Version: 1.0.0 **Target:** 0.131.0-Beta

Implementation Plan - Phase 1.5: Visualization Standardization (Plotly Cluster)

1. Overview

This plan executes **Phase 1.5** of the visualization update roadmap. It establishes a centralized title generation utility (`_report_utils.py`) that enforces the "Three-Line Title" and "Smart Scoping" standards defined in `CLAReportsStyleGuide v1.3.0`. It then systematically refactors all existing Plotly-based reports to use this utility and the `PlotlyStyleManager` footer, ensuring consistent branding and metadata across the visual stack. The WRTC Animation migration is deferred to Phase 5.

2. Proposed Changes

A. Core Utilities: `contest_tools/reports/_report_utils.py`

- **Add:** `get_standard_title_lines(report_name, logs, band_filter, mode_filter, modes_present_set=None)`
- **Logic:**
  1. Extracts Year, Contest, EventID, and Callsigns from `logs`.
  2. Constructs Line 2 (Context).
  3. Constructs Line 3 (Scope) handling "All Bands" and "Smart Scoping" for modes (suppressing "(All Modes)" if `modes_present_set` has length 1).
  4. Returns list of strings `[Line1, Line2, Line3]`.

B. Standard Plotly Reports (Refactoring)

All listed reports will be updated to:

1. Calculate `modes_present` from their data (where applicable).
2. Call `get_standard_title_lines` to generate the header.
3. Call `get_cty_metadata` to generate the footer.
4. Pass `footer_text` to `PlotlyStyleManager.get_standard_layout`.

Target Files:

1. `contest_tools/reports/plot_qso_rate.py` (Refactor to use utility)
2. `contest_tools/reports/plot_point_rate.py`
3. `contest_tools/reports/plot_cumulative_difference.py`
4. `contest_tools/reports/plot_comparative_band_activity.py`
5. `contest_tools/reports/plot_comparative_run_sp.py`
6. `contest_tools/reports/chart_point_contribution.py`
7. `contest_tools/reports/chart_point_contribution_single.py`

8. `contest_tools/reports/chart_qso_breakdown.py`
9. `contest_tools/reports/plot_interactive_animation.py`

3. Surgical Change Verification

File: `contest_tools/reports/_report_utils.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -95,4 +95,39 @@ """ if not logs: return "CTY File: Unknown"

```
path = getattr(logs[0], 'cty_dat_path', '')
if not path or not os.path.exists(path):
    return "CTY File: Unknown"

# Try to extract version from filename (cty_wt_mod_3504.dat)
version_match = re.search(r'(\d{4})', os.path.basename(path))
version_str = f"CTY-{version_match.group(1)}" if version_match else "CTY-Unknown"

# Try to extract date
date_obj = CtyLookup.extract_version_date(path)
date_str = date_obj.strftime('%Y-%m-%d') if date_obj else "Unknown Date"

return f"CTY File: {date_str} {version_str}"
```

-

+def get_standard_title_lines(report_name: str, logs: list, band_filter: str = None, mode_filter: str = None, modes_present_set: set = None) -> list:

- """
- Generates the standard 3-Line Title components (Name, Context, Scope).
- Applies Smart Scoping logic for modes.
- """
- if not logs: return [report_name, "", ""]
-
-

--- Line 2: Context ---

- metadata = logs[0].get_metadata()
- df = get_valid_dataframe(logs[0])
- year = df['Date'].dropna().iloc[0].split('-')[0] if not df.empty else "----"
- contest_name = metadata.get('ContestName', '')

- event_id = metadata.get('EventID', '')
- all_calls = sorted([l.get_metadata().get('MyCall', 'Unknown') for l in logs])
- callsign_str = ", ".join(all_calls)
-
- line2 = f"{year} {event_id} {contest_name} - {callsign_str}".strip().replace(" ", " ")
-
-

--- Line 3: Scope ---

-

Band Logic

- is_single_band = len(logs[0].contest_definition.valid_bands) == 1
- if band_filter == 'All' or band_filter is None:
- `band_text = logs[0].contest_definition.valid_bands[0].replace('M', ' Meters') if is_`
- else:
- `band_text = band_filter.replace('M', ' Meters')`
-
-

Mode Logic (Smart Scoping)

- if mode_filter:
- `mode_text = f" ({mode_filter})"`
- else:
- `# Only show (All Modes) if multiple modes actually exist in the data`
- `if modes_present_set and len(modes_present_set) > 1:`
- `mode_text = " (All Modes)"`
- `else:`
- `mode_text = ""`

-
- line3 = f"{band_text}{mode_text}"
-
- return [report_name, line2, line3] --- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_qso_rate.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -23,7 +23,7 @@ from ..contest_log import ContestLog from .report_interface import ContestReport -from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part, get_cty_metadata +from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines from ..data_aggregators.time_series import TimeSeriesAggregator from ..styles.plotly_style_manager import PlotlyStyleManager

@@ -190,26 +190,14 @@)

```
    # 4. Styling & Layout
```

- ```
    metadata = self.logs[0].get_metadata()
  ```
- ```
    year = get_valid_dataframe(self.logs[0])['Date'].dropna().iloc[0].split('-')[0] if n
  ```
- ```
    contest_name = metadata.get('ContestName', '')
  ```
- ```
    event_id = metadata.get('EventID', '')
  ```
- ```
    # Smart Scoping: Collect unique modes from all dfs
  ```
- ```
    modes_present = set()
  ```
- ```
    for df in dfs:
  ```
- ```
        if 'Mode' in df.columns:
  ```
- ```
            modes_present.update(df['Mode'].dropna().unique())
  ```
- ```
    is_single_band = len(self.logs[0].contest_definition.valid_bands) == 1
  ```
-
- ```
    # Determine Band Text
  ```
- ```
    if band_filter == 'All':
  ```
- ```
        if is_single_band:
  ```
- ```
            # ARRL 10M Case: "All" really means the one valid band
  ```
- ```
            band_text = self.logs[0].contest_definition.valid_bands[0].replace('M', ' Me
  ```
- ```
        else:
  ```

```
            band_text = "All Bands"
    else:
        band_text = band_filter.replace('M', ' Meters')

    # Determine Mode Text
    if mode_filter:
        mode_text = f" ({mode_filter})"
    else:
        # Smart Scoping: Only show (All Modes) if there actually ARE multiple modes
        modes_present = set()
        for df in dfs:
            if 'Mode' in df.columns:
                modes_present.update(df['Mode'].dropna().unique())

        mode_text = " (All Modes)" if len(modes_present) > 1 else ""

    # Line 3 Scope construction
    scope_line = f"{band_text}{mode_text}"

    callsign_str = ", ".join(all_calls)

    title_line1 = f"{self.report_name}"
    title_line2 = f"{year} {event_id} {contest_name} - {callsign_str}".strip().replace("
    final_title = f"{title_line1}<br><sub>{title_line2}<br>{scope_line}</sub>"
    title_lines = get_standard_title_lines(self.report_name, self.logs, band_filter, mod
    final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"

    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"
```

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_point_rate.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -34,7 +34,7 @@ from ..contest_log import ContestLog from .report_interface import ContestReport -from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part +from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines from ..data_aggregators.time_series import TimeSeriesAggregator from ..styles.plotly_style_manager import PlotlyStyleManager

@@ -160,19 +160,14 @@)

```
    # --- Layout & Styling ---
-   metadata = self.logs[0].get_metadata()
-   year = get_valid_dataframe(self.logs[0])['Date'].dropna().iloc[0].split('-')[0] if m
-   contest_name = metadata.get('ContestName', '')
-   event_id = metadata.get('EventID', '')
-
-   is_single_band = len(self.logs[0].contest_definition.valid_bands) == 1
-   band_text = self.logs[0].contest_definition.valid_bands[0].replace('M', ' Meters') i
-   mode_text = f" ({mode_filter})" if mode_filter else ""
-   callsign_str = ", ".join(all_calls)
-   # Smart Scoping: Collect unique modes from all dfs
-   modes_present = set()
-   for df in dfs:
-       if 'Mode' in df.columns:
-           modes_present.update(df['Mode'].dropna().unique())
-   title_line1 = f"{self.report_name}{mode_text}"
-   title_line2 = f"{year} {event_id} {contest_name} - {callsign_str}".strip().replace("
-   final_title = f"{title_line1}<br>{title_line2}" # Use <br> for Plotly title
+   title_lines = get_standard_title_lines(self.report_name, self.logs, band_filter, mod
+   final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"
+
+   footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"
```

```
        # Apply standard layout
```

- `    layout_config = PlotlyStyleManager.get_standard_layout(final_title)`
- `    layout_config = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`
  `fig.update_layout(layout_config)`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_cumulative_difference.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -36,7 +36,7 @@ from ..contest_log import ContestLog from .report_interface import ContestReport -from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part +from .report_utils import create_output_directory, get_valid_dataframe, save_debug_data, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines from ..data_aggregators.time_series import TimeSeriesAggregator from ..styles.plotly_style_manager import PlotlyStyleManager

@@ -131,18 +131,16 @@ fig.add_hline(y=0, line_width=1, line_color="gray")

```
    # --- Metadata & Titles ---
```

- `    metadata = log1.get_metadata()`
- `    year = get_valid_dataframe(log1)['Date'].dropna().iloc[0].split('-')[0] if not get_v`
- `    contest_name = metadata.get('ContestName', '')`
- `    event_id = metadata.get('EventID', '')`
- `    # Smart Scoping: Collect unique modes (using TimeSeriesAggregator data is tricky, as`
- `    # Since this uses aggregated data, we check if mode_filter is set. If not, we assume`
- `    modes_present = set()`
- `    for log in self.logs:`
- `        df = get_valid_dataframe(log)`
- `        if 'Mode' in df.columns:`
- `            modes_present.update(df['Mode'].dropna().unique())`
- `    is_single_band = len(log1.contest_definition.valid_bands) == 1`
- `    band_text = log1.contest_definition.valid_bands[0].replace('M', ' Meters') if is_sin`
- `    mode_text = f" ({mode_filter})" if mode_filter else ""`
- `    callsign_str = f"{call1} vs. {call2}"`
-
- `    title_line1 = f"{self.report_name} - {metric_name}{mode_text}"`

- `    title_line2 = f"{year} {event_id} {contest_name} - {callsign_str}".strip().replace("`
- `    final_title = f"{title_line1}<br>{title_line2}" # Use <br> for Plotly title break`
- `    title_lines = get_standard_title_lines(f"{self.report_name} ({metric_name})", self.l`
- `    final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"`

  `    # --- Layout Standardization ---`
- `    layout = PlotlyStyleManager.get_standard_layout(final_title)`
- `    layout = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`
  `fig.update_layout(layout)`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_comparative_band_activity.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -37,7 +37,7 @@ from ..contest_log import ContestLog from .report_interface import ContestReport -from ._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part +from ._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines from ..data_aggregators.matrix_stats import MatrixAggregator from ..styles.plotly_style_manager import PlotlyStyleManager

@@ -133,18 +133,13 @@

`    # --- 4. Layout & Styling ---`
`    # Prepare Titles`

- `    metadata = log1.get_metadata()`
- `    df_first_log = get_valid_dataframe(log1)`
- `    year = df_first_log['Date'].dropna().iloc[0].split('-')[0] if not df_first_log.empty`
- `    contest_name = metadata.get('ContestName', '')`
- `    event_id = metadata.get('EventID', '')`
- `    modes_present = set()`
- `    for log in [log1, log2]:`
- `        df = get_valid_dataframe(log)`
- `        if 'Mode' in df.columns:`
- `            modes_present.update(df['Mode'].dropna().unique())`

- `mode_title_str = f" ({mode_filter})" if mode_filter else ""`
- `title_line1 = f"{self.report_name}{mode_title_str}"`
- `context_str = f"{year} {event_id} {contest_name}".strip().replace("  ", " ")`
- `calls_str = f"{call1} (Up) vs. {call2} (Down)"`
- `title_line2 = f"{context_str} - {calls_str}"`
- `full_title = f"{title_line1}<br>{title_line2}"`
- `title_lines = get_standard_title_lines(self.report_name, [log1, log2], "All Bands",`
- `final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata([log1, log2])}"`
- `layout_config = PlotlyStyleManager.get_standard_layout(full_title)`
- `layout_config = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`
  `fig.update_layout(layout_config)`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_comparative_run_sp.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -35,7 +35,7 @@ from ..contest_log import ContestLog from .report_interface import ContestReport -from ._report_utils import get_valid_dataframe, create_output_directory, save_debug_data +from ._report_utils import get_valid_dataframe, create_output_directory, save_debug_data, get_cty_metadata, get_standard_title_lines from ..data_aggregators.matrix_stats import MatrixAggregator from ..styles.plotly_style_manager import PlotlyStyleManager

@@ -155,18 +155,16 @@)

```
    # --- Formatting ---
```

- `year = get_valid_dataframe(self.logs[0])['Date'].dropna().iloc[0].split('-')[0]`
- `contest_name = log1_meta.get('ContestName', '')`
- `event_id = log1_meta.get('EventID', '')`
- `modes_present = set()`
- `for log in self.logs:`
- `    df = get_valid_dataframe(log)`
- `    if 'Mode' in df.columns:`
- `        modes_present.update(df['Mode'].dropna().unique())`

- `    mode_title_str = f" ({mode_filter})" if mode_filter else ""`
- `    callsign_str = f"{call1} vs. {call2}"`
-
- `    title_line1 = f"{self.report_name}{mode_title_str}{page_title_suffix}"`
- `    title_line2 = f"{year} {event_id} {contest_name} - {callsign_str}".strip().replace("`
- `    title_lines = get_standard_title_lines(f"{self.report_name}{page_title_suffix}", sel`
- `    final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"`

`    # Apply standard layout`

- `    layout_cfg = PlotlyStyleManager.get_standard_layout(f"{title_line1}<br><sub>{title_l`
- `    layout_cfg = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`

`    # Customize for this specific report`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/chart_point_contribution.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -32,7 +32,7 @@ from contest_tools.contest_log import ContestLog from contest_tools.data_aggregators.categorical_stats import CategoricalAggregator from contest_tools.styles.plotly_style_manager import PlotlyStyleManager -from contest_tools.reports._report_utils import create_output_directory +from contest_tools.reports._report_utils import create_output_directory, get_cty_metadata, get_standard_title_lines, get_valid_dataframe

class Report(ContestReport): """ @@ -102,17 +102,15 @@ fig.add_trace(trace, row=1, col=i+1)

`    # Standard Title Construction`

- `    first_log = self.logs[0]`
- `    year = first_log.get_processed_data()['Date'].iloc[0].split('-')[0]`
- `    contest_name = first_log.get_metadata().get('ContestName', 'Unknown')`
- `    all_calls = sorted([l.get_metadata().get('MyCall') for l in self.logs])`
- `    modes_present = set()`
- `    for log in self.logs:`
- `        df = get_valid_dataframe(log)`

- `        if 'Mode' in df.columns:`
- `            modes_present.update(df['Mode'].dropna().unique())`
- `    chart_title = f"Comparative Point Contribution: {year} {contest_name} - {', '.join(a`
- `    title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non`
- `    final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"`

`    # Apply standard layout`

- `    layout_config = PlotlyStyleManager.get_standard_layout(chart_title)`
- `    layout_config = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`
  `fig.update_layout(**layout_config)`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/chart_point_contribution_single.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -31,7 +31,7 @@ from contest_tools.contest_log import ContestLog from contest_tools.data_aggregators.categorical_stats import CategoricalAggregator from contest_tools.styles.plotly_style_manager import PlotlyStyleManager -from contest_tools.reports._report_utils import get_valid_dataframe, create_output_directory +from contest_tools.reports._report_utils import get_valid_dataframe, create_output_directory, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -132,9 +132,13 @@ fig.add_trace(trace, row=row, col=col)

`    # 5. Styling and Output`

- `    title_text = f"Point Contribution Breakdown: {callsign}"`
- `    layout_config = PlotlyStyleManager.get_standard_layout(title_text)`
- `    modes_present = set(df['Mode'].dropna().unique())`
- `    title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non`
- `    final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"`
-

- `layout_config = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`

  `# Merge dynamic size into standard layout`
  `layout_config.update({`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/chart_qso_breakdown.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -31,7 +31,7 @@ from contest_tools.contest_log import ContestLog from contest_tools.data_aggregators.categorical_stats import CategoricalAggregator from contest_tools.styles.plotly_style_manager import PlotlyStyleManager -from contest_tools.reports._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part +from contest_tools.reports._report_utils import get_valid_dataframe, create_output_directory, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines

class Report(ContestReport): """ @@ -134,16 +134,13 @@)

`# Standard Layout Application`

- `metadata = self.log1.get_metadata()`
- `year = df1['Date'].dropna().iloc[0].split('-')[0] if not df1.empty else "----"`
- `contest_name = metadata.get('ContestName', '')`
- `event_id = metadata.get('EventID', '')`
- `call1 = self.log1.get_metadata().get('MyCall')`
- `call2 = self.log2.get_metadata().get('MyCall')`
- `modes_present = set(df1['Mode'].dropna().unique()) | set(df2['Mode'].dropna().unique`
- `title_line1 = self.report_name`
- `title_line2 = f"{year} {event_id} {contest_name} - {call1} vs. {call2}".strip().repl`
- `final_title = f"{title_line1}<br>{title_line2}"`
- `title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non`
- `final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"`
-
- `footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"`
- `layout_config = PlotlyStyleManager.get_standard_layout(final_title)`
- `layout_config = PlotlyStyleManager.get_standard_layout(final_title, footer_text)`
  `fig.update_layout(layout_config)`

--- END DIFF --- **CODE_BLOCK**

File: `contest_tools/reports/plot_interactive_animation.py`

___CODE_BLOCK___diff --- BEGIN DIFF --- @@ -17,7 +17,7 @@ from typing import List, Dict, Any, Tuple from pathlib import Path

from .report_interface import ContestReport -from ._report_utils import create_output_directory, _sanitize_filename_part +from ._report_utils import create_output_directory, _sanitize_filename_part, get_cty_metadata, get_standard_title_lines, get_valid_dataframe from ..data_aggregators.time_series import TimeSeriesAggregator from ..data_aggregators.matrix_stats import MatrixAggregator from ..styles.plotly_style_manager import PlotlyStyleManager @@ -211,12 +211,13 @@ fig.update_xaxes(title_text="Band", row=2, col=2)

```
    # Construct Standard Title
-   log1_meta = self.logs[0].get_metadata()
-   year = kwargs.get('year', '2024') # Fallback
-   if not kwargs.get('year'):
-        year = "2024"
-
-   contest = log1_meta.get('ContestName', 'Contest')
-   title_line1 = self.report_name
-   title_line2 = f"{year} {contest} - {', '.join(callsigns)}"
-   modes_present = set()
-   for log in self.logs:
-       df = get_valid_dataframe(log)
-       if 'Mode' in df.columns:
-           modes_present.update(df['Mode'].dropna().unique())
-
-   title_lines = get_standard_title_lines(self.report_name, self.logs, "All Bands", Non
-
-   # For animation, we can add help text to the title
-   final_title = f"{title_lines[0]}<br><sub>{title_lines[1]}<br>{title_lines[2]}</sub>"

    # Animation Settings
```

@@ -257,8 +258,8 @@ }])

```
-   # Update Title with Standard Formatting
```

```
    fig.update_layout(title_text=f"{title_line1}<br><sub>{title_line2}</sub>")
    # Update Title with Standard Formatting and Footer
    footer_text = f"Contest Log Analytics by KD4D\n{get_cty_metadata(self.logs)}"

    # Merge with existing layout properties (since animation has complex layout)
    # We inject footer as annotation manually because we aren't using get_standard_layou
    fig.update_layout(title_text=final_title)
    fig.add_annotation(
        x=0.5, y=-0.25, # Push footer way down below controls
        xref="paper", yref="paper",
        text=footer_text.replace('\n', '<br>'),
        showarrow=False,
        font=dict(size=12, color="#7f7f7f"),
        align="center",
        valign="top"
    )

    # 6. Save
    sanitized_calls = "_".join([_sanitize_filename_part(c) for c in callsigns])
```

--- END DIFF --- **CODE_BLOCK**

4. Pre-Flight Check

- **Architecture Compliance:** All modified reports now delegate title and footer logic to shared utilities (`_report_utils.py` and `PlotlyStyleManager`), enforcing consistent standards.
- **Smart Scoping:** The `get_standard_title_lines` utility implements the logic to suppress "(All Modes)" for single-mode datasets, as verified in Phase 1.1.
- **Footer Metadata:** All visual reports now include the Branding and CTY version footer.
- **Animation Layout:** `plot_interactive_animation.py` needed custom handling for the footer to avoid overlapping the slider controls; it uses `y=-0.25` instead of the standard `-0.15`.

Next Action: I will issue the standardized prompt for **Confirmed**.