1. Dataset Selection & Loading

I used the **CWRU bearing dataset**, which contains real vibration data collected from motors with normal and faulty bearings. The files are `.mat` format and were recorded at **12,000 Hz sampling rate**.

---

2. Signal Segmentation

Each raw vibration signal was split into **5-second windows**, resulting in segments with **60,000 data points** each. This helped break down the long signals into manageable chunks.

---

3. Spectrogram Generation

I converted each segment into a **2D spectrogram** (shape: 128×128), which visually represents how the frequency content of the signal changes over time — like turning sound into an image.

---

4. Labeling the Data

The type of fault was extracted from the filename:

- `IR` → Inner Race Fault
- `OR` → Outer Race Fault
- `B0` → Ball Fault
- `97` or `Normal` → Healthy

These were mapped to labels: 1, 2, 3, and 0 respectively.

---

5. Data Augmentation

To increase the dataset size and improve robustness:

- I **added Gaussian noise** (simulates sensor noise)
- I **applied time shift (rolling)** to simulate slight timing changes

This tripled the amount of data.

The number of samples is 444

---

6. CNN Model

I built a **Convolutional Neural Network (CNN)** with this architecture:

```
Input: (128, 128, 1)  → grayscale spectrogram
Conv2D(32 filters, 3x3) + ReLU
MaxPooling2D(2x2)
Conv2D(64 filters, 3x3) + ReLU
MaxPooling2D(2x2)
Flatten
Dense(64) + ReLU
Dense(4) + Softmax  → (4 classes)
```

> **ReLU** stands for *Rectified Linear Unit*. It introduces non-linearity by turning all negative values into 0.

---

7. Training and Evaluation

To check how well the model generalizes, I used **5-fold stratified cross-validation**. This means:

- The data was split into 5 parts.
- In each round, 4 parts were used for training and 1 for testing.
- This was repeated 5 times so that each part got tested once.

> I tracked accuracy and label distribution in each fold.

8. Results (Sample)

These are example accuracies I achieved in different folds:

- Fold 1: **98.8%**
- Fold 2: **97.7%**
- Fold 3: **100%**
- Fold 4: **99%**
- Fold 5: **37%** *(possibly due to class imbalance)*

Average Accuracy: ~87%

---

9. Why This Matters

This project simulates how **smart factories** or **e-government maintenance systems** can automatically detect faults in rotating machines, reducing the chances of breakdowns.

🧾 Reference Paper

I studied and followed the concepts from this IEEE paper:
"Bearing Fault Detection and Diagnosis Using CWRU Dataset With Deep Learning Approaches"
[Neupane & Seok, 2020, IEEE Access]