Project Charter Report - ML Model for E-Healthcare Monitoring

ML-Based Real-Time E-Healthcare Monitoring System

Project Title:

ML-Based Real-Time E-Healthcare Monitoring System

Project Objective:

To develop a smart healthcare monitoring system that uses machine learning models to detect health anomalies in real-time, enabling remote and proactive care, especially in underserved regions.

What the Model Will Do:

1. Classify health parameters as normal or abnormal.

2. Detect anomalies (e.g., hypoxia, fever, irregular heart rate).

3. Predict future health deterioration.

4. Trigger alerts to healthcare providers and patients.

5. Suggest preliminary health actions.

Model Inputs (Features):

- Heart Rate (beats per minute)

- SpO2 (Oxygen Saturation %)

- Body Temperature (C)

- Time-series derived metrics (e.g., HRV, temperature trends)

Model Outputs:

- Binary Class: Normal / Abnormal

- Condition Labels: e.g., Hypoxia, Tachycardia

- Risk Score (0-100)

- Action: Send Alert / Suggest Rest / No Action

Why ML is Used:

- To identify patterns and early warning signs not easily captured by fixed thresholds.

- To provide proactive and automated healthcare alerts.

- To personalize monitoring based on user trends.

- To reduce burden on hospitals and enable remote care.

How the Model Works:

1. Sensor Data Collection via IoT devices (ESP32 + sensors)

2. Data Preprocessing: Cleaning, Normalization, Feature Extraction

3. ML Model Stack:

   - Logistic Regression for simple binary classification

   - LSTM for time-series forecasting

   - Random Forest for multi-feature risk prediction

4. Alerting System: Cloud-based + On-device (TensorFlow Lite)

5. Output Integration: Real-time dashboard, SMS/email alerts

Model Performance (Targeted):

- Heart Rate Anomaly Detection: 94% accuracy

- Temperature Forecasting: 96% accuracy

- SpO2 Risk Classification: 92% accuracy

- Latency: < 0.8 seconds for prediction

- Alert Response Time: < 2 seconds

Tools and Libraries:

- Python, TensorFlow Lite, Scikit-learn, React.js, Firebase, Arduino IDE

Deployment Plan:

- On-device inference using TensorFlow Lite (for offline alerts)

- Cloud deployment for dashboard analytics and continuous learning

- Scalable Firebase backend for data storage and real-time sync

Prepared on: 2025-06-08