Below is one example of a software specification that defines a set of Python API functions for executing a broad range of visualization‐and‐reporting instructions in semiconductor chip test data analytics. In the design below, similar visualization tasks are grouped into common functions so that a single API can serve multiple instructions. Each function specification lists a “Title,” the chosen function name, a unique “ID” (or IDs when shared across instructions), a brief “Purpose” description, a sample Python “Signature” (with a header and a docstring), and the list of instructions (by Workflow ID and Instruction ID) that the function is intended to support.

**Function 1: Visualize Yield Improvement and Classification**

**Function Name**: visualize\_yield\_improvement\_and\_classification  
**ID**: 1\_7  
**Purpose**:  
Generate visualizations (e.g., bar charts or line plots) to display yield recovery after re‐probing by showing both yield improvement and changes in test bin classifications.  
**Signature**:

def visualize\_yield\_improvement\_and\_classification(data, chart\_type="bar", \*\*kwargs):

"""

Generate visualizations to display yield improvement and changes in test bin classifications after re-probing.

Parameters:

data (DataFrame): The input dataset with yield and classification info.

chart\_type (str): Type of chart to generate (e.g., 'bar', 'line').

\*\*kwargs: Additional options such as labels, title, etc.

"""

pass

**Used For**:  
– Workflow 1, Instruction 7

**Function 2: Visualize Probe Configuration Comparison**

**Function Name**: visualize\_probe\_configuration\_comparison  
**ID**: 4\_6, 17\_7  
**Purpose**:  
Compare yield distributions across different probe configurations by generating comparative charts or graphs.  
**Signature**:

def visualize\_probe\_configuration\_comparison(data, chart\_type="comparative", \*\*kwargs):

"""

Generate comparative visualizations to depict yield distributions and differences across probe configurations.

Parameters:

data (DataFrame): Data containing yield and probe configuration details.

chart\_type (str): The style of comparison chart (e.g., 'side-by-side', 'box').

\*\*kwargs: Additional visualization parameters.

"""

pass

**Used For**:  
– Workflow 4, Instruction 6  
– Workflow 17, Instruction 7

**Function 3: Visualize Test House Performance**

**Function Name**: visualize\_test\_house\_performance  
**ID**: 5\_6, 32\_5  
**Purpose**:  
Display yield performance for each test house, facilitating side‐by‐side comparisons through charts or graphs.  
**Signature**:

def visualize\_test\_house\_performance(data, chart\_type="bar", \*\*kwargs):

"""

Generate visualizations to display and compare yield performance across different test houses, including target thresholds.

Parameters:

data (DataFrame): Data with yield metrics per test house.

chart\_type (str): Type of chart to use (e.g., 'bar', 'line').

\*\*kwargs: Additional parameters for customization.

"""

pass

**Used For**:  
– Workflow 5, Instruction 6  
– Workflow 32, Instruction 5

**Function 4: Visualize Time-Series Trends**

**Function Name**: visualize\_time\_series\_trends  
**ID**: 6\_6, 28\_6, 61\_6, 34\_9, 69\_7, 70\_6  
**Purpose**:  
Plot time‐series charts to observe trends in metrics such as voltage measurement aggregates, lot-level yields, performance metrics over specified periods, and yield trends with annotated events.  
**Signature**:

def visualize\_time\_series\_trends(data, x\_axis, y\_axis, title, \*\*kwargs):

"""

Plot time-series charts to observe trends over time in various semiconductor test metrics.

Parameters:

data (DataFrame): The dataset containing time-series data.

x\_axis (str): The column name for the time axis.

y\_axis (str): The column name for the metric to plot.

title (str): Title of the chart.

\*\*kwargs: Additional styling or annotation parameters.

"""

pass

**Used For**:  
– Workflow 6, Instruction 6  
– Workflow 28, Instruction 6  
– Workflow 61, Instruction 6  
– Workflow 34, Instruction 9  
– Workflow 69, Instruction 7  
– Workflow 70, Instruction 6

**Function 5: Visualize Failure Correlations**

**Function Name**: visualize\_failure\_correlations  
**ID**: 7\_8, 21\_7, 29\_7, 57\_9, 75\_7, 31\_8, 49\_7  
**Purpose**:  
Generate scatter plots (with optional trendlines) and other comparative charts to illustrate correlations between yield performance and various failure metrics (logical, parametric, or measurement variance).  
**Signature**:

def visualize\_failure\_correlations(data, x\_axis, y\_axis, chart\_type="scatter", \*\*kwargs):

"""

Generate visualizations to illustrate correlations between yield and failure metrics using scatter plots and trend lines.

Parameters:

data (DataFrame): The dataset containing yield and failure information.

x\_axis (str): The field representing failure metrics.

y\_axis (str): The field representing yield performance.

chart\_type (str): Type of plot to generate (default is 'scatter').

\*\*kwargs: Other parameters for visual customization.

"""

pass

**Used For**:  
– Workflow 7, Instruction 8  
– Workflow 21, Instruction 7  
– Workflow 29, Instruction 7  
– Workflow 57, Instruction 9  
– Workflow 75, Instruction 7  
– Workflow 31, Instruction 8  
– Workflow 49, Instruction 7

**Function 6: Visualize Wafer Map and Anomalies**

**Function Name**: visualize\_wafer\_map\_and\_anomalies  
**ID**: 13\_6, 44\_7, 37\_8, 73\_6, 59\_8  
**Purpose**:  
Generate visualizations of wafer maps to highlight detected patterns, anomaly areas, failure concentration regions, and repeating patterns.  
**Signature**:

def visualize\_wafer\_map\_and\_anomalies(wafer\_map\_data, highlight\_fields, chart\_type="heatmap", \*\*kwargs):

"""

Generate wafer map visualizations highlighting detected patterns, anomaly areas, and failure concentrations.

Parameters:

wafer\_map\_data (DataFrame): Data containing wafer map coordinates and failure info.

highlight\_fields (list): List of features to highlight (e.g., anomaly areas).

chart\_type (str): Type of visualization to generate (e.g., 'heatmap', 'annotated\_map').

\*\*kwargs: Additional chart options.

"""

pass

**Used For**:  
– Workflow 13, Instruction 6  
– Workflow 44, Instruction 7  
– Workflow 37, Instruction 8  
– Workflow 73, Instruction 6  
– Workflow 59, Instruction 8

**Function 7: Visualize Measurement Distributions**

**Function Name**: visualize\_measurement\_distributions  
**ID**: 23\_6, 24\_8, 30\_8, 66\_5  
**Purpose**:  
Generate histograms, box plots, or scatter plots to assess differences in measurement distributions (e.g., E-test measurements, test variance) and compare groups.  
**Signature**:

def visualize\_measurement\_distributions(data, measurement\_field, group\_field=None, chart\_type="histogram", \*\*kwargs):

"""

Generate visualizations to assess differences in measurement distributions across groups.

Parameters:

data (DataFrame): Input dataset containing measurement data.

measurement\_field (str): The field name for measurement values.

group\_field (str, optional): The field name to group by for comparison.

chart\_type (str): The type of chart to generate (e.g., 'histogram', 'box').

\*\*kwargs: Additional parameters for customization.

"""

pass

**Used For**:  
– Workflow 23, Instruction 6  
– Workflow 24, Instruction 8  
– Workflow 30, Instruction 8  
– Workflow 66, Instruction 5

**Function 8: Visualize Control Chart**

**Function Name**: visualize\_control\_chart  
**ID**: 40\_7  
**Purpose**:  
Create a control chart displaying historical yield data with defined control limits and marking the current lot’s yield.  
**Signature**:

def visualize\_control\_chart(data, yield\_field, control\_limits, current\_lot, \*\*kwargs):

"""

Create a control chart to display historical yield data, control limits, and highlight the current lot's yield.

Parameters:

data (DataFrame): Historical yield data.

yield\_field (str): Column name for yield values.

control\_limits (tuple): Lower and upper control limits.

current\_lot (float): Yield value of the current lot.

\*\*kwargs: Additional chart customization options.

"""

pass

**Used For**:  
– Workflow 40, Instruction 7

**Function 9: Generate Summary Report**

**Function Name**: generate\_summary\_report  
**ID**: 18\_8, 22\_7, 41\_9, 77\_6  
**Purpose**:  
Compile a comprehensive report that integrates visualizations (charts, tables) with textual summaries to present analysis findings and recommendations.  
**Signature**:

def generate\_summary\_report(report\_data, charts, report\_title, \*\*kwargs):

"""

Compile and generate a comprehensive summary report that includes charts and visualizations.

Parameters:

report\_data (DataFrame): Data used for report generation.

charts (list): List of chart objects or visualization outputs.

report\_title (str): The title of the report.

\*\*kwargs: Additional report formatting options.

"""

pass

**Used For**:  
– Workflow 18, Instruction 8  
– Workflow 22, Instruction 7  
– Workflow 41, Instruction 9  
– Workflow 77, Instruction 6

**Function 10: Visualize Statistical Analysis**

**Function Name**: visualize\_statistical\_analysis  
**ID**: 78\_8  
**Purpose**:  
Generate visualizations (e.g., ROC curves, scatter plots, bar charts) to illustrate statistical significance and correlation between test metrics and yield outcomes.  
**Signature**:

def visualize\_statistical\_analysis(data, analysis\_field, chart\_types, \*\*kwargs):

"""

Generate visualizations to illustrate statistical correlations and significance, including ROC curves and scatter plots.

Parameters:

data (DataFrame): The dataset containing analysis metrics.

analysis\_field (str): The field for which statistical analysis is performed.

chart\_types (list): List of chart types to generate.

\*\*kwargs: Additional parameters for statistical annotations.

"""

pass

**Used For**:  
– Workflow 78, Instruction 8

**Function 11: Visualize Report Table**

**Function Name**: visualize\_report\_table  
**ID**: 9\_7, 55\_6, 65\_6  
**Purpose**:  
Generate a table or chart-based visualization for presenting summary data (such as weekly wafer yields or test bin counts) in an easy-to-read format.  
**Signature**:

def visualize\_report\_table(data, table\_type="table", title="", \*\*kwargs):

"""

Generate a table or chart visualization for reporting purposes.

Parameters:

data (DataFrame): Data to be formatted into a table or chart.

table\_type (str): Type of report visualization (e.g., 'table', 'chart').

title (str): Title for the visualization.

\*\*kwargs: Additional formatting options.

"""

pass

**Used For**:  
– Workflow 9, Instruction 7  
– Workflow 55, Instruction 6  
– Workflow 65, Instruction 6

**Function 12: Visualize Logical Failure Distribution**

**Function Name**: visualize\_logical\_failure\_distribution  
**ID**: 28\_8  
**Purpose**:  
Visualize the distribution of logical test failures across soft bins and compile a report of the findings.  
**Signature**:

def visualize\_logical\_failure\_distribution(data, logical\_test\_field, soft\_bin\_field, \*\*kwargs):

"""

Generate charts/graphs to visualize the distribution of logical test failures across soft bins.

Parameters:

data (DataFrame): The dataset containing logical test failure data.

logical\_test\_field (str): Column for logical test outcomes.

soft\_bin\_field (str): Column for soft bin counts.

\*\*kwargs: Additional visualization options.

"""

pass

**Used For**:  
– Workflow 28, Instruction 8

**Function 13: Visualize Center Fail Correlation**

**Function Name**: visualize\_center\_fail\_correlation  
**ID**: 26\_5  
**Purpose**:  
Illustrate the correlation between center-fail patterns and specific foundries or manufacturing batches using bar charts or heatmaps.  
**Signature**:

def visualize\_center\_fail\_correlation(data, pattern\_field, foundry\_field, batch\_field, \*\*kwargs):

"""

Generate visualizations to illustrate the correlation between center-fail patterns and specific foundries or batches.

Parameters:

data (DataFrame): Data containing failure pattern and manufacturing info.

pattern\_field (str): Column for center-fail patterns.

foundry\_field (str): Column for foundry identifiers.

batch\_field (str): Column for manufacturing batch information.

\*\*kwargs: Additional chart options.

"""

pass

**Used For**:  
– Workflow 26, Instruction 5

**Function 14: Visualize Test Failure Comparison**

**Function Name**: visualize\_test\_failure\_comparison  
**ID**: 9\_6  
**Purpose**:  
Create comparative visualizations (e.g., bar charts or tables) that display the differences in failure counts or rates between parametric and logical test failures.  
**Signature**:

def visualize\_test\_failure\_comparison(data, parametric\_field, logical\_field, chart\_type="bar", \*\*kwargs):

"""

Generate comparative visualizations to display failure counts or rates for parametric versus logical test failures.

Parameters:

data (DataFrame): The dataset containing failure counts.

parametric\_field (str): Column for parametric test failures.

logical\_field (str): Column for logical test failures.

chart\_type (str): The type of comparative chart (default 'bar').

\*\*kwargs: Additional styling options.

"""

pass

**Used For**:  
– Workflow 9, Instruction 6

**Function 15: Visualize Tester Yield Comparison with Factors**

**Function Name**: visualize\_tester\_yield\_comparison\_with\_factors  
**ID**: 14\_8  
**Purpose**:  
Compare yield performance between testers by generating visualizations (e.g., box plots, scatter plots, timeline charts) that also overlay factors such as test measurement correlations, calibration differences, or load board changes.  
**Signature**:

def visualize\_tester\_yield\_comparison\_with\_factors(data, tester\_field, yield\_field, factors, \*\*kwargs):

"""

Generate visualizations to compare yield performance between testers and analyze influencing factors like test measurement correlations, calibration, and load board changes.

Parameters:

data (DataFrame): Dataset containing tester yield data.

tester\_field (str): Column indicating tester identity.

yield\_field (str): Column for yield performance.

factors (dict): Dictionary specifying additional factors (e.g., calibration changes, load board configuration).

\*\*kwargs: Other visualization options.

"""

pass

**Used For**:  
– Workflow 14, Instruction 8

**Function 16: Visualize Last10 Wafer Yield**

**Function Name**: visualize\_last10\_wafer\_yield  
**ID**: 19\_5  
**Purpose**:  
Generate visualizations (e.g., line plots or charts) to display the wafer-level yield for the most recent 10 lots.  
**Signature**:

def visualize\_last10\_wafer\_yield(data, lots, yield\_field, chart\_type="line", \*\*kwargs):

"""

Generate visualizations to display wafer-level yield for the last 10 lots.

Parameters:

data (DataFrame): Data containing wafer yield per lot.

lots (list): List of the last 10 lot identifiers.

yield\_field (str): Column name for yield values.

chart\_type (str): Type of chart (default 'line').

\*\*kwargs: Additional parameters for chart customization.

"""

pass

**Used For**:  
– Workflow 19, Instruction 5

**Function 17: Visualize E-test Temporal Correlation**

**Function Name**: visualize\_etest\_temporal\_correlation  
**ID**: 34\_9  
**Purpose**:  
Create time series and scatter plots to illustrate temporal trends and the correlation between E-test site means and average parametric test values for a failing measurement family.  
**Signature**:

def visualize\_etest\_temporal\_correlation(data, etest\_field, parametric\_field, \*\*kwargs):

"""

Generate time series and scatter plots to illustrate the temporal trends and correlation between E-test site means and parametric test values.

Parameters:

data (DataFrame): Data containing E-test and parametric measurement values.

etest\_field (str): Column for E-test site means.

parametric\_field (str): Column for average parametric test values.

\*\*kwargs: Additional plotting options.

"""

pass

**Used For**:  
– Workflow 34, Instruction 9

**Function 18: Visualize Repeating Wafer Patterns**

**Function Name**: visualize\_repeating\_wafer\_patterns  
**ID**: 37\_7  
**Purpose**:  
Detect and illustrate repeating wafer map patterns among failing wafers over a specified time window using annotated maps, heatmaps, or frequency charts.  
**Signature**:

def visualize\_repeating\_wafer\_patterns(wafer\_data, pattern\_criteria, chart\_type="annotated\_map", \*\*kwargs):

"""

Generate visual representations to illustrate repeating wafer map patterns among failing wafers.

Parameters:

wafer\_data (DataFrame): Data containing wafer map details.

pattern\_criteria (dict): Criteria to identify repeating patterns.

chart\_type (str): Type of visualization (default 'annotated\_map').

\*\*kwargs: Additional customization options.

"""

pass

**Used For**:  
– Workflow 37, Instruction 7

**Function 19: Visualize E-test Variation Yield Drop**

**Function Name**: visualize\_etest\_variation\_yield\_drop  
**ID**: 30\_7  
**Purpose**:  
Generate visualizations (e.g., scatter plots) to illustrate the correlation between variations in E-test site measurements and yield drops on recent wafers.  
**Signature**:

def visualize\_etest\_variation\_yield\_drop(data, etest\_variation\_field, yield\_field, chart\_type="scatter", \*\*kwargs):

"""

Generate visualizations to illustrate the correlation between E-test site variation and yield drop.

Parameters:

data (DataFrame): The dataset containing E-test variation and yield data.

etest\_variation\_field (str): Column for E-test variation metrics.

yield\_field (str): Column for yield performance.

chart\_type (str): Type of plot to generate (default 'scatter').

\*\*kwargs: Additional options.

"""

pass

**Used For**:  
– Workflow 30, Instruction 7

This specification provides a modular API design where each function is responsible for one or more visualization/reporting tasks. In practice, the functions can be extended with additional parameters and error‐handling as needed.