Grenze International Journal of Engineering and Technology, June Issue

Smart Milk Testing for Milk Quality Monitoring

Harshavardhan M[1], Rakshitha G D[2], Pavithra K[3], Vedashri P[4] and Pooja M R[5]
[1-5]Vidyavardhaka College of Engineering, Mysuru, Karnataka, India.
Email: harshamahesh49@gmail.com rakshithagd571@gmail.com pavithrak12122003@gmai.com vedashrip4@gmail.com pooja.mr@vvce.ac.in

***Abstract*— The adulteration of milk is one of the major challenges to food safety, as the addition of adulterants affects its nutritional quality. Traditional methods for detection have remained dominantly laboratory-based and hence not suitable for real-time procurement environments. This paper proposes an IoT-enabled milk quality assessment system, comprising sensors such as temperature, pH, electrical conductivity, turbidity, and RGB color. The data from multiple sensors is processed to calculate a milk purity percentage, which is used to create a pricing recommendation that will be transparent and unbiased toward the farmers. The proposed system will introduce greater accountability and enhance trust along the value chain of dairy products, thereby offering a scalable solution for intelligent monitoring of the quality of milk.**

***Keywords*— Internet of Things (IoT), Real-time monitoring/evaluation, Multiple Sensors , Sensor Integration, System Design, Requirement Specifications.**

I. INTRODUCTION

The dairy sector plays a vital role in the agricultural economy, especially in countries like India. Milk quality directly affects consumer health and farmer income. However, existing systems rely heavily on manual testing methods using chemical reagents and basic instruments, which are inefficient and error-prone.
In many rural areas, the lack of transparency in milk quality testing leads to mistrust between farmers and dairy collection centers. Additionally, manual calculation of payments increases the chances of financial discrepancies. To overcome these challenges, this project introduces an IoT-based solution that automates milk quality analysis and accounting using simulated sensor data. The system eliminates manual intervention, improves accuracy, and provides real-time monitoring and reporting.

II. LITERATURE SURVEY

- Several IoT-based systems have been developed for milk quality monitoring using sensors such as pH, temperature, and turbidity sensors. These systems provide real-time monitoring and improve accuracy in milk quality assessment. Technologies like Arduino, ESP32, and Raspberry Pi are commonly used for data collection and transmission.
- Existing systems mainly focus on quality detection and cloud monitoring but require expensive hardware and regular maintenance. Some studies also explored AI and Machine Learning techniques for predicting milk quality and spoilage using sensor data.
- However, many existing solutions lack integrated accounting and payment management features. They also provide limited support for simulation-based testing.

Grenze ID: 01.GIJET.12.2.673_26

- To overcome these limitations, the proposed system uses simulated IoT data for milk quality analysis along with automated accounting and reporting, providing a cost-effective and scalable solution.

III. Functional Requirements

Functional requirements describe the main operations and services provided by the system. These requirements are based on the Python simulator and React-based web application.

A. User Authentication and Management

- The system shall provide secure login access for Administrators and Milk Suppliers.
- Admin users shall manage suppliers, thresholds, and transaction records, while suppliers can access only their personal dashboard and payment details.
- The system shall allow supplier registration with unique IDs and contact details.
- User passwords shall be securely stored in the database.

B. IoT Data Simulation

- The system shall generate simulated sensor values for Temperature, pH, Turbidity, RGB Frequency, and Weight.
- Users shall be able to configure sensor ranges using the Tkinter interface.
- Generated data shall be formatted in JSON format for communication.
- The simulator shall support configurable publishing intervals for real-time data generation.

C. Data Communication (MQTT)

- The system shall use the MQTT protocol for transmitting sensor data.
- The simulator shall connect to the HiveMQ public broker.
- Sensor data shall be published to a predefined topic for the React application.
- The system shall provide connection status and error handling during data transmission.

D. Quality Analysis and Grading

- The application shall compare sensor values with predefined thresholds.
- Milk quality shall be classified as Excellent, Good, Average, or Rejected.
- Abnormal values shall be detected as possible adulteration cases.
- Quality analysis shall be performed in real time.

E. Automated Accounting and Settlement

- The system shall calculate payment based on milk quality and quantity.
- Administrators shall manage pricing rates for each quality grade.
- The system shall support Daily, Weekly, and Monthly settlements.
- All transactions shall be stored with supplier and payment details.
- Suppliers shall be able to request payment withdrawals.

F. Reporting and Visualization

- The React dashboard shall display real-time sensor data and system status.
- Users shall be able to view supply history and transaction logs.
- The system shall generate financial summaries and reports.
- Reports shall be displayed in tabular format within the application.

G. Hardware Requirements

Since this project utilizes a Simulated IoT approach, the hardware requirements focus on the development and execution environment rather than physical sensors. However, the system is designed to be compatible with physical hardware in future iterations.

Recommended System Requirements

- Processor: Intel Core i5 (10th Generation) or higher
- RAM: 8 GB or higher
- Storage: 512 GB SSD
- Display Resolution: 1920 × 1080 (Full HD)
- Network: High-Speed Broadband Connection

- Input Devices: Standard Keyboard and Mouse

Server Requirements
- Localhost or Cloud VPS for deployment
- Minimum 20 GB free storage
- Stable internet connection for MQTT communication

H. Future Hardware Support
- Microcontroller: ESP32 or Arduino Uno
- Sensors: pH, Temperature, Turbidity, and Load Cell Sensors
- Communication Module: Wi-Fi or GSM Module

I. Software Stack
- Operating System: Windows 10/11 – Host Environment
- Python 3.8+: Data Generation & MQTT
- React.js 18+: User Interface
- MySQL 8.0: Data Storage
- MQTT 3.1.1: Data Transmission
- VS Code (Latest): Development
- HiveMQ Public Broker: Message Routing

J. Data Flow
- Sensor data is generated in Python
- Data is sent via MQTT broker
- React app receives and processes data
- Quality is analyzed
- Payment is calculated
- Data is stored in database

IV. Data Flow Diagrams Includes

A. Context Diagram (Level 0 DFD)
- The Context Diagram represents the overall working of the Smart Milk Quality System and its interaction with Admin, Farmer, and Simulator modules.

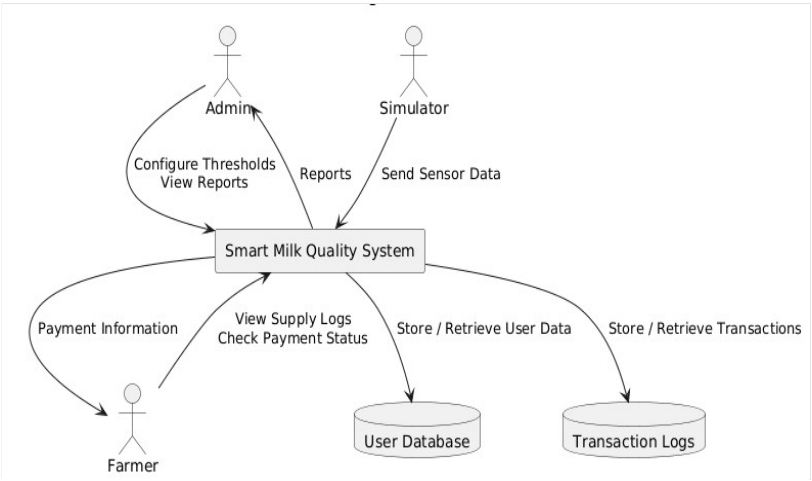

Fig 1

B. Level 1 DFD

The Level 1 DFD illustrates the internal processes of the system including authentication, data simulation, quality analysis, accounting, and reporting.

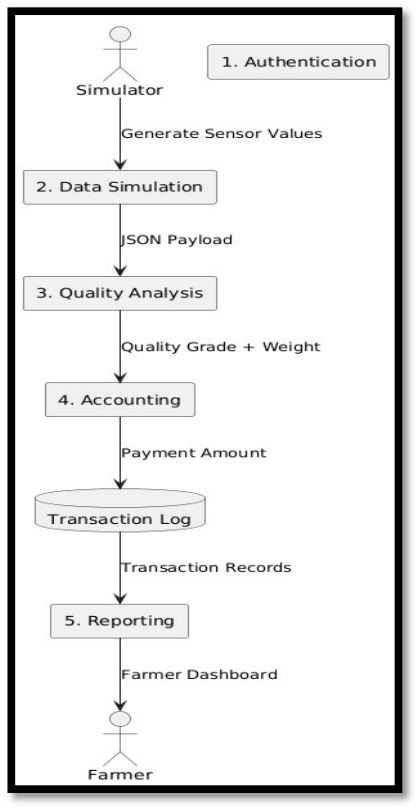

Fig 2

C. Use Case Diagram

The Use Case Diagram shows the interaction between Administrator, Farmer, and System modules for login, quality analysis, report viewing, and payment processing.

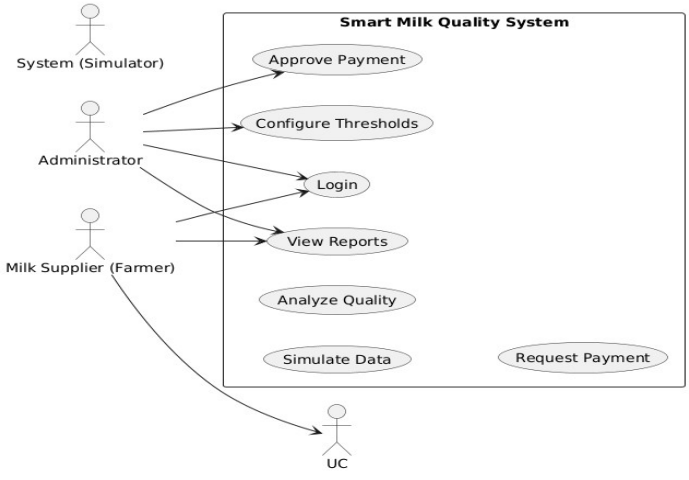

Fig 3

D. Class Diagram

The Class Diagram represents the structure of the system including user management, sensor data handling, MQTT communication, and transaction processing classes.

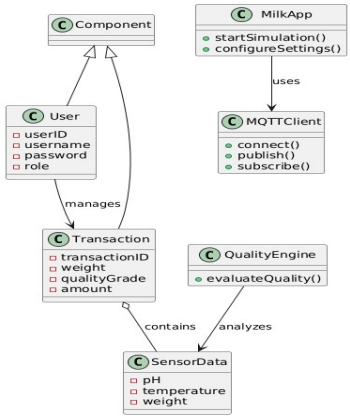

Fig 4

E. Sequence Diagram

The Sequence Diagram illustrates the flow of sensor data from the simulator to the React application through MQTT, followed by quality analysis and database storage.

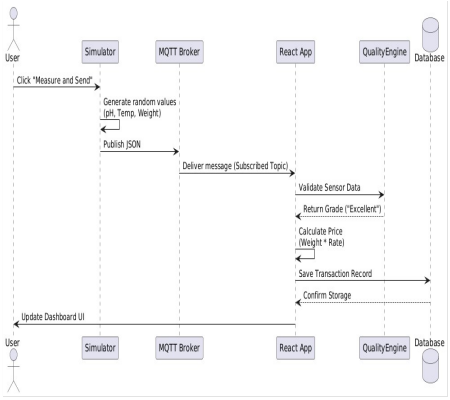

Fig 5

F. Activity Diagram

The Activity Diagram describes the workflow of farmer login, viewing supply records, and requesting payment settlements.

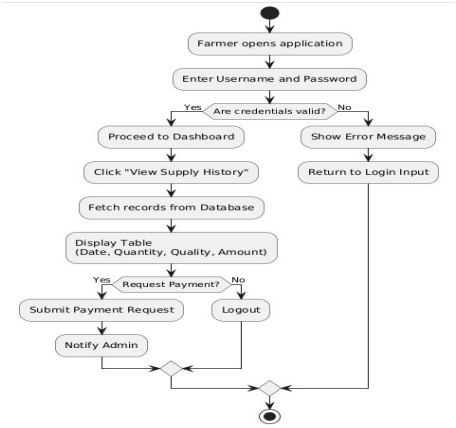

Fig 6

G. Entity Relationship Diagram

The ER Diagram represents the database structure of the system including users, suppliers, transactions, sensor logs, and payment requests.

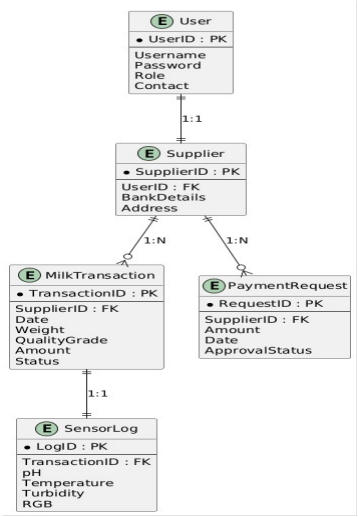

Fig 7

V. IMPLEMENTATION STEPS

- Install Python, Node.js, MySQL
- Configure development environment
- Create database tables
- Develop Python simulator
- Build React frontend
- Integrate MQTT communication
- Connect frontend with database
- Test and deploy system

A. Development Environment Setup

The development environment was configured to ensure all tools and libraries were properly installed.

- Python (3.8+) installed with libraries like paho-mqtt, tkinter, and json
- Node.js and npm used for React setup
- MySQL installed with phpMyAdmin for database management
- VS Code configured with required extensions
- HiveMQ public MQTT broker used for communication testing

B. Database Implementation

The database stores user data, transactions, and sensor records.

- Created database milk_management
- Tables: users, suppliers, transactions, sensor_logs, payment_requests
- Used primary and foreign keys for data integrity
- Inserted initial admin and sample data for testing

C. Simulator Module Implementation

The simulator generates IoT data using Python.

- Tkinter GUI created for input ranges
- Random sensor values generated using function
- MQTT used to send data in JSON format
- Settings saved in file for reuse

D. Frontend Implementation

The UI was developed using React.js.

- Components: Login, Dashboard, SupplyTable, PaymentHistory
- Used React Hooks for state management
- MQTT subscription for real-time data
- Axios used for backend communication
- Styled using CSS and Bootstrap

E. Integration and Deployment

All modules were integrated and tested.

- Verified data flow from Simulator → MQTT → React → Database
- Tested quality grading logic
- Verified payment calculations
- Deployed system on localhost

F. Algorithms

Algorithm 1: Milk Quality Grading

- Input: pH, Temperature, Turbidity
- Process:
  - Check pH range
  - Check temperature
  - Evaluate turbidity
- Output: Quality grade

Algorithm 2: Payment Calculation

- Input: Weight, Quality
- Process:
  - Select rate based on quality
  - Multiply with weight
- Output: Total amount

Algorithm 3: MQTT Data Transmission

- Generate JSON data
- Connect to broker
- Publish data
- Disconnect

G. Testing

Types of Testing

- Unit Testing
- Integration Testing
- System Testing
- Functional Testing

Test Cases

- Sensor data generation → PASS
- MQTT communication → PASS
- Login system → PASS
- Payment calculation → PASS

Performance Testing

- Fast response time
- Handles multiple users
- Real-time updates within seconds

VI. Results and Discussion

The system was successfully implemented and tested, showing effective simulation of IoT-based milk quality monitoring. The Python simulator generated realistic sensor data, and MQTT enabled smooth real-time data transmission to the web application.

The system accurately classified milk quality into different grades using threshold values, ensuring consistent and reliable results. The automated payment calculation reduced manual effort and minimized errors in financial processing.
Additionally, the system improved transparency by allowing farmers to view their data and earnings in real time, reducing disputes with administrators.
Overall, the project proved to be efficient, cost-effective, and scalable for dairy management.

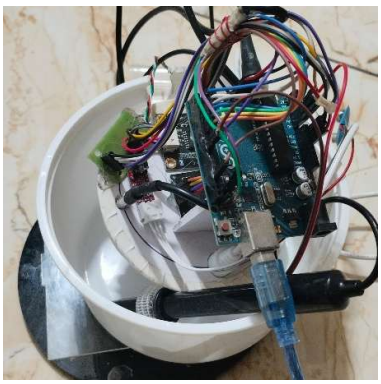

Fig 8

A. Challenges Faced

- MQTT connection failures caused interruptions in real-time data transmission
- Difficulty in handling real-time data updates in React
- Data synchronization issues between simulator, frontend, and database
- Unrealistic sensor values due to completely random data generation

B. Solutions

- Added proper error handling to manage MQTT connection issues
- Implemented data validation to ensure accurate and consistent data
- Improved state management in React for smooth real-time updates
- Used controlled random ranges to generate realistic sensor values

VII. CONCLUSION

The project titled "Smart Milk Quality Analysis using AI-ML & Automated Accounting Using Simulated IoT" was successfully designed and developed to address challenges in traditional milk collection systems.
The dairy industry is important for many farmers, but existing systems suffer from lack of transparency, manual errors, and inefficient payment processes. These issues often lead to financial losses and mistrust.
The proposed system overcomes these problems using a software-based IoT simulation approach. By using Python for data simulation, MQTT for communication, React.js for the interface, and MySQL for storage, the system provides an efficient and cost-effective solution for milk quality analysis and automated accounting.

REFERENCES

[1] Vaidya, S., & Deshpande, P., "IoT Based System for Milk Quality Assessment and Adulteration Detection", Journal/Conference Name, Year.
[2] Ashwini, et al., "Milk Quality Analysis and Grading Using IoT", Journal/Conference Name, Year.
[3] "IoT Based Dairy Data Management System Using Raspberry Pi", International Journal of Engineering Research & Technology, Year.
[4] Research papers on Artificial Intelligence and Machine Learning in Dairy Farming, Various Authors, Year.
[5] Python Official Documentation – https://www.python.org
[6] React.js Official Documentation – https://reactjs.org
[7] MySQL Official Documentation – https://www.mysql.com
[8] MQTT Documentation – https://mqtt.org