PROJECT 5 – ECG Signal Explorer & Heart Rhythm Analyzer

Final Project for Introduction to Python Programming

40 marks — ~10 hours of work

BACKGROUND – THE BIOLOGY & HEALTH SCIENCE

The electrocardiogram (ECG or EKG) records the electrical activity of the heart. Health professionals analyze ECG waves to detect:

- Heart rate (beats per minute)
- Arrhythmias (irregular rhythms)
- R-peaks (major spikes in the QRS complex)
- HRV (heart rate variability)

In this project you will analyze simulated ECG recordings of 6 patients. Each contains 10 seconds of ECG signal sampled at 250 Hz (2500 data points).

You will:

- Clean and preprocess ECG signals
- Detect R-peaks using simple numerical methods
- Compute HR (heart rate) and HRV metrics
- Create 4+ visualizations
- Build an animation showing ECG scrolling over time
- Export summary tables

This is similar to entry-level biomedical engineering and cardiology analysis.

PROJECT TASKS (Overview)

Your team will:

- Load CSV of ECG values for each patient

• Use strings, lists, NumPy arrays, pandas

• Compute:

– filtered ECG

– R-peak positions

– heart rate (BPM)

– RR-intervals and HRV (standard deviation)

• Create at least 4 PNG figures:

1) Raw ECG line plot

2) Filtered ECG plot

3) Scatterplot of RR-intervals

4) Histogram of heart rates

• Create 1 GIF animation:

ecg_scrolling.gif — ECG waveform scrolling across screen

• Export:

ecg_summary.csv

rr_intervals.csv

GROUP ROLES (Recommended)

Student 1 – Data Engineer

• Load ECG CSV files

• Structure DataFrame: PatientID, Time, ECG

Student 2 – Signal Analyst

• Compute filtered signal (moving average)

• R-peak detection using threshold logic

Student 3 – Visualization

• Create 4 PNG figures

Student 4 – Animation + Report

• Scrolling animation

• Group presentation

WEEK 1

(2 hrs) Load ECG data + inspect

(3 hrs) Filter signal + detect R-peaks

WEEK 2

(3 hrs) PNG visualizations

(2 hrs) Animation + presentation

DELIVERABLES

Your submission folder must include:

1. ecg_project.py
2. Figures:
   - raw_ecg.png
   - filtered_ecg.png
   - rr_scatter.png
   - hr_hist.png
3. Animation:
   - ecg_scrolling.gif
4. CSV files:
   - ecg_summary.csv
   - rr_intervals.csv
5. Professional Project Presentation (PDF or pptx)

LEARNING OUTCOMES

• Use NumPy arrays for signal processing

• Use pandas for timeseries analysis

• Create biomedical visualizations with matplotlib & seaborn

- Produce animations showing signal dynamics
- Understand ECG interpretation basics

Good luck

Final Project Rubric (40 marks)

IMP 1. Only materials covered in class lectures, labs, and assignments. if you are using any reference outside the following textbook chapters (i.e., Chapters 1-9, Chapters 14-15) get approval from the teacher before use.

IMP 2. Research to work with real data files from scientific domains.

IMP 3. Due to the possibilities of using Generative AI, and third-party code solutions, even full-functioning code may not qualify for evaluation and may be discarded with a zero (0) mark. To avoid such evaluation, strongly follow point IMP 1.

A) Data Processing (12 marks)

- ECG loaded correctly (2)
- Filter applied correctly (3)
- R-peak detection functional (4)
- HR & HRV computed correctly (3)

B) Visualizations (10 marks)

- Raw ECG line plot (2)
- Filtered ECG plot (2)
- RR-interval scatterplot (2)
- Heart rate histogram (2)
- Clear labeling and axes (2)

C) Code Quality (8 marks)

- NumPy used correctly (2)
- Pandas used correctly (2)
- Modular functions (2)
- Commenting and readability (2)

D) Animation (5 marks)

- ECG scroll animation functions (3)
- Smooth, readable GIF (2)

E) Team Presentation (5 marks)

- Heart rhythm interpretation (3)
- Clarity and professionalism (2)

starter.py

```
# ECG_Project_starter.py
# Starter File — Complete all TODOs

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib import animation
DATAFILE = "ECG_sample.csv"
FS = 250  # sampling frequency
def load_ecg(path):
    # TODO load CSV, convert to dataframe
    pass
def filter_signal(values):
    # TODO moving average smoothing
    pass
def detect_r_peaks(values):
    # TODO simple threshold-based R-peak detection
    pass
def compute_metrics(df):
    # TODO compute BPM and HRV per patient
    pass
def create_plots(df):
    # TODO raw_ecg.png
```

```
    # TODO filtered_ecg.png
    # TODO rr_scatter.png
    # TODO hr_hist.png
    pass
def make_animation(df):
    # TODO horizontal scrolling ECG animation
    pass
def export_results(df):
    # TODO ecg_summary.csv and rr_intervals.csv
    pass
if __name__ == "__main__":
    df = load_ecg(DATAFILE)
    # Apply filtering
    # Detect R-peaks
    # Visualizations
    # Animation
    print("Complete all TODOs!")
```

Special Note on ECG data files

#ECG_sample.csv

For plotting only (not real ECG analysis):

- 200–300 samples
  You can see ~1–2 heartbeats clearly.

For school-level ECG visualization:

- 500 samples
  You can show multiple beats and QRS peaks.

For any real analysis:

- 1000–2500 samples
  10 seconds at 100–250 Hz is the global standard.

For clinical-grade accuracy:

- 2500–5000 samples
  Usually sampled at 250–500 Hz.

Why the sample size is tied to sampling rate (Nyquist Theory)

ECG signals contain frequencies up to:

- 40 Hz for normal signals
- 150 Hz needed for accurate QRS detection
- Recommended sample sizes:

Purpose	Minimum Samples	Best Choice	Explanation
Simple animation	100	300	Smooth line movement
Multiple cycles	300	600	Shows full ECG waveform
Approx scientific	600	1000	Covers ~4–5 heartbeats
Realistic	1000	2500	Matches clinical ECG