HEALTH SCORE GRADING SYSTEM

Problem Statement Breakdown

We are building a **Health Scoring System** that evaluates a person's overall health based on medical test results. The system should:

1. Assess individual health indicators (also called vitals), such as Blood Pressure, Glucose Level, Oxygen Saturation (SpO2), Temperature, etc.
2. Compare each vital against the World Health Organization **(WHO)** standard ranges to determine how healthy the user's result is.
3. Award a score between 0 and 100 for each vital, where:

    - **100 points** mean the value is within the WHO standard range.
    - **Less than 100 points** mean the value is outside the healthy range (with greater deviation leading to lower scores).

4. Ignore missing vitals—only include the ones provided by the user.
5. Calculate a cumulative health score by combining the individual scores and classifying the user’s health into categories such as "Excellent," "Good," "Average," or "Poor."

Defined WHO Standard Ranges

VITALS	WHO STANDARD RANGE	UNIT
Glucose (Fasting)	70-100	Mg/dL
SpO2	95-100	%
Blood Pressure (Systolic)	90-120	mmHg
Blood Pressure (Diastolic)	60-80	mmHg
Weight (BMI-based)	18.5-24.9	kg/m²
Temperature	36.5-37.5	°C
ECG (Heart Rate)	60-100	BPM
Malaria	Negative	Binary

Widal Test	Negative	Binary
Hepatitis B	Negative	Binary
Voluntary Serology	Negative	Binary

For vitals that are numerical (like blood pressure, glucose, etc.), we check whether the user's value is inside or outside the range. For binary tests (like Malaria, Widal, etc.), a Negative result gets a full score of 100, while a Positive result gets 0.

Calculating Score for Each Vital

Formula for Numerical Vitals

We compare the user's value to the WHO standard range and assign a score using this formula:

Score= max(0,100 − Deviation Factor ×|User Value − Ideal Value|)

Where:

- **User Value** = the test result provided by the user
- **Ideal Value** = the midpoint of the WHO range
- **Deviation Factor** = a number that controls how fast the score decreases when a value moves away from the ideal range

Example Calculation:

- If the WHO range for **Glucose** is **70–100 mg/dL**, the midpoint is:

$$\text{Ideal Value} = \frac{70 + 100}{2} = 85$$

- If the user's Glucose level is **110 mg/dL**, and the Deviation Factor is **1.5**, the score is:

$$\begin{aligned} \text{Score} &= 100 - 1.5 \times |110 - 85| \\ &= 100 - 1.5 \times 25 \\ &= 100 - 37.5 = 62.5 \end{aligned}$$

This means the user gets **62.5 points** for Glucose.

This formula is used to quantify performance or accuracy by penalizing deviations from an ideal value. It assigns a score between 0 and 100, where higher scores indicate closer alignment with the ideal value, and larger deviations lead to greater penalties.

It is a **penalty-based scoring function**, aligning with **regularization in machine learning** and **scoring in optimization** by penalizing deviations from an ideal reference while ensuring scores remain interpretable and bounded.

Alignment with Penalty-Based Loss Functions (Regularization & Scoring Systems)

1. **Penalty for Deviation** (Similar to L1/L2 Regularization)

   - The absolute difference **|User Value−Ideal Value||** acts like an **L1 loss** (absolute error), ensuring larger deviations receive proportionally larger penalties.
   - The **Deviation Factor** controls the **penalty strength**, similar to how regularization parameters (e.g., lambda in Ridge/Lasso regression) control model complexity.

2. **Hard Cutoff at 0 (Non-Negativity Constraint)**

   - Ensures scores never drop below zero, preventing negative or misleading values.
   - This is similar to **clipping loss functions** in machine learning to avoid extreme penalties.

3. **Scoring Systems & Regularization Analogy**

   - Like **regularization in ML**, this formula discourages extreme deviations (overfitting to incorrect values).
   - It acts as a **scoring function in assessments or decision models**, similar to penalty-based loss functions used in optimization.

Binary Vital Scoring

For tests like Malaria or Hepatitis B**:**

- If the result is Negative, the score is **100**.
- If the result is Positive, the score is **0**.

Computing Total Health Score And Classification

To get the final health score, we take the average of all the individual scores (excluding missing vitals):

$$\text{Total Score} = \frac{\sum \text{Vital Scores}}{\text{Number of Provided Vitals}}$$

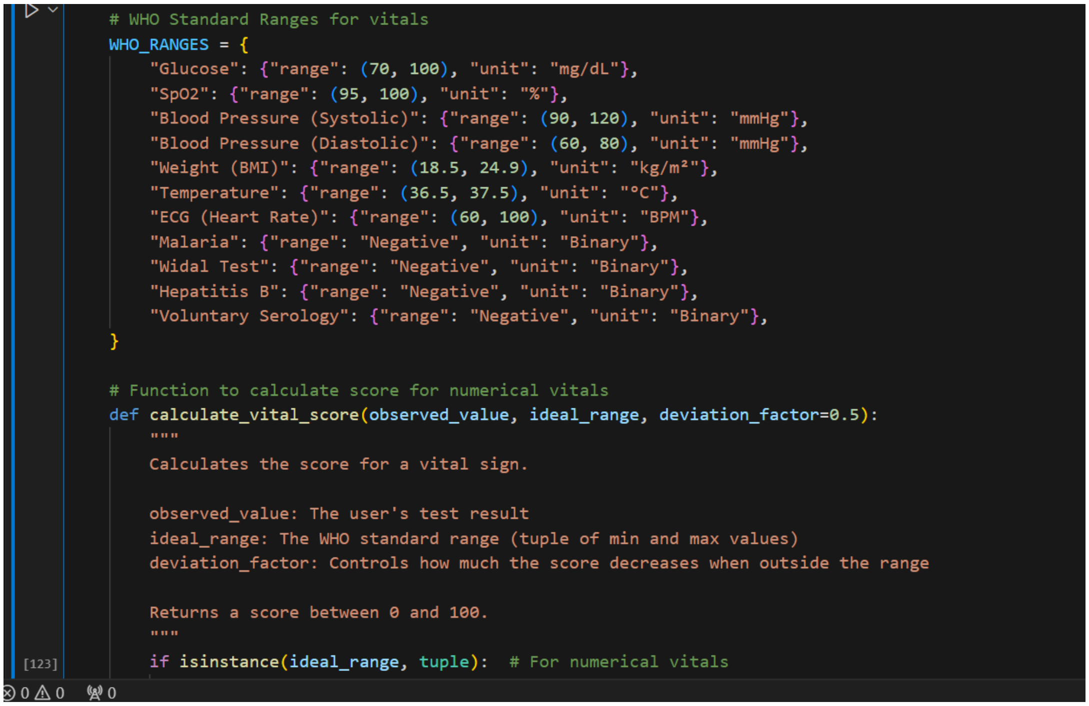
Classify Health Based on Score

We categorize the total score into four health groups:

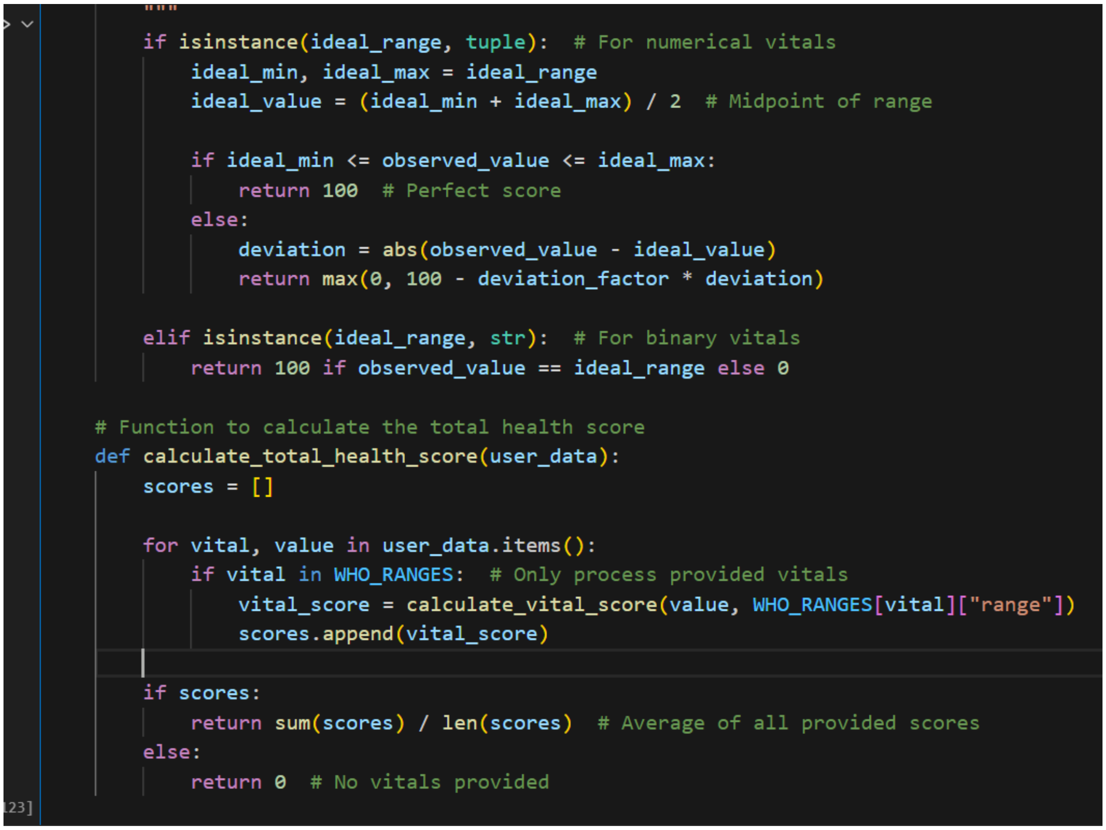
Score Range	**Health Status**
90-100	Excellent
70-89	Good
50-69	Average
0-49	Poor

Python Code Explanation

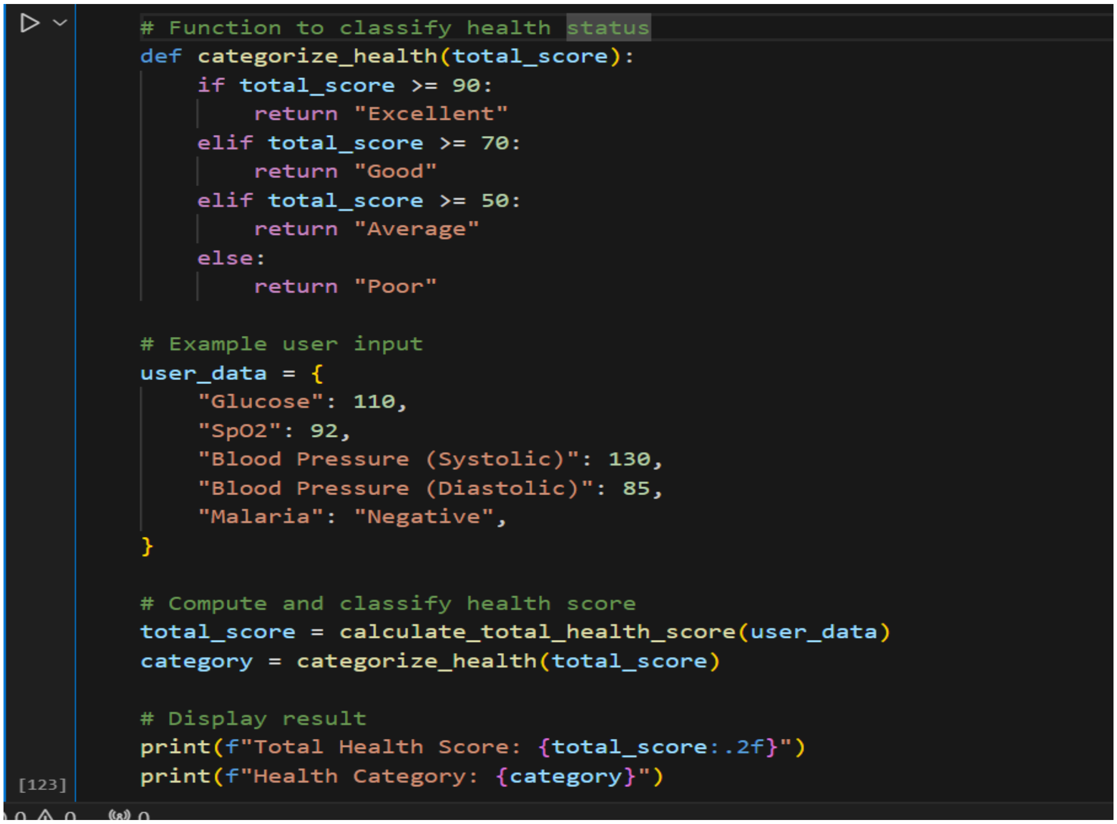
Below is a snapshot of the python code for the Health Score System



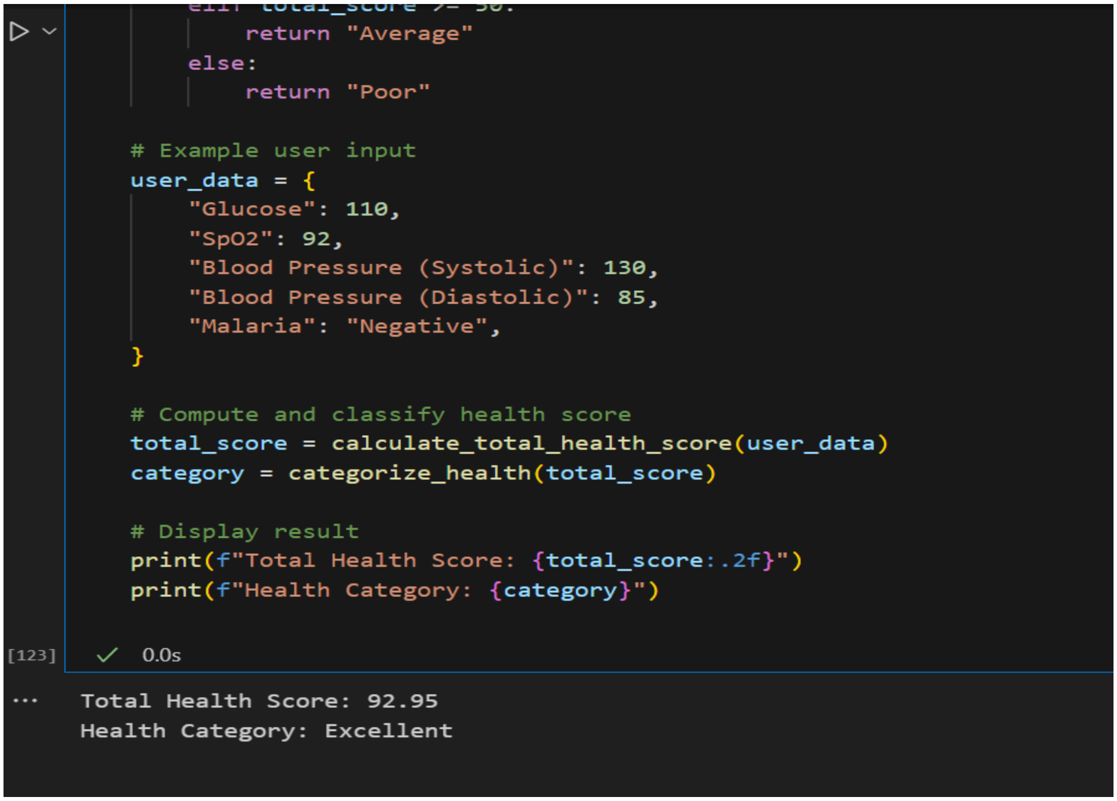
```python
# WHO Standard Ranges for vitals
WHO_RANGES = {
    "Glucose": {"range": (70, 100), "unit": "mg/dL"},
    "SpO2": {"range": (95, 100), "unit": "%"},
    "Blood Pressure (Systolic)": {"range": (90, 120), "unit": "mmHg"},
    "Blood Pressure (Diastolic)": {"range": (60, 80), "unit": "mmHg"},
    "Weight (BMI)": {"range": (18.5, 24.9), "unit": "kg/m²"},
    "Temperature": {"range": (36.5, 37.5), "unit": "°C"},
    "ECG (Heart Rate)": {"range": (60, 100), "unit": "BPM"},
    "Malaria": {"range": "Negative", "unit": "Binary"},
    "Widal Test": {"range": "Negative", "unit": "Binary"},
    "Hepatitis B": {"range": "Negative", "unit": "Binary"},
    "Voluntary Serology": {"range": "Negative", "unit": "Binary"},
}

# Function to calculate score for numerical vitals
def calculate_vital_score(observed_value, ideal_range, deviation_factor=0.5):
    """
    Calculates the score for a vital sign.

    observed_value: The user's test result
    ideal_range: The WHO standard range (tuple of min and max values)
    deviation_factor: Controls how much the score decreases when outside the range

    Returns a score between 0 and 100.
    """
    if isinstance(ideal_range, tuple):  # For numerical vitals
```



```python
    """
    if isinstance(ideal_range, tuple):  # For numerical vitals
        ideal_min, ideal_max = ideal_range
        ideal_value = (ideal_min + ideal_max) / 2  # Midpoint of range

        if ideal_min <= observed_value <= ideal_max:
            return 100  # Perfect score
        else:
            deviation = abs(observed_value - ideal_value)
            return max(0, 100 - deviation_factor * deviation)

    elif isinstance(ideal_range, str):  # For binary vitals
        return 100 if observed_value == ideal_range else 0

# Function to calculate the total health score
def calculate_total_health_score(user_data):
    scores = []

    for vital, value in user_data.items():
        if vital in WHO_RANGES:  # Only process provided vitals
            vital_score = calculate_vital_score(value, WHO_RANGES[vital]["range"])
            scores.append(vital_score)

    if scores:
        return sum(scores) / len(scores)  # Average of all provided scores
    else:
        return 0  # No vitals provided
```



```python
# Function to classify health status
def categorize_health(total_score):
    if total_score >= 90:
        return "Excellent"
    elif total_score >= 70:
        return "Good"
    elif total_score >= 50:
        return "Average"
    else:
        return "Poor"

# Example user input
user_data = {
    "Glucose": 110,
    "SpO2": 92,
    "Blood Pressure (Systolic)": 130,
    "Blood Pressure (Diastolic)": 85,
    "Malaria": "Negative",
}

# Compute and classify health score
total_score = calculate_total_health_score(user_data)
category = categorize_health(total_score)

# Display result
print(f"Total Health Score: {total_score:.2f}")
print(f"Health Category: {category}")
```



```python
    elif total_score >= 50:
        return "Average"
    else:
        return "Poor"

# Example user input
user_data = {
    "Glucose": 110,
    "SpO2": 92,
    "Blood Pressure (Systolic)": 130,
    "Blood Pressure (Diastolic)": 85,
    "Malaria": "Negative",
}

# Compute and classify health score
total_score = calculate_total_health_score(user_data)
category = categorize_health(total_score)

# Display result
print(f"Total Health Score: {total_score:.2f}")
print(f"Health Category: {category}")
```

```
Total Health Score: 92.95
Health Category: Excellent
```

Conclusion

- This model automatically excludes missing vitals from the calculation.
- It adjusts scores based on how far a user's value deviates from WHO standards.
- The total score gives an overall picture of health.
- The system can be integrated into a mobile/web app for easy user interaction.