# Precision Livestock Farming System

## Code 1: Livestock Health Monitoring

import random  
import time  
  
# Simulated sensor data for a group of animals  
livestock\_data = [  
 {"id": "Cow 001", "temperature": 38.5, "activity": 70, "heart\_rate": 65},  
 {"id": "Cow 002", "temperature": 39.2, "activity": 30, "heart\_rate": 80},  
 {"id": "Cow 003", "temperature": 40.1, "activity": 20, "heart\_rate": 95}, # abnormal  
 {"id": "Cow 004", "temperature": 37.8, "activity": 85, "heart\_rate": 60}  
]  
  
# Thresholds for abnormal health  
TEMP\_THRESHOLD = 39.5  
ACTIVITY\_THRESHOLD = 25  
HEART\_RATE\_THRESHOLD = 90  
  
# Function to analyze data  
def analyze\_livestock(data):  
 alerts = []  
 for animal in data:  
 abnormal = []  
 if animal["temperature"] > TEMP\_THRESHOLD:  
 abnormal.append("High Temp")  
 if animal["activity"] < ACTIVITY\_THRESHOLD:  
 abnormal.append("Low Activity")  
 if animal["heart\_rate"] > HEART\_RATE\_THRESHOLD:  
 abnormal.append("High Heart Rate")  
 if abnormal:  
 alerts.append({  
 "id": animal["id"],  
 "issues": abnormal,  
 "status": "Alert"  
 })  
 else:  
 alerts.append({  
 "id": animal["id"],  
 "issues": ["Normal"],  
 "status": "OK"  
 })  
 return alerts  
  
# Display results  
def display\_dashboard(results):  
 print("=== Precision Livestock Monitoring Dashboard ===\n")  
 for result in results:  
 print(f"Animal ID : {result['id']}")  
 print(f"Status : {result['status']}")  
 print(f"Issues : {', '.join(result['issues'])}")  
 print("----------------------------------------------")  
  
# Simulate a reading every few seconds  
if \_\_name\_\_ == "\_\_main\_\_":  
 print("Starting Livestock Health Monitoring...\n")  
 time.sleep(1)  
 analysis = analyze\_livestock(livestock\_data)  
 display\_dashboard(analysis)

## Code 2: Real-time Sensor Monitoring and Alerts

import time  
import random  
import matplotlib.pyplot as plt  
  
# Simulated sensor reading function  
def read\_sensors():  
 temperature = random.uniform(20.0, 35.0) # Ambient temp (simulated)  
 humidity = random.uniform(20.0, 60.0) # Humidity (simulated)  
 animal\_temp = random.uniform(36.5, 40.0) # Animal body temp (simulated)  
 return temperature, humidity, animal\_temp  
  
# Function to send alerts  
def send\_alert(message):  
 print(f'ALERT: {message}')  
  
# Initialize data lists  
temperature\_data = []  
humidity\_data = []  
animal\_temp\_data = []  
  
# Real-time monitoring loop (simulated)  
run = True  
while run:  
 temperature, humidity, animal\_temp = read\_sensors()  
  
 temperature\_data.append(temperature)  
 humidity\_data.append(humidity)  
 animal\_temp\_data.append(animal\_temp)  
  
 if temperature > 30:  
 send\_alert('High ambient temperature detected!')  
 if humidity < 30:  
 send\_alert('Low humidity detected!')  
 if animal\_temp >= 39:  
 send\_alert('High animal temperature detected!')  
  
 # Plotting real-time data  
 plt.clf()  
 plt.plot(temperature\_data, label='Ambient Temperature')  
 plt.plot(humidity\_data, label='Humidity')  
 plt.plot(animal\_temp\_data, label='Animal Temperature')  
 plt.legend()  
 plt.pause(1)  
  
 if len(temperature\_data) > 50:  
 run = False # Stop after 50 readings (for simulation)

## Code 3: Same as Code 2 (Repeated for clarity)

import time  
import random  
import matplotlib.pyplot as plt  
  
# Simulated sensor reading function  
def read\_sensors():  
 temperature = random.uniform(20.0, 35.0) # Ambient temp (simulated)  
 humidity = random.uniform(20.0, 60.0) # Humidity (simulated)  
 animal\_temp = random.uniform(36.5, 40.0) # Animal body temp (simulated)  
 return temperature, humidity, animal\_temp  
  
# Function to send alerts  
def send\_alert(message):  
 print(f'ALERT: {message}')  
  
# Initialize data lists  
temperature\_data = []  
humidity\_data = []  
animal\_temp\_data = []  
  
# Real-time monitoring loop (simulated)  
run = True  
while run:  
 temperature, humidity, animal\_temp = read\_sensors()  
  
 temperature\_data.append(temperature)  
 humidity\_data.append(humidity)  
 animal\_temp\_data.append(animal\_temp)  
  
 if temperature > 30:  
 send\_alert('High ambient temperature detected!')  
 if humidity < 30:  
 send\_alert('Low humidity detected!')  
 if animal\_temp >= 39:  
 send\_alert('High animal temperature detected!')  
  
 # Plotting real-time data  
 plt.clf()  
 plt.plot(temperature\_data, label='Ambient Temperature')  
 plt.plot(humidity\_data, label='Humidity')  
 plt.plot(animal\_temp\_data, label='Animal Temperature')  
 plt.legend()  
 plt.pause(1)  
  
 if len(temperature\_data) > 50:  
 run = False # Stop after 50 readings (for simulation)