**Task 1 – Edge AI Prototype**

**Deliverable:** Code + Report with Accuracy Metrics and Deployment Steps

**◈ Theoretical Explanation**

Edge AI brings computation closer to the data source (IoT devices, wearables, or sensors), reducing latency and increasing efficiency. In this project, we demonstrate Edge AI by training a simple **health-monitoring model** on clinical data and deploying it for inference on-device.

**◈ Colab Code (Simplified Example)**

```
# Task 1: Edge AI Prototype

import pandas as pd

from sklearn.model_selection import train_test_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy_score


# Load dataset (already uploaded in Colab)

df = pd.read_csv("clinical_data(labels).csv")


# Split features/target

X = df.drop("label", axis=1)

y = df["label"]


# Train/test split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)


# Model training

model = RandomForestClassifier(n_estimators=100, random_state=42)

model.fit(X_train, y_train)


# Evaluation

y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)

print(f"Model Accuracy: {accuracy:.2f}")
```

```
# Save model for deployment
import joblib
joblib.dump(model, "edge_ai_model.pkl")
```

### ◈ Accuracy Metrics

Suppose after running the above, the accuracy is around:
**Model Accuracy: 0.87 (87%)**

### ◈ Deployment Steps

1. Export model (edge_ai_model.pkl)
2. Use **TensorFlow Lite** or **ONNX** to convert for mobile/IoT devices.
3. Deploy to edge hardware (e.g., Raspberry Pi or mobile app).
4. Run real-time inference locally without depending on cloud.