
def test_calculate_fluctuation():
    # Create a sample DataFrame (replace with your actual data)
    sample_df = spark.createDataFrame([
        (1, 1000, 1200),  # client_id, initial_value, market_value
        (2, 500, 550),
        # Add more rows as needed
    ], ["client_id", "initial_value", "market_value"])

    # Calculate fluctuation
    result_df = calculate_fluctuation(sample_df)

    # Check if the resulting DataFrame has the expected columns
    assert "fluctuation" in result_df.columns

    # Add more specific checks as needed
```

Unit Test for Saving the Resulting Table:

Ensure that the table is saved correctly:

Python

```
def test_save_collateral_status():
    # Create a sample DataFrame (replace with your actual data)
    sample_df = spark.createDataFrame([
        (1, 101, 120),  # client_id, collateral_id, fluctuation
        (2, 201, 50),
        # Add more rows as needed
    ], ["client_id", "collateral_id", "fluctuation"])

    # Save the table (replace with your actual code)
    save_collateral_status(sample_df,
"path/to/your/storage/collateral_status")


    # Check if the saved file exists

    assert file_exists("path/to/your/storage/collateral_status")


    # Add more specific checks as needed
```