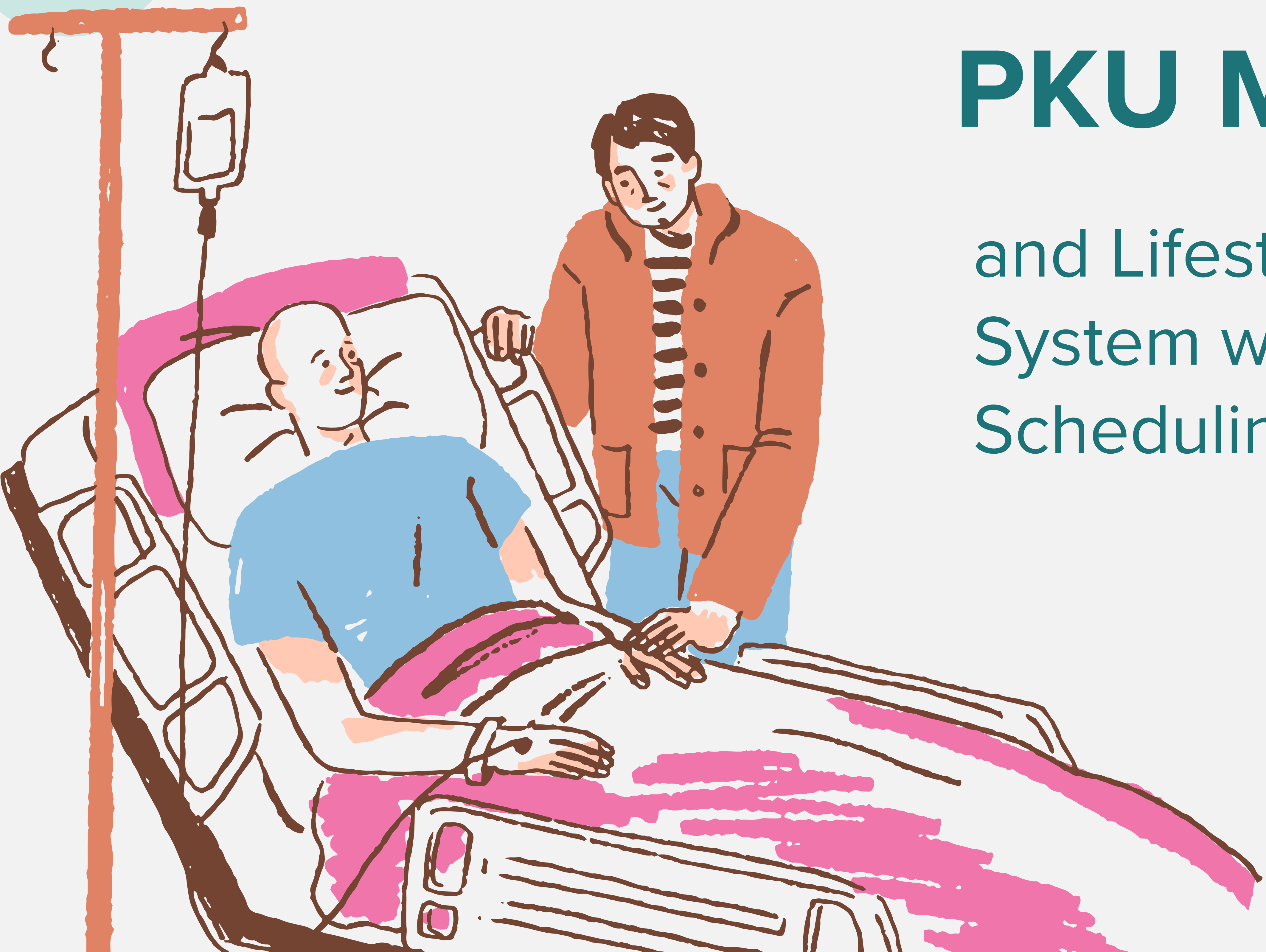
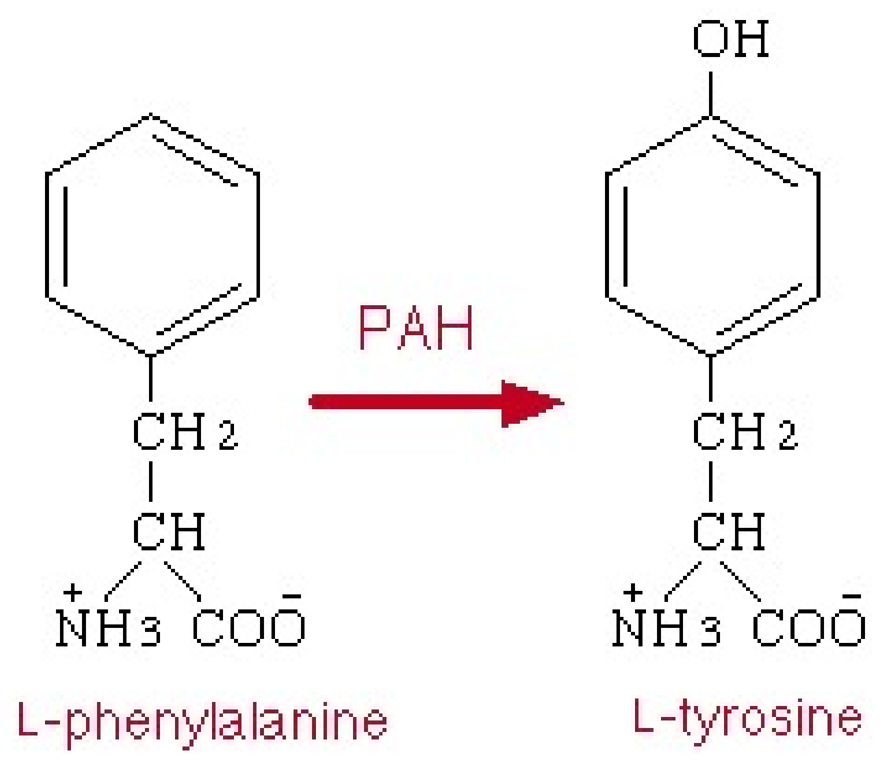


FOG-ENABLED SMART PKU MONITORING

and Lifestyle Recommendation System with Priority-Based Task Scheduling

INTRODUCTION

- Phenylketonuria (PKU) is a genetic metabolic disorder where the body can’t break down phenylalanine properly.
- If unmanaged, phenylalanine buildup can lead to severe brain damage, seizures, and intellectual disability.
- Patients must monitor phenylalanine and tyrosine levels continuously and take enzyme injections or diet changes at proper intervals.



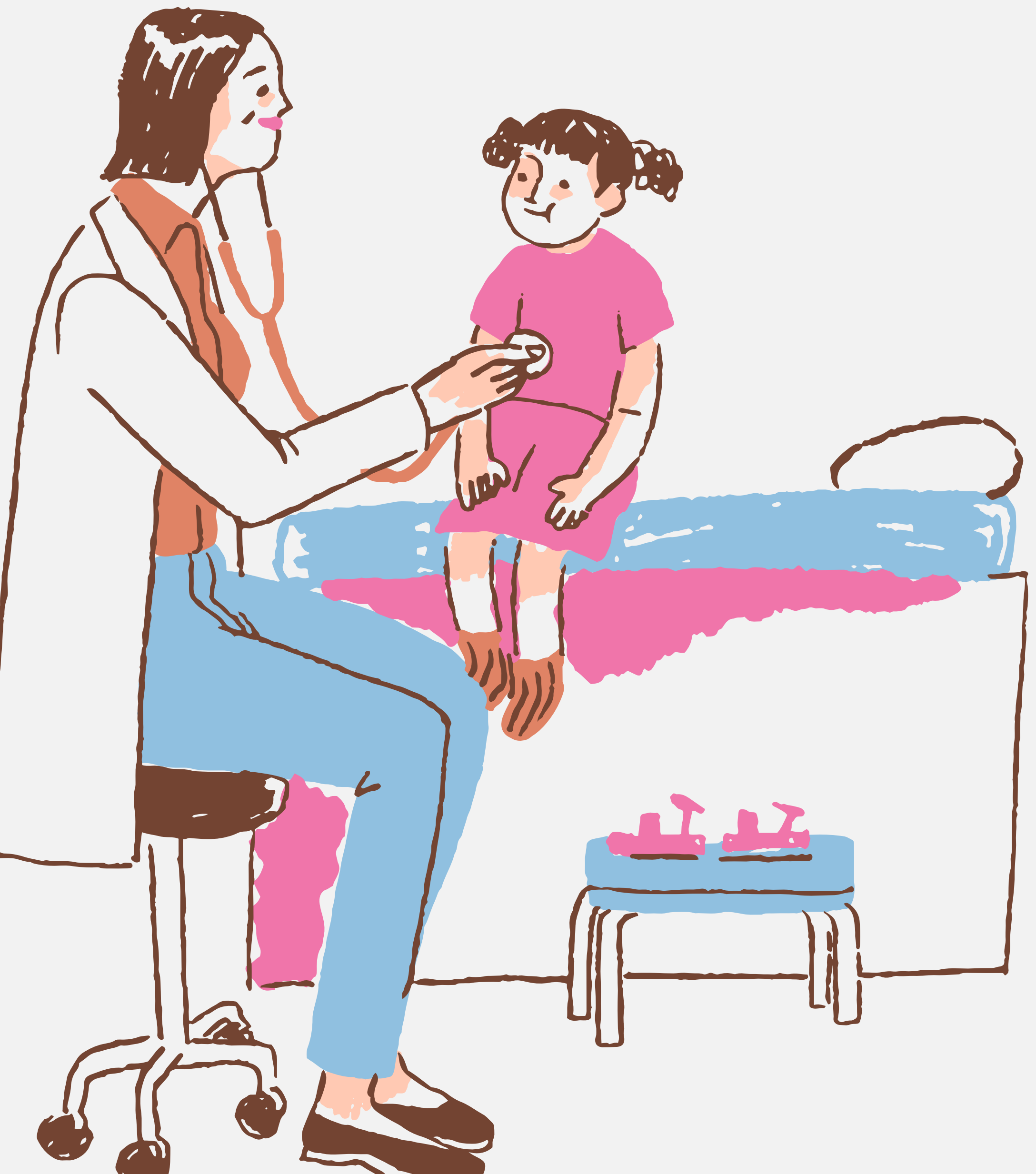
The enzyme phenylalanine hydroxylase converts the amino acid phenylalanine to tyrosine.

$$A \xrightarrow{C} B$$

Healthy Person

$$A \not\to B$$

PKU Person



- Current methods are:
  - Manual and invasive (e.g., regular blood tests).
  - Lack real-time monitoring.
  - Cause fatigue and mental stress due to constant vigilance.
- In remote/rural areas, access to hospital-based monitoring and timely intervention is limited.
- There's no intelligent system to prioritize alerts, like dangerous hormone levels vs. low battery.
- Also, internet connectivity isn't always reliable, risking failure in sending critical health data.
- Hence, we need a real-time, intelligent, and fail-safe edge–fog-based PKU monitoring system.

STRATEGY

- Wearable Patch (ESP32 + Biosensors) monitors phenylalanine and tyrosine le in real time.
- Two-Level Threshold System:
- Threshold 1 (Mild Spike):
  - Data sent to main node (fog).
  - Dietary suggestions are generated and sent to the patient.
- Threshold 2 (Critical Spike):
  - Immediate alert sent to patient (via BLE).
  - Hospital notified via fog node (via LoRa/NB-IoT).
- Backup Node Mechanism:

In case of fog node failure, data reroutes to a secondary (backup) node to ensure reliability and continuous operation.

- End-of-Day Review Report-
- Sent from device to fog node.
- Stored temporarily for offline access and summary review.

JUSTIFICATION

1. **Is this an edge-based project? How?**
   - The ESP32 wearable node does on-device threshold checks (Level 1 & 2), local decision-making, and only sends relevant data — which defines edge computing.
   - The edge reduces the need for continuous cloud communication by filtering and acting locally.
2. **Why do we need edge here?**
   - Low latency for alerts: Immediate response (e.g., BLE alert to patient) without waiting for server.
   - Bandwidth efficiency: Sends only important data (threshold breach or daily report).
   - Offline operation: Works even without internet temporarily, storing data in flash.

JUSTIFICATION

3. How is it scalable?
   - Each edge device works independently, only needing minimal config to join the system.
   - Fog node handles multiple devices, and new ones can be added easily.
4. Additional Justifications
   - Two-level thresholding reduces false alarms and avoids flooding the system.
   - Daily report strategy ensures periodic backup to Fog for medical records.
   - Backup/Standby Node ensures fault tolerance — it takes over if the primary edge node fails.

HOW ARE WE SOLVING TASK SCHEDULING HERE?

Event-driven hormone check

- When the wearable detects a spike in phenylalanine, a BLE notification is sent to the smartphone.

Immediate patient notification

- If phenylalanine is above the threshold, the app instantly alerts the patient with a warning message.
- It also provides diet suggestions to bring the level down (e.g., avoid protein-rich food).

Diet tracking task

- After every meal input (from patient), the app checks if the food may affect Phe levels

Sleep monitoring task

- Wearable tracks sleep duration and quality.

Data sync task

- The wearable and app keep track of the patient's health. Normally, the data is synced with the hospital every few hours or daily, but only when internet is available. This upload doesn’t disturb urgent tasks.

Task priority

- Tasks like Phe spike alerts and low battery notifications are given the highest priority.
- Diet and sleep suggestions are scheduled during low-load periods to avoid delay in urgent tasks

Fog Node Task Scheduling

- **Emergency alert handling**
  - If the edge device flags a high-risk Phe spike, the fog node immediately sends alerts to the hospital or caregiver.
  - This is the highest priority task in the fog system.
- **Batch ML processing**
  - If edge devices send requests for advanced ML analysis, the fog node processes them in batches to reduce load.
  - Scheduled to run during non-peak times unless triggered by an event.
- **Standby node synchronization**
  - In multi-node setups, fog nodes sync data with standby or backup nodes regularly (e.g., every 6 hours) to ensure fault tolerance.

- **Report storage and logging**
    - End-of-day reports from edge devices are stored in the fog node for long-term access or forwarding to hospital databases.

- **Battery audit logs**
    - Battery performance data from edge devices is collected every 12 hours and logged for maintenance tracking.

- **Priority queue-based scheduling**
    - All fog tasks are managed through a priority queue, ensuring urgent alerts are processed first and less critical tasks are deferred.

- **Supports multithreading (if hardware allows)**
    - Tasks may run in parallel threads for faster execution when using capable fog nodes (like Raspberry Pi or Jetson Nano).

PAPER:OVERVIEW

Real-time Personal Healthcare Data Analysis Using Edge Computing for Multimodal Wearable Sensors

Core Idea:
Development of a flexible, multimodal sensor patch that collects vital signals (ECG, respiration, temperature, humidity, activity) and analyzes them in real-time using a smartphone—with no need for cloud or internet.

Problem Addressed:

- Traditional wearable devices:
    - Generate large, complex data
    - Depend on cloud for analysis (slow, less private)
- Need for real-time, offline, low-power health monitoring
- Difficulty in integrating multimodal flexible sensors with efficient ML algorithms

Proposed Solution:

- A flexible multimodal sensor patch (ECG, respiration, temp, humidity, motion)
- Edge-computing architecture - processing done on a smartphone, not cloud
- Machine learning via:
- Echo State Network (ESN) - a lightweight reservoir computing model, Needs very little training
- Detects events like:

  Arrhythmia
  Cough
  Falls
  Activity & posture

WHAT WE ADOPTED FROM THE PAPER?

Feature in Paper	How We Applied It in Our PKU Project
Edge-based wearable health system	We use ESP32 wearable with bio-sensing
ML-based edge prediction	ML used for dietary suggestion, alerts and sleep analysis
Real-time analysis on edge	Fog and Edge nodes do low-latency alerting
Smartphone interface	App used for food logging/alerting the patient

MODIFICATIONS AND INNOVATIONS IN OUR PROJECT

- PKU-specific chemical sensors (Phe/Tyr)
- Task scheduling
- Standby node for load/failure handling
- Remote patient gateway (LoRa)
- Hospital alert system
- Diet based ML suggestion model

RELATED WORK ON PKU WEARABLE MONITORING

- **Parrilla et al., Biosens. Bioelectron. (2022)**- https://pubmed.ncbi.nlm.nih.gov/34753096
- **Messina et al., ACS Sensors (2023)**- https://www.researchgate.net/publication/375032516_Fully_Integrated_Point-of-Care_Platform_for_the_Self-Monitoring_of_Phenylalanine_in_Finger-Prick_Blood
- **Nature Commun. (2024) – sweat-Phe biochip**- https://www.nature.com/articles/s41467-024-44751-z

PLAN OF ACTION

1. Problem Definition and Requirement Analysis.
2. Define Edge-Fog Architecture.
3. Simulate Sensor Data Generation.
4. Implement 2-Level Threshold Logic on Edge.
5. Machine Learning Simulation.
6. End-of-Day Report Simulation.
7. Simulate Communication Between Nodes.
8. Edge System Without Internet.

CONCLUSION

This project presents a real-time healthcare monitoring system using edge and fog computing. By simulating a sensor patch we ensured realistic input conditions. The smartphone acts as the edge device, the fog computing layer comprising a main hospital node and backup nodes that handles task scheduling, deeper ML analysis, alerts and daily reports. This end-to-end setup ensures fast, reliable, and scalable health monitoring system.