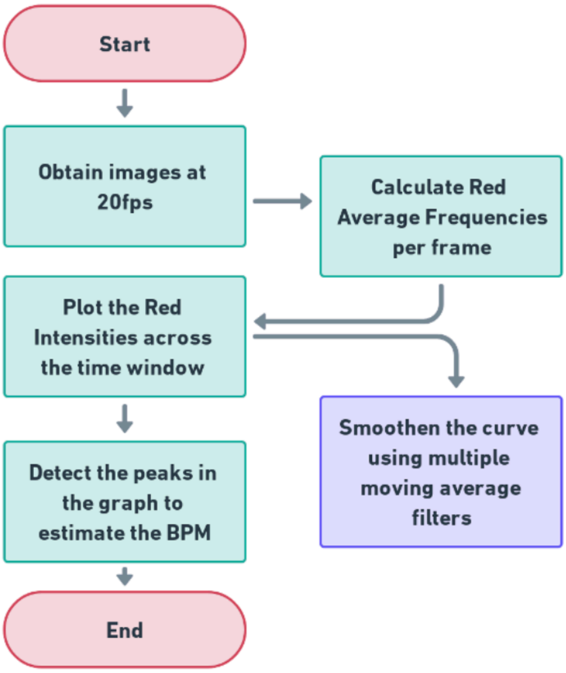
METHODOLOGY

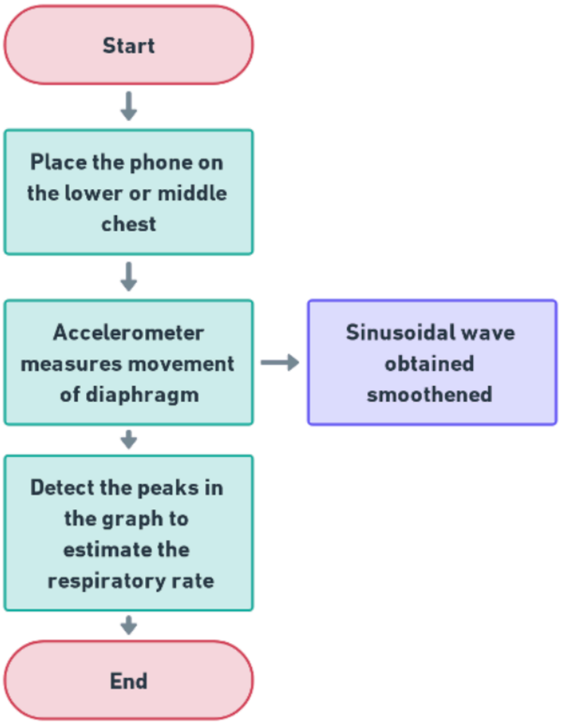
Heart Rate


Start
Obtain images at 20fps
Calculate Red Average Frequencies per frame
Plot the Red Intensities across the time window
Smoothen the curve using multiple moving average filters
Detect the peaks in the graph to estimate the BPM
End

1.**Data Collection**: Capture a video or sequence of images at a consistent frame rate, such as 20 frames per second (fps). These images should focus on a region where the user's skin tone is visible, often around the fingertip or the face.
2.**Image Preprocessing**: For each frame, process the image to extract the red channel (or other color channels that emphasize blood flow changes). This might involve filtering or segmentation techniques to isolate the region of interest (ROI) containing the skin tone.
3.**Frequency Analysis**: Calculate the frequency content of the red channel's intensity using techniques like the Fast Fourier Transform (FFT). This helps to identify the dominant frequency component associated with heart rate.
4.**Frequency-to-BPM Conversion**: Convert the dominant frequency obtained from the FFT to beats per minute (BPM). This conversion can be done using the relationship between frequency and heart rate.
5.**Time Series Plotting**: Plot the red channel intensity values across time to create a time series. This graph will show the variation in red intensity due to blood flow changes.
6.**Smoothing the Curve**: Apply moving average filters to smoothen the time series curve. This can help reduce noise and make it easier to detect peaks accurately. You can experiment with different window sizes for the moving average filters.
7.**Peak Detection**: Implement a peak detection algorithm to identify the peaks in the smoothed curve. A common approach is to find local maxima that are significantly higher than their neighbouring points.
8.**BPM Calculation**: Calculate the heart rate in beats per minute based on the time intervals between the detected peaks.

9.**Validation and Calibration**: Validate your approach against known heart rate measurements to ensure accuracy. You might need to calibrate the method for different lighting conditions, skin tones, and camera setups.
9.**Machine Learning Enhancement** : For improved accuracy, you can consider training a machine learning model on a labeled dataset of heart rate data captured using medical-grade equipment. This model could learn to predict heart rate directly from the image features, potentially eliminating the need for some of the manual signal processing steps.

Respiratory Rate


Start
Place the phone on the lower or middle chest
Accelerometer measures movement of diaphragm
Sinusoidal wave obtained smoothened
Detect the peaks in the graph to estimate the respiratory rate
End

1.**Data Collection**: Place the smartphone on the lower or middle chest of the person. The smartphone's accelerometer will measure the movement caused by the expansion and contraction of the chest during breathing.
2.**Data Capture**: Use the smartphone's accelerometer to collect continuous acceleration data over a specific time duration.
3.Signal Preprocessing:

-**Data Filtering**: Apply appropriate filtering techniques, such as low-pass filters, to remove high-frequency noise from the accelerometer data.

-**Baseline Correction**: Remove any static or slow-varying components from the signal to focus on the oscillatory respiratory component.

4.Frequency Analysis:

-**FFT or Wavelet Transform**: Apply the Fast Fourier Transform (FFT) or Wavelet Transform to convert the accelerometer signal from the time domain to the frequency domain.

-**Dominant Frequency**: Identify the dominant frequency component in the frequency spectrum. This frequency corresponds to the respiratory rate.

5.Peak Detection:

-**Smoothing**: Smooth the frequency-domain signal to make peak detection more accurate.

-**Peak Detection Algorithm**: Use peak detection algorithms (e.g., finding local maxima) to identify the peak corresponding to the dominant respiratory frequency.

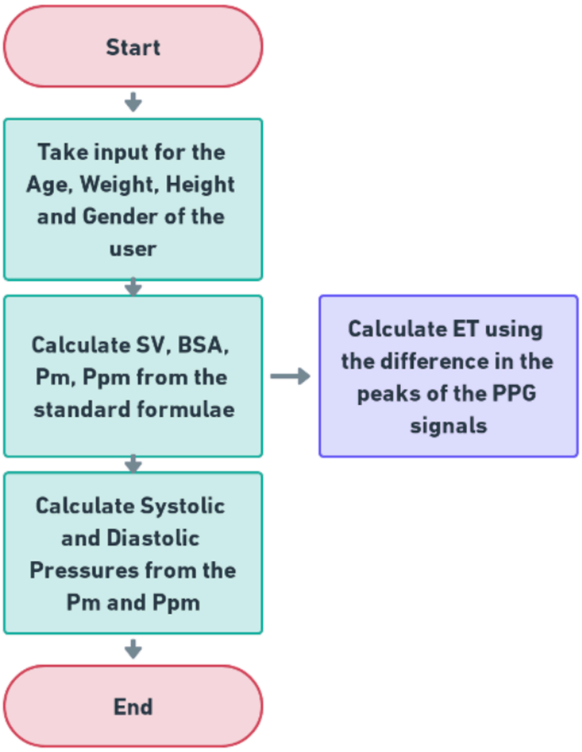
6.**Respiratory Rate Calculation**: Convert the frequency of the peak to the respiratory rate. The relationship between frequency and respiratory rate can be established through calibration using reference measurements.

7.**Validation and Calibration**: Validate your method by comparing the calculated respiratory rate with measurements obtained using established medical techniques.

8.**Visualization**: Optionally, you can visualize the accelerometer data, the frequency spectrum, and the detected peak to gain insights into the breathing pattern.

9.**User Experience**: Ensure the comfort and stability of the smartphone placement on the chest for accurate measurements. Consider user guidance to ensure consistent data collection.

Blood Pressure


Start
Take input for the Age, Weight, Height and Gender of the user
Calculate SV, BSA, Pm, Ppm from the standard formulae
Calculate ET using the difference in the peaks of the PPG signals
Calculate Systolic and Diastolic Pressures from the Pm and Ppm
End

Measuring blood pressure using a smartphone and photoplethysmography (PPG) signals combined with physiological parameters involves a comprehensive approach

1.User Information Input:

Collect user information like age, weight, height, and gender.

These parameters are important as they can influence physiological characteristics used in blood pressure estimation formulas.

2.Calculate Stroke Volume (SV):

Use an appropriate formula to estimate stroke volume based on age, weight, and gender.

SV estimation from PPG signals can be challenging due to various factors influencing PPG waveforms.

3.Calculate Body Surface Area (BSA):

Use relevant formulas (e.g., Dubois-Dubois formula) to calculate body surface area from weight and height.

BSA is used in some blood pressure estimation methods.

4.Calculate Pulse Pressure Multiplier (PPM):

Estimate PPM based on age and gender.

PPM is used in some estimation models.

5.Photoplethysmography (PPG):

Collect PPG signals from the smartphone's camera and flashlight, often by placing a fingertip over the camera and light source.

Process the PPG signals to extract relevant features, such as the pulse waveform.

6.Calculate Ejection Time (ET):

Calculate the ejection time based on the time difference between the peaks of the PPG waveform.

ET is used in some blood pressure estimation methods.

7.Calculate Mean Arterial Pressure (MAP):

Use appropriate formulas that involve SV, BSA, and PPM to estimate Mean Arterial Pressure.

MAP is an important parameter in blood pressure estimation.

8.Estimate Systolic and Diastolic Pressures:

Use calculated parameters (SV, BSA, PPM, ET, etc.) in relevant formulas to estimate systolic and diastolic pressures.

These formulas can vary depending on the methodology being used.

9.Validation and Calibration:

Validate your method against accurate blood pressure measurements obtained from certified medical devices.

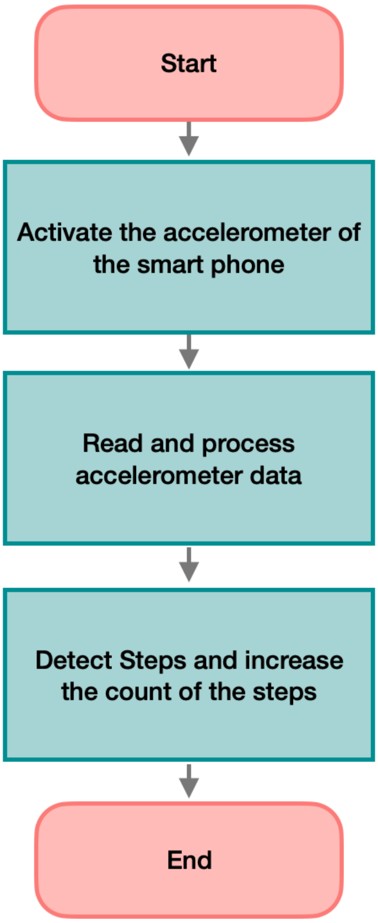
Calibration is crucial to ensure accuracy across different individuals and conditions.

10.Considerations:

Variability: Blood pressure estimation can be influenced by many factors, and individual responses to PPG signals can vary.

Accuracy: Accurate blood pressure measurement requires careful calibration and validation against established methods.

Terrestrial Locomotion


Start
Activate the accelerometer of the smart phone
Read and process accelerometer data
Detect Steps and increase the count of the steps
End

1.Permission and Setup:

Request necessary permissions to access the smartphone's accelerometer data. In most platforms, you'll need to request the SENSOR_ACCELEROMETER permission.

2.Reading Accelerometer Data:

Start the accelerometer sensor to continuously receive data.
Register a listener (callback) to receive accelerometer data updates.
Accelerometer data usually includes acceleration values in the X, Y, and Z directions. These values change as the device moves.

3.Processing Data:

Apply filtering and noise reduction techniques to the raw accelerometer data to obtain more accurate measurements. Common techniques include low-pass filtering and data smoothing.
Convert acceleration values to obtain the magnitude of the acceleration vector. This helps in detecting motion more effectively.

4.Step Detection Algorithm:

Choose or design a step detection algorithm. A common approach is to use peak detection:

Find local peaks in the acceleration magnitude signal.

Apply thresholding and heuristics to differentiate between steps and other movements.

Consider factors like the magnitude of the peaks and the timing between them.

5.Counting Steps:

Implement a step counting logic that increases the step count each time a valid step is detected.

To avoid counting the same step multiple times, introduce a delay or a window in which subsequent steps are ignored.

6.Updating the User Interface:

Display the step count in your app's user interface in real-time.

You can also provide additional features like a progress bar or daily step goals.

7.Background and Battery Optimization:

Implement mechanisms to optimize battery usage, such as adjusting the sensor sampling rate and managing sensor callbacks when the app is in the background.

8.Calibration and Testing:

Calibrate your step detection algorithm for different walking speeds and user behaviors.

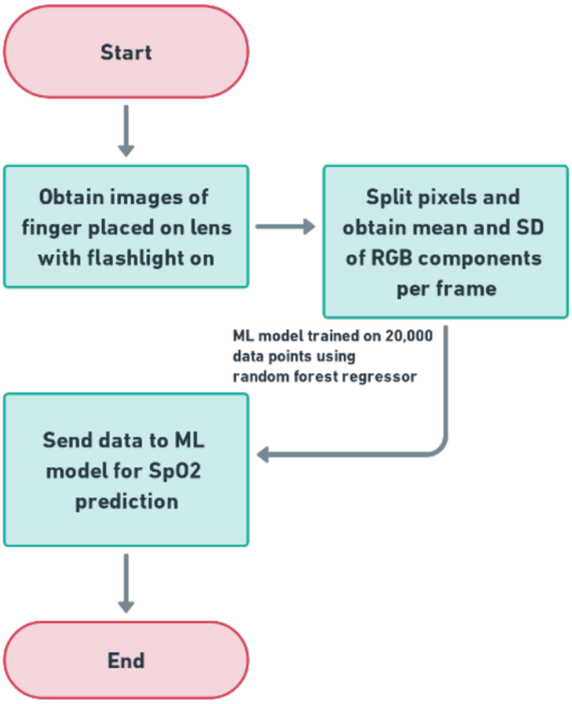
Test your app in various scenarios (e.g., walking, running, slow movement) to ensure accuracy and reliability.

9.Error Handling and User Experience:

Implement error handling mechanisms in case the sensor is not available or fails to provide data.

Provide clear instructions to users on how to ensure accurate step counting (e.g., holding the phone in hand while walking).

Blood Oxygen (SpO2)


Start
Obtain images of finger placed on lens with flashlight on
Split pixels and obtain mean and SD of RGB components per frame
ML model trained on 20,000 data points using random forest regressor
Send data to ML model for SpO2 prediction
End

1.Data Collection:

Gather a dataset of images of fingers placed on the smartphone's camera lens with the flashlight turned on. Capture these images in different lighting conditions, finger positions, and skin tones to ensure a diverse dataset.

2.Image Processing:

Process the images to extract the region of interest (ROI) containing the fingertip. This might involve using image segmentation techniques to focus on the area that provides the best signal for SpO2 measurement.

3.Pixel Analysis:

For each frame, split the pixels within the ROI into their RGB components. Calculate the mean and standard deviation (SD) of the RGB values. These values will serve as features for your machine learning model.

4.Dataset Preparation:

Prepare your dataset with features (mean and SD of RGB components) and target values (actual SpO2 measurements, which you'll need to collect using a certified medical device).

5.Model Training:

Train your machine learning model using the prepared dataset. You mentioned using a random forest regressor, which can be a suitable choice due to its ability to handle non-linear relationships and noisy data.

6.Model Evaluation and Validation:

Evaluate your trained model's performance using metrics such as mean squared error (MSE) or root mean squared error (RMSE) on a validation set. Perform cross-validation to ensure that your model generalizes well to unseen data.

7.Deployment:

Integrate the trained model into your app.

Develop a user-friendly interface that guides users through the process of placing their finger on the camera lens and flashlight.

Provide clear instructions for accurate measurement, such as proper finger placement and lighting conditions.

8.Testing and Validation:

Test your app on a diverse set of users to gather data for further validation.

Compare the app's measurements with those from certified medical devices to assess its accuracy and reliability.

9.Regulatory Considerations:

Keep in mind that any medical-related app might fall under regulatory guidelines depending on your country. Make sure to understand and follow the necessary regulations for medical apps.

10.Continuous Improvement:

Continuously gather feedback and data from users to refine and improve your app's accuracy and user experience.