ECG Quality Analysis and measurement parameters

Goal

1. Analyze the quality of a 12-lead ECG signal to determine whether it is diagnostically usable. The system should provide a user-friendly recommendation about signal issues, indicating which electrode is affected and the likely cause of noise. Validate the quality of a 12-lead ECG before final submission to determine whether the signal is diagnostically usable. If problems are detected — provide a clear explanation to the user indicating which electrodes produced a poor signal, why, and how to fix it.
2. Provided that the ECG validation has passed, the system should analyze and provide basic measurements.

- HartRate (bpm)
- R-R(ms)
- P (ms)
- PR (ms)
- QRS(ms)
- QT(ms)
- QTc(ms)
- Axis (P,QRS and T)

Current Architecture

Client (Mobile Device / Application)

- Records an ECG for a duration of X seconds.
- Applies built-in filters:
  - **Low-pass**, **high-pass**, **notch**, possibly **baseline correction**
- Generates the following data:
  - **Raw ECG signal** (digital format)
  - **Metadata**: type and parameters of filters
  - **ECG image** (PDF for report or preview PNG/JPEG)
- Sends data to the server.

Server

- Receives data from the client:
  - Raw ECG signals
  - Image representation
- Performs:
  - Displays ECG in the form of a report

What Needs to Be Implemented

Signal Validation Procedure Before Submission:

- User taps **"Start Recording"**
- After recording completes (but before final submission/storage):
    - The device sends an ECG "draft" to the server for validation:
        - Raw data + used filter metadata
- **Server actions:**
    - Decode ECG from raw data
    - Validation user's filter set
        - Full signal reconstruction (Based on user's filter configuration)
        - ECG quality analysis
        - If successful → Return answer ECG quality OK, ready to get mesaurements
        - If failed →
            - Get from the organization the minimum quality ECG filters
            - ECG quality analysis
            - If successful → Return answer the ECG is readable but a user can improve the quality by re-recording from the beginning
            - Else -> Return answer: the ECG is not readable and the user should record the ECG from the beginning.
    - Return **Response** and **recommendations to user**
- **Server Response Includes:**
    - Signal quality assessment
    - Recommendations for improvement
    - Identification of noisy leads and noise sources
    - Visual aids explaining:
        - Location of problematic electrodes / poor contact
        - Proper glove usage
        - Optimal posture for recording

What the Server Checks - Categories of Signal Quality Checks

Check	**Detection**	**Possible Causes**
Drift (Baseline Wander)	Floating baseline	Breathing, patient movement, poor contact
50/60Hz Noise	Sinusoidal interference	Electrical noise, weak grounding
Muscle Noise	High-frequency noise in QRS	Muscle tension, patient movement
Loss of Contact	Leads with near-zero amplitude	Poor or absent electrode contact
Movement Artifact	Sudden spikes, clipped peaks	Electrode displacement or detachment

General clinical information

Electrodes and Lead Formation

Lead Calculation Rules:

- **Lead II = Lead I + Lead III**
- **WCT (Wilson Central Terminal) = (RA + LA + LL) / 3**

Lead Definitions:

- Lead I = LA − RA
- Lead II = LL − RA
- Lead III = LL − LA
- aVR = RA − (LA + LL)/2
- aVL = LA − (RA + LL)/2
- aVF = LL − (RA + LA)/2
- **V1–V6 = Vx – WCT**

Lead Dependency on Electrodes

Electrode	Affected Leads	Impact on Signal Quality
RA (Right Arm)	I, II, aVR, aVL, aVF	Complex distortion, noise, or signal loss
LA (Left Arm)	I, III, aVR, aVL, aVF	Complex distortion, noise, or signal loss
LL (Left Leg)	II, III, aVR, aVL, aVF	Complex distortion, noise, or signal loss
RL (Right Leg – ground)	All leads (shared ground)	Increased noise and interference in all leads. RL is not directly involved in measurement of any lead. Its **role is purely to stabilize the amplifier reference** and reduce common-mode noise. It doesn't directly alter lead voltages unless it fails catastrophically
V1–V6	Corresponding lead (V1–V6)	Complex distortion, noise — also affected by WCT , directly influenced by RA, LA, LL (the reference node), not just "slightly."

Lead-to-Electrode Dependency Matrix

Lead/El	RA	LA	LL	RL	V1	V2	V3	V4	V5	V6
L1	yes	yes	no	part	no	no	no	no	no	no
L2	yes	no	yes	part	no	no	no	no	no	no
L3	no	yes	yes	part	no	no	no	no	no	no
aVR	yes	yes	yes	part	no	no	no	no	no	no
aVL	yes	yes	yes	part	no	no	no	no	no	no
aVF	yes	yes	yes	part	no	no	no	no	no	no
V1	no	no	no	part	yes	no	no	no	no	no
V2	no	no	no	part	no	yes	no	no	no	no
V3	no	no	no	part	no	no	yes	no	no	no
V4	no	no	no	part	no	no	no	yes	no	no
V5	no	no	no	part	no	no	no	no	yes	no
V6	no	no	no	part	no	no	no	no	no	yes

Main ECG Noise Types

1. **Baseline Wander**:
    - Low-frequency drift (< 0.5 Hz) of the isoelectric line
    - Typically caused by respiration or patient movement
2. **Artifacts & Interference**:
    - **Muscle Artifact**:
        - High-frequency noise (> 40 Hz) from muscle contractions
        - Overlaps with ECG signal spectrum
    - **Motion Artifacts**:
        - Caused by breathing, coughing, electrode movement
        - Sudden spikes, sometimes clipped due to saturation
        - Frequency: 1–10 Hz
3. **Powerline Interference**:
    - 50/60 Hz electromagnetic noise
    - Caused by nearby electrical equipment or poor grounding
4. **Poor Electrode Contact**:
    - **No Contact / Dry Electrode**:
        - Flat signal, low amplitude (< 0.1 mV), missing QRS
    - **Partial Contact**:
        - High-frequency noise in affected leads only

Диагностика плохого сигнала

· Снижение амплитуды комплекса - повышенное сопротивление контакта: высох гель, электрод неплотно прилегает. При этом все зубцы остаются узнаваемыми, но их высота аномально мала. Проблема решается повторным смачиванием или заменой электродов, улучшением контакта с кожей

.

Signal Quality Diagnostics (Per Lead)

Each ECG lead is evaluated with a percentage score across the following metrics:

EcgLeadQuality

1. **Baseline_Drift**
2. **Muscle_Artifact**
3. **Powerline_Interference**
4. **Bad_Electrode_Contact**
5. **Low_SNR**
6. **QRS_Amplitude**
7. **SNR_dB**

Metric Definitions

- **Muscle_Artifact**:
  Power in 40–100 Hz (excluding 50/60 Hz). Present if >10% of total signal power.
- **Bad_Electrode_Contact**:
  Excessive power in 0.01–0.5 Hz band.
- **Powerline_Interference**:
  Spikes at 50/60 Hz. Present if >5–10% of total power.
- **Baseline_Drift**:
  Dominance of low frequencies (< 0.5 Hz) in the spectrum.
- **Low_SNR**:
  Signal-to-noise ratio < 10 dB.
- **QRS_Amplitude**:
  Peak-to-peak QRS amplitude.
  - < 0.1 mV → Flat signal (possible electrode issue)
  - Normal: 0.5–2.0 mV
- **SNR_dB**:
  Signal-to-noise ratio in decibels.
  - ≥ 20–25 dB is considered acceptable for clinical quality

Library Selection for ECG Quality Assessment and Feature Extraction

Comparison: WFDB (MIT-LCP) vs NeuroKit2

Characteristic	**WFDB (MIT-LCP)**	**NeuroKit2**
anguage	ython, C	ython
urpose	ccess and processing of linical ECG recordings	eature extraction from biosignals
2-lead ECG Support	ull 12-lead support	nly by manually processing ndividual leads
er-lead Signal Quality	imited: manual analysis of mplitude/noise	euristic only via ecg_clean()Not vailable
verall Signal Quality Score	ot available	vailable via ecg_quality() (basic euristic)
QRST Interval Measurement	o built-in solution	vailable via ecg_process()
eak Detection (P, Q, R, S, T)	anual or external scripts	utomatic
eart Rate / HRV	equires annotations	uilt-in

nnotation Support	upports MIT WFDB format	nly post-processing labels
oise Filtering Support	anual filtering using SciPy	uilt-in via cg_clean(method='neurokit')
ustom Filter Support	ull control (open pipeline)	imited (mostly black-box nplementation)
upport for Clinician nnotations	es (standard clinical formats)	o

Reasons to Choose NeuroKit2

- Easier integration into Python workflows
- Prior experience with the library
- Built-in interval measurements and basic quality scoring
- Actively maintained (while MIT WFDB has seen no major updates in 9+ years)

Lead quality

Weighted Importance Coefficients for ECG Leads (*Clinical Use*)

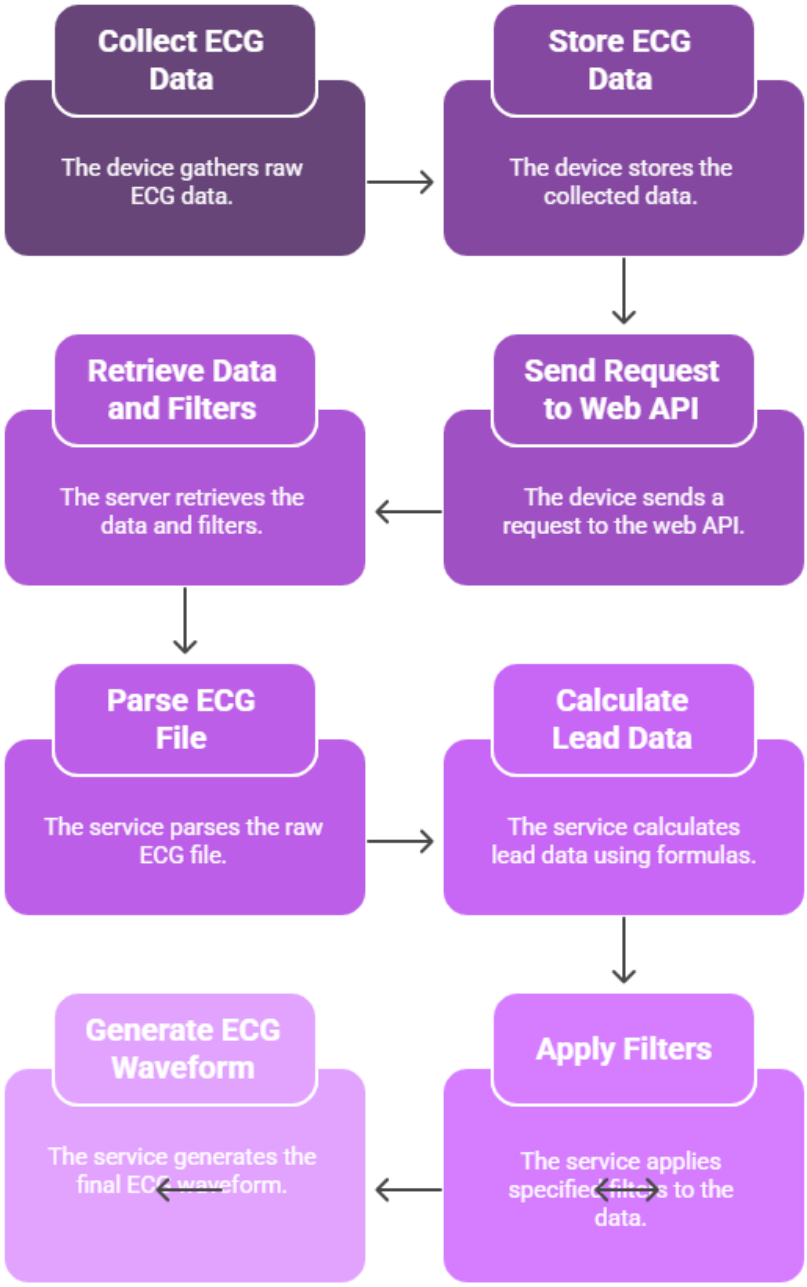
Lead	**Clinical Significance**	**Weight (Importance)**
I	Lateral wall; infarctions, LBBB detection	0.07
II	Primary rhythm analysis; commonly used in monitoring	0.12
III	Inferior wall; less stable but informative	0.06
aVR	Less informative routinely; important in pericarditis, toxicity	0.04
aVL	Lateral wall; useful in infarction detection	0.06
aVF	Inferior wall; frequently used	0.09

V1	Right heart, rhythm, bundle branch blocks	0.10
V2	Septum; key for anterior MI	0.10
V3	Anterior wall	0.10
V4	Apex of the left ventricle	0.08
V5	Lateral left ventricle	0.09
V6	Lateral wall of the left ventricle	0.09

Weighted Importance Coefficients for Ambulance Use (*Prehospital ECG*)

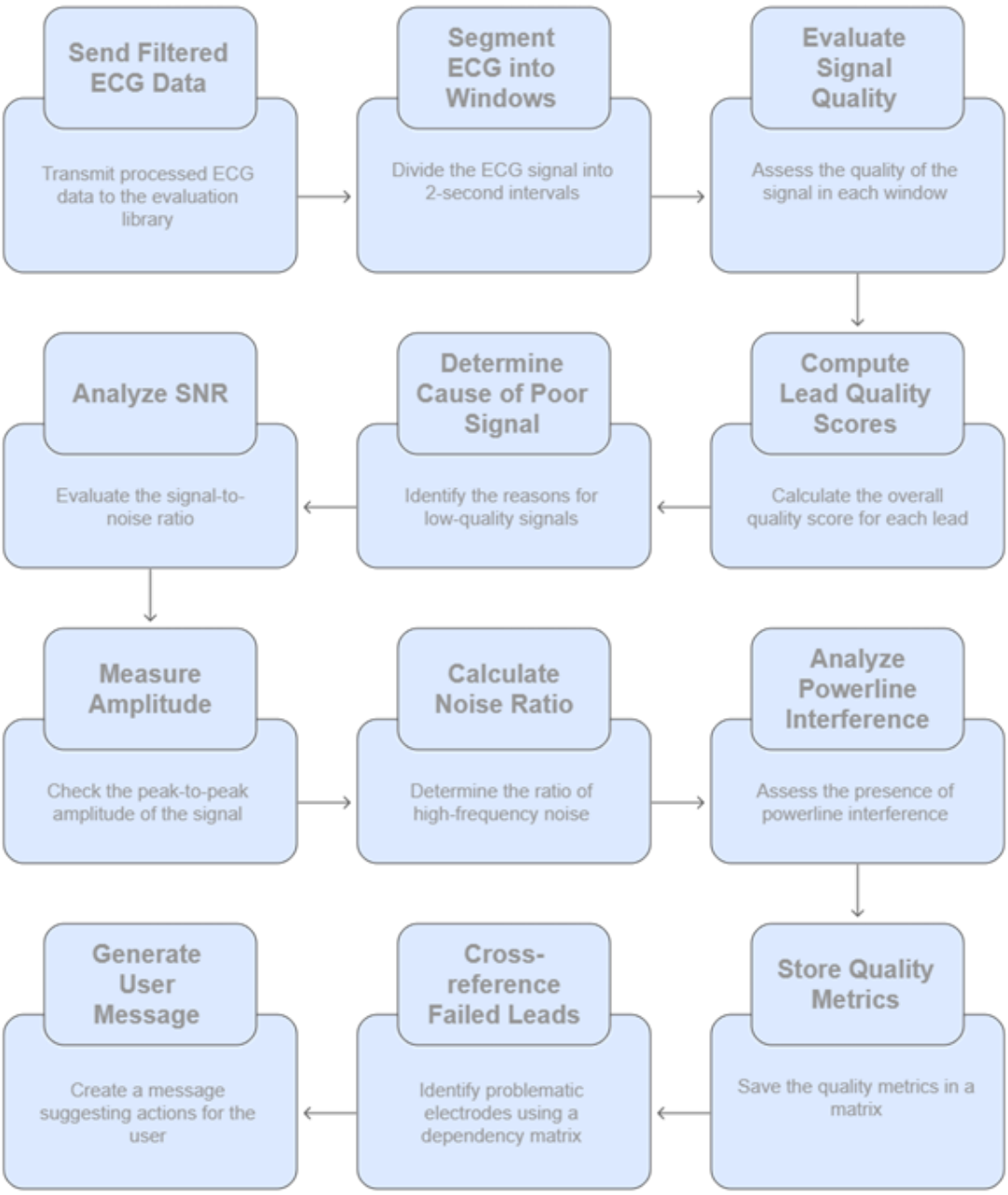
Lead	**Prehospital Importance & Use Case**	**Weight (Ambulance)**
I	Basic lateral view; supportive info	0.06
II	**Main rhythm lead** (used in monitors, AEDs)	0.20
III	Inferior view; complements Lead II	0.07
aVR	Rarely used in emergency decisions	0.03
aVL	Lateral; useful if infarction suspected	0.05
aVF	Inferior wall; common ischemic area	0.10
V1	**Critical** for rhythm (e.g., VT, BBB); P waves in junctional rhythm	0.12
V2	Septal MI, early ischemia	0.10
V3	Anterior wall	0.08
V4	Anterior/Apical infarcts	0.07
V5	Left lateral infarcts	0.06
V6	Left ventricle; supports V5	0.06

ECG Data Processing Sequence



ECG Quality Evaluation Process

ECG Quality Evaluation Process



General Workflow for ECG Signal Quality Assessment

Step-by-Step Process

1. **Device (Client-side):**
    - Collect and store raw ECG data
    - Store basic onboard filtering
2. **Device → Web API:**
    - Send a request to GetECGQuality endpoint
3. **Web (Server-side):**
    - Retrieve the raw ECG data file and associated filter set from the request
4. **Service (Processing Backend):**
    - Parcering raw ECG file.
    - Get Lead data + use PhysioGlove Lead compensation formula
    - Add calculated lead
    - Applying the all specified filters (Reconstruct all ECG leads
    - Generate a ECG waveform (as seen in the clinical interface)
5. **ECG Quality Evaluation per Lead():**

5.1. Send filtered ECG data to the evaluation library for quality .

Segment the 10-second ECG into three 2-second windows. Evaluate the signal quality within each window for each lead. Calculate the average quality score across the 5 windows for each lead. This final value becomes the lead-specific quality score**.**

5.2. Compute lead quality scores:

Select (predefine by organization) the Weighted Importance Coefficients

Formula ECG Quality Assessment $Q_{total} = \sum(Q_{lead} \times weight_{lead})$

- TotalQuality > 0.8 → Good signal
- TotalQuality < 0.8 → Questionable
- TotalQuality < 0.5 → Not usable without further filtering

5.3. If signal is not good, determine the cause:

- **5.3.1. Low Signal-to-Noise Ratio (SNR):**
    - Apply denoising and compare raw vs cleaned signal
    - Estimate SNR as the ratio of clean to noisy power
- **5.3.2. No Signal (Flat or Weak):**
    - Measure peak-to-peak amplitude
    - Very low amplitude may indicate poor electrode contact
- **5.3.3. Muscle Artifact:**
    - Calculate high-frequency noise ratio in two bands:
        - 35–49 Hz and 51–99 Hz
    - Express as % of total signal power
- **5.3.4. Powerline Interference (50/60 Hz):**
    - Analyze FFT in the 49–51 Hz range
    - Express as % of total spectral power

6. **Store per-lead quality metrics in a quality matrix**
    - Thresholds and classification rules will be improve empirically
    - Used for diagnostic logic and signal validation
7. **For any leads that fail validation:**

7.1. Cross-reference failed leads using the lead-electrode dependency matrix

- Identify potentially problematic electrodes

7.2. For each suspect electrode:

- Match failure pattern to known noise types via dependency mapping

7.3. Generate a user-facing message:

- Suggest specific actions (e.g., “Check RA electrode: poor contact suspected due to low amplitude in Leads I, II, aVR”)

все параметры ЭКГ (PR, QRS, QT, P-wave, T-wave, HRV) могут **различаться между отведениями**. **Поэтому каждый параметр нужно анализировать либо в ключевом лиде (обычно II), либо брать максимум** NeuroKit2 **анализирует только 1 отведение за раз**. **QRS-интервал** может **отличаться между отведениями**, особенно V1–V3 (там шире)

Параметр	Отличается по лидам?	Как правильно обрабатывать
QRS duration	✅ Сильно	➤ Брать **максимум** среди всех➤ Или использовать **лид II / V5**
QT interval	✅ Может отличаться	➤ Брать **максимум или медиану**➤ Для автоматов — медиана
PR interval	✅ Бывает разный	➤ Часто используют **лид II** (он “длиннее” и удобнее)
P-wave duration	✅ Может меняться	➤ Брать **лид II или V1**
T-wave duration	✅ Меняется	➤ Лучше брать **V5/V6** или **усреднённый подход**
R-R (HR, HRV)	❌ Почти одинаково	➤ Любой лид с чёткими R-пиками
R-пики (HRV)	❌ Нет разницы, если пики точные	➤ Выбрать лид с наилучшим качеством сигнала

12 сигналов (I, II, III, aVR, aVL, aVF, V1–V6).

ecg_process() делает для каждого лида :

R-пики, P, Q, S, T, PR, QRS, QT-интервалы , HR и HRV с DataFrame с таймкодами

1. **Нужно Использовать лид II *(или V5/V6).* чаще всего используется для измерения QRS, т.к. он даёт хорошую амплитуду QRS-комплекса. Альтернатива: V5 или V6, если II слишком зашумлён или плоский**

    a. **Выбираешь один "опорный" канал (например, II).**

    b. **Прогоняешь его через neurokit2.ecg_process().**

    c. **Извлекаешь ECG["ECG"]["QRS_Duration"].**

2. **Анализировать все 12 и выбрать максимальный QRS, По стандартам (например, AHA), QRS-интервал измеряется как максимальное значение среди всех отведений, особенно если подозреваются патологии (например, блокада ножки пучка Гиса).**

    a. **Прогоняешь каждый лид через ecg_process().**

    b. **Извлекаешь ECG["ECG"]["QRS_Duration"] для каждого.**

    c. **Берёшь максимум из этих значений.**

Функция ecg_quality() — оценивает шум и “разборчивость” пиков.

Функция ecg_clean() — использует встроенные фильтры и может выдать «cleaned ECG».

Code Example Признаки, что **лид зашумлён**

```
import numpy as np
from scipy.signal import butter, filtfilt, welch

def analyze_lead_quality(signal, sampling_rate=500):
    results = {
        "Muscle_Artifact": False,
        "Bad_Electrode_Contact": False,
        "Powerline_Interference": False,
```

```
        "Baseline_Drift": False,
        "Low_SNR": False,
        "QRS_Amplitude": None,
        "SNR_dB": None
}

# Normalize signal
signal = signal - np.mean(signal)

# 1. Check for muscle artifact (high-frequency noise > 40 Hz)
freqs, psd = welch(signal, fs=sampling_rate)
hf_power = np.sum(psd[(freqs > 40) & (freqs < 100)])
total_power = np.sum(psd)
if hf_power / total_power > 0.1:
        results["Muscle_Artifact"] = True

# 2. Bad electrode contact → high variance baseline (slow shifts)
low_freq_power = np.sum(psd[(freqs > 0.01) & (freqs < 0.5)])
if low_freq_power / total_power > 0.2:
        results["Bad_Electrode_Contact"] = True

# 3. Powerline interference at 50 or 60 Hz
pl_50hz = np.sum(psd[(freqs > 49) & (freqs < 51)])
pl_60hz = np.sum(psd[(freqs > 59) & (freqs < 61)])
if (pl_50hz + pl_60hz) / total_power > 0.05:
    results["Powerline_Interference"] = True

# 4. Baseline drift — excessive low-frequency movement
drift = low_freq_power
```

```
    if drift / total_power > 0.1:
        results["Baseline_Drift"] = True

    # 5. Signal-to-noise ratio (SNR)
    signal_amplitude = np.max(signal) - np.min(signal)
    noise = signal - filtfilt(*butter(2, [0.5, 40], btype='bandpass', fs=sampling_rate), signal)
    noise_power = np.mean(noise**2)
    snr = 10 * np.log10(signal_amplitude**2 / noise_power)
    results["SNR_dB"] = snr
    if snr < 10:
        results["Low_SNR"] = True

    # QRS amplitude
    results["QRS_Amplitude"] = signal_amplitude

    return results

# Example test with synthetic noisy signal
fs = 500
t = np.linspace(0, 10, fs * 10)
synthetic_signal = 0.5 * np.sin(2 * np.pi * 1 * t) + 0.05 * np.random.randn(len(t))  # baseline + noise
analyze_lead_quality(synthetic_signal, sampling_rate=fs)
```

Code Example ECG parameters

```
import numpy as np
import neurokit2 as nk
```

```
def analyze_ecg_all_leads(leads, sampling_rate=500):
        results = {
      "Lead_Results": [],
        "Summary": {
        "Best_Lead_Index": None,
        "Best_SNR_dB": -np.inf,
        "QRS_Mean": None,
        "QRS_Max": None,
        "QT_Mean": None,
        "QT_Max": None,
        "PR_Mean": None,
        "PR_Max": None
        }
        }
        qrs_list = []
        qt_list = []
        pr_list = []
        snr_list = []

        for i, lead in enumerate(leads):
        try:
        signals, info = nk.ecg_process(lead, sampling_rate=sampling_rate)
        qrs = info["ECG"].get("QRS_Duration", np.nan)
        qt = info["ECG"].get("QT_Interval", np.nan)
        pr = info["ECG"].get("PR_Interval", np.nan)

        signal_clean = nk.ecg_clean(lead, sampling_rate=sampling_rate)
        noise = lead - signal_clean
        noise_power = np.mean(noise**2)
```

```
        signal_amp = np.max(lead) - np.min(lead)
        snr = 10 * np.log10(signal_amp**2 / (noise_power + 1e-10))

        qrs_list.append(qrs)
        qt_list.append(qt)
        pr_list.append(pr)
        snr_list.append(snr)

        results["Lead_Results"].append({
            "Lead_Index": i,
            "QRS": qrs,
            "QT": qt,
            "PR": pr,
            "SNR_dB": snr
        })

        if snr > results["Summary"]["Best_SNR_dB"]:
            results["Summary"]["Best_SNR_dB"] = snr
            results["Summary"]["Best_Lead_Index"] = i

        except Exception as e:
        results["Lead_Results"].append({
            "Lead_Index": i,
            "Error": str(e)
        })

results["Summary"]["QRS_Mean"] = np.nanmean(qrs_list)
results["Summary"]["QRS_Max"] = np.nanmax(qrs_list)
results["Summary"]["QT_Mean"] = np.nanmean(qt_list)
```

```
results["Summary"]["QT_Max"] = np.nanmax(qt_list)

results["Summary"]["PR_Mean"] = np.nanmean(pr_list)

results["Summary"]["PR_Max"] = np.nanmax(pr_list)


    return results
```