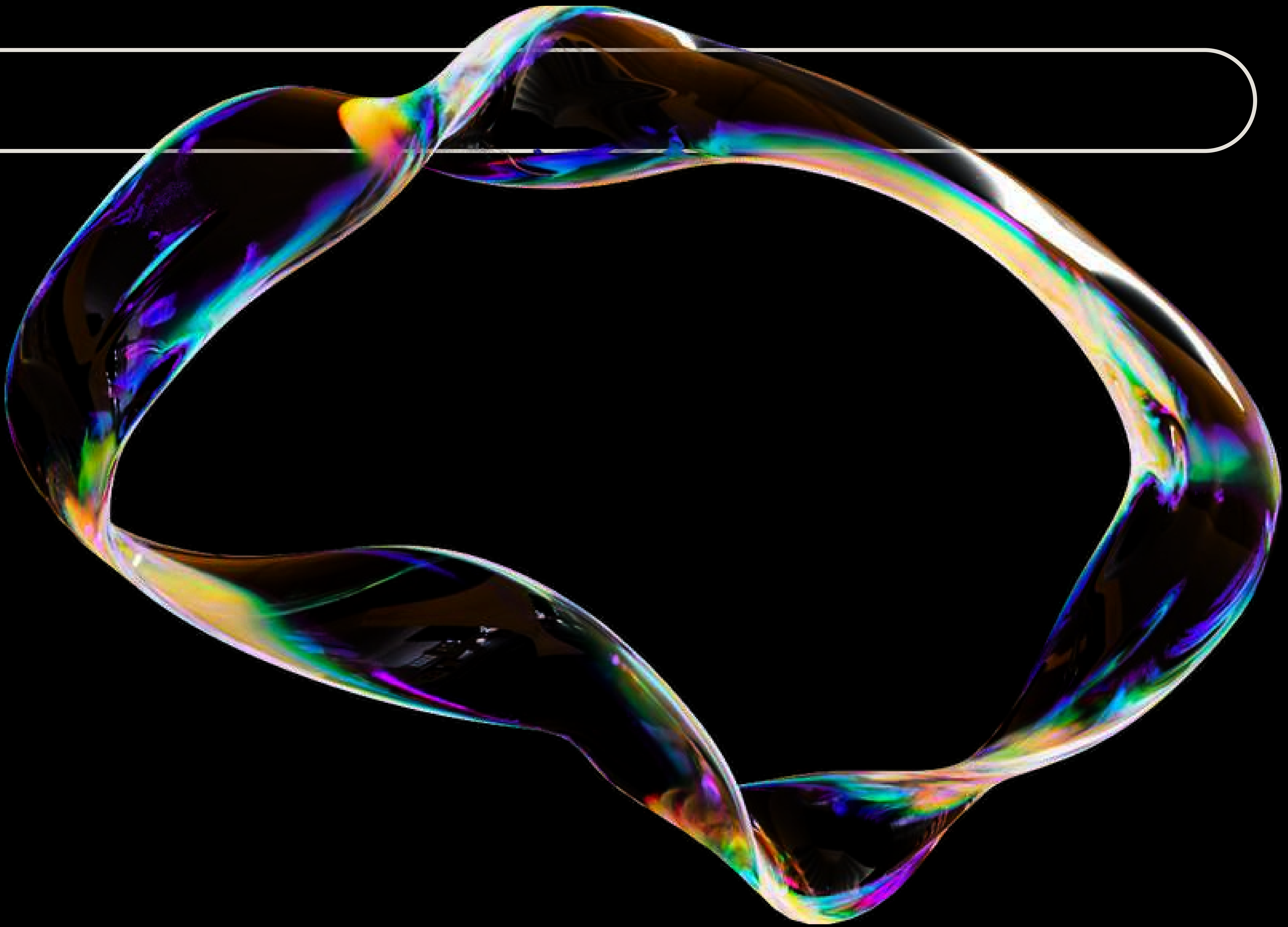
Capstone Project



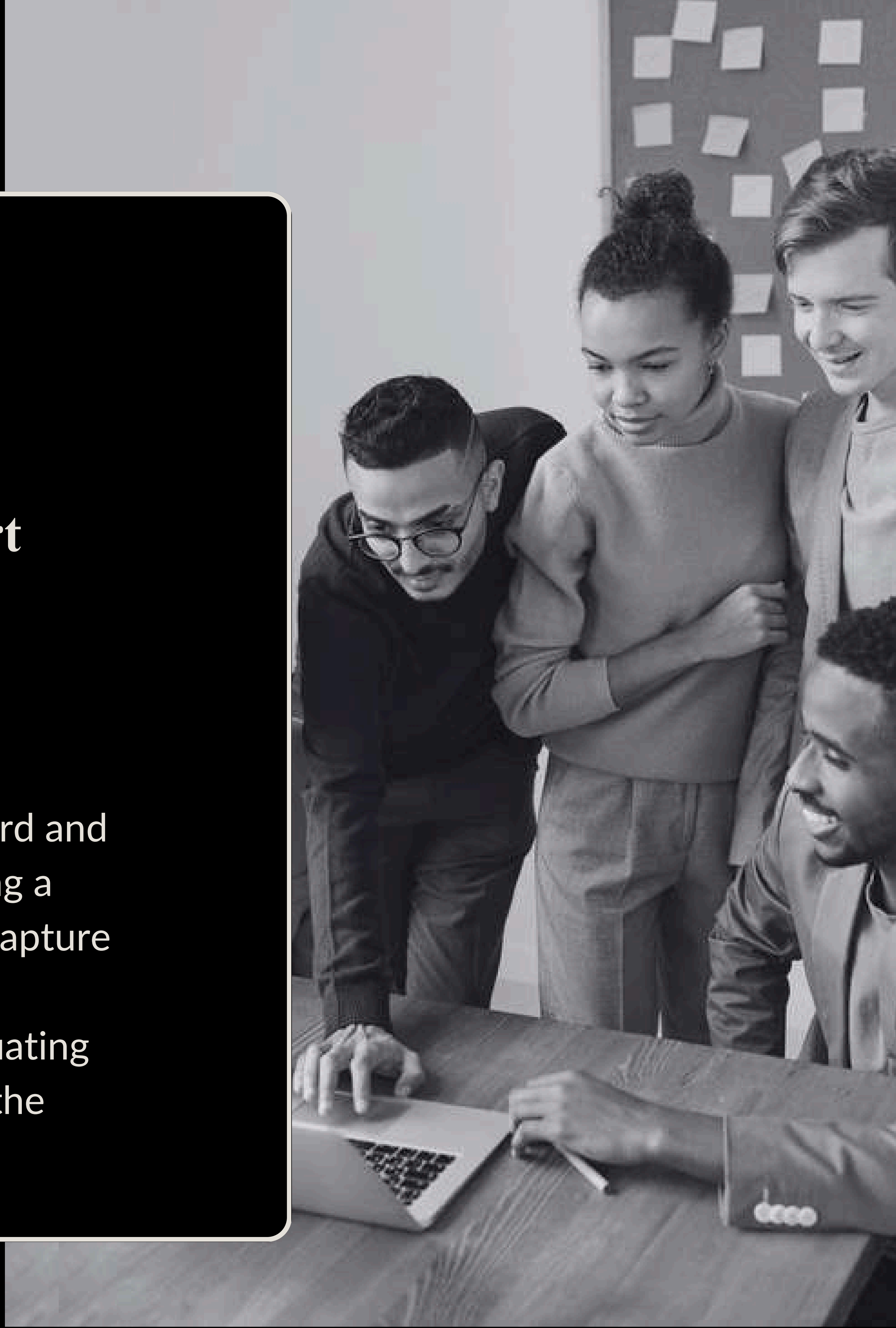
Team Members

Sairam Deepak Varun Manogna Chathura

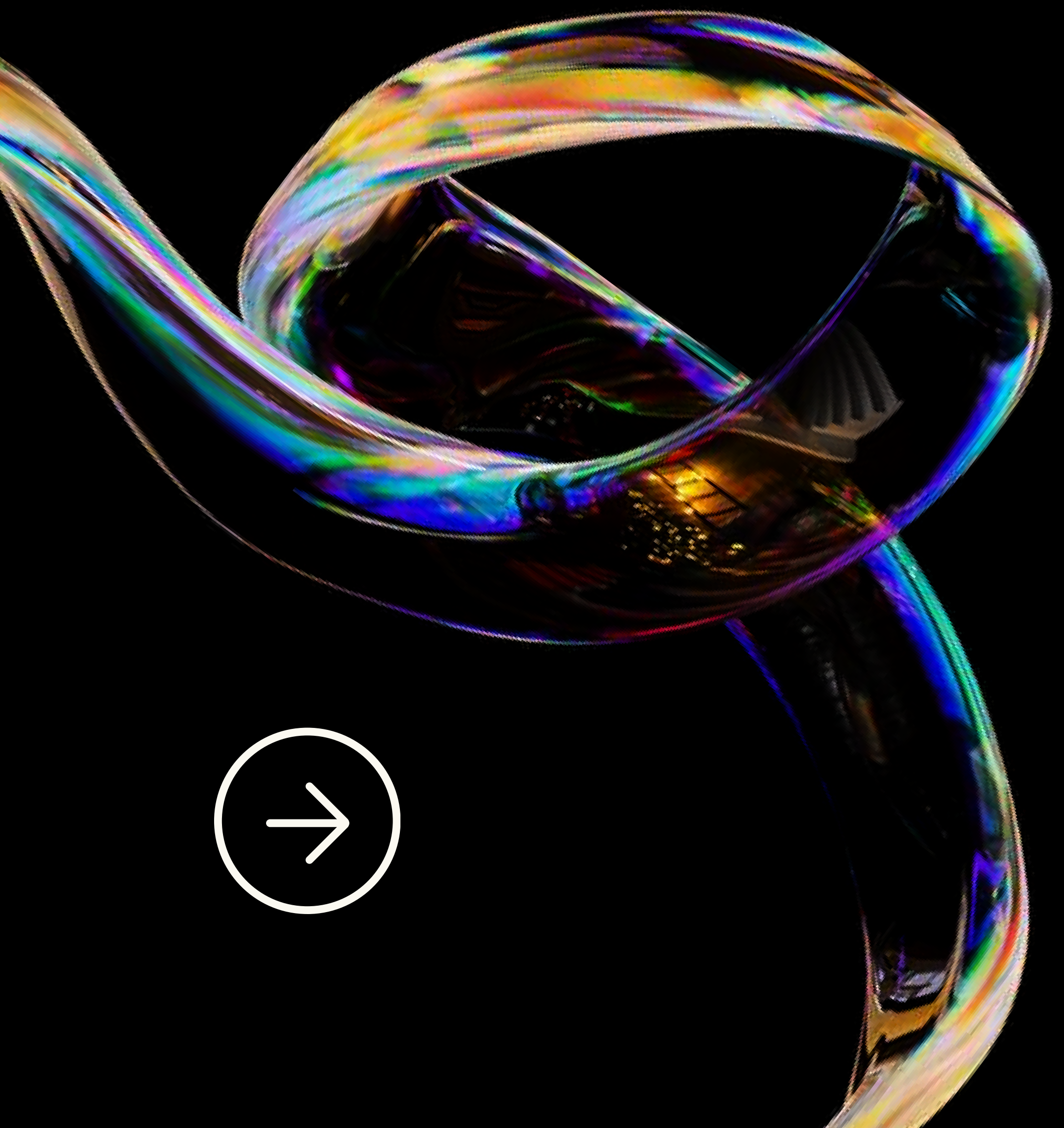
INTRODUCTION

Objective:Phonocardiography Circuit to Record Heart Sounds and Suppress Noise Using Machine Learning Algorithms

Phonocardiography is a diagnostic technique used to record and analyze the sounds produced by the heart. It involves using a specialized microphone or sensor placed on the chest to capture heart sounds, which are then displayed as a graphical representation. This method helps in diagnosing and evaluating heart conditions by providing detailed information about the timing, intensity, and quality of heart sounds.



HOW DOES A DEVICE DETECT NOISE?



- Transducer (Diaphragm)
- Microphone
- Amplifier
- Filters
- ADC(Analog to Digital convertor)
- Noise Cancellation using ML

HARDWARE COMPONENTS

Microphone/Sensor:

ADMP504 MEMS microphone -Using a high-sensitivity piezoelectric microphone.

Role: The microphone is responsible for capturing the acoustic heart sounds and converting them into electrical signals.

Working: MEMS (Micro-Electro-Mechanical Systems) microphones use a tiny diaphragm that vibrates in response to sound waves, causing a change in capacitance, which is converted to an electrical signal.

Operational Amplifier (Op-Amp):

LM358 Dual Operational Amplifier-to boost weak signals from the microphone. Ensuring it has a gain sufficient to amplify heart sounds (20-40 dB).

Role: Amplifies the weak electrical signals from the microphone to a level that can be processed by the ADC.

Working: An operational amplifier takes the difference between its two inputs (inverting and non-inverting) and multiplies it by a gain factor. This amplification makes the heart sound signals strong enough for further processing.

HARDWARE COMPONENTS

Analog-to-Digital Converter (ADC):

ADS1115 16-bit ADC- A 16-bit ADC with a sampling rate of at least 44.1 kHz to accurately digitize the heart sounds.

Role: Converts the amplified analog signal from the microphone into a digital signal that can be processed by the microcontroller.

Working: The ADC samples the analog signal at a specific rate and converts it into a binary number that represents the signal amplitude at each sample point.

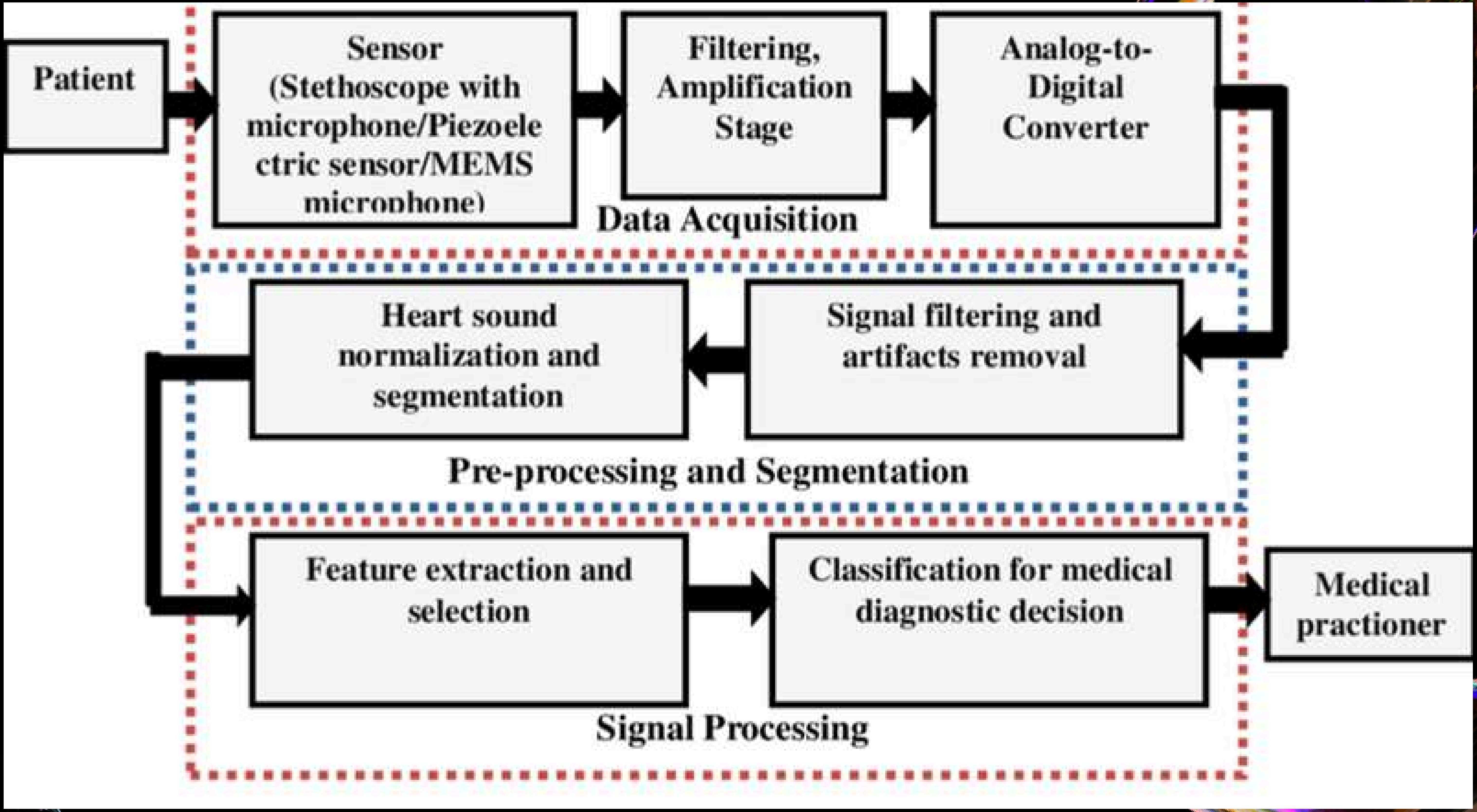
Microcontroller:

Arduino Uno will be used as the microcontroller.

Role: The microcontroller handles data acquisition from the ADC, processes the signals, and interfaces with any external software (e.g., for machine learning).

Working: It runs a program that continuously reads digital data from the ADC, processes it in real-time, and sends it to a computer for further analysis or storage

BLOCK DIAGRAM OF THE CIRCUIT



Patient
Sensor (Stethoscope with microphone/Piezoelectric sensor/MEMS microphone)
Filtering, Amplification Stage
Analog-to-Digital Converter
Data Acquisition
Signal filtering and artifacts removal
Heart sound normalization and segmentation
Pre-processing and Segmentation
Feature extraction and selection
Classification for medical diagnostic decision
Signal Processing
Medical practioner

STEPS INVOLVED IN DESIGNING AND DEVELOPING A CIRCUIT

- After procurement of desired components we will proceed with the circuit design and assembly .Design the interface circuit to connect the sensor to the pre-amplifier.Determine Desired Gain:Decide how much amplification is needed based on the strength of the heart sound signals and the requirements of the subsequent stages (filtering and ADC).

- Use required Op-Amp device the one with the low noise(TL072).Design the preamplifier circuit with appropriate gain settings.Validate the preamplifier's performance using a signal generator.
- Design filters with cutoff frequencies that effectively isolate the heart sound range and then Assemble and connect filters to the preamplifier output.

- **Choose a cutoff frequency just below the lowest frequency of interest in heart sounds to remove low-frequency noise for high pass filter.**
- **Choose a cutoff frequency just above the highest frequency of interest in heart sounds to remove high-frequency noise for low pass filter.**

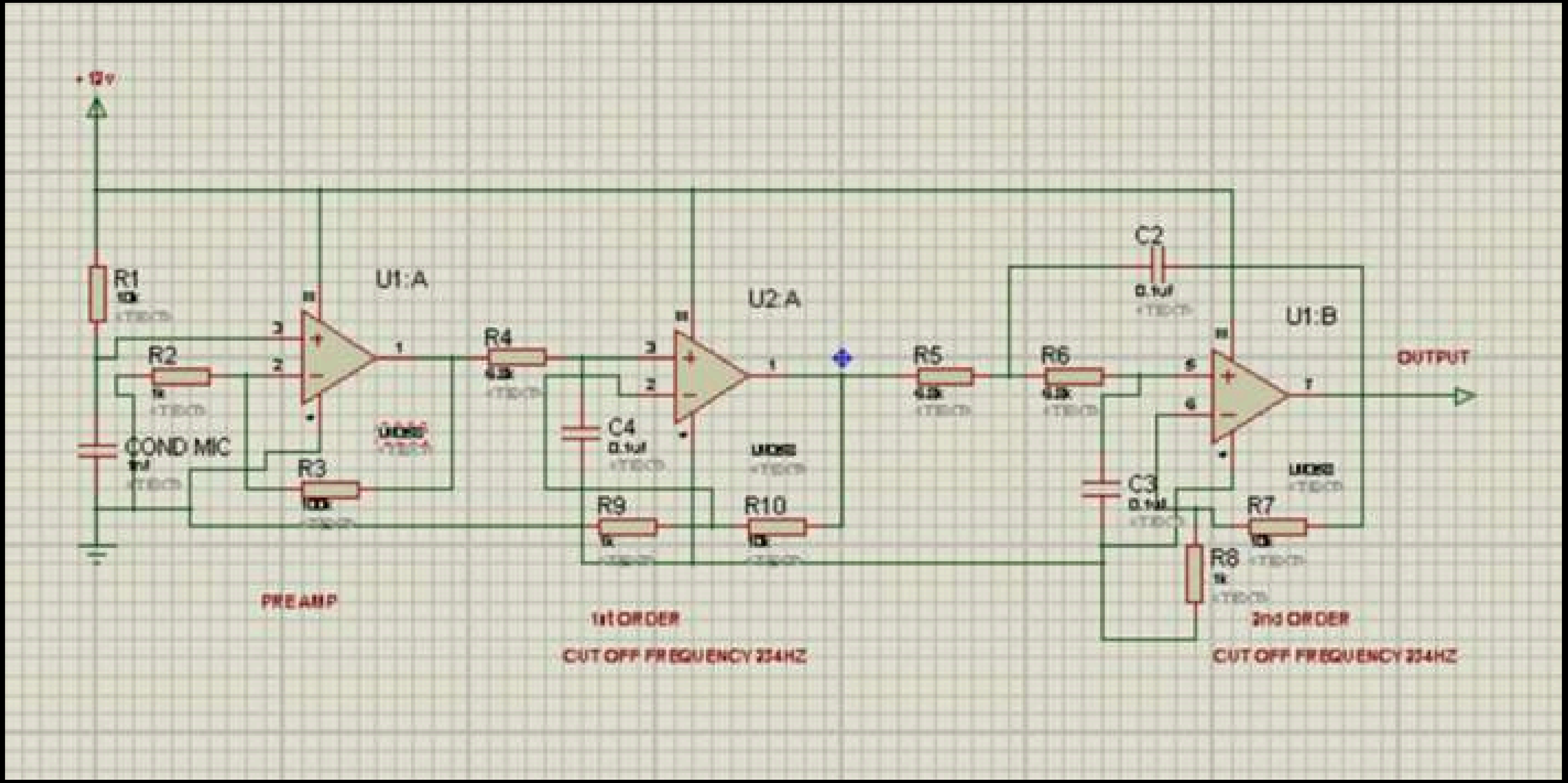
STEPS INVOLVED IN DESIGNING AND DEVELOPING A CIRCUIT

- Use multiple filter stages if necessary to achieve the desired frequency response.
- Ensure that the filters effectively attenuate unwanted frequencies while preserving the heart sound frequencies

- The preamplifier amplifies weak heart sounds to a usable level.
- Filters clean the signal, making it suitable for ADC conversion and further processing.Iit rejects the external noices.
- To digitize the analog heart sound signals ADC is used.

- **The output from the filtering stage is connected to the ADC input of the microcontroller.**
- **And we will Ensure proper grounding and signal integrity by minimizing noise and interference in the connections.**

CIRCUIT DESIGN AND EXPLANATION



1. MICROPHONE (MEMS MICROPHONE

Connection:

- The MEMS microphone is connected to the power supply and ground.
- The output of the microphone is connected to the non-inverting input of the op-amp.

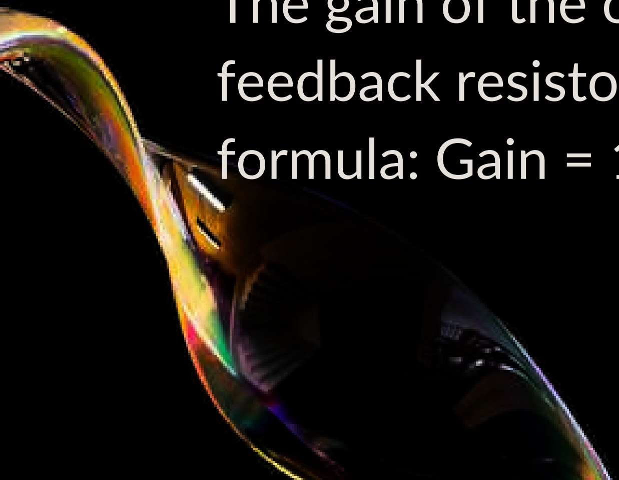
The microphone converts the acoustic energy from heartbeats into a small voltage signal.MEMS microphones have a diaphragm that vibrates in response to sound waves, causing changes in capacitance that generate an electrical signal.

2. OPERATIONAL AMPLIFIER (LM358)

Connection:

- The non-inverting input of the op-amp is connected to the microphone output.
- A feedback resistor (Rf) is connected between the output and inverting input of the op-amp, and an input resistor (Rin) is connected to the inverting input.
- The output of the op-amp is connected to the input of the ADC.
- Power supply (typically 5V) and ground are connected to the op-amp.

The op-amp amplifies the weak signal from the microphone to a level that the ADC can effectively digitize.
The gain of the op-amp is determined by the ratio of the feedback resistor (Rf) to the input resistor (Rin), following the formula: Gain = 1 + (Rf/Rin).



AMPLIFIER

AMPLIFIER

Without distorting the signal , we increase the weak signal into a level suitable for further processing.This increases the power of the signal.

COMPONENT 1: INPUT STAGE

- This takes small voltage signal from transducer.
- This has high impedence value.

COMPONENT 2: GAIN STAGE

- Here actual amplification occurs.
- Uses transistors or OP-AMPS to increase the signal amplitude.

A(gain) = A(output)/A(input)

COMPONENT 3: OUTPUT STAGE

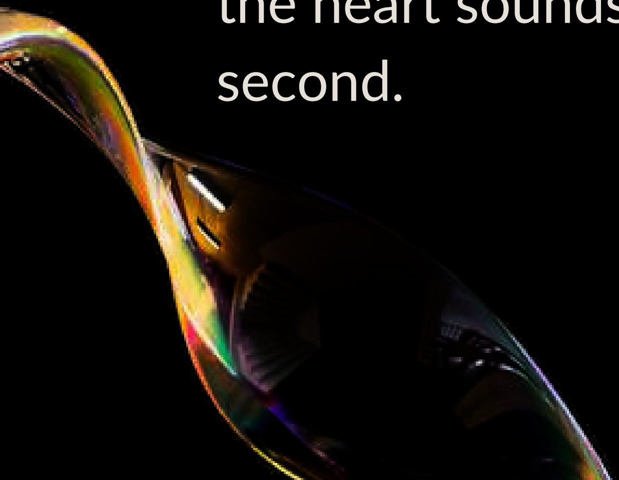
- Delivers the amplitude signal to the next part.
- This stage usually has low output impedance to efficiently drive the load.

3. ANALOG-TO-DIGITAL CONVERTER

Connection:

- The output of the op-amp is connected to one of the analog input channels of the ADC.
- The ADC is connected to the microcontroller via the I2C interface (SDA and SCL lines).
- Power supply and ground are connected to the ADC.

The ADC samples the analog signal at a specified rate and converts it into a digital value.
The ADS1115 is a 16-bit ADC, which provides high precision, meaning it can distinguish 65,536 levels of signal amplitude.
The sampling rate is set based on the expected frequency of the heart sounds, typically around 500 to 860 samples per second.



4. MICROCONTROLLER (ARDUINO UNO)

Connection:

- The I2C lines (SDA and SCL) from the ADC are connected to the corresponding pins on the microcontroller.
- The microcontroller is connected to a power source (typically USB for Arduino) and ground.
- Optionally, a connection is made from the microcontroller to a computer or display module for data visualization.

The microcontroller reads the digital signal from the ADC, processes it, and stores or sends the data for further analysis.

CIRCUIT WORKING PROCESS

a. Signal Capture and Amplification

- The MEMS microphone captures the heart sound and generates a small voltage signal.
- This small signal is then fed into the op-amp, where it is amplified to a level suitable for digital conversion.

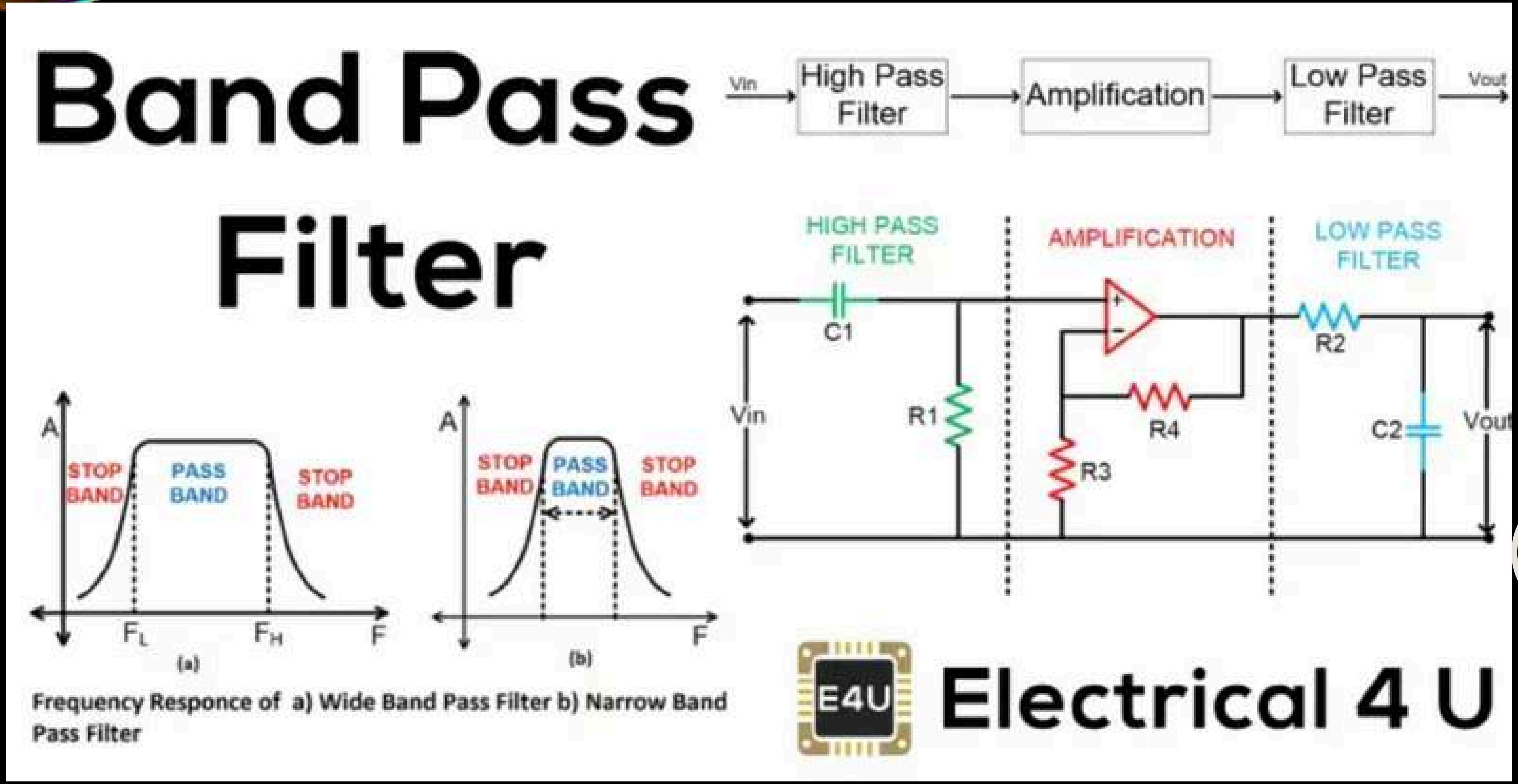
b. Analog-to-Digital Conversion

- The amplified analog signal is sent to the ADC, which converts it into a 16-bit digital signal.
- This digital signal represents the amplitude of the heart sound at each sample point, capturing the waveform in a format that can be processed by digital systems.

c. Data Processing and Output

- The microcontroller reads the digital data from the ADC, processes it and stores or transmits the processed data for further analysis.
- The processed data can be visualized as a waveform on a computer or used as input for machine learning algorithms to classify the heart sounds.

SIGNAL FILTERING



BAND PASS FILTERING

- Objective: To filter out frequencies that are not part of the heart sound signal.
- Procedure: Use a bandpass filter that typically passes frequencies between 20 Hz and 2,000 Hz, which covers the frequency range of heart sounds while removing high-frequency noise (like electronic noise) and low-frequency noise (like breathing)

Adaptive Filtering

- Objective: To dynamically adjust the filter parameters based on the detected noise level.
  Procedure: Adaptive filters can change their characteristics in real-time to better filter out noise, especially when the noise characteristics change (e.g., a patient moves).

ANALOG TO DIGITAL

- Analog Signal
- Sampling
- Quanatization
- Encoding
- Passing the Digital Signal

Analog to Digital Conversion

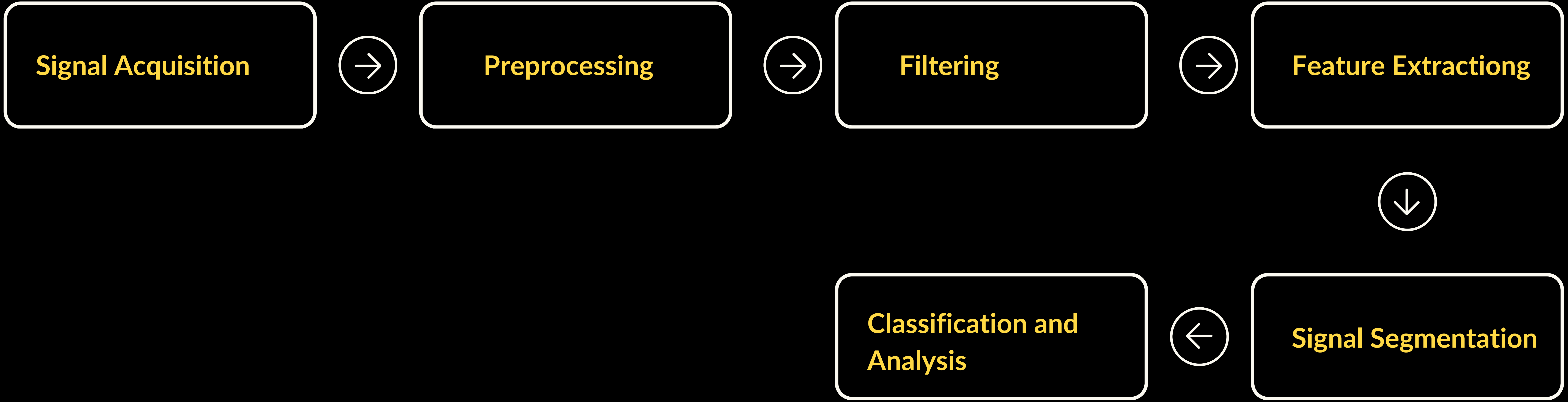
Sampling: The filtered analog signal is sampled at regular intervals (e.g., 4,000 Hz) to convert it into a discrete-time signal.

Quantization: Each sampled value is mapped to the nearest level within a set range, creating a discrete digital signal.

Encoding: The quantized values are converted into binary numbers (e.g., 16-bit) for digital representation

Digital Signal: The resulting digital signal can now be processed, analyzed, and stored for further use.

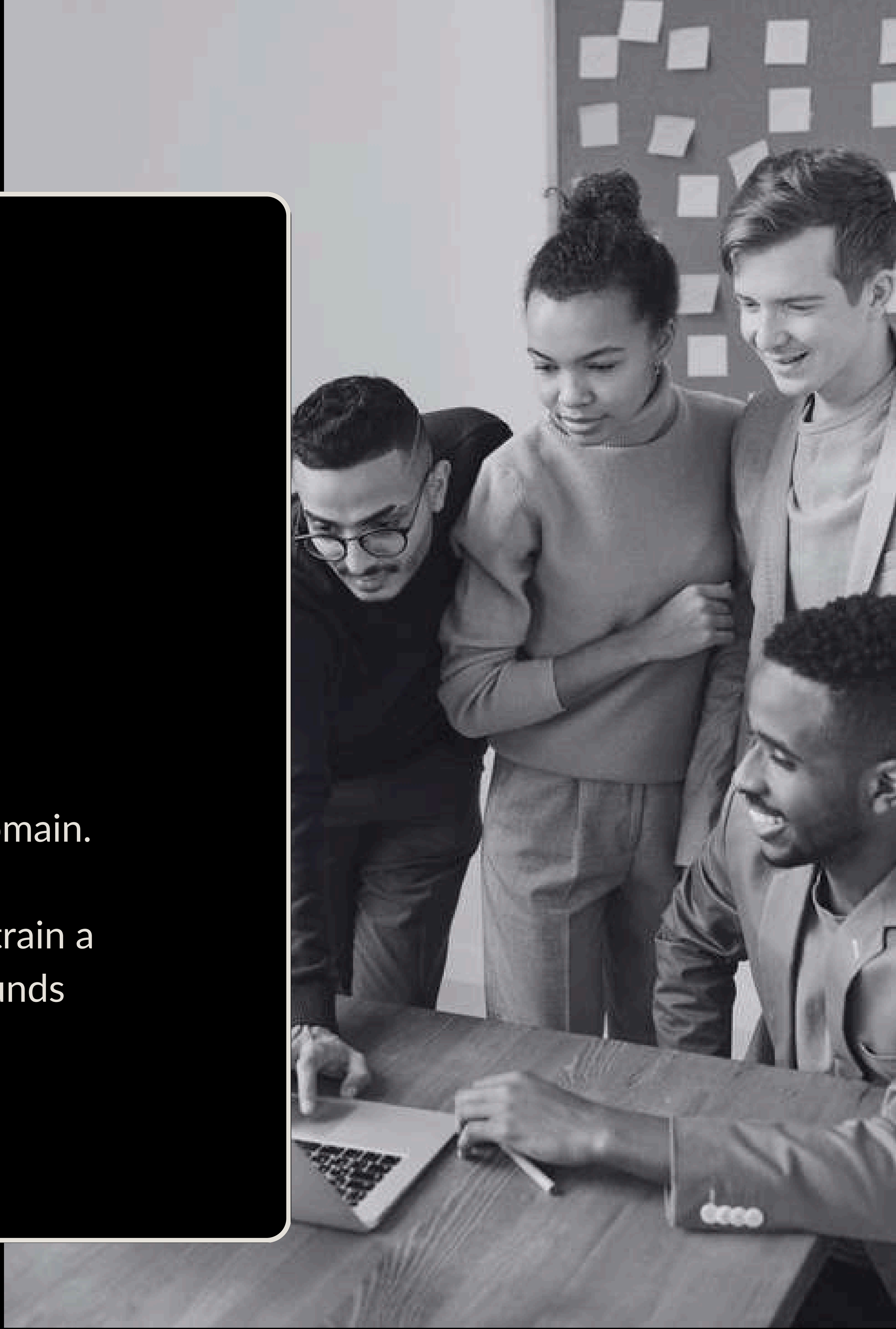
SIGNAL PROCESSING



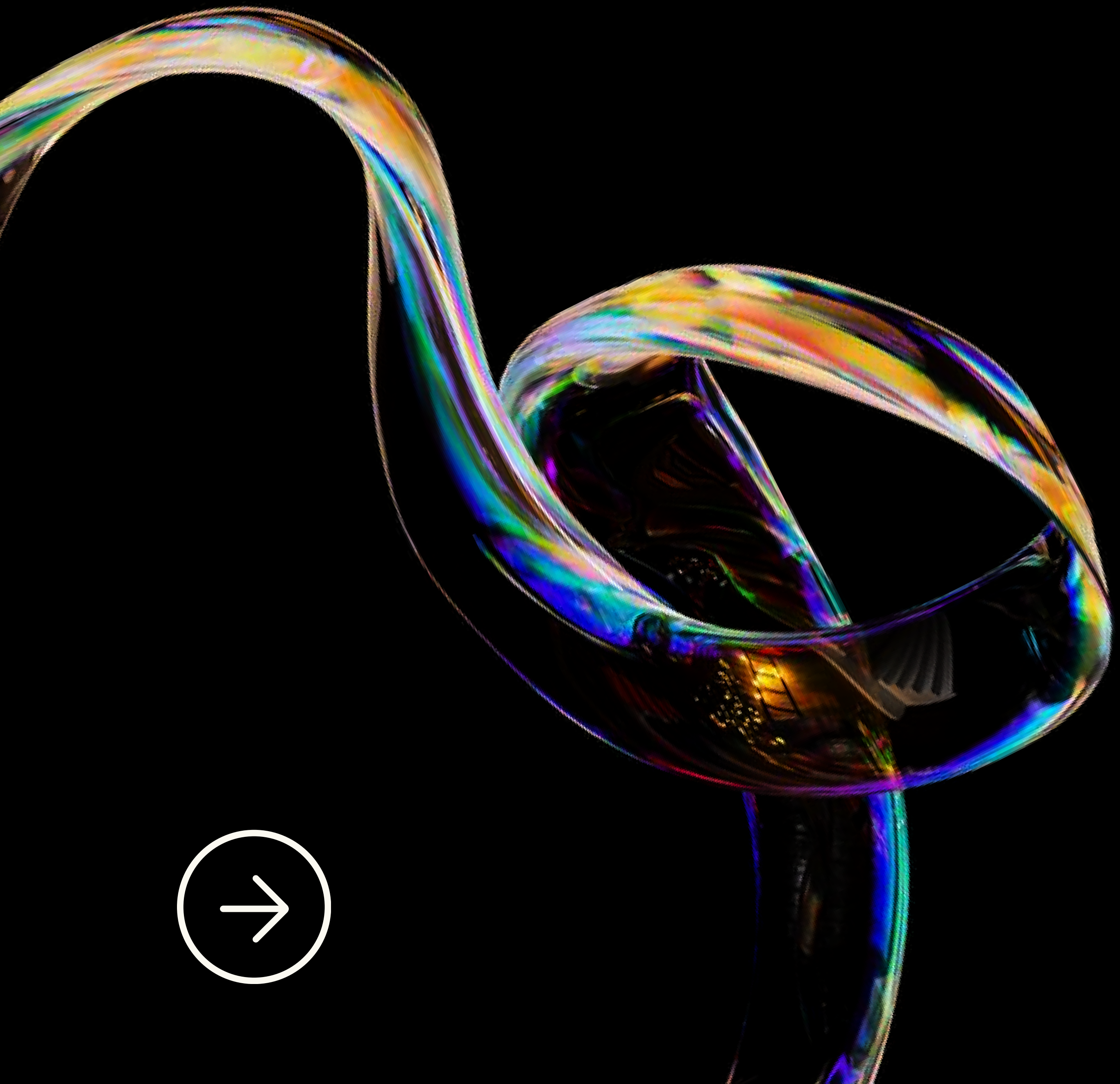
NOICE CANCELLATION WITH ML

Denoising of Phonocardiogram Signals Using Time-Frequency Analysis and U-Nets

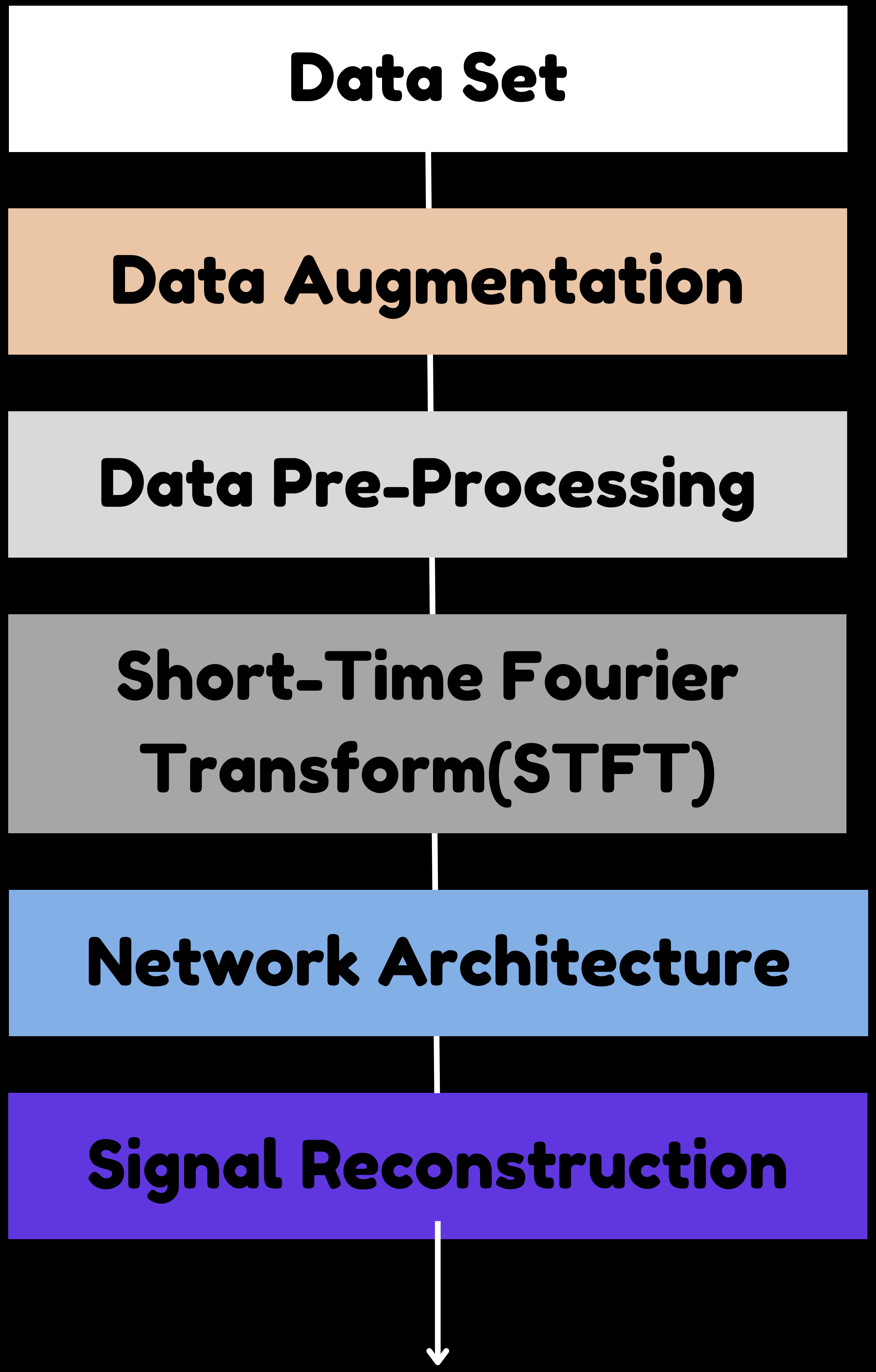
- The denoising process occurs in the time–frequency domain. More specifically, we compute the Short-Time Fourier Transform (STFT) of the contaminated PCG signal and train a U-Net to recognize normal and pathological cardiac sounds from noise.



FOUR DIFFERENT NOISE SOURCES:



- Additive white Gaussian noise (AWGN)
- Additive pink Gaussian noise (APGN)
- PhysioNet Noise
- Brownian Noise



Methodology

DATASET AND DATA AUGMENTATION

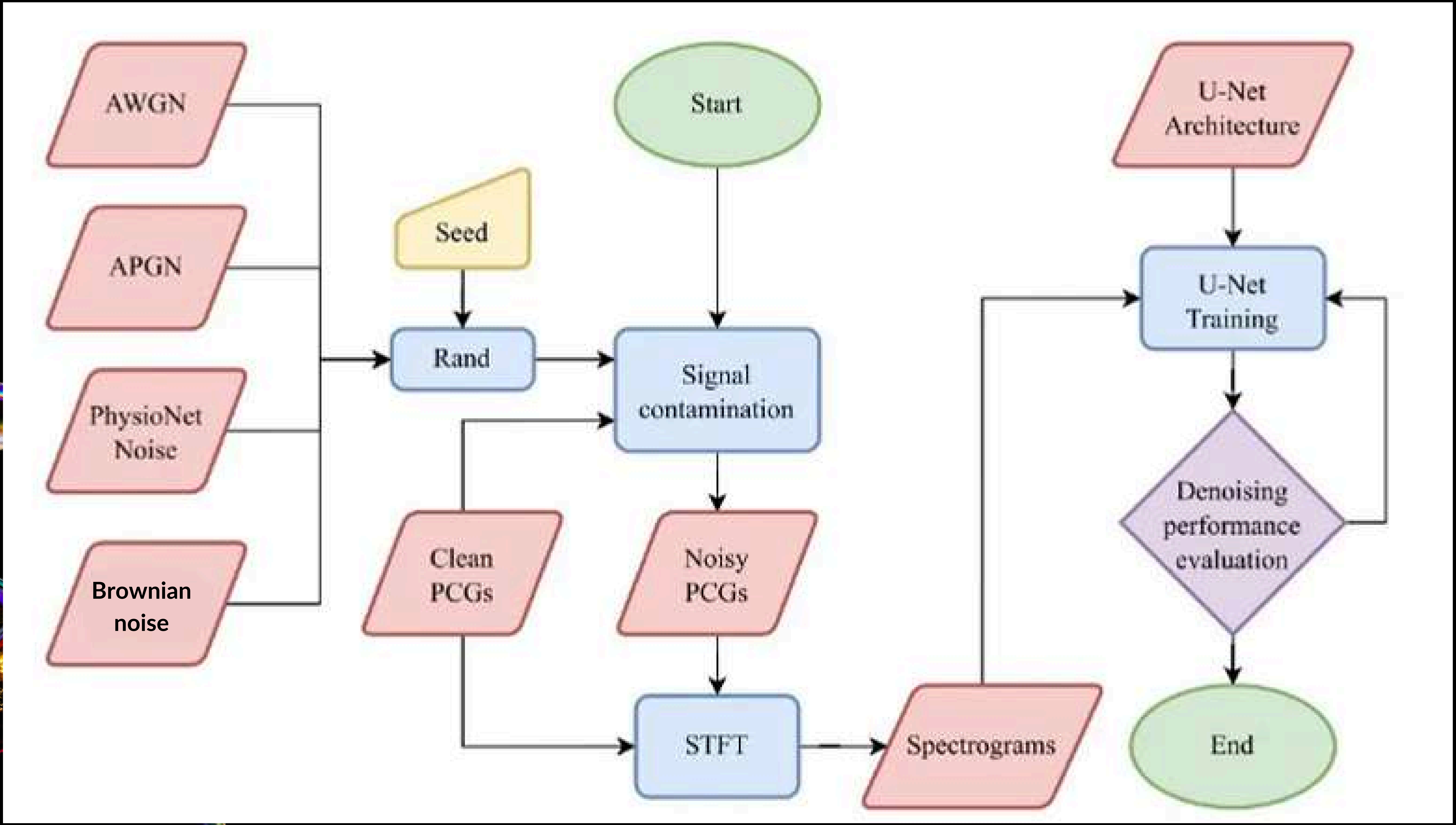
- Datasets are available on internet.
- Our method uses a well-established technique called Analysis by-Synthesis/Overlap-Add for data augmentation(ABS/OLA)

DATA PRE-PROCESSING

- This work used 10-fold cross-validation; the total signals were split into ten subsets.
- Then each subset was utilized for training and testing, 90% were utilized for training the network, and the other 10% to perform validation.
- This process was repeated ten times, selecting a new validation subset every time.
- The noise databases were also split similarly to the clean PCG dataset. That is, 90% of the noise recordings were utilized to contaminate the training signals and 10% as test signals during the denoising validation.

SHORT-TIME FOURIER TRANSFORM(STFT)

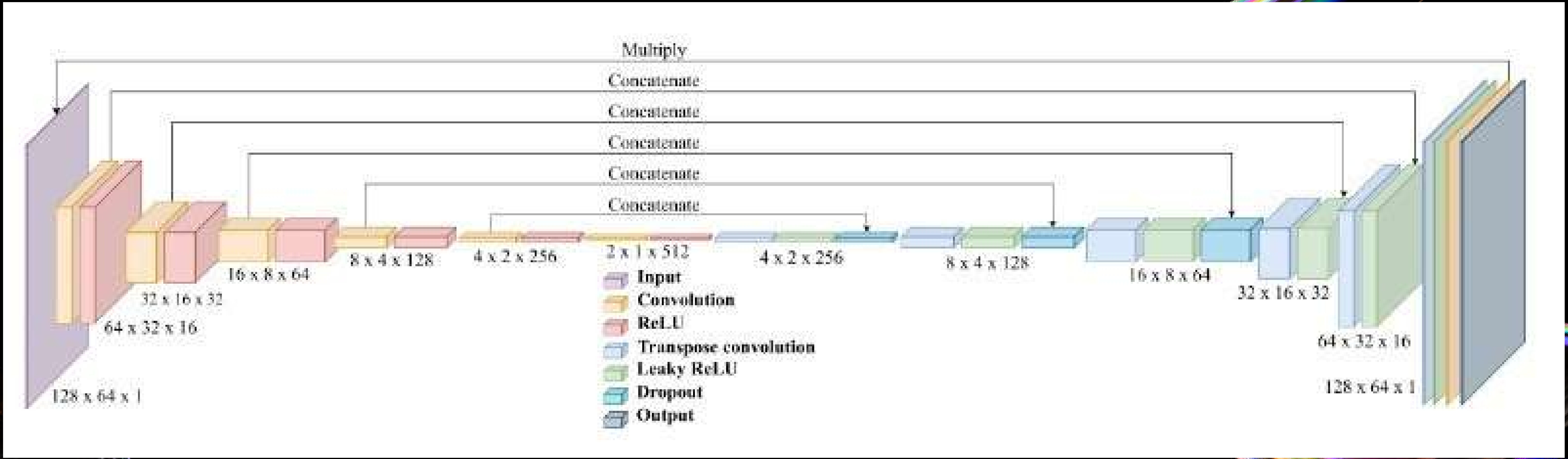
- We suggest utilizing the short-time FourierTransform (STFT) to generate the spectrograms of the audio signals and to use them as images to train the network.
- The spectograms for both clean and contaminated signals were necessary to train the network.



Flowchart diagram of the PCG audio pre-processing and network training

NETWORK ARCHITECTURE

- In this work, we use a U-Net architecture as shown below figure
- The input and output images of the network have a size of (128, 64, 1)



SIGNAL RECONSTRUCTION

- The phase of each spectrogram obtained during the STFT calculation is stored for reconstruction later on.
- The spectrograms then proceeded to be processed with the trained network to obtain the mask, then multiplied element by element with the original spectrograms; then using the phase to be reconstructed into audios with the inverse short-time Fourier Transform and combine each audio partition into a denoised cardiac sound signal.

Timeline

Weeks 1-2: Initial Research & Planning:

- Researching relevant datasets for heart sound classification and denoising, identifying suitable machine learning frameworks and Drafting initial circuit schematics
- Ordering Components

Weeks 3-4: Circuit Design, Simulation, & ML Data Collection;

- Designing the phonocardiography circuit including microphones, preamplifiers, and analog-to-digital converters (ADC). and Simulating the circuit design using appropriate tools.
- Collecting and preprocessing a dataset of heart sounds with and without noise.
- Implementing basic data preprocessing techniques, such as normalization and segmentation.

Weeks 5-6: Breadboard Testing & Initial ML Model Development

Assembing and testing the circuit on a breadboard, maki adjustments as needed.

- Verifying the performance of the preamplifier and filter wi heart sound recordings.
- Developing an initial ML model for noise reduction, focusing model architecture and training.
- Training the model on the preprocessed dataset and evaluate performance.

Weeks 7-8: PCB Design & ML Model Training

- Finalizing the PCB layout based on breadboard results and prepare for manufacturing.
- Continuation of training and refining the ML model, experimenting with different architectures and hyperparameters.
- Testing the model on real-world noisy heart sound data, improving its accuracy and robustness.

Weeks 9-10: PCB Assembly & ML Model Testing:

- Assembling the PCB and test each circuit stage to ensure it works as expected.
- Integrating the ML model into the software pipeline, allowing for real-time noise reduction.
- Begin testing the ML model with data acquired from the assembled PCB.

Weeks 11-12: Microcontroller Integration & Parallel ML Model Tuning:

- Writing and testing the microcontroller code for data acquisition, storage, and communication with the ML processing unit.
- Continuing to refining the ML model based on feedback from real-time tests, focusing on reducing latency and improving accuracy.
- Optimizing the model for deployment on the chosen hardware platform

Weeks 13-14: Hardware-Software Integration & ML Testing:

- Integrating the hardware components with the software, ensuring smooth data flow from signal acquisition to ML processing.
- Testing the integrated system, running the ML model on real-time data to evaluate its performance in various noise conditions.
- Making necessary adjustments to both the hardware and software to ensure optimal performance.

Weeks 15-16: Final Refinement & Calibration:

System Optimization:

- Fine-tuning the entire system, optimizing the hardware and software for different environments (quiet vs. noisy).
- Calibrating the device to ensure consistent performance across different users.

ML Model Finalization:

- Finalizing the ML model, ensuring it is robust, accurate, and efficient.
- Conducting thorough testing and validation, comparing the denoised output with ground truth data.

Weeks 17-18: Final Testing & Documentation

- Conducting extensive testing to validate the entire system, ensuring it meets all project requirements.
- Collecting data on the system’s performance in real-world conditions and make final adjustments if necessary.
- Documenting the entire project, including circuit designs, software code, ML model development, testing results, and user manuals.

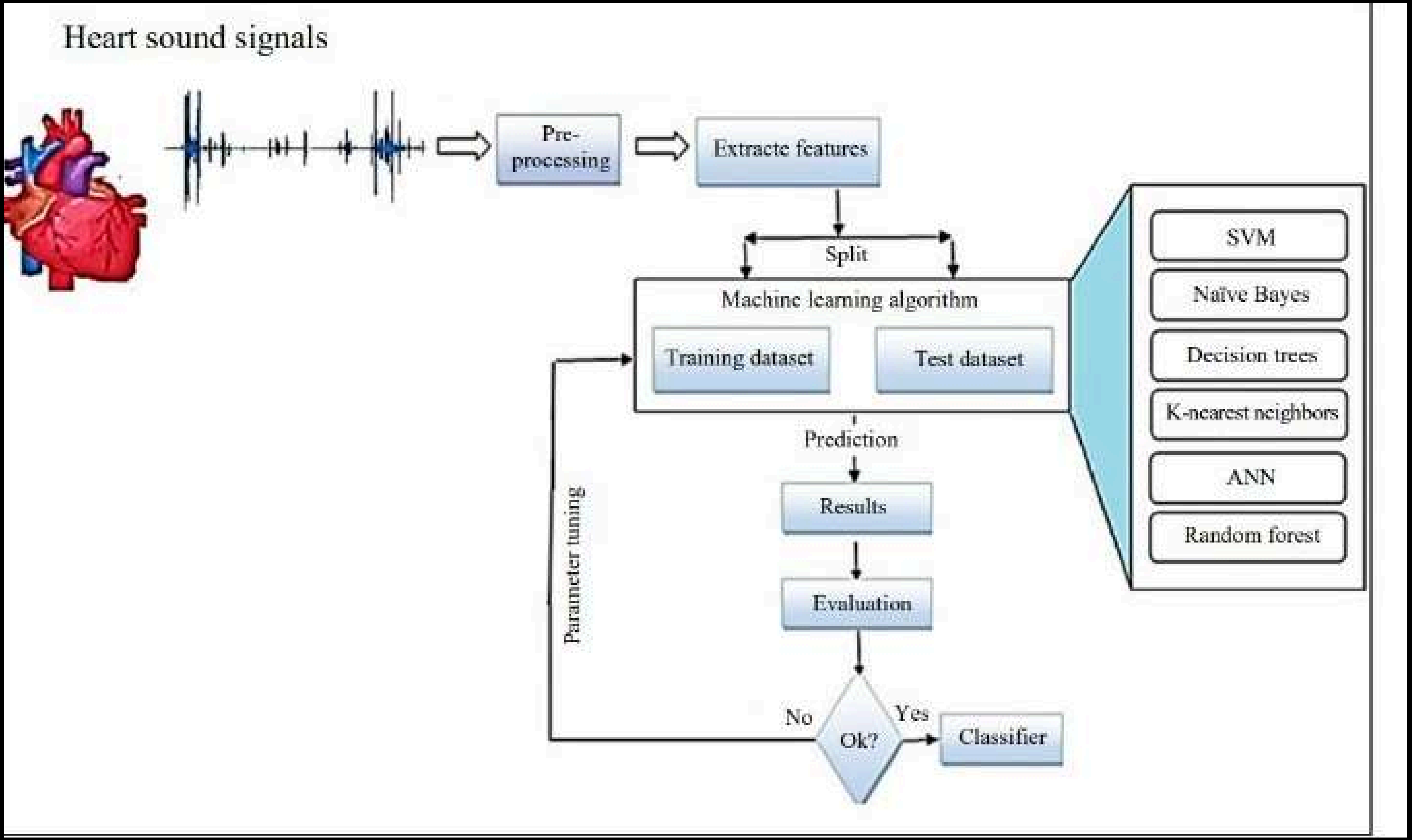
Weeks 19-20: User Testing & Project Presentation

- User Testing:Testing the device with different users, gathering feedback on usability, comfort, and accuracy.
- Making any last-minute changes based on user feedback.
- Presenting the project

THANK YOU

ML Algorithms For Automatic Classifications

BEYOND THE SCOPE OF THE GIVEN PROJECT



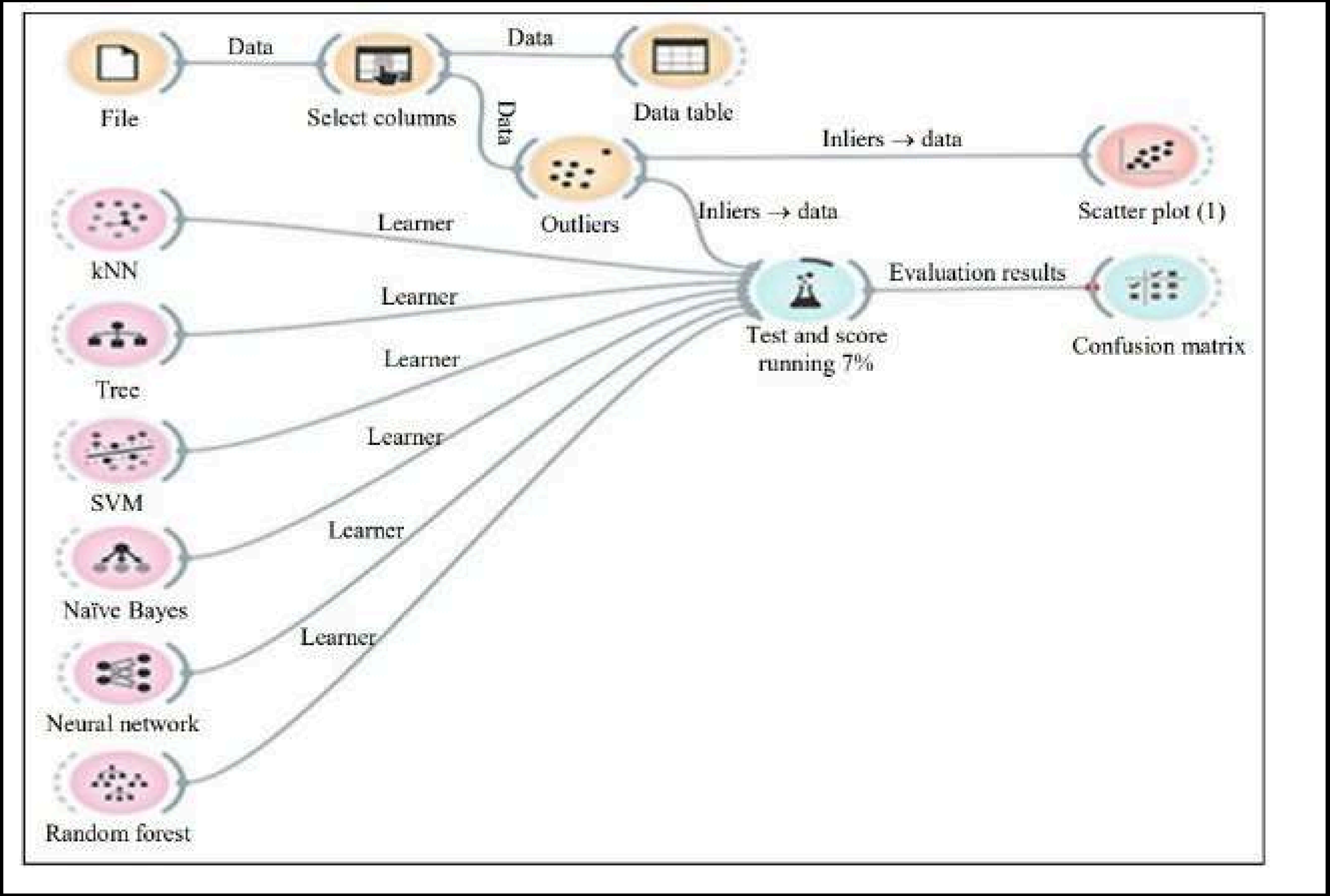
Flow diagram for Proposed Method

FEATURE EXTRACTION

- Mel Frequency Cepstral Coefficient (i.e., MFCCs), Filter Bank coefficients (i.e., FBANK), delta MFCC and combined of MFCC with FBANK were considered as the features (i.e., attributes with examples) separately to detect the heart disease.
- A software-based simulation using Python source code was performed for features extraction and then converted to .CSV (i.e., comma delimited file)

Classification of Cardiac sounds

- Normal
- Murmur
- Extra heart sounds
- Artifacts



ML Model for Heartbeat sound classification