# CSCI262 Assignment 3 Report

1. Initial Input

a. Data Storage

- Events: Stored in a dictionary events, where each event name maps to attributes like type, min, max, and weight.
- Statistics: Stored in a dictionary statistics, where each event name maps to mean and standard deviation.

b. Consistency Checks

- Validation: The program compares Events.txt and Stats.txt for:
    1. Matching event counts.
    2. Event names aligning across files.
    3. Mean values within event-defined ranges.
- Issues: Discrepancies (e.g. mismatched names) are flagged with specific error messages.

2. Activity Engine and Logs

a. Event Generation

- Events are generated to align with Stats.txt distributions:
    - Continuous Events: Gaussian distribution, constrained by min/max, rounded to two decimals.
    - Discrete Events: Integer-converted Gaussian values within the specified range.

b. Logs

- Format: JSON files (day_X.log), ensuring readability and compatibility.
- Reasoning: JSON is easily processable for analysis and debugging.
- Content: Daily logs include the day number and generated values for each event.

3. Analysis Engine

a. Baseline Data

- Daily Totals: Computed for each event.
- Statistics: Mean and standard deviation calculated using all days' data.

b. Reporting

- Results are stored in JSON format for consistency and usability.

4. Alert Engine

a. Consistency Check Workflow

1. Input Setup: User loads a new statistics file and specifies days for event generation.
2. Data Generation: Events are simulated using the same process as the baseline.
3. Anomaly Detection:
    - Deviation from the baseline mean is calculated in terms of standard deviations.
    - Deviations are weighted using values from Events.txt.
    - Threshold:  2×sum of weights is used to detect anomalies.

b. Reporting

- Daily Results: The program outputs the threshold, anomaly counter, and a status ("okay" or "flagged").
- Repetition: The process can be repeated with different statistics files or exited by the user.