Here’s the **full detailed version** of your Proof of Concept report, ready for copy-pasting into Word (includes all steps, explanations, code, and results).

**Lung Cancer Detection via Cough Spectrograms – Proof of Concept Report**

**Student:** Saud Abdullah  
**Program:** MSc Computer Science  
**Date:** July 2025

**1. Introduction**

This report documents the **Proof of Concept (PoC)** stage for the lung cancer detection project using **cough audio recordings converted into spectrogram images**.

The goal of this stage was to:

* Verify that the **data processing pipeline** (audio → spectrograms) works correctly.
* Confirm that **metadata creation** and **dataset handling** are functional.
* Train a **ResNet-18 baseline model** on a small subset of the data to validate the **end-to-end workflow**.
* Produce **evaluation metrics (confusion matrix)** to confirm correct classification.

This PoC serves as a foundation for scaling up to the full dataset and eventually extending the model to **multi-class classification** (multiple respiratory diseases).

**2. Dataset Preparation**

Four main data sources were used:

1. **Coswara Dataset** – Healthy cough and breath recordings (audio).
2. **COUGHVID Dataset** – Healthy cough audio samples with metadata.
3. **Kaggle CSI Dataset** – Contained **30 lung cancer cough sound images (CSI)** and 24 normal samples.
4. (Later for expansion) Additional Kaggle-provided categories for multi-class classification (COVID-19, Pneumonia, etc.).

**2.1 Folder Organization**

All data was extracted and organized into the following structure for preprocessing:

data/

├── raw/ # Original datasets

├── processed/ # Converted spectrogram images

│ ├── healthy\_coswara/

│ ├── healthy\_coughvid/

│ ├── healthy\_kaggle/

│ └── cancer/

**2.2 Spectrogram Generation**

All .wav files from Coswara and COUGHVID were converted into **256×256 Log-Mel spectrograms** using the librosa library.

* Corrupted or unreadable .wav files were skipped and logged separately.
* The final processed dataset consisted of **3,601 total samples** (including healthy and cancer).

**3. Metadata Creation**

A single **metadata.csv** file was generated, containing:

* **Filepath** (absolute path to spectrogram image)
* **Label** (healthy or cancer)
* **Source** (Coswara, COUGHVID, Kaggle)

A PoC subset was then sampled from this:

* **30 cancer samples** (from Kaggle CSI)
* **50 healthy samples** (randomly drawn from all healthy sources)

This subset (80 total samples) was saved as **metadata\_poc.csv** for testing the pipeline.

**4. Proof of Concept Model**

**4.1 Why ResNet-18?**

For a quick validation, **ResNet-18** (a smaller, lightweight CNN) was chosen because:

* It is pre-trained on ImageNet, allowing faster convergence.
* It runs quickly on CPU (no GPU dependency for PoC).
* It validates the workflow without long training times.

**4.2 Model Modifications**

* The **first convolution layer** was kept for 3-channel RGB (as spectrograms are saved as RGB images).
* The **final fully connected layer** was replaced to output **2 classes** (healthy, cancer).

**5. Training Setup**

* **Epochs:** 2 (for quick test)
* **Batch Size:** 8
* **Learning Rate:** 0.0001
* **Optimizer:** Adam
* **Loss Function:** CrossEntropyLoss
* **Device:** CPU (ready to scale to GPU later)

**6. Results**

**6.1 Training Performance**

* **Epoch 1:** Loss = 0.3105, Accuracy = 88.75%
* **Epoch 2:** Loss = 0.0445, Accuracy = 98.75%

The model quickly learned the patterns due to the **small dataset (80 samples)**.

**6.2 Confusion Matrix**

On the PoC dataset (80 samples):

* **50 Healthy samples** – 50 classified correctly
* **30 Cancer samples** – 30 classified correctly
* **100% accuracy** (expected overfitting on tiny dataset)

**7. Key Code Snippets**

Below are the main code sections used during the PoC, with explanations.

**7.1 Spectrogram Generation**

def audio\_to\_mel\_spectrogram(audio\_path, output\_path, sample\_rate=22050):

y, sr = librosa.load(audio\_path, sr=sample\_rate)

y, \_ = librosa.effects.trim(y)

S = librosa.feature.melspectrogram(y=y, sr=sr, n\_mels=128)

S\_dB = librosa.power\_to\_db(S, ref=np.max)

plt.figure(figsize=(2.56, 2.56), dpi=100)

plt.axis('off')

librosa.display.specshow(S\_dB, sr=sr, cmap='inferno')

plt.savefig(output\_path, bbox\_inches='tight', pad\_inches=0)

plt.close()

This function converts a .wav file into a **256×256 Log-Mel spectrogram image** for training.

**7.2 PoC Dataset Creation**

df\_full = pd.read\_csv("metadata.csv")

df\_cancer = df\_full[df\_full['label']=='cancer'].sample(n=30, random\_state=42)

df\_healthy = df\_full[df\_full['label']=='healthy'].sample(n=50, random\_state=42)

df\_poc = pd.concat([df\_cancer, df\_healthy]).sample(frac=1, random\_state=42)

df\_poc.to\_csv("metadata\_poc.csv", index=False)

This creates a **balanced subset** (80 images) for quick PoC testing.

**7.3 ResNet-18 Training**

model = models.resnet18(pretrained=True)

model.conv1 = nn.Conv2d(3,64,kernel\_size=(7,7),stride=(2,2),padding=(3,3),bias=False)

model.fc = nn.Linear(model.fc.in\_features, 2)

criterion = nn.CrossEntropyLoss()

optimizer = optim.Adam(model.parameters(), lr=0.0001)

for epoch in range(2):

running\_loss = 0.0

correct, total = 0, 0

for inputs, labels in dataloader\_poc:

inputs, labels = inputs.to(device), labels.to(device)

optimizer.zero\_grad()

outputs = model(inputs)

loss = criterion(outputs, labels)

loss.backward()

optimizer.step()

running\_loss += loss.item()

\_, preds = torch.max(outputs, 1)

correct += (preds == labels).sum().item()

total += labels.size(0)

print(f"Epoch {epoch+1}/2 - Loss: {running\_loss/len(dataloader\_poc):.4f} - Accuracy: {(correct/total)\*100:.2f}%")

This trains the **ResNet-18 model for 2 epochs**, reporting loss and accuracy.

**7.4 Confusion Matrix Evaluation**

from sklearn.metrics import confusion\_matrix, ConfusionMatrixDisplay

model.eval()

all\_labels, all\_preds = [], []

with torch.no\_grad():

for inputs, labels in dataloader\_poc:

outputs = model(inputs.to(device))

\_, preds = torch.max(outputs, 1)

all\_labels.extend(labels.cpu().numpy())

all\_preds.extend(preds.cpu().numpy())

cm = confusion\_matrix(all\_labels, all\_preds, labels=[0,1])

disp = ConfusionMatrixDisplay(confusion\_matrix=cm, display\_labels=["Healthy","Cancer"])

disp.plot(cmap="Blues")

plt.show()

Generates the confusion matrix visual for evaluation.

**8. Insights**

* The **pipeline is fully functional** (data → spectrograms → model → evaluation).
* The PoC model **achieves 100% accuracy** on a very small subset, confirming correct setup (though overfitting is expected).
* Ready to **scale to the full dataset (~3,600 samples)** and then to **multi-class classification**.

**9. Next Steps**

1. **Prepare the full dataset** with a **70/20/10 stratified train/val/test split**.
2. Train **ResNet-50** and later **Audio Spectrogram Transformer (AST)** with PEFT on the binary classification task.
3. Expand to **multi-class classification** (adding categories like Pneumonia, COVID-19, etc.) using the same spectrogram pipeline.
4. Implement **augmentation (SpecAugment) and weighted loss functions** to handle class imbalance.
5. Log experiments and prepare visualizations (ROC curves, confusion matrices) for the dissertation report.

Would you like me to:

* **Add the actual confusion matrix image here in the Word version** (so it’s ready to present)?
* And **also make a version with full logs (errors, failed conversions, file counts) as an appendix** so you can hand that to your supervisor?

Which one do you want — **clean version**, **detailed (with logs)**, or **both**?