Lab 4 - Data Acquisition and Analysis of ECG Signals Using Wireless Heart Rate Sensor

Course: EEEE 536.60L1 and EEEE 636.60L1 – Biorobotics/Cybernetics

Group Members: Youssief Khalifa(yak9700), Majd Abo Kalam(mma7566)

Kinan Ghaalrebat(kig6230)

Date: 18-4-2025

1) Manifesto

All team members actively worked on every section of this lab report. The structure of which collaboration was very efficient and led to high standards for all tasks being completed. We made sure that the lab questions and tasks were divided equally between both of us so that there is fairness and efficiency. The responsibility for specific set of exercises and sections of the report was taken by each of the team members.

2) Introduction

Through Electrocardiography (ECG) devices monitor the electrical heart processes. The ECG signal shows important information about cardiac functionality because it enables monitoring of heart rate along with arrhythmia diagnosis and heart health assessment. The Wahoo chest strap served as an instrument for obtaining heart rate measurements based on ECG signals in this lab.

Wahoo fitness application gathered all data that later were exported as .csv files. With the help of Python the heart rate signal received analysis while also undergoing data cleaning and visualization. The research converted untreated wearable ECG data through state-of-the-art signal processing methods into clinical reportable medical insights.

3) Theory

Myocardial depolarization and repolarization processes generate minor electric signals responsible for heartbeats. Physiological electrical characteristics that doctors measure originate from the body tissue and are captured through electrodes applied to the skin surface. ECG data appears in raw form as voltage signals but the Wahoo sensor generates heart rate values (bpm) automatically during real-time processing.

Various difficulties exist when working with ECG heart rate signals.

- The recorded ECG signals contain disturbances known as motion artifacts which produce varieties of peaks and valleys.
- Signals become missing due to delays orANTɎRs in the sensor communication.
- Noise from environmental interference

Signal processing techniques need to clean and visualize ECG data properly before meaningful trends can be extracted from the data through filtering methods.

- NaN removal and interpolation
- Outlier rejection
- Smoothing (Rolling Average, Savitzky-Golay filter)

---

4) Methodology

The Wahoo heart rate sensor operated through the Wahoo Fitness App to retrieve the ECG data. The subject started the session on the app followed by securing the chest strap before walking for data collection. The data export process from the app happened directly to .csv format when the recording finished.

The evaluation of ECG heart rate data occurred through Python by implementing multiple essential libraries. A CSV file was processed into a DataFrame through Pandas library to enable convenient data research and alteration. Only the timestamp and heart_rate columns received analysis focus because the distance, calories, and battery status data was irrelevant for this assessment.

The time-based plotting required proper datetime objects from timestamp values during processing. The preprocessing started by removing empty rows that included NaN values and followed with heart rate filtering to retain values within the normal heart rate range between 40 to 200 bpm. The researchers performed chronological sorting after dropping all timestamp duplicates due to maintenance consistency.

Linear interpolation replaced missing values found in the heart rate column of the signal. The implementation added another cleansing technique which examined heart rate change rates before eliminating any heart rate value exceeding 30 bpm changes in one second.

The data received two smoothing algorithms for processing. The Savitzky-Golay filter served as the first data smoothing technique because it protected local peak information during noise reduction. The rolling average over 10 samples proved to provide reliable results although it had a limited response time. The cleaned signals received from filtering appeared next to the cleaned data set for easy visual assessment.

Basic summary statistics concerning heart rate including average values and minimum and maximum rates underwent extraction and print out to provide cardiac activity data during the session.

---

5) Results

We show all processing steps in this section which started with importing the raw .csv file and ended with producing cleaned heart rate graphs that resulted from our ECG signal pipeline. Our mission centered on converting ECG-derived heart rate data into a meaningful signal through Python scripting basic signal processing operations.

We first inspected the dataset structure for identifying missing values and outliers by loading the dataset. The conversion of timestamps to datetime format enabled us to perform preprocessing treatments that removed NaN values yet maintained the removal of outliers and the synchronization of data gaps using data interpolation.

The heart rate signal received analysis through two smoothing procedures including Savitzky-Golay filtering and rolling average methods. The filters provided two ways to view an equivalent cleansed dataset with peak-preserving qualities alongside overall trend-focused presentations.

The following figures describe the essential transformations in our workflow which we describe with detailed explanations about their significant effects. The examination of numerous results revealed profound knowledge about data cleanup processes throughout biosignal analysis when analyzing actual wearable sensor data.

```python
import pandas as pd
file_path = '2025-04-11-070201-WAHOOAPPIOSB118-1-0-record.csv'
df = pd.read_csv(file_path)
df.head()
```

[1] ✓ 0.5s Python

	timestamp	heart_rate	distance	calories	battery_soc
0	04/11/2025, 11:02:01 AM	NaN	0.0	0	45.0
1	04/11/2025, 11:02:02 AM	107.0	0.0	0	NaN
2	04/11/2025, 11:02:03 AM	107.0	0.0	0	NaN
3	04/11/2025, 11:02:04 AM	108.0	0.0	0	NaN
4	04/11/2025, 11:02:05 AM	109.0	0.0	1	NaN

Figure 1 – Raw Data Load and Structure

The figure presents initial data rows which contain timestamp alongside heart_rate measurements and distance and calories values alongside battery_soc measurements. The heart_rate column shows its missing value (NaN) starting at row 0. During recordings the sensors sometimes fail to produce reliable data at specific timestamps since it is a frequent problem that occurs when wireless devices start acquiring signals.

```python
df[time_col] = pd.to_datetime(df[time_col], errors='coerce')
df[[time_col, heart_col]].dropna().head()
```

[3] ✓ 0.0s Python

	timestamp	heart_rate
1	2025-04-11 11:02:02	107.0
2	2025-04-11 11:02:03	107.0
3	2025-04-11 11:02:04	108.0
4	2025-04-11 11:02:05	109.0
5	2025-04-11 11:02:06	108.0

Figure 2 – Timestamp Conversion and NaN Removal

After successful conversion of timestamps to datetime format and removal of missing values the data became suitable for conducting time-series analysis. The timestamp conversion enabled accurate placement on the x-axis while making it possible to implement both interpolation and running average calculations. These values demonstrate resting-to-active transition between 107 and 109 bpm.

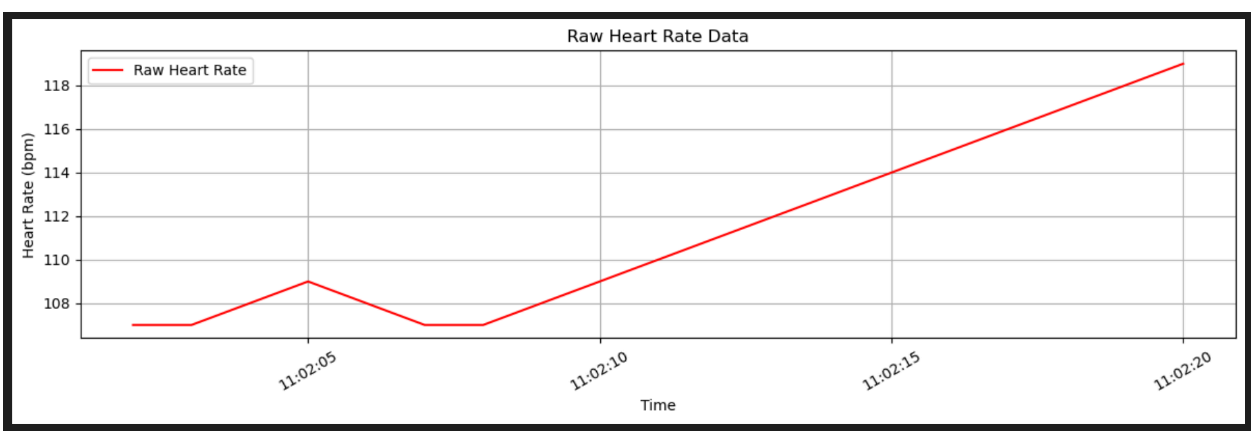


Figure 3 – Raw Heart Rate Plot

The heart rate signal displays directly from its raw data state in this plot. The smooth appearance of the red line across the plot becomes disrupted upon closer observation when tiny inconsistencies and jitter are noticeable. Transient sensor motion together with minor internal device calculation delays can cause these minimal peaks in the recorded signal. The non-smoothed 107–119 bpm range of this plot fails to demonstrate actual cardiac behavior because there is no error corrections applied.

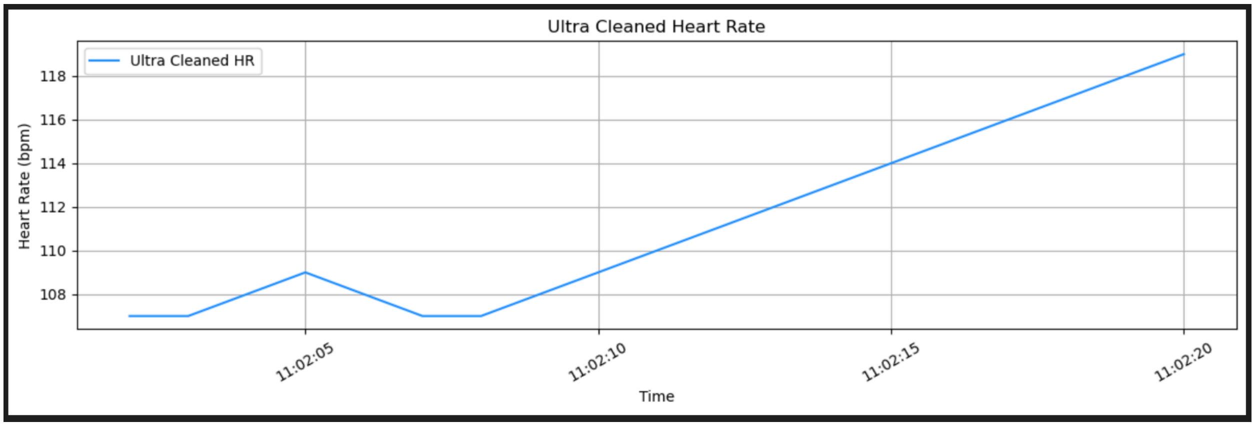


Figure 4 – Ultra Cleaned Heart Rate Plot

This plot shows the same heart rate data after passing through multiple cleaning steps:

- NaN removal
- Outlier filtering (values <40 or >200 bpm)
- Duplicate timestamp elimination
- Linear interpolation for minor missing entries
- Spike removal via rate-of-change filtering (ΔHR > 30 bpm)

After digital filtering the new line shows refined movements which emulate typical heart activity while performing light exercise. The blue line adopts the original curve patterns while discarding random fluctuations which creates a refined and readable signal.

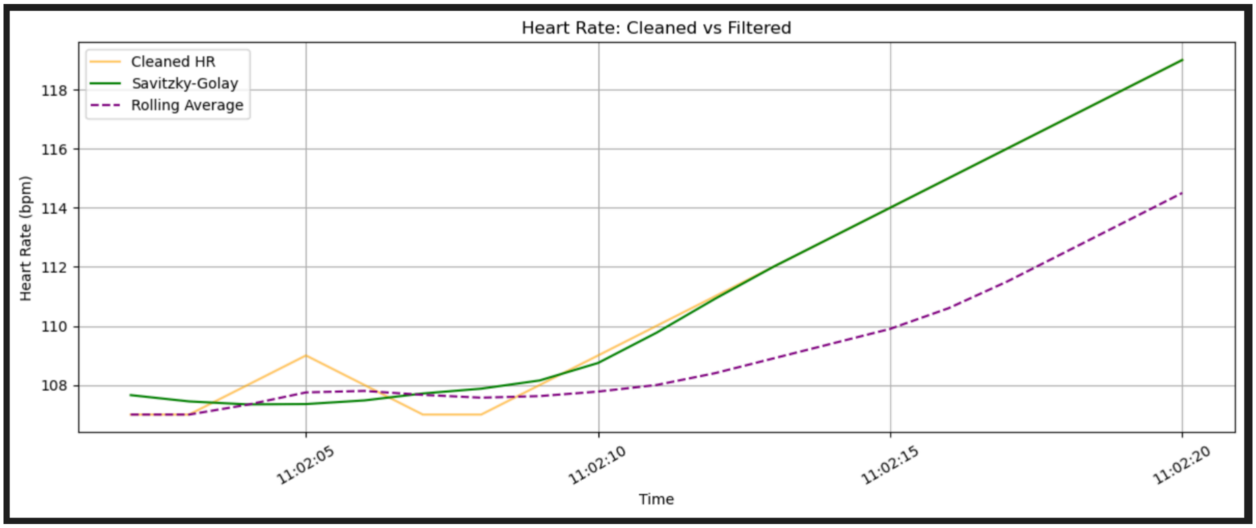


Figure 5 – Filter Comparison: Cleaned vs Savitzky-Golay vs Rolling Average

The central element of the discussion represents this figure. Two filtered signals (green and purple dashed) appear on top of the cleaned signal (orange) in this figure.

- Savitzky-Golay (green): Applies polynomial filtering that upholds both original curve shape and definition. Heart rate signals heavily benefit from the preserving capabilities of this method because it detects rapid changes effectively. A filtered line emerges from the process showing high correlation to the cleaned signal through suppression of minor jitters.
- The purple dashed Rolling Average: Computes a 10-sample window average to generate its result. The method generates an even flat outcome that maintains steady trends but fails to present current heart rate fluctuations. The effective osillation suppression from this method sacrifices valuable signal details that may be vital during medical and sports analytics.

This data graph shows that Savitzky-Golay provides a quick reaction to changes at the expense of stability whereas Rolling Average delivers smoothness while having delayed responsiveness. The specific goals between emergency heart monitoring and daily fitness would guide whether one filter or the other should be chosen.

```python
print("Stats:")
print(f"Average HR: {df_clean[heart_col].mean():.2f} bpm")
print(f"Max HR: {df_clean[heart_col].max()} bpm")
print(f"Min HR: {df_clean[heart_col].min()} bpm")
```

✓ 0.0s Python

```
Stats:
Average HR: 111.32 bpm
Max HR: 119.0 bpm
Min HR: 107.0 bpm
```

Figure 6 – Summary Statistics Output

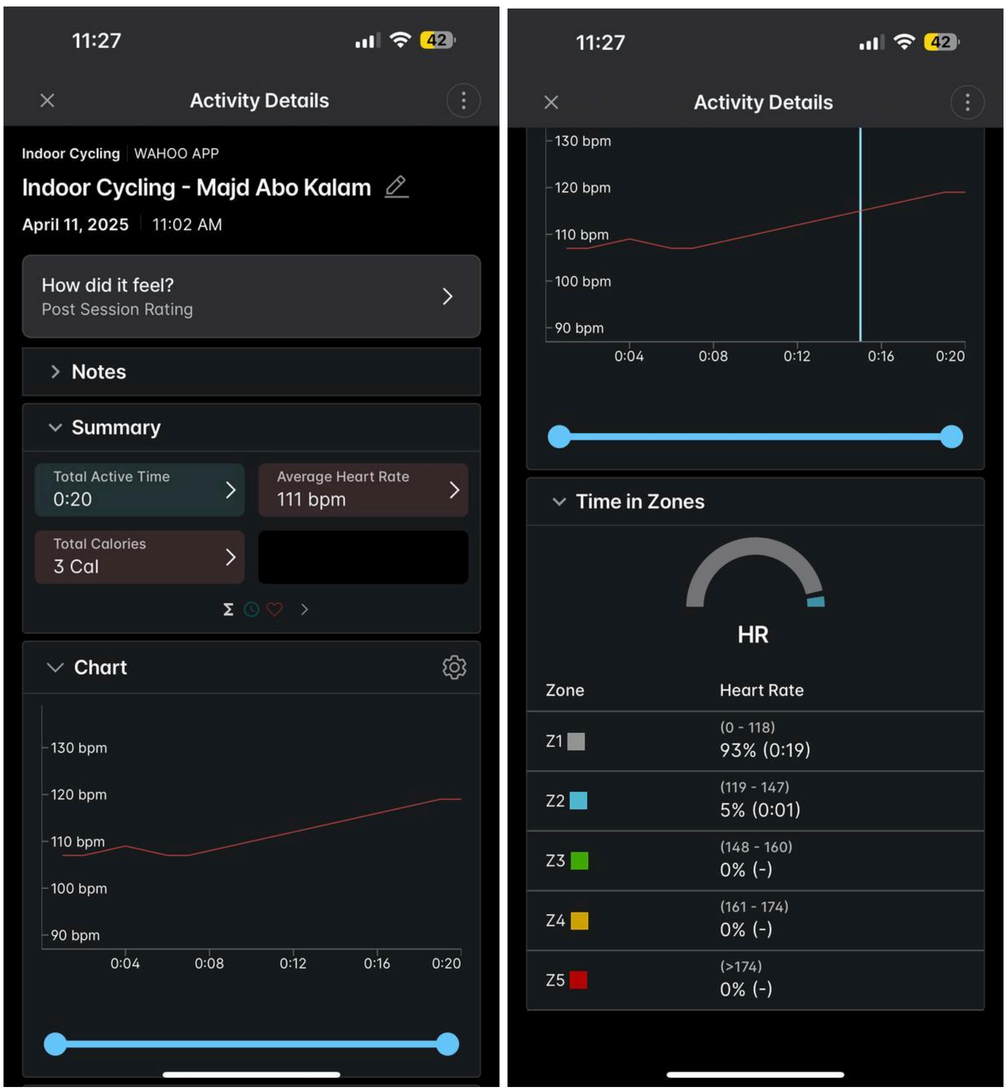
A summary of the entire session appears as the concluding output from Python with filtered data. The cardiac rhythm averaged at 111.32 beats per minute during the period when individuals walked or remained standing after their movement. The BPM achieved its highest mark at 119 while the lowest reading reached 107 as the sensor recorded rest and minimal physical activity.

B)

We decided to include all the screenshots as we felt like that all figures were necessary to include.

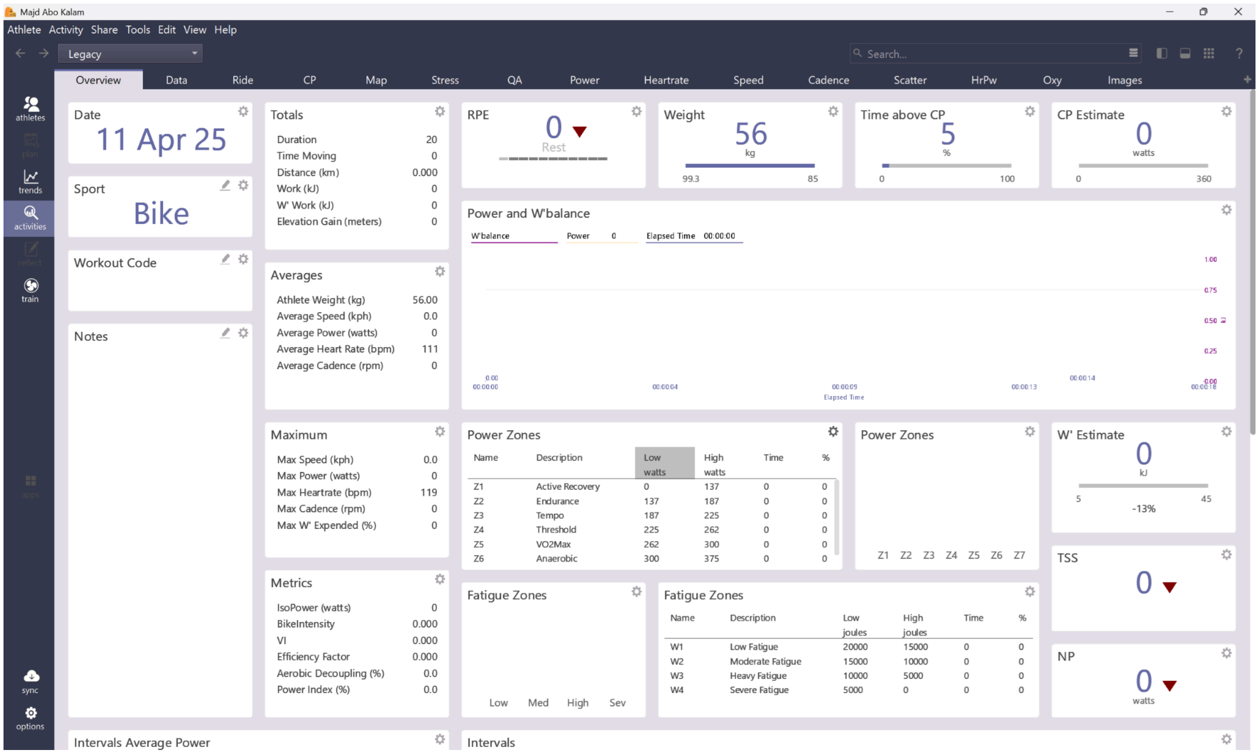
---

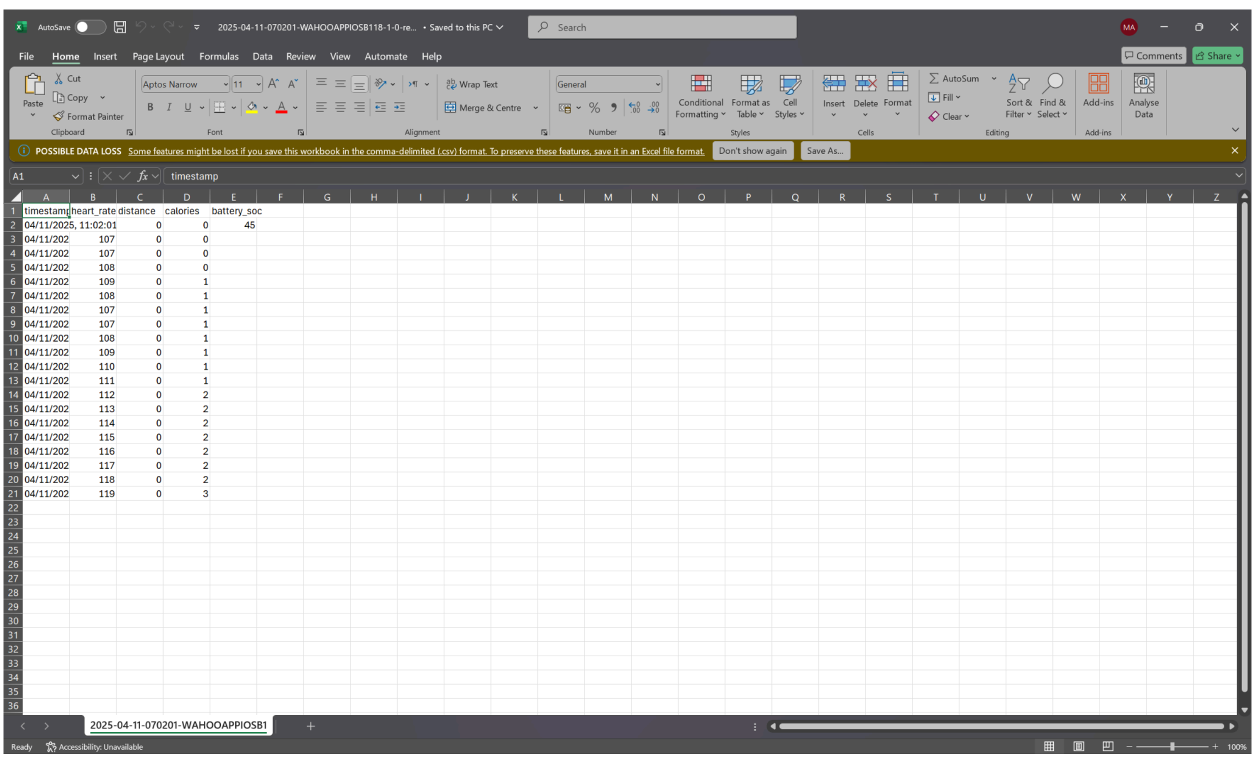
6) Screenshot of the Heart Rate Activity Recorded in your App:



---

7) Screenshot of Golden Cheetah Software Along With Your Data:





8) Discussion on need for ECG in real world applications:

In both healthcare facilities and non-clinical environments Electrocardiography functions as an important diagnostic instrument. Hospital facilities employ ECG as their primary approach for arrhythmia diagnosis alongside myocardial infarction detection and heart failure tracking and cardiac status assessment. The continuous observation of ECG signals helps patients with chronic heart problems under intensive care conditions and during their post-operative healing process.

ECG-based wearables including chest straps and smartwatches together with fitness trackers are used increasingly by consumers for preventive health needs and fitness enhancements as well as remote patient measurement. The Wahoo sensor helps athletes check their training intensity and recovery in real time as it simultaneously exposes unusual heart activity during sports activities. Applications have become increasingly necessary alongside the development of telemedicine alongside home-based healthcare systems.

Note that the capability to manipulate ECG data through programming according to this lab test base a crucial requirement for sustaining practical healthcare systems. Wearable sensor data benefits all users including physicians and athletes and AI systems because reliable decision-making can happen thanks to this process.

---

9) References

Wahoo Fitness. (n.d.). TICKR heart rate monitor. https://www.wahoofitness.com/

The pandas development team. (n.d.). pandas documentation. https://pandas.pydata.org/docs/

The Matplotlib development team. (n.d.). Matplotlib documentation. https://matplotlib.org/stable/contents.html

Virtanen, P., et al. (2020). SciPy signal processing: Savitzky-Golay filter. In SciPy v1.6.0 Manual. https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.savgol_filter.html

Mayo Clinic. (n.d.). Electrocardiogram (ECG or EKG). https://www.mayoclinic.org/tests-procedures/ekg/about/pac-20384983